

Entwicklung und Test einer eingebetteten Elektronik für einen innovativen Schaltaktor

Malte Breitenbach, Johannes Faupel, Jonas Tautz, Johanna Vetter



**TECHNISCHE
UNIVERSITÄT
DARMSTADT**

Institut für Mechatronische Systeme im
Maschinenbau
Prof. Dr.-Ing. Stephan Rinderknecht

31. März 2019

Kurzfassung

Die vorliegende Arbeit befasst sich mit der Entwicklung und der Realisierung einer eingebetteten Elektronik für einen linearen Tauchspulenaktor. Dieser wird am Prüfstand des Instituts für Mechatronische Systeme eingesetzt um die Schaltgabel eines Getriebes translatorisch zu bewegen. In vorangegangenen Arbeiten wurde bereits eine Regelung der Schaltgabelposition entworfen. Auf Basis dessen erfolgt zunächst die Implementierung der bisherigen Programmierung mittels einer geeigneten Toolchain in Matlab-Simulink auf einem Mikrocontroller, der von nun an die Steuerung der Schaltgabel übernehmen soll. Mithilfe von CAN-Kommunikation soll der Mikrocontroller einerseits Statusmeldungen und besonders Fehlzustände übermitteln können, andererseits bietet die CAN-Schnittstelle die Möglichkeit, dem Mikrocontroller Befehle zu senden. Im nächsten Schritt wird die für die Regelstrecke benötigte Elektronik, die Aktoransteuerung, Sensorik sowie Spannungsversorgung, umfasst an den Aktor integriert, indem die gesamte Schaltung auf einer eigens entworfenen Platine verwirklicht wird, welche anschließend in einem Gehäuse liegend am Motor verschraubt wird. Zusätzlich wird ein Temperatursensor integriert, der Übertemperaturen erkennen kann sowie einige bauteilinternen Funktionen, beispielsweise eine Strommessung innerhalb der verwendeten Halbbrücken, genutzt. Das Ergebnis der Arbeit ist ein Smart Actuator inklusive kompakter eingebetteter Elektronik, der auf einfachen Befehl über die Benutzeroberfläche Controldesk selbstständig die zwei Gänge einlegen oder in Neutralstellung schalten kann.

Schlagwörter: Smart Actuator, eingebettetes System, Linearaktor, Platinendesign

Abstract

This thesis deals with the evaluation and implementation of an embedded electronics for a linear plunger coil actuator. This actuator is used at the test bench of the Institute of Mechatronic Systems to move the shift fork of a gearbox. In previous work a control of the shift fork position has already been developed. On the foundation of this, the implementation of the previous programming is carried out using a suitable toolchain in Matlab-Simulink on a microcontroller, which henceforth will take over the control of the shift fork. With the use of CAN communication, the microcontroller should be able to transmit status messages and especially error states but also the CAN interface gives the opportunity to send commands directly to the microcontroller. In the next step, the electronics required for the controlled system, including the actuator control, sensor technology and power supply, are integrated into the actuator by realizing the entire circuit on a specially designed circuit board, which is then mounted to the motor in a case. In addition, a temperature sensor, which can detect overtemperatures, is integrated as well as some internal functions, for example the current measurement within the used half bridges. The result of the work is a Smart Actuator including compact embedded electronics, which can engage the two gears independently or shift to neutral position with a simple command via the user interface Control Desk.

Keywords: smart actuator, embedded system, linear actuator, PCB design



Abbildungsverzeichnis

2.1	Prüfstand [adp]	3
2.2	Fahrzeuggetriebe	4
2.3	Querschnitt Tauchspulenaktor [Hahn2018]	4
2.4	Kraft-Weg-Strom Kennlinien des Tauchspulenaktors [adp]	5
2.5	Einbauposition PLCD Sensor an der Schaltgabel [adp]	6
2.6	Sprungantwort Schaltgabelposition [adp]	7
2.7	Blockschatzbild eines Mikrocontrollers [Bernstein2015]	8
2.8	Aufbau CAN-Bus [manual]	9
2.9	Beispielhaftes PWM Signal	10
3.1	V-Modell nach [Boehm79]	12
3.2	Klassifikationsbaum am Beispiel des Motortreibers	13
3.3	Bildliche Darstellung des Prototypings (Iterationsprozesses)	14
4.1	Benötigte Ebenen der Verschaltung	15
4.2	Minimalbeschaltung des STM32F405RGT7	16
4.3	Pin-Belegung des Mikrocontrollers	17
4.4	Anschluss externer Quarz	18
4.5	CAN-Transceiver Verschaltung	18
4.6	Blockschatzbild LDO	19
4.7	Schaltplan Spannungsversorgung	20
4.8	Anschlusspins Stecker	20
4.9	Vereinfachter Aufbau einer H-Brücke	21
4.10	Blockdiagramm der BTN8982 Halbbrücke	21
4.11	Funktionsweise des IS-Pins	22
4.12	Schematischer Aufbau und Logiktabelle des CD74HCT125 Leitungsverstärkers	23
4.13	Strommessung mit und ohne Leitungsverstärker bei unterschiedlichen PWM-Signalen	24
4.14	Schematischer Aufbau der H-Brücke	24
4.15	(a):Simulationsergebnisse des durchgeschaltetet Stroms aufgetragen über der Zeit,(b): Welligkeit des durchgeschalteten Stroms	25
4.16	Wheatstone-Brückenschaltung	26
4.17	Schaltung zum Auslesen des Thermistors	27
5.1	Derzeitiger Entwurf des Elektronikgehäuses	29
5.2	Rückseite des Elektronikgehäuses zur Darstellung der Aktoranbindung	30
5.3	Platzierungsgrafik der Bauteile	31
5.4	Dimensionierungsauswirkungen Temperatur und Leitungsquerschnitt	32
5.5	Darstellung einer zwei Layer Platine inklusive Via	34
5.6	Logikboard zur Endplatine inklusive Versorgungsleitungen	34
5.7	Darstellung des Top-Layers	35
5.8	Darstellung des Bottom-Layers	36
5.9	Top-Layer nach dem Fertigungsprozess	37
5.10	Bottom-Layer nach dem Fertigungsprozess	37
6.1	Vereinfachte Darstellung der eingesetzten Toolchain	39
6.2	Drei-Ebenenarchitektur nach Rasmussen (vereinfacht)	40
6.3	Darstellung des Gesamtsystems	41
6.4	Hauptzustandsautomat (MAIN), Außenansicht	41
6.5	Hauptzustandsautomat (MAIN), vereinfachter Aufbau	43
6.6	Schaltgabelpositionsermittlung	43
6.7	CAN-Kommunikation	44
6.8	Sensorik	44

6.9 Fehlererkennung	45
6.10 Kalibrierung (Lagesensor)	45
6.11 Regelung	46
6.12 Auswahlglied für PWM-Signal	46
6.13 Motortreiber	47
6.14 Taktbaum generiert in STM32 CubeMX	48
6.15 Gesamtkonzept der CAN-Kommunikation	49
7.1 Aufteilung der Kosten für die Stückzahlen 1 (links) und 1000 (rechts)	55
7.2 Position, Regelabweichung und Stellgröße für das Schalten in Gang 2	57
7.3 Position, Regelabweichung und Stellgröße für das Schalten in Neutral	58
7.4 Strommessung im Schaltvorgang von Neutral in Gang 2	59
8.1 Ergebnis der eingebetteten Elektronik und Schnittstellen	61

Inhaltsverzeichnis

Kurzfassung	i
Abbildungsverzeichnis	iv
Inhaltsverzeichnis	vi
Nomenklatur	vii
1 Einleitung	1
1.1 Motivation und Ziele der Arbeit	1
1.2 Anforderungsliste	1
1.3 Aufbau der Arbeit	1
2 Aufbau des Prüfstands und Grundlagen	3
2.1 Grundlagen des Prüfstandes	3
2.1.1 Getriebe allgemein	3
2.1.2 Getriebe des Prüfstands	3
2.1.3 Aktor	4
2.1.4 Aktoransteuerung	5
2.1.5 Autobox	5
2.1.6 Positionsmessung und -regelung	6
2.2 Grundlagen Löten	6
2.3 Mikrocontroller	7
2.4 Eingebettete Systeme und Smart Actuators	7
2.5 Controller Area Network (CAN)	8
2.6 Pulsweitenmodulation	9
2.7 Zustandsautomaten	10
2.8 Serial Wire Debug	10
3 Vorgehensweise und Testverfahren	11
3.1 Entwicklungsmodell	11
3.1.1 Das V-Modell	11
3.1.2 Konkretes Vorgehen mit inkrementellem Entwicklungsansatz	13
4 Auswahl elektronischer Komponenten und Verschaltung	15
4.1 Anforderungen an die Komponenten	15
4.1.1 Passive Bauteile	15
4.1.2 Aktive Bauteile	15
4.2 Komponentenauswahl	16
4.2.1 Mikrocontroller	16
4.2.2 CAN-Transceiver	18
4.2.3 Spannungsversorgung	18
4.3 Platinenanschluss	20
4.4 H-Brücke	20
4.5 Sensorik	25
4.5.1 Temperatursensor	25
4.5.2 Lagesensor	27
4.5.3 Sensorik Eingangsspannung	28

5 Platinenentwurf	29
5.1 Elektronikgehäuse und Anforderungen an die Platinendimensionierung	29
5.2 Komponentenplatzierung auf der Platine	30
5.3 Dimensionierung der Leiterbahnen	31
5.3.1 Anwendung auf realer Platine	33
5.4 Routen der Platine	33
5.4.1 Design-Regeln und EMV	33
5.4.2 Leiterbahnverlegung des Entwicklungsboards	35
5.5 Fertigung	37
6 Software	39
6.1 Toolchain	39
6.2 Vorstellung des Hauptprogramms	39
6.2.1 Gesamtsystem	40
6.2.2 Hauptzustandsautomat (MAIN)	41
6.2.3 Positionsermittlung (POS)	43
6.2.4 CAN-Kommunikation (CAN)	44
6.2.5 Sensorik (SEN)	44
6.2.6 Fehlererkennung (ERR)	45
6.2.7 Kalibrierung (CAL)	45
6.2.8 Regelung (CONS/CONH)	46
6.2.9 Auswahlglied für PWM-Signal (PWM)	46
6.2.10 Motortreiber (MOT)	47
6.3 Codegenerierung und Performanceoptimierung	47
6.4 Einleitung in benutzerseitige CAN Schnittstelle	48
7 Analyse und Performance	53
7.1 Produktvergleich mit Anforderungsliste	53
7.2 Kostenaufstellung	54
7.3 Platinendesign-Analyse	55
7.4 Regelergebnisse	56
7.4.1 Schalten in Gang 2	56
7.4.2 Schalten in Neutral	56
7.4.3 Diskussion der Regelergebnisse	56
7.4.4 Schaltgabelkraft am Anschlag	58
8 Fazit und Ausblick	61
8.1 Ausblick	62

Nomenklatur

Abkürzungen

ADC- Analog/Digital Wandler
ADP – Advanced Design Project
AEC- Automotive Electronics Council
BF – Bereichsforderung
CAN – Controller Area Network
CPU- central processing unit
CSMA/CR- carrier sense multiple access/collision resolution
EMV- Elektromagnetische Verträglichkeit
FF – Festforderung
FPGA- Field Programmable Gate Array
GND- Ground
IC- Integrated Circuit
ID- Identifier
IEEE- Institute of Electrical and Electronics Engineers
IMS – Institut für Mechatronische Systeme
IPC- Association Connecting Electronics Industries
JTAG- Joint Test Action Group
LDO- Low Dropout
MCU – microcontroller unit
MOSFET- Metall-Oxid-Halbleiter-Feldeffekttransistor
NKW- Nutzkraftwagen
NTC- Negative Temperature Coefficient
PC- Personal Computer
PCB- printed circuit board
PID- proportional-integral-derivative (controller)
PKW- Personenkraftwagen
PLCD – permanentmagnetische linear contactless displacement
PWM- Pulsweitenmodulation
RAM- random access memory
ROM- read only memory
RTC- real-time clock
SMD- surface-mounted device
SRAM- Static random-access memory
STM- STMicroelectronics
SWD- Serial Wire Debugging
THT- through hole technology
W- Wunsch
ZF – Zielforderung

1 Einleitung

1.1 Motivation und Ziele der Arbeit

Dass elektrifizierte Antriebssysteme auf lange Sicht stark an Bedeutung gewinnen werden, steht für die Akteure der Automobilbranche außer Frage. Doch obwohl Elektroautos schon seit Jahren auf dem Markt vertreten sind, bleibt die Nachfrage gering. Besonders bezüglich der Aspekte Kosten und Reichweite bestehen noch große Mängel, die die Akzeptanz und Verbreitung einschränken, obwohl seitens Politik und Industrie immer weiter in die Elektromobilität investiert wird. Um die Defizite der E-Mobilität zu beheben wird seit Jahren ausgiebig an der Effizienz und den Kosten von elektrischen Automobilen geforscht und gearbeitet. Auch das Institut für Mechatronische Systeme (IMS) der TU Darmstadt nimmt sich dieser Aufgabe an und arbeitet im Rahmen des Projektes Speed4E (Nachfolgerprojekt von Speed2E), das einen elektrifizierten Antriebsstrang mit Peak-Antriebsdrehzahlen von bis zu 50.000 min^{-1} zum Ziel hat, an einem innovativem Schaltaktor. Das Projekt testete dabei einen neuartigen Antriebsstrang, bestehend aus zwei elektrischen Antriebeinheiten, deren Leistung sich über zwei parallelen Teilgetriebe, von denen eines schaltbar ist, summiert. Denn nicht nur in konventionellen Antrieben, sondern auch in elektrifizierten und hybriden Antrieben optimieren Getriebe und die damit einhergehende Einstellung der Drehzahl die Fahrleistung und die Effizienz [Tsch14]. Im Verlauf vorheriger Arbeiten wurden bereits ein linearer Tauchspulenaktor für das Zwei-Gang-Teilgetriebe ausgelegt sowie eine Elektronik und Positionsregelung dafür entwickelt. Das Ziel dieser Arbeit ist es nun, die bisherigen Funktionen auf einen Mikrocontroller zu implementieren sowie eine eingebettete Elektronik zu entwerfen, die den Aktor zu einem Smart Actuator transformiert. Die bisher zum Steuern benötigte Autobox soll im Rahmen des ADPs ersetzt, und die zugehörige Elektronik auf einer Platine untergebracht werden. Die Vorteile hierbei sind die starke Bauraum- und Gewichtsreduzierung sowie verringerte Kabellängen, die geringere Komplexität für Systemintegratoren und die gesunkenen Kosten. Weiterhin sind die bessere Kontrollierbarkeit und die kürzere Installationsdauer zu nennen. Das Gesamtsystem ist leichter anzusteuern und besteht nicht mehr aus mehreren einzelnen Komponenten. Ein weiteres Ziel ist das Einrichten von Schutzmechanismen und Überwachungsfunktionen für den Aktor. Fehlzustände sollen erkannt und über CAN-Kommunikation übermittelt werden, sodass das übergeordnete System entsprechend reagieren kann. Der Smart Actuator kann somit seinen Status diagnostizieren und in Echtzeit auf Störungen reagieren.

1.2 Anforderungsliste

Um das übergeordnete Ziel weiter zu spezifizieren wurde zunächst eine Anforderungsliste (Tabelle 1.1) erstellt. In dieser sind alle Forderungen an das Endprodukt gesammelt, sie dient damit als Basis und Referenz für die Produktentwicklung. Die Liste ist hierbei dynamisch, das heißt sie kann im Verlauf des Entwicklungsprozesses verändert oder ergänzt werden. Die formulierten Anforderungen werden nach ihrer Priorität kategorisiert und einer der vier folgenden Anforderungsarten zugeordnet [2013a]. Festforderungen (FF) sind unter allen Umständen einzuhalten. Eine Erfüllung ist für eine erfolgreiche Lösung notwendig. Bereichsforderungen (BF) geben einen Toleranzbereich an, innerhalb dessen sich der schlussendlich erreichte Wert befinden muss. Zielforderungen (ZF) geben an, welcher Wert (auch im Hinblick auf spätere Entwicklungen) angestrebt wird. Wünsche (W) sollten nach Möglichkeit erfüllt werden, sind aber keine Voraussetzung.

1.3 Aufbau der Arbeit

Nachdem in der Anforderungsliste nach Tabelle 1.1 die Ziele der vorliegenden Arbeit definiert wurden, soll nun das weitere Vorgehen zum Erreichen der Zielsetzung erläutert werden. Zunächst werden in Kapitel 2 der Stand des Prüfstands vor Beginn des Projektes sowie wichtige Grundlagen als Basis für die weitere Bearbeitung und zum besseren Verständnis dargelegt. Kapitel 3 stellt das allgemeine methodische Vorgehen der Arbeit vor und erklärt die Entwicklungsmodelle und Testkonzepte, die angewendet wurden. In Kapitel 4 werden die verschiedenen Komponenten, die für die Elektronik benötigt werden, und ihre Funktionen beschrieben. In Kapitel 5 soll schließlich der endgültige Platinenentwurf inklusive Methoden zur EMV-Verbesserung vorgestellt werden. In Kapitel 6 wird nach der Vorstellung der entwickelten Software auch auf Optimierungsmöglichkeiten der Programmierung eingegangen. Anschließend wird in Kapitel 7 das Endprodukt auf seine Performance analysiert sowie die anfangs gestellten Anforderungen überprüft. Zum Abschluss wird in Kapitel 8 ein Fazit gezogen sowie ein Ausblick für weitere Forschungsarbeiten gegeben.



Relevanz	Anforderung	Erläuterung
FF	Benutzerfreundliche Kommunikation durch CAN Schnittstelle	Empfang von Befehlen, Senden von Statusmeldungen
FF	Nichtflüchtige Kalibrierung	Eine Kalibrierung ist nur einmalig und zur Rekalibrierung notwendig
BF	Schaltzeit	< 100 ms (Latenz zwischen Senden des Befehls und vollständig ausgeführtem Gangwechsel)
FF	Selbstständige Fehlererkennung	Überstrom, Temperatur, Eingangsspannungsreich, Dekalibrierung
FF	Schnittstellen	CAN, 8-16VDC Versorgung, Programmierschnittstelle (für Updates & Bugfixes)
W	Wartbarkeit	Sicherung wechselt im eingebauten Zustand
BF	kompakte Baugröße	88,8x50 mm
BF	Effizienz (gemittelt über einen Schaltvorgang)	elektrischer Wirkungsgrad > 90 %
FF	Temperaturbeständigkeit	bis 105°C
BF	Aktorüberschwingen	Toleriert, solange kein unbeabsichtigter Gangwechsel
W	Schaltgabelkraft am Anschlag	möglichst gering
FF	Standby	Standbyleistungsaufnahme < 2W

Tabelle 1.1: Anforderungsliste

2 Aufbau des Prüfstands und Grundlagen

In diesem Kapitel wird der verwendete Schaltaktorikprüfstand des IMS vorgestellt, an dem die Entwicklung des Smart Actuators stattgefunden hat. Beschrieben wird der Aufbau des Prüfstands, wie er vor den Änderungen im Rahmen dieses Projektes vorlag.

2.1 Grundlagen des Prüfstandes

Die Konstruktion des Prüfstandes erfolgte in vorangegangenen Arbeiten und wurde seitdem stetig weiterentwickelt. An ihm werden Schaltaktoriksysteme für Fahrzeugantriebe untersucht. Abbildung 2.1 zeigt die in dieser Arbeit verwendeten Subsysteme des Prüfstandes. Es erfolgt zunächst die Vorstellung des mechanischen Aufbaus, woraufhin der elektronische Aufbau anschließt.



Abbildung 2.1: Prüfstand [adp]

2.1.1 Getriebe allgemein

Ein Getriebe ist im Automobil dafür zuständig, die Drehzahl des Motors in ein Drehmoment umzuwandeln, welches die Räder antreibt. Da Motoren nur einen kleinen Bereich von Motordrehzahlen abdecken, werden mehrstufige Getriebe verwendet die verschiedene Raddrehzahlen durch unterschiedliche Übersetzungsverhältnisse bereitstellen können. Das Einstellen des jeweiligen Ganges kann dabei per Hand (Handschaltgetriebe) oder automatisiert über einen Aktor (Schaltaktorik) erfolgen. Im Fahrzeuggetriebe, beispielhaft dargestellt in Abbildung 2.2 ist die Eingangswelle, welche durch den Motor angetrieben wird, über eine Zahnradverbindung fest mit der Vorgelegewelle verbunden. Auf ihr sind noch weitere fest fixierte Zahnräder angebracht, deren Anzahl mit den verfügbaren Gängen übereinstimmt. Diese greifen jeweils in Losräder auf der Abtriebswelle. Um einen bestimmten Gang einzulegen muss nun das jeweilige Losrad für den Moment fest mit der Abtriebswelle verbunden werden, sodass nur diese Zahnverbindung ein Drehmoment überträgt. Dies geschieht über eine formschlüssige Verbindung mit einer Schaltmuffe, die über die durch den Aktor angetriebene Schaltgabel in Position gebracht wird.

2.1.2 Getriebe des Prüfstands

Der Prüfstand besitzt zwei Gänge, in die über eine Schaltgabel geschaltet werden kann. Eine Bewegung der Schaltgabel nach links legt Gang 1 über eine mechanische Synchronisierung ein, während mit Hilfe einer Bewegung nach rechts die Schaltung des Ganges 2 durch eine Klauenkupplung erfolgt. Somit können sowohl Schaltaktoriksysteme mit als auch

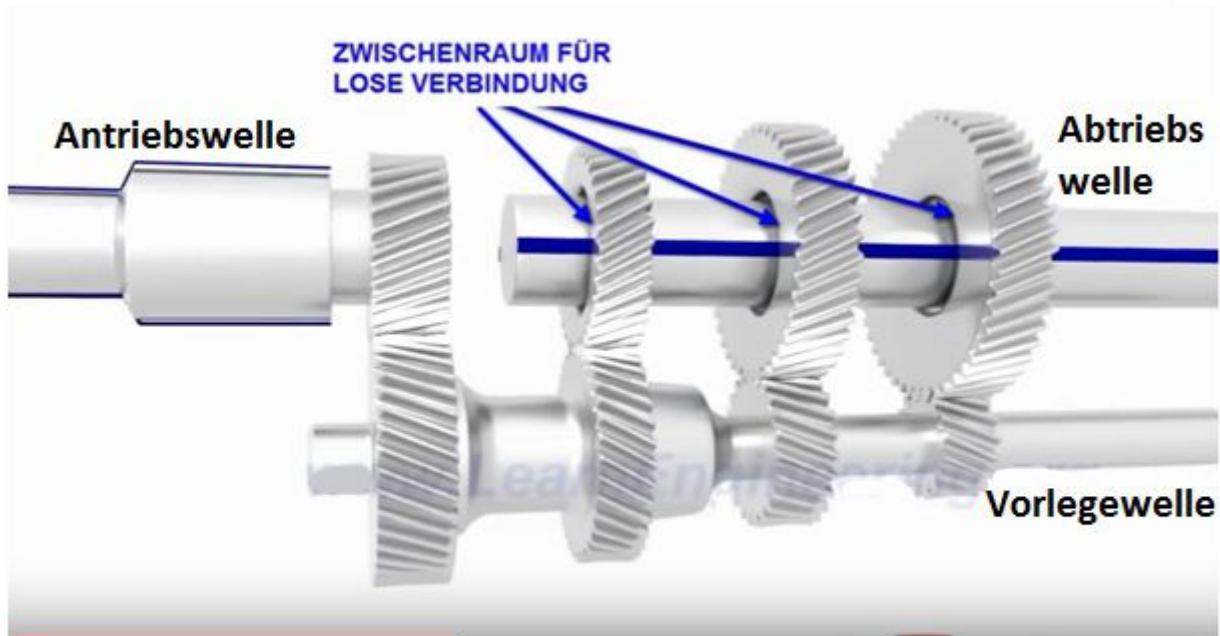


Abbildung 2.2: Fahrzeuggetriebe

ohne Synchronring untersucht werden. Die Bewegung der Schaltgabel wird durch einen Linearaktor ermöglicht, welcher von außen an das Item-Profil verschraubt ist. In diesem ist eine Tauchspule verbaut, die die benötigten Kräfte auf die Läuferstange aufbringt. Über eine starre Wellenkupplung sind Läuferstange und Schaltgabel miteinander verbunden, wodurch die Kräfte auf die Schaltgabel übertragen werden und Schaltvorgänge ermöglicht werden.

2.1.3 Aktor

Der momentan in dem Prüfstand verbaute Tauchspulenaktor wurde von Oliver Hahn im Rahmen seiner Bachelorthesis entwickelt und konstruiert. Er übernimmt im automatisierten Schaltvorgang die Aufgabe, die Schaltgabel über die Schaltstange translatorisch zu verschieben, welche ursprünglich mit einem Schalthebel per Hand ausgeführt wurde. Die Übertragung des Schaltbefehls an den Getriebeaktor erfolgt dabei durch ein elektrisches Signal (*shift by wire*). Sein Querschnitt ist in folgender Abbildung schematisch dargestellt.

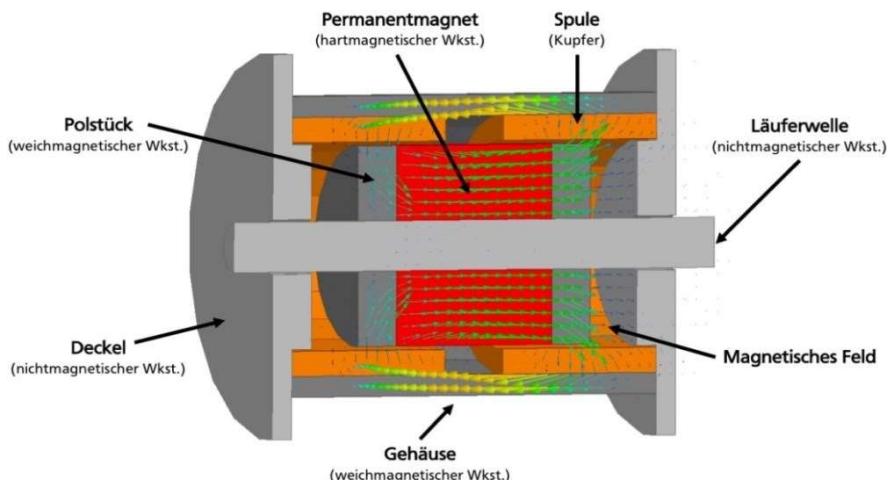


Abbildung 2.3: Querschnitt Tauchspulenaktor [Hahn2018]

Der Aufbau ist zylindrisch und kann in den ortsfesten Stator und den beweglichen Läufer unterteilt werden. Der Stator des Aktors besteht aus zwei in Reihe geschalteten Kupferspulen, welche fest in dem Gehäuse aus Weicheisen liegen und nach oben und unten mit Deckeln aus Aluminium fixiert werden. Der Läufer besteht aus einer nichtmagnetischen Läuferwelle, auf der sich fünf Permanentmagneten aus Neodym-Eisen-Bor befinden, welche mit Hilfe von zwei Polstücken aus Weicheisen axial auf der Läuferstange montiert sind. Werden nun die Kupferspulen von Strom durchflossen, so wirkt eine vom Magnetfeld der Permanentmagneten induzierte Lorentzkraft orthogonal auf sie. Diese Kraft ist abhängig von der Stromstärke I , der magnetischen Flussdichte der Permanentmagneten B und der vom Magnetfeld durchsetzten Leiterlänge l :

$$F = I \cdot l \cdot B \quad (2.1)$$

Da die Spulen jedoch fest im Gehäuse verbaut sind, wirkt eine entgegengerichtete Kraft auf die Permanentmagneten, die auf der axial verschiebbaren Läuferwelle lagern. Diese Kraft bewirkt dann eine translatorische Bewegung der Welle und somit auch der Schaltgabel. Die Richtung der translatorischen Bewegung kann dabei über die Richtung des in den Spulen fließenden Stromes, der Betrag der Kraft über den Betrag des Stromes eingestellt werden. Anhand von Abbildung 2.4 ist zu erkennen, dass die Kraft-Weg-Kennlinien des Aktor innerhalb des Nutzungsbereichs von -10 Millimeter bis +10 Millimeter bei verschiedenen Stromstärken annähernd linear verlaufen.

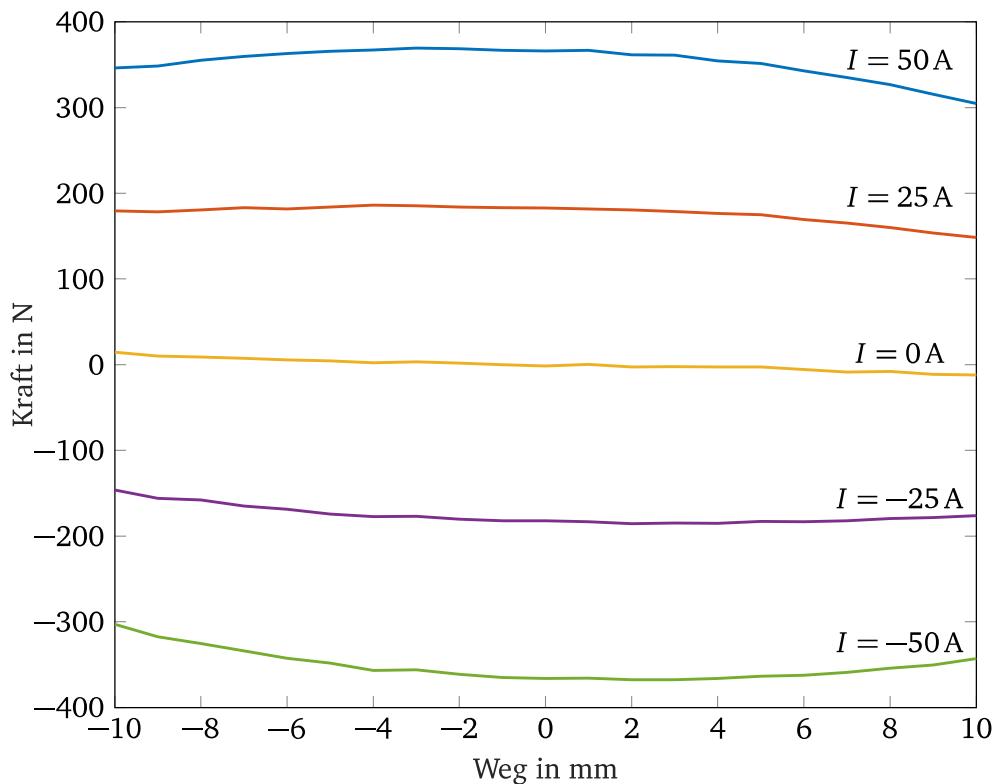


Abbildung 2.4: Kraft-Weg-Strom-Kennlinien des Tauchspulenaktors [adp]

2.1.4 Aktoransteuerung

Zur Aktoransteuerung wird ein Arduino IBT2 Motortreiber verwendet, die Stromversorgung erfolgt über ein Manson SBC-2130 Battery Charger im Power Supply Mode. Der Motortreiber besteht aus zwei BTS7960 MOSFETs, die je nach Vorgabe des pulsdauermodulierten Signals (PWM-Frequenz) eine Spannung von bis zu 13,8V an den Aktor durchschalten.

2.1.5 Autobox

Die bisherigen Funktionen am Prüfstand laufen auf einer MicroAutoBox II 1401/1513 der dSpace GmbH. Das Programm zum Schalten des Aktors und der Positionsregelung, welches in MATLAB/SIMULINK entwickelt wurde, wird auf die AutoBox geflasht und diese über die Software dSpace ControlDesk gesteuert.

2.1.6 Positionsmessung und -regelung

Die Position der Läuferstange wird über einen PLCD-25M Sensor gemessen, dessen berührungslose PLCD (permanentmagnetic linear contactless displacement) Technologie die magnetische Sättigung zur Positionsbestimmung verwendet. Der weichmagnetische Kern des Sensors ist über seine komplette Länge von einer Primärspule umgeben. Auf der Schaltgabel ist ein Permanentmagnet befestigt, welcher je nach Position zu einer lokalen Sättigung des weichmagnetischen Kerns führt. Über zwei Auswertungsspulen am Rand des Sensors kann die Position der Sättigungszone entlang der Sensorachse über die induzierte Spannung bestimmt werden.

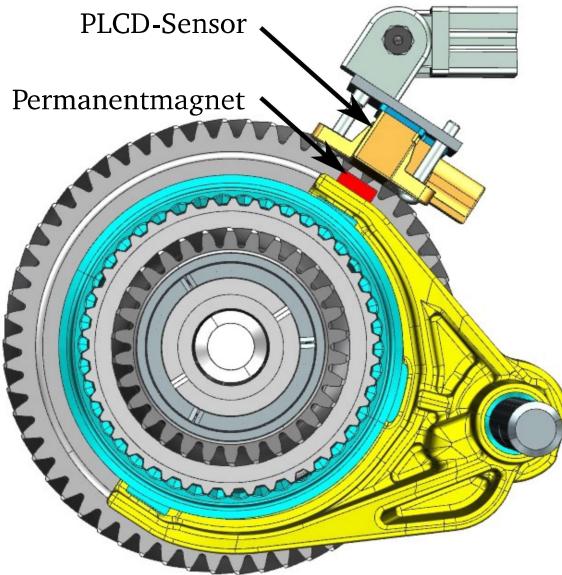


Abbildung 2.5: Einbauposition PLCD Sensor an der Schaltgabel [adp]

Die Kalibrierung wurde in [messtechnik] entwickelt und erfolgt bisher bei jedem Start. Zu Beginn des Einfahrvgangs der Schaltwalze wird dabei das Spannungssignal U_0 ausgelesen, welches der Ausgangslage von $x_0 = 0 \text{ mm}$ entspricht. Anschließend werden die Spannungssignale U_1 und U_2 an beiden Anschlägen gemessen. Die Wegdifferenz zwischen den beiden Anschlagspositionen beträgt $x_{\text{ges}} = 16.97 \text{ mm}$. Nun kann eine Geradengleichung zur Bestimmung der aktuellen Schaltgabelposition aufgestellt werden. U_a entspricht dabei immer dem aktuellen Ausgangssignal.

$$x = \frac{U_a - U_0}{U_1 - U_2} \cdot x_{\text{ges}} \quad (2.2)$$

Die bisherige Regelung der Schaltgabelposition wurde in einem vorangegangenem Advanced Design Project ausgelegt. Dazu wurden mithilfe des Ziegler-Nichols-Verfahren zunächst Parameter für einen PID-Regler ermittelt, welcher allerdings noch zu große Totzeiten und ein Überschwingen aufwies. In einer anschließenden iterativen Optimierung mit Störgrößenkompensation wurden die Regelparameter zu $K_p = 0.026 \frac{\text{V}}{\text{mm}}$, $K_I = 0.2 \frac{\text{V}}{\text{s}\text{mm}}$ und $K_D = 0.016 \frac{\text{Vs}}{\text{mm}}$ bestimmt. Der Verlauf der Sprungantwort ist in nachstehender Abbildung (2.6) dargestellt.

2.2 Grundlagen Löten

Löten gehört zu den Fügeverfahren und bezeichnet das Verbinden zweier Metalle durch eine Metallegierung unter Einfluss von Wärme [loeten]. Die Metallegierung wird dabei als Lot bezeichnet und hat eine geringere Schmelztemperatur (Liquidustemperatur) als die beiden zu verbindenden Metalle (Solidustemperatur). Durch das Löten entsteht eine feste, korrosionsbeständige sowie strom- und wärmeleitende Verbindung. Lötverfahren werden nach Arbeitstemperatur eingeteilt in Weichlöten (bis 450°C), Hartlöten (450°C - 900°C) und Hochtemperaturlöten (über 900°C). Im Rahmen dieses Projektes wird das Verfahren Weichlöten mit Lötkolben angewendet, um die elektrischen Bauteile auf der Platine anzubringen. Dazu können die vom Institut bereitgestellten Lötstationen und -materialien verwendet werden.

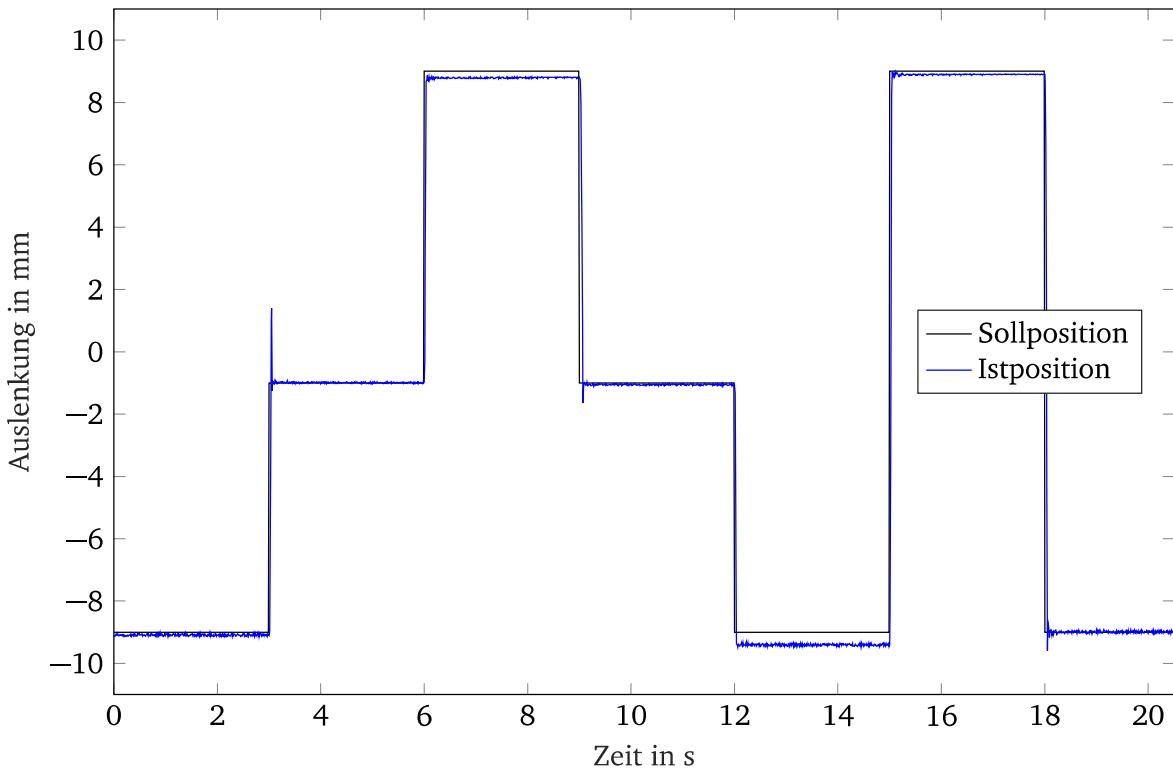


Abbildung 2.6: Sprungantwort Schaltgabelposition [adp]

2.3 Mikrocontroller

Ein Mikrocontroller (*MCU: microcontroller unit*) ist ein hochintegrierter Halbleiterchip, der ein komplettes Mikrorechner-System enthält. Prozessoren, Speicher, Ein- und Ausgabegeräte sind somit auf einem kleinen Chip enthalten, der zum Ziel hat, Steuerungs- und Kommunikationsaufgaben möglichst simpel mit wenig Bauelementen zu bearbeiten.

In Abbildung 2.7 ist der typische Aufbau eines Mikrocontrollers dargestellt. Die Schnittstellen zur Peripherie sind durch einen Betriebsspannungsanschluss, einen Takteingang sowie die Portleitungen gegeben. Bei den Ports wird zwischen Eingangskanälen (*Input Port*), die digitale Signale lesen, und Ausgangskanälen (*Output Port*), die digitale Signale setzen beziehungsweise löschen, unterschieden. Des Weiteren können I/O Pins digitale oder analoge Signale auslesen und stellen, wobei analoge Signale mithilfe eines Analog/Digital Wandlers (*ADC*) zuerst in digital Signale umgewandelt werden müssen, damit der Mikrocontroller sie verarbeiten kann. Eine Spezialform von Eingängskanälen sind die Interrupt-Pins, die bei bestimmten Ereignissen Unterbrechungen des laufenden Programms verursachen, um temporär einen anderen Vorgang zu bearbeiten.

Intern sind die einzelnen Bausteine, die im folgenden kurz erläutert werden, über ein Bussystem verbunden. Der Prozessor (*CPU: central processing unit*) führt Berechnungen und logische Operationen durch. Der Arbeitsspeicher (*RAM: random access memory*) speichert temporär Daten, die aber spätestens nach dem Entfernen der Betriebsspannung verloren gehen. Der Festspeicher (*ROM: read only memory*) behält seinen Speicherinhalt auch nach dem Entfernen der Betriebsspannung und enthält aufgrund dessen das Programm sowie Einstellungen und wichtige Daten. Im Vergleich zum RAM hat er eine langsame Schreibgeschwindigkeit. Ein sogenannter Timer hilft dabei, das Auftreten von Ereignissen zu zählen oder Zeitabstände zu messen, indem beispielsweise Spannungswechsel an einem Eingangskanal gezählt werden.

Die genaue Ausführung des Chips kann je nach Aufgabentyp variieren, sodass eine Vielzahl an verschiedenen Mikrocontrollern erhältlich ist. Diese unterscheiden sich in der Größe des Speichers, in der Anzahl der Anschlüsse beziehungsweise Schnittstellen, in der Bitbreite, in den Taktraten sowie in der Bauform. Typischerweise werden Mikrocontroller in eingebetteten Systemen (*embedded systems*) verwendet, bei denen die Steuereinheit direkt im System selbst integriert ist. Übliche Anwendungen für Mikrocontroller sind Robotik, Handys, Temperaturregler oder Motorsteuerungen [**Brinkschulte**].

2.4 Eingebettete Systeme und Smart Actuators

Eingebettete Systeme werden definiert als Rechenmaschinen, die für den Anwender weitgehend unsichtbar in einem technischen System eingebettet sind und mit diesem in Wechselwirkung stehen [**Gessler2014**]. Der Rechner kann dabei

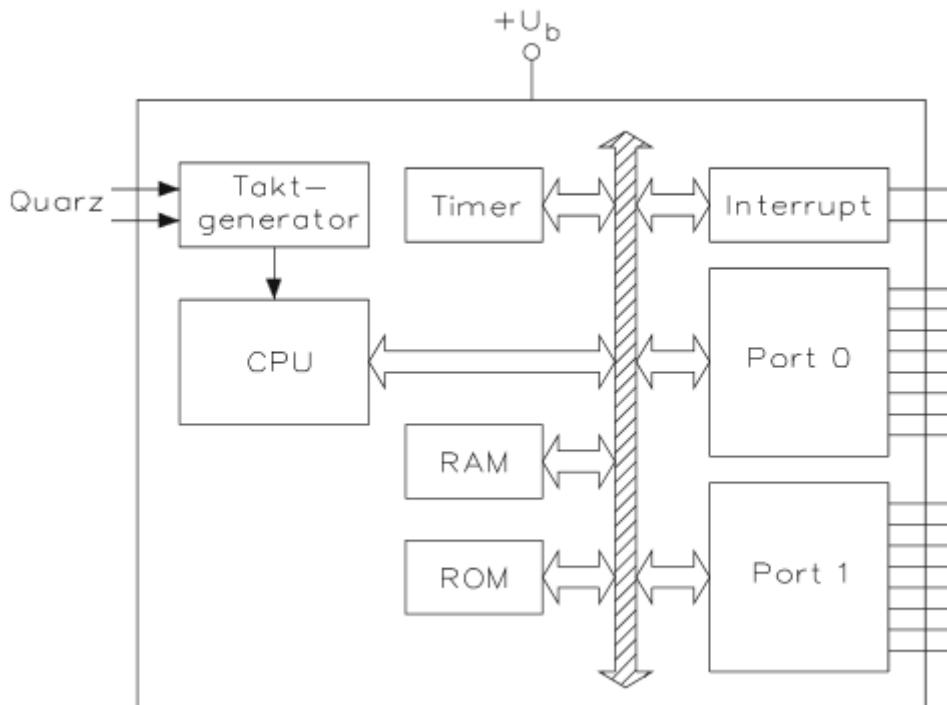


Abbildung 2.7: Blockschaltbild eines Mikrocontrollers [Bernstein2015]

für Regelungs-, Steuer- oder Überwachungsaufgaben und/ oder für Signal- und Datenverarbeitung zuständig sein. Der allgemeine Zweck ist es, die Stellglieder als Reaktion auf Eingangssignale, die von Sensoren oder manuell vorgegeben werden, zu steuern [Broy2003]. Eingebettete Systeme werden oft für speziell eine isolierte Aufgabe entwickelt und angepasst. Ein Aktor, beziehungsweise Aktuator (aus dem englischen: *actuator*) wird genutzt um elektrischen Signale in eine Bewegung oder Kraft umzusetzen und bildet somit das Stellglied. Ein intelligenter Aktor (*smart actuator*) ist definiert als das integrierte Stellglied inklusive aller Komponenten wie Motor, Steuerung, Sensorik und Kommunikationseinheit [**smartactuator**]. Das Konzept besteht darin, den Aktor um Informationsverarbeitungs- und Kommunikationsmöglichkeiten zu erweitern, sodass der Aktor die eingebettete Elektronik enthält. Smart Actuators reduzieren beziehungsweise ersetzen die benötigte Interaktion mit einem Menschen oder externen Rechner und können somit effektiver und schneller arbeiten. Außerdem kann oft viel Bauraum eingespart werden, es gibt weniger Komponenten und deutlich weniger Verkabelung, womit der Einbau in ein Gesamtsystem komfortabler ist. Hervorzuheben sind weiterhin die verringerten Kosten, die Überwachung und Diagnose des eigenen Zustands sowie die verbesserte Präzision. Einbußen müssen Smart Actuators allerdings in Bezug auf flexible Anwendungen machen, da sie oft für eine spezielle Aufgabe ausgelegt sind und ihr Verhalten stark vorgegeben ist [**smartaktor**].

2.5 Controller Area Network (CAN)

Um eine Kommunikation zwischen zwei Geräten (z.B. Sensoren, Aktoren und Steuergeräten) zu ermöglichen, muss ein System zur Datenübertragung vorhanden sein. Dabei hat sich bei vielen PKW- und NKW-Herstellern das Controller Area Network (CAN) durchgesetzt [Werner2014]. CAN ist ein von Bosch für den Automobilbau entwickeltes Bussystem, mit dem mehrere gleichberechtigte Teilnehmer verbunden werden und miteinander kommunizieren können [Woern2006]. Für ein CAN-Netzwerk werden zwei Leitungen CAN-High und CAN-Low benötigt, an denen alle zu verbindenden Teilnehmer parallel angeschlossen sind. Die beiden Leitungen werden an beiden Enden mit Wellenwiderständen von 120 Ohm verbunden. Der schematische Aufbau eines CAN-Buses mit zwei Teilnehmern ist in Abbildung 2.8 dargestellt.

Sollten mehrere Teilnehmer gleichzeitig versuchen eine Nachricht zu senden und somit ein gleichzeitiger Buszugriff vorliegt, kommt es zu einer Kollision. Um diese Aufzulösen wird ein CSMA/CR-Verfahren (*Carrier Sense Multiple Access/Collision Resolution*) verwendet, bei dem die höher gewichtete (dominante) Nachricht die niedriger gewichtete (rezipiente) Nachricht überschreibt, sodass der Übertragungsversuch beendet wird. Die Einteilung der Relevanz einer Nachricht erfolgt auf Basis der sogenannten Identifier(ID)-Bits, wobei niedrige ID-Bits eine höhere Relevanz als höhere ID-Bits haben. Dadurch wird eine Hierarchie der Nachrichten erzeugt, wodurch die Nachricht mit der niedrigsten ID immer gesendet werden darf. Dementsprechend werden wichtigen Nachrichten niedrige IDs zugeteilt [Lawrenz2010].

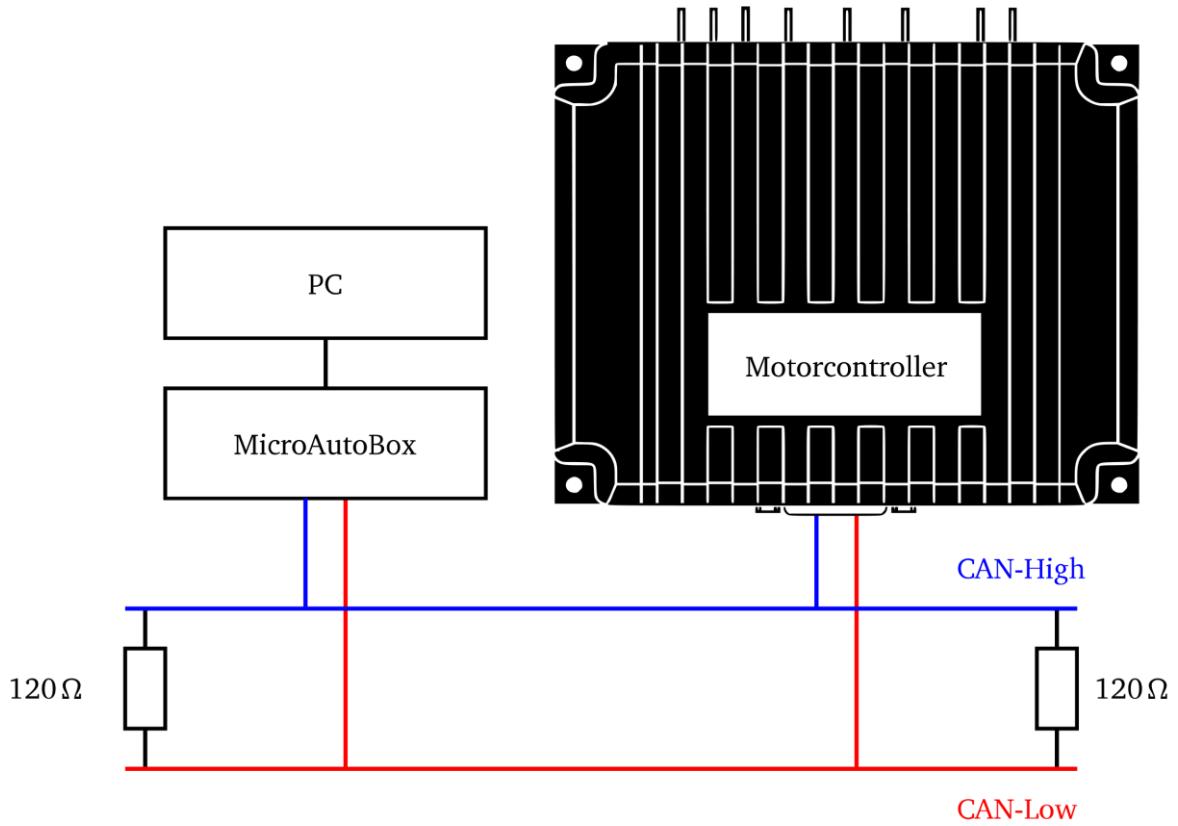


Abbildung 2.8: Aufbau CAN-Bus [manual]

Die Geschwindigkeit der Datenübertragung zwischen den Teilnehmer ist dabei von der Länge der Leitungen abhängig und beträgt maximal 1Mbit/s. Nach [Werner2014] kann die Bitrate überschlagsmäßig mit

$$\text{Buslänge} \leq 40 \dots 50 \text{m} \cdot \frac{1 \text{MBit/s}}{\text{Bitrate}} \quad (2.3)$$

berechnet werden.

In Abschnitt 6.4 wird speziell auf die in dieser Arbeit eingerichtete CAN-Kommunikation eingegangen. Es wird ein CAN-Bus verwendet, mit dem lediglich zwei Teilnehmer miteinander verbunden werden. Diese sind zum einen die MicroAutoBox und zum anderen der Mikrocontroller des Smart Actuators. Die Länge der Leitungen zwischen den beiden ist wesentlich kürzer als 3m, wodurch nach Formel (2.3) die maximale Bitrate von 1Mbit/s möglich wird.

2.6 Pulsweitenmodulation

Pulsweitenmodulation (PWM) bietet die Möglichkeit, ein analoges Signal anhand einer digitalen Quelle zu bilden, indem das Tastverhältnis eines Rechteckimpulses bei gleichbleibender Frequenz moduliert wird. Dabei wird der Effekt ausgenutzt, dass sich ein digitales Signal, welches schnell genug und in einem gewissen Tastverhältnis seinen Zustand wechselt, sich verhält wie ein analoges Signal mit konstanter Spannung. So ist es mithilfe von PWM zum Beispiel möglich mit einem Mikrocontroller, der nur einen gewissen Spannungsbetrag liefern kann, verschiedene hohe Spannungssignale am Endgerät zu erzeugen um es zu betreiben. Es muss allerdings darauf geachtet werden, dass die Pulse im Endeffekt nicht mehr als einzelne Pulse wahrgenommen werden können, das heißt die Frequenz der Pulse muss höher sein als die Abtastrate des Empfängers.

Das Tastverhältnis, oder der Tastgrad p , beschreibt dabei das Verhältnis zwischen der Einschaltzeit t_{ein} und der Periodendauer T :

$$p = \frac{t_{\text{ein}}}{T} = \frac{t_{\text{ein}}}{t_{\text{ein}} + t_{\text{aus}}} \quad (2.4)$$

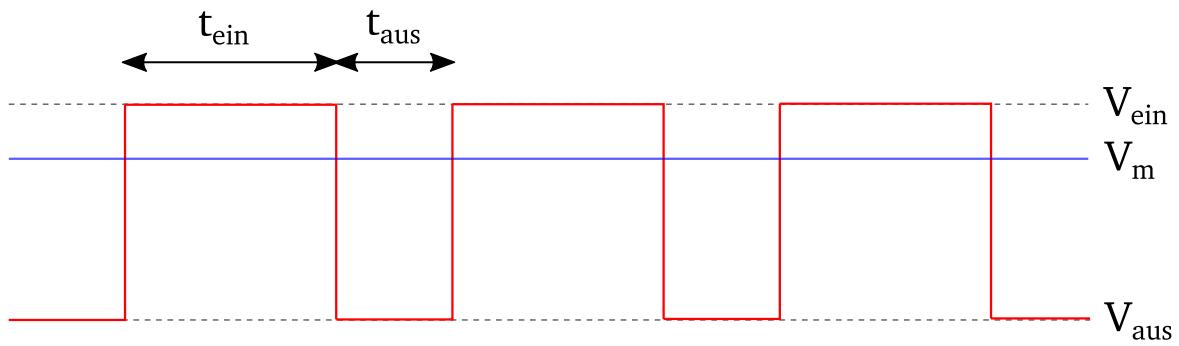


Abbildung 2.9: Beispielhaftes PWM Signal

Das Tastverhältnis kann nur Werte zwischen 0 und 1 annehmen.

Der zeitliche Mittelwert der Spannung U_m ergibt sich dann bei einer Betriebsspannung V_b zu:

$$U_m = V_b \cdot p \quad (2.5)$$

In Abbildung 2.9 ist ein pulsweitenmoduliertes Signal mit dem entsprechendem zeitlichen Mittelwert aufgetragen. Das Tastverhältnis entspricht hier $p = 0,75$.

2.7 Zustandsautomaten

Zustandsautomaten, auch endliche Automaten genannt, eignen sich zur Beschreibung von Systemen, in denen eine endliche Anzahl an Ereignissen eintreten können. In ihnen treten Zustände, Zustandsübergänge und Aktionen auf, aus dem das Verhalten eines Systems modelliert wird. Zustandsübergänge sind logische Operatoren, die erfüllt sein müssen, damit ein System von Zustand X in Zustand Y wechselt. Dementsprechend hat ein Übergang immer einen Zustand als Ein- und Ausgang. Aktionen können durch den Eingang in einen Zustand, den Ausgang aus einem Zustand oder durch eine Eingabe ausgelöst werden.

Eine visuelle Darstellung von Zustandsautomaten kann über Zustandsübergangsdiagramme erfolgen. Dabei werden die Zustände mit Pfeilen (Zustandsübergänge) verbunden. Diese Pfeile werden mit Bedingungen beschriftet, die erfüllt sein müssen, damit der Zustand in Richtung des Pfeils gewechselt wird. In den Zuständen selbst stehen die Aktionen die bei Ein- und Ausgang aus dem Zustand, sowie bei bestimmten Eingaben ausgeführt werden.

2.8 Serial Wire Debug

Serial Wire Debug (SWD) ist eine Weiterentwicklung des IEEE-Standards 1149.1 (JTAG), welcher eine Debugging- und Programmiermethode für Mikrocontroller oder FPGAs auf Leiterplatten ist. JTAG wurde ursprünglich entwickelt um die Signalkommunikation zwischen mehreren Chips in einem elektronischen System zu überprüfen und wurde dann zunehmend zur Programmierung eingesetzt [swd]. Während JTAG 5 Pins zur Programmierung benötigt (TCK, State Machine Control, Data In, Data Out, Reset) genügen SWD bei den selben Mikrocontroller-Ausgängen zwei Signale (SW-CLK, SWDIO). Die geringe Anzahl der Ports basiert dabei auf dem bidirektionalen SWDIO Pin, welcher die Rolle der Data In und Data Out Ports gleichzeitig übernehmen kann [swd]. Unter Debugging versteht man die geführte Ausführung eines Programms, vornehmlich um Fehlerzustände zu lokalisieren. Programmierung bezeichnet den Vorgang, bei dem ein Programm von einer Programmierumgebung auf die Zielhardware übertragen wird. In diesem ADP wird eine SWD-Verbindung aufgebaut um den Mikrocontroller der Platine zu programmieren.

3 Vorgehensweise und Testverfahren

Das folgende Kapitel thematisiert, wie während des Projekts vorgegangen wird, welche Entwicklungsmodelle zum Einsatz kommen und welche Testkonzepte angewendet werden.

3.1 Entwicklungsmodell

Sowohl für die Entwicklung eines Softwaresystems als auch für die Entwicklung von Elektronik haben sich Modelle etabliert, die eine Systematik in den Entwicklungsprozess bringen. Dadurch soll eine hohe Qualität des Entwicklungsprodukts sichergestellt werden. Unter Qualität, insbesondere bei Softwaresystemen, werden nach [BasSof] bzw. ISO-Norm 25010 [ISO_25010] Eigenschaften verstanden wie Funktionalität, Zuverlässigkeit, Benutzbarkeit, Effizienz und Änderbarkeit.

Für die vorliegende Aufgabenstellung ist ein Softwaresystem zur

- Verarbeitung der Sensorik,
- Anwendung des Regelgesetzes,
- Ansteuerung der Aktorik,
- nichtflüchtige Kalibrierung und
- selbstständige Fehlererkennung

notwendig. Gleichermassen muss jedoch eine Elektronik entwickelt werden, auf welcher das Softwaresystem ausgeführt wird und durch welche die Schnittstelle zu notwendigen Komponenten hergestellt wird. Um beide Systeme parallel zu entwickeln wird das V-Modell gewählt, das in Abbildung 3.1 dargestellt ist.

3.1.1 Das V-Modell

Das V-Modell ist ein Vorgehensmodell, dass in verschiedenen Ausführungen existiert. Das hier vorgestellte und angewandte Modell entspricht dem Modell nach [Boehm79]. Als weitere Quelle wird [BasSof] herangezogen und im Folgenden referenziert. Der linke Zweig stellt dabei die konstruktive Entwicklung dar, während jede Aktivität eine dazu korrespondierende Testaktivität im rechten Zweig besitzt.

Konstruktive Entwicklung (linker Zweig)

Zunächst wird demnach eine Anforderungsliste im Zuge der *Anforderungsdefinition* erstellt, die die Entwicklungsaufgabe genauer spezifiziert und später eine Überprüfungsgrundlage bietet inwiefern das fertige Gesamtsystem des Wunschsystems entspricht. Daran anschließend wird ein *funktionaler Systementwurf* durchgeführt. Hierbei werden die Anforderungen in Funktionen überführt, die das Gesamtsystem erfüllen muss, um den Anforderungen gerecht zu werden. Darauf folgend findet der *technische Systementwurf* statt. Dafür wird das System in unabhängige Teilsysteme unterteilt und Schnittstellen zur Umwelt ermittelt. Daran logisch anknüpfend wird die *Komponentenspezifikation* durchgeführt. Dafür wird für jede Komponente (elementares Teilsystem) die Aufgabe, das gewünschte Verhalten und Schnittstellen zu anderen Teilsystemen definiert.

Den letzten reinen Entwicklungsschritt bildet die *Komponentenentwicklung*. Dabei werden schließlich die einzelnen Komponenten nach der zuvor erarbeiteten Spezifikation entwickelt. Eine Komponente stellt dabei ein Teilsystem dar, das eine feste Funktion erfüllt, und in der Regel aus einem Softwareteil und einem Hardwareteil besteht. Beide Teile werden parallel in genauer Abstimmung entwickelt, da die gewünschte Funktion nur durch beide Systeme im Zusammenspiel erfüllt werden kann. Das Softwaresystem besteht dabei in der Regel aus einem Treiber, der die komplexe Hardware-Schnittstelle für das Restsystem versteckt und komfortable Schnittstellen bereitstellt. Ein Beispiel dafür stellt der Treiber für den Tauchspulenaktor dar, der eine Pulsweite in Prozent und ein Aktivierungssignal entgegen nimmt. Intern werden dann daraus Steuersignale für die Halbbrücken generiert und über die Elektronikschaltung dem Aktor zugeführt. Neben

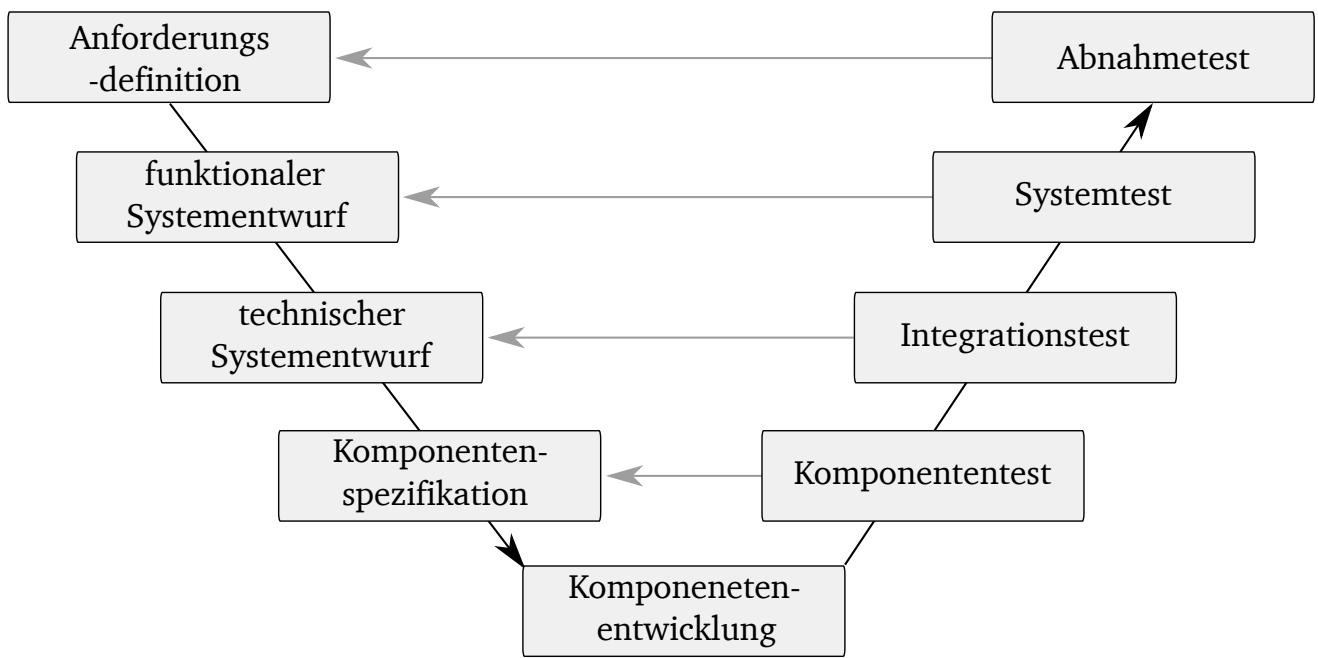


Abbildung 3.1: V-Modell nach [Boehm79i]

Elektronikschaltungen mit zugehörigem Treiber existieren auch *reine Softwarekomponenten*, die unabhängig der Elektronik entwickelt werden wie bspw. eine Verhaltenslogik für ein- und ausgehende CAN-Nachrichten. Ebenso existieren *reine Elektronikkomponenten*, die keinen direkten Bezug zur Software aufweisen und daher unabhängig zur Software entwickelt werden können.

Testaktivitäten (rechter Zweig)

Alle folgenden Informationen sowie detaillierte Ausführungen bezüglich des Testverfahrens können [BasSof] entnommen werden.

Nachdem die Entwicklung aller Komponenten abgeschlossen ist, müssen die einzelnen Komponenten auf die geforderte Funktionalität überprüft werden. Dazu wird jede Komponente so isoliert wie möglich vom Restsystem getestet, ggf. unter Einsatz sogenannter Testtreiber und Platzhalter, um die zu testenden Komponenten mit Testdaten oder Testfällen auszuführen. Durch die Isolation wird sichergestellt, dass die einzelne Komponente als solche funktioniert und Fehlerzustände leichter eingegrenzt und lokalisiert werden können. Jeder Testfall enthält dabei feste Vorbedingungen, Eingabewerte und ein gefordertes Sollverhalten bzw. geforderte Ausgabewerte. Die Auswahl der Testfälle wird durch die Art der Komponenten unterschieden. *Treiber mit zugehöriger Hardwarekomponente* bieten Schnittstellen, die beliebige Zahlenwerte und Kombinationen entgegen nehmen. Daraus resultiert eine quasi unendliche Menge an möglichen Testfällen, die nicht praktikabel getestet werden können. Vollständiges Testen ist daher nicht möglich. Aus diesem Grund werden die Testfälle aus einem Klassifikationsbaum mit anschließender Grenzwertanalyse gewonnen. Dabei wird jeweils mindestens ein Testwert aus jeder Gruppe ausgewählt. Die Gruppierung wird so vorgenommen, dass von jeder Gruppe aus Testwerten gleiches Verhalten angenommen wird. Als Beispiel kann wieder die Ansteuerung des Tauchspulenaktors herangezogen werden. Es wird davon ausgegangen, dass alle Werte zwischen -100 und 0 eine Bewegung in eine feste Richtung bewirken. Dagegen bewirken Werte von 0 bis 100 eine Bewegung in die Gegenrichtung. Werte außerhalb des Zahlenbereichs werden auf entsprechend -100 oder +100 abgerundet. Durch eine Grenzwertanalyse werden diese Gruppen (auch Äquivalenzklassen genannt) noch in Untergruppen unterteilt, in denen die oft kritischen Grenzwerte (-100, 0 und 100) noch eigene Gruppen bilden. Eine Sonderstellung besitzt die Null, weil hier keine Reaktion erwartet wird. Durch kombinatorische Paarbildungen werden darüber hinaus auch beliebige Kombinationen aus abhängigen Eingangswerten, bspw. Aktivierungssignal (enable) und Pulsbreite (PWM [%]), berücksichtigt. Um die Testfallgenerierung durch Klassifikationsbäume zu verdeutlichen, ist in Abbildung 3.2 der Klassifikationsbaum grafisch dargestellt. Die untere Zeile aus konkreten Werten stellt beispielhafte Testwerte aus der darüberliegenden Äquivalenzklasse dar. Darunter ist über die Gitterstruktur ange deutet wie aus der paarweisen Kombination der einzelnen Testwerte Testfälle bestimmt werden. *Testfall 1* entspricht dem Aufrufen des Treibers mit einer PWM von -100 % und einem aktivierte enable-Signal. Es wird daher erwartet, dass der Aktor voll in „negative“ Bewegungsrichtung bestromt bzw. beschleunigt wird. Ob die Elektronik als solche den Erwartun

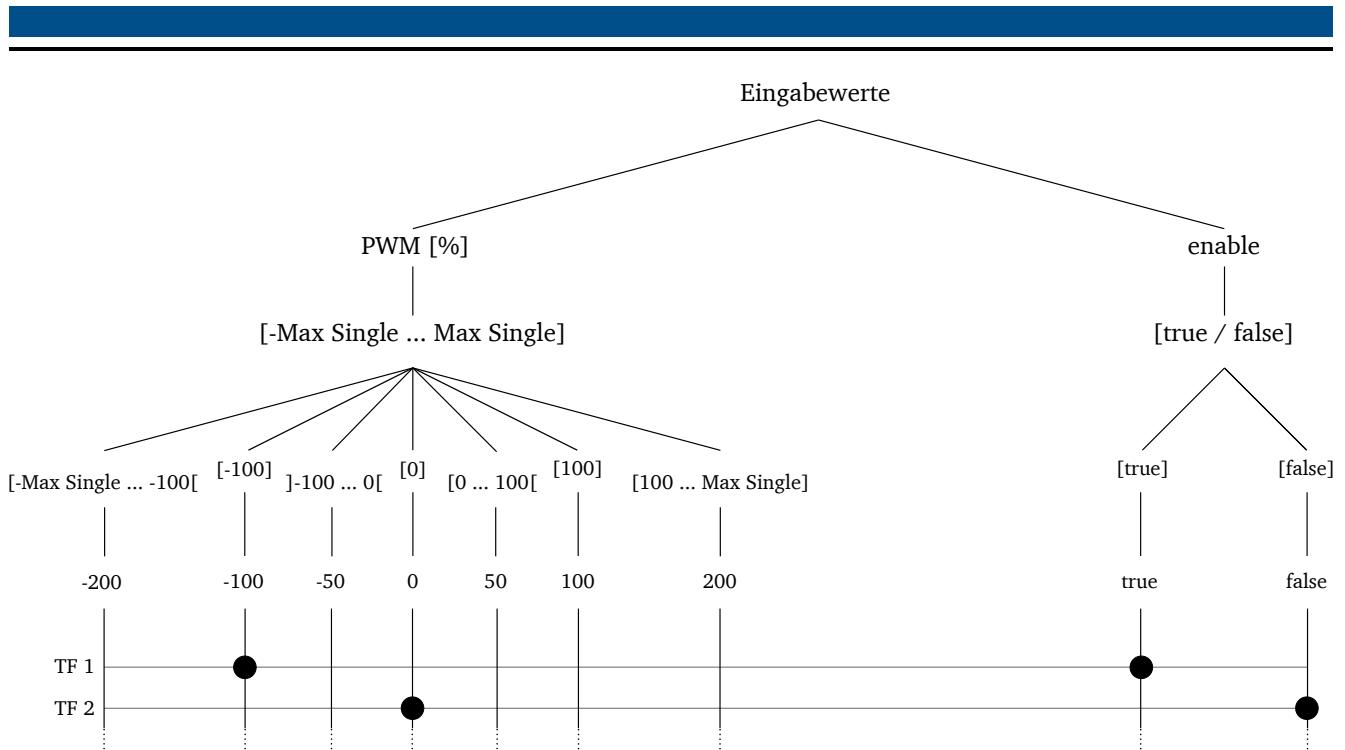


Abbildung 3.2: Klassifikationsbaum am Beispiel des Motortreibers

gen bzw. der Spezifikation entspricht, kann durch Strom-/Spannungs- sowie Temperaturmessungen verifiziert werden. Für *reine Softwarekomponenten* hingegen werden Testfälle auf eine geeignete Weise generiert. Bei Softwarekomponenten auf Basis eines Zustandsautomaten werden Testfälle aus einer zustandsbasierten Testfallgenerierung gewonnen. Dabei wird als Testziel vorausgesetzt, dass alle Entscheidungen (Kanten) eines Zustandsautomaten mindestens einmal ausgeführt werden, was als Entscheidungsüberdeckungskriterium bezeichnet wird. Dieses Kriterium schließt auch automatisch ein, dass alle Zustände (Knoten) besucht worden sind. Wie viele Testfälle getestet werden müssen hängt dabei stark von der Größe und Komplexität des Zustandsautomaten ab.

Nach dem Test der einzelnen Komponenten wird eine Integration vollzogen. Unter Integration versteht man dabei das Zusammenführen der Komponenten zu einem Gesamtsystem. Es stehen viele Varianten zur Verfügung, dazu zählen *Bottom Up*-, *Top Down*- und *Big Bang*-Integration. Letztere Variante sieht vor, dass alle Komponenten in einem Zug ins Gesamtsystem integriert werden. Auftretende Fehler können jedoch in diesem Fall nur schwierig zugeordnet werden. Günstiger ist daher die *Bottom Up*-Integration, wobei nach und nach Komponenten hinzugefügt werden und übergeordnete Komponenten, die mit mehreren Einzelkomponenten Schnittstellen haben, später integriert werden. Da im vorangehenden *Komponententest* bereits die Komponenten getestet sind, ist im anschließenden *Integrationstest* mit Fehlern zu rechnen, die durch Schnittstellen zwischen den Komponenten hervorgerufen werden. Es kann davon ausgegangen werden, dass die Komponenten intern funktionieren und ausschließlich das Zusammenspiel der Komponenten fehlerhaft ist.

Zuletzt entsteht somit ein Gesamtsystem, das darauf getestet werden muss, ob die im *funktionalen Systementwurf* festgelegten Funktionen erfüllt werden. Um dies zu testen werden *Use-Case*-Szenarios erstellt, die eine übliche Nutzung simulieren. Dabei wird von mehreren Akteuren ausgegangen, die mit dem System in der realen Anwendung agieren werden. Der Schaltaktor wird bspw. über ein übergeordnetes Steuergerät (*electrical Control Unit*) angesteuert. Die Schnittstelle wird durch CAN realisiert, sodass ein entsprechendes Steuergerät über CAN Signale/Befehle einen Schaltvorgang anfordern kann oder eine Fehlermeldung auslesen kann. Einen typischen *Use-Case* würde bspw. ein Schaltbefehl vom neutralen in den zweiten Gang darstellen. Dieser kann erfolgreich ablaufen oder durch eine mechanische Blockade verhindert werden. Abschließend erfolgt der Abnahmetest, der in der Regel durch den Auftraggeber durchgeführt wird. Hier wird explizit verglichen, ob die versprochenen Leistungen im Pflichtenheft auch erreicht werden.

Diese und weitere Informationen für den interessierten Leser finden sich in [**BasSof**].

3.1.2 Konkretes Vorgehen mit inkrementellem Entwicklungsansatz

Neben dem V-Modell fließen auch inkrementelle Ansätze in die Entwicklung ein. So wird das V-Modell mehrmals durchlaufen, während nach jeder Iteration ein ausführbares und testbares Entwicklungsprodukt vorliegt. Diese Zwischenprodukte entsprechen der Anforderungsliste unter Umständen nur teilweise, sodass ggf. mehrere Iterationen bis zu einem zufriedenstellenden Produkt notwendig sind. Für das vorliegende Projekt wird in der ersten Iteration ein Prototyp angestrebt, der bereits die Grundfunktionalitäten bereitstellt. Dies umfasst die Möglichkeit einen Schaltvorgang

Iteration	Schaltzeit	Nichtflüchtige Kalibrierung	kompakte Baugröße	Effizienz	Fehlererkennung	...
1	✓	✓	✗	✗	✗	...
2	✓	✓	✗	✗	✓	...
3	✓	✓	✗	✓	✓	...
...						

Tabelle 3.1: Beispielhafter Ausschnitt aus Systemtest-Ergebnissen über mehrere Iterationen

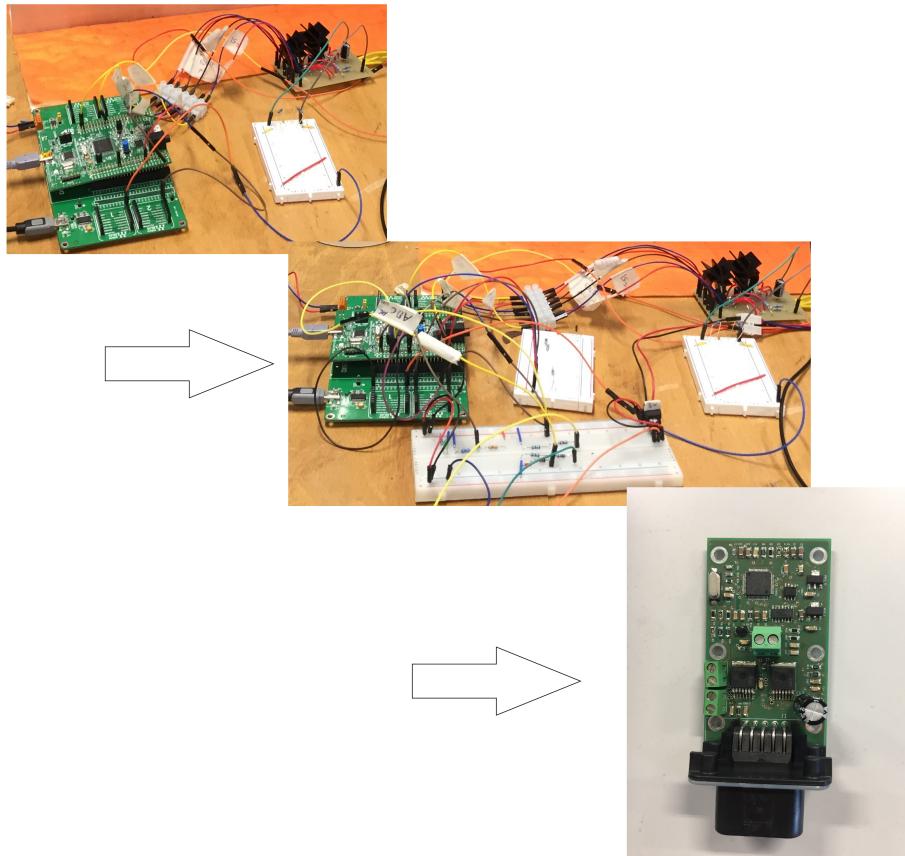


Abbildung 3.3: Bildliche Darstellung des Prototypings (Iterationsprozesses)

nach Anforderungsspezifikation durchzuführen. Dafür muss unter Anderem die Aktorik ansteuerbar sein und die Sensorik notwendige und hinreichend genaue Regelgrößen liefern. Anforderungen an Größe, Effizienz oder Integrierbarkeit sind zunächst nur sekundär oder überhaupt nicht verfolgt worden. Durch das Ergebnis werden Erkenntnisse gewonnen, die in der zweiten Iteration berücksichtigt werden. Bei folgenden Iteration wird versucht über geeignete Maßnahmen (bspw. durch eine angepasste Komponentenspezifikation) bisherige Ergebnisse zu verbessern und weitere Anforderungen zu erfüllen, wie die Fähigkeit Fehlerzustände (Überstrom, etc.) zu erkennen. Einen wesentlichen Iterationsschritt stellt der Übergang von Steckbrett- auf Platinenbasis dar. Durch diesen Übergang treten weitere Anforderungen wie Baugröße und Integrierbarkeit in den Vordergrund. Nach dem letzten Iterationsschritt sollten alle Anforderungen nachweislich erfüllt sein. In Abbildung 3.3 sind die Iterationen bildlich dargestellt, während in Tabelle 3.1 ein Beispielhafter Ausschnitt aus dem Ergebnis des jeweiligen *Systemtests* pro Iteration dargestellt sind.

4 Auswahl elektronischer Komponenten und Verschaltung

Im Folgenden wird auf die Auswahl der Platinenkomponenten, sowie deren Beschaltung eingegangen.

4.1 Anforderungen an die Komponenten

Die benötigten elektronischen Komponenten werden üblicherweise in „Passive Bauelemente“ und „Aktive Bauelemente“ untergliedert. Unter passiven Bauelementen versteht man Komponenten, die das Eingangssignal ohne Verstärkung übertragen. Aktive Bauelemente benötigen meist eine Hilfsquelle und können somit ein Eingangssignal verstärken [haendschke]. Für den Einsatz elektronischer Komponenten im Automobilbereich hat sich für passive Bauteile die Zertifizierung nach AEC-Q200 und für aktive Bauteile nach AEC-Q100 als Qualitätsstandard etabliert. Anhand dieser Zertifizierungen wird eine nachweisliche Belastbarkeit der Bauelemente nach dem jeweiligen Standard definiert. Für die Anforderungen an die Platine werden die passiven Bauelemente nach AEC-Q200 und die aktiven Bauelemente nach AEC-Q100 mindestens in Grade 2 eingestuft, um den Temperaturanforderung bis 105 °C gerecht zu werden [aecq].

4.1.1 Passive Bauteile

Unter die benötigten passiven Bauteile fallen lediglich Kondensatoren und Widerstände, welche nach dem AEC-Q200 Standard ausgewählt werden. Anforderung ist danach eine hohe thermische Belastbarkeit von mindestens 105 °C. Um eine geringe Bauteilgröße für die Platine zu bekommen sollten sowohl Widerstände als auch Kondensatoren als SMD (*surface-mounted device*) gekauft werden. Für Abblockkondensatoren werden nach [ldo] Keramikkondensatoren mit XR7 präferiert. Als Baugröße bei manueller Bestückung bieten sich die Standardgrößen 1206 oder 0805 (Angabe in $\frac{1}{100}$ Zoll), welche noch per Hand lötbar sind. Die Widerstände für die Sensorik müssen mit sehr hoher Präzision, geringem Rauschen und niedriger Temperaturabhängigkeit gewählt werden um Messfehler aufgrund falscher Widerstandswerte zu vermeiden. Im Anwendungsfall bieten sich daher Dünnschichtwiderstände an.

4.1.2 Aktive Bauteile

Die aktiven Bauelemente der Platine müssen dafür ausgelegt werden, um im späteren alle Anforderungen aus der Anforderungsliste gerecht zu werden. Demnach müssen sie in ihrer logischen Verschaltung einerseits in der Lage sein den Tauchspulenaktor anzusteuern und auszuregeln, andererseits aber auch die Bedingungen über Leistungsaufnahme etc. erfüllen. Kern der Platine ist ein Mikrocontroller, welcher schnell genug sein muss um Gangwechsel in weniger als 0,1s durchzuführen. Weiterhin muss der Mikrocontroller die CAN-Kommunikation mit der MicroAutobox unterstützen. Da der Mikrocontroller nicht über die 13.8 V der Autobatterie gespeist werden kann, wird ein Spannungsregler benötigt, welcher auf die Betriebsspannung des Mikrocontrollers regelt. Um aus den physikalischen Bussignalen CAN-HIGH und CAN-LOW für den Mikrocontroller verwertbare Signale zu erhalten wird ein CAN-Transceiver benötigt. Über ihn ist eine Kommunikation des Mikrocontrollers mit der CAN-Peripherie erst möglich. Zum Regeln des Tauchspulenaktors ist ein Motortreiber nötig, welcher vom Mikrocontroller angesteuert wird um die Spannung an den Tauchspulenaktor durchzuschalten. Eine Ansteuerung des Motortreibers mittels PWM-Signal sollte möglich sein. Der Motortreiber muss dabei für zukünftige Aktorkonfigurationen für Ströme bis zu 55A ausgelegt sein. Des Weiteren wird eine Spannungsebene von 5 V benötigt um die Lagesensorik des Aktors zu unterstützen, wofür ein zweiter Spannungsregler eingeplant werden muss.

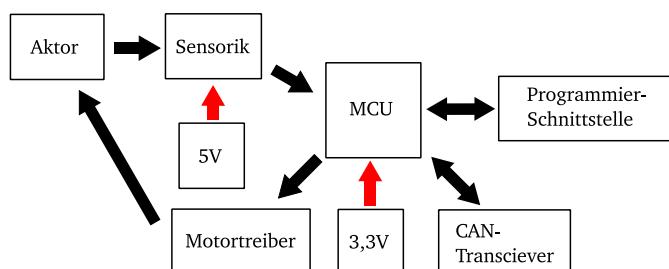


Abbildung 4.1: Benötigte Ebenen der Verschaltung

4.2 Komponentenauswahl

Anhand der Anforderungen an die Platinenelemente sind die endgültigen Komponenten der Platine zu wählen. Es müssen Mikrocontroller, Spannungsregler, CAN-Transceiver und Motortreiber ausgewählt werden. Weiterhin wird eine Temperaturbeständigkeit bis 105 °C gefordert.

4.2.1 Mikrocontroller

Die Recheneinheit zur Regelung des Tauchspulenaktors ist ein STM32F405RGT7. Dieser ist mit verschiedenen Pin-Anzahlen erhältlich und auf der Platine aus Platzgründen in der Ausführung LQFP64 mit 64 Pins verbaut. Der Mikrocontroller zeichnet sich durch seine hohe CPU-Geschwindigkeit von 168 MHz und seine große Programm-Speichergröße von 1 MB aus. Die Produktbezeichnung „T7“ steht dabei für die zulässige Betriebstemperatur von -40...105 °C. Weiterhin besitzt dieser Mikrocontroller die geforderten CAN-Schnittstellen, anhand derer mit der MicroAutobox kommuniziert werden soll. In Abbildung 4.2 ist die Minimalbeschaltung des Mikrocontrollers zu sehen. Diese ist dem Datenblatt des STM32 entnommen [stm32]. Die Pins mit den Bezeichnungen VDD...VDD_4 sind die Versorgungspins des Mikrocontrollers. An diesen Pins liegt die Versorgungsspannung an, welche nach dem Datenblatt zwischen 1,8 und 3,6 V liegt. Diese ist aufgrund der Versorgungsspannung anderer Bauteile zu VDD = 3,3 V gewählt. Die VDD Pins werden durch die Keramikkondensatoren C7-C11 entkoppelt, welche nach dem Datenblatt dimensioniert sind und möglichst nah an den dem jeweiligen VDD-Pin plaziert werden sollen. Die Entkopplung wird benötigt um Welligkeiten der Spannungsversorgung zu filtern und parasitäre Induktivitäten zu entkoppeln [decoupling]. Der Pin VBAT wird unter Anderem für die Versorgung der Echtzeituhr (RTC), der Backup-Register und des Backup-SRAMs genutzt und wird, wenn keine zweite Spannungsversorgung neben der Hauptversorgung vorhanden ist, ebenfalls an VDD angeschlossen [stm32]. VDDA ist der Pin für die Spannungsversorgung des Analog-Digital-Converters (ADC) und wird ebenfalls mit der Versorgungsspannung von VDD angeschlossen. Die Kapazitäten C5 und C6 sind nach den Herstellerangaben dimensioniert und sorgen ebenfalls für eine Entkopplung der Spannungsversorgung. Die Induktivität L1 bietet die Möglichkeit, je nach Welligkeit der Versorgungsspannung, diese über einen LC-Filter zu filtern um eine bessere Auflösung des ADCs zu gewährleisten. Wenn dieser Filter nicht benötigt wird, sollte L1 durch einen 0 Ω Widerstand ersetzt werden. VCAP_1 und VCAP_2 sind die Ausgänge des internen Voltage Regulators des STM32 und C5 bzw. C6 werden zur Glättung der intern geregelten Spannung genutzt [stm32]. Die Pins VSS, VSS_2, VSSA sind die GND Anschlüsse für den Mikrocontroller und den internen ADC.

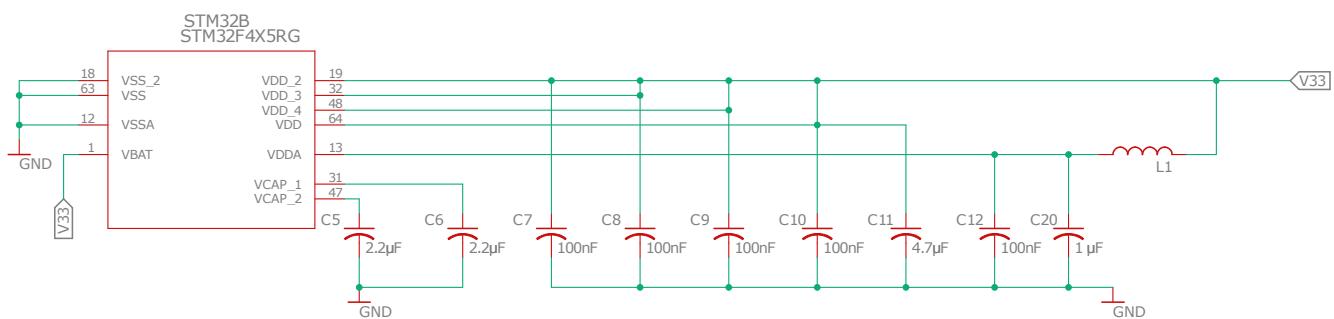


Abbildung 4.2: Minimalbeschaltung des STM32F405RGT7

In Abbildung 4.3 sind die Pin-Belegungen des Mikrocontrollers dargestellt, welche durch Tabelle 4.1 genauer beschrieben sind. Unter Sensorik fallen die Ausgänge für Lagesensorik, Temperatursensorik und Messung der Versorgungsspannung. Die UART Pins können zum debuggen auf der endgültigen Platine genutzt werden. Zur CAN-Kommunikation werden die Pins CAN1-TX(PA11) und CAN1-RX(PA12) benötigt, womit der Mikrocontroller über einen CAN-Transceiver Nachrichten an die MicroAutobox senden und empfangen kann. Zum Programmieren des Mikrocontrollers wird eine SWD-Schnittstelle aufgebaut, welche sich mittels externem ST-Link/V2 mit dem Computer verbinden lässt. Dazu werden die jeweiligen Pins über den Platinenstecker nach außen geführt. Weiterhin müssen Pins zum Beschalten der H-Brücke und zum Auslesen der Strommessungen belegt werden. Die Pins OSC_IN und OSC_OUT sind zum Anschließen eines externen Quarzes, welcher als Taktgeber für den Mikrocontroller genutzt wird. Der Pin NRST wird für den Fall eines notwendigen Resets herausgeführt.

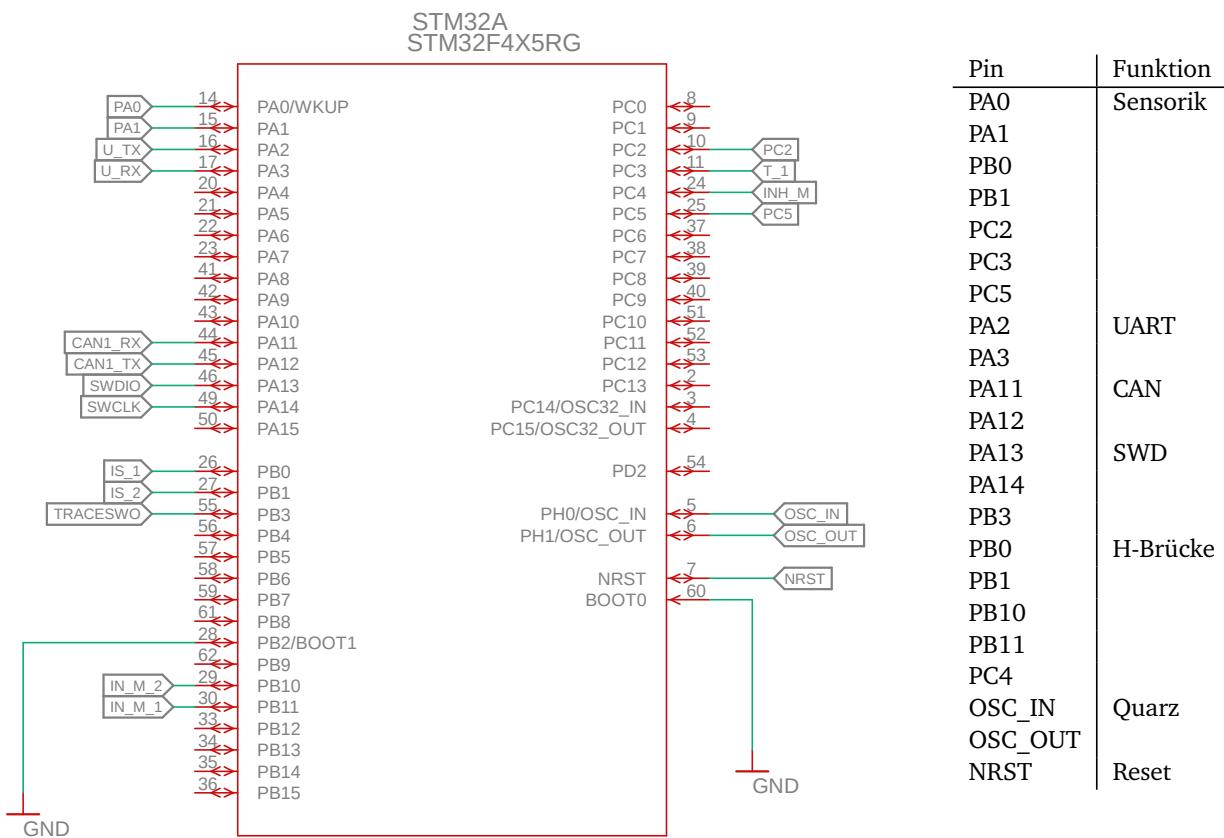


Abbildung 4.3: Pin-Belegung des Mikrocontrollers

Tabelle 4.1: Pin-outs

Wichtig bei der Beschaltung ist außerdem die Belegung von BOOT0 und BOOT1, welche den Speicherbereich definieren aus dem der Mikrocontroller beim Startvorgang sein Programm lädt. Nach Tabelle 4.2 muss für den Start im Main Flash memory BOOT0 auf GND-Niveau gesetzt werden, während die Belegung von BOOT1 undefiniert ist und somit mit GND belegt werden kann.

BOOT1	BOOT0	Boot mode	Aliasing
x	0	Main Flash memory	Main Flash memory is selected as the boot space
0	1	System memory	System memory is selected as the boot space
1	1	Embedded SRAM	Embedded SRAM is selected as the boot space

Tabelle 4.2: Boot modes des STM32F405RG7 nach [stmref]

Um möglichst akurate Taktraten zu haben und somit Schaltvorgänge und Sensorikdaten genau und reproduzierbar zu gestalten, wird wie vom Hersteller des Mikrocontrollers empfohlen ein externer Taktgeber benutzt[stmref]. Auf der Platine wird hierbei ein ABLS-8.000MHZ-K4T von Abracon genutzt, welcher eine Nennfrequenz von 8MHz hat. Abbildung 4.4 zeigt die letztendliche Verschaltung auf der Platine. Die Lastkapazitäten CQ1 und CQ2 von 22 pF sind dabei nach dem Datenblatt des Herstellers gewählt. Die Widerstände R19 und R20 sind einerseits verbaut, um den externen Quarz vom Mikrocontroller trennen zu können, andererseits wird R20 benutzt, um den Stromfluss des Quarzes zu begrenzen [stmquarz]. Nach [stmquarz] lässt sich eine Abschätzung von R20 über

$$R_{20} = \frac{1}{2\pi f C_{Q1}}$$

gewinnen. Damit ergibt für R20 sich ein Richtwert von etwa 900Ω bei $C_{Q1} = 22\text{pF}$. Ein zu niedriger Widerstand erhöht die Verlustleistung über den Quarz, während ein höherer Widerstand zum Stillstand der Oszillation führen kann. Auf der Platine ist ein Widerstand von $1\text{k}\Omega$ gewählt.

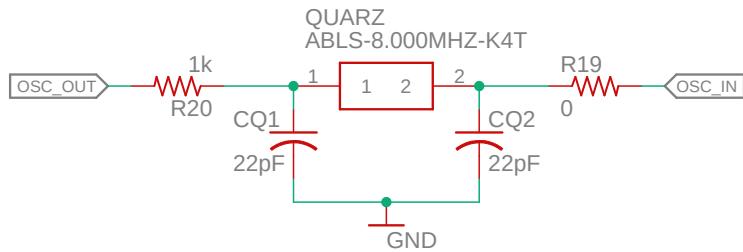


Abbildung 4.4: Anschluss externer Quarz

4.2.2 CAN-Transceiver

Zur Kommunikation des Mikrocontrollers mit der MicroAutobox ist eine Signalübertragung über CAN vorgesehen. Da der Mikrocontroller die physikalischen Bussignale nicht verarbeiten kann, wird die Kommunikation des Mikrocontrollers mit dem Bussystem über einen CAN-Transciever gestaltet. Dieser sorgt dafür, dass den Mikrocontroller nur für ihn lesbare Daten erreichen und übersetzt gleichzeitig die vom Mikrocontroller ins Bussystem gesendete Nachrichten. Die Pins des Mikrocontrollers für diese Aufgaben sind CAN1-RX zum Erhalten von Informationen und CAN1-TX zum Senden von CAN-Nachrichten. Auf Busebene gibt es die Signale CAN-HIGH und CAN-LOW, welche in Kapitel 2 beschrieben wurden. In Abbildung 4.5 ist die Verschaltung des CAN-Transcievers SNHVD230QD von Texas Instruments zu sehen. An Pin VCC wird die Spannungsversorgung von 3.3 V angeschlossen. VREF ist ein Ausgangspin mit halber VCC-Spannung beispielsweise zum Entwerfen einer Split-Termination. Pin D ist der Anschlusspin für die gesendeten Nachrichten des Mikrocontrollers über CAN1-TX. Pin R sendet die übersetzten CAN-Nachrichten des Bussystems an CAN1-RX.

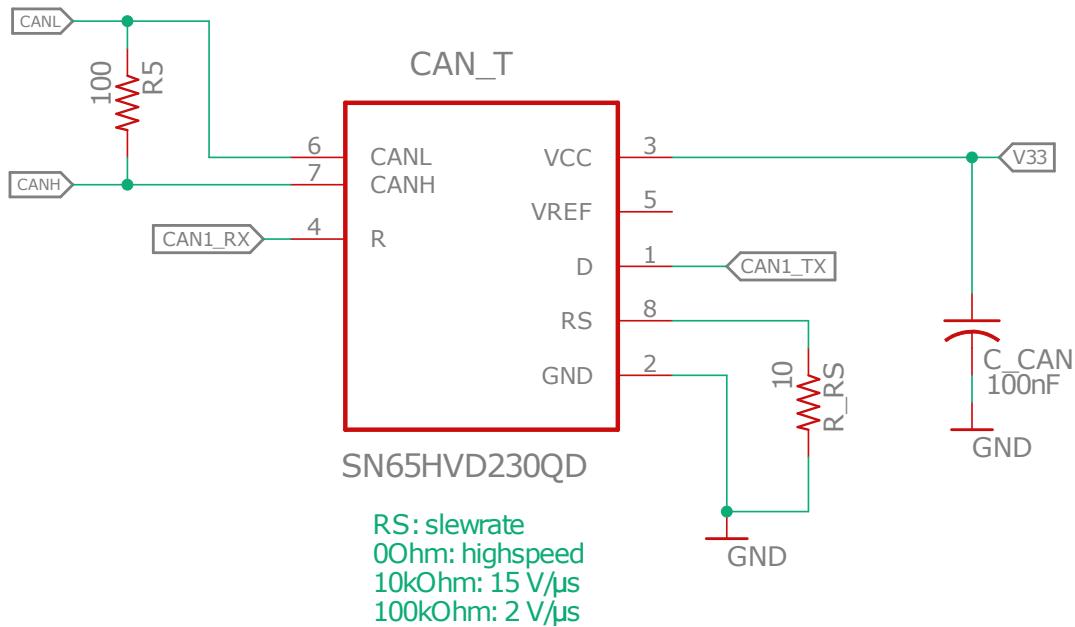


Abbildung 4.5: CAN-Transciever Verschaltung

Über den Pin RS lassen sich durch den Widerstand R_RS verschiedene *slew rates* einstellen und damit, wie schnell der Ausgangstransistor bei der Übermittlung von Nachrichten durchschaltet [cantrans]. Für einen Widerstandswert von 0Ω ist die Geschwindigkeit maximal, während sie bei $10\text{k}\Omega$ etwa $15\frac{\text{V}}{\mu\text{s}}$ beträgt. Zwischen CAN-HIGH und CAN-LOW wird nach dem Datenblatt ein 120Ω geschaltet. Bei der Wahl von R5 und R_RS wurde sich an der bestehenden Konfiguration des STM-Discovery-Shields orientiert, mit der die CAN-Kommunikation aufgebaut wurde.

4.2.3 Spannungsversorgung

Die Elektronik wird weiterhin mit dem bisher verwendeten Manson SBC-2130 Battery Charger versorgt. Dieser stellt eine konstante Spannung von 13.8 V. Da die verschiedenen Komponenten jedoch Versorgungsspannungen von 3,3 Volt

und 5 Volt benötigen, muss die Schaltung durch einen Spannungsregler erweitert werden. Zusätzlich werden dadurch Schwankungen in der Eingangsspannung geglättet.

Low Dropout Spannungsregler

Ein LDO ist ein Festspannungsregler, der eine festgelegte und somit invariable Ausgangsspannung liefert, die sich auch dann nicht ändert, wenn die Eingangsspannung schwankt. Die Schaltung eines LDO-Reglers besteht aus einer Referenzspannungsquelle, einem Differenzverstärker und einem Stellglied in Form eines Leistungstransistors. Die hier verwendeten Ausführungen sind P-Kanal-MOSFET-basierte Regler. Das Blockschaltbild eines solchen LDOs ist in folgender Abbildung schematisch dargestellt.

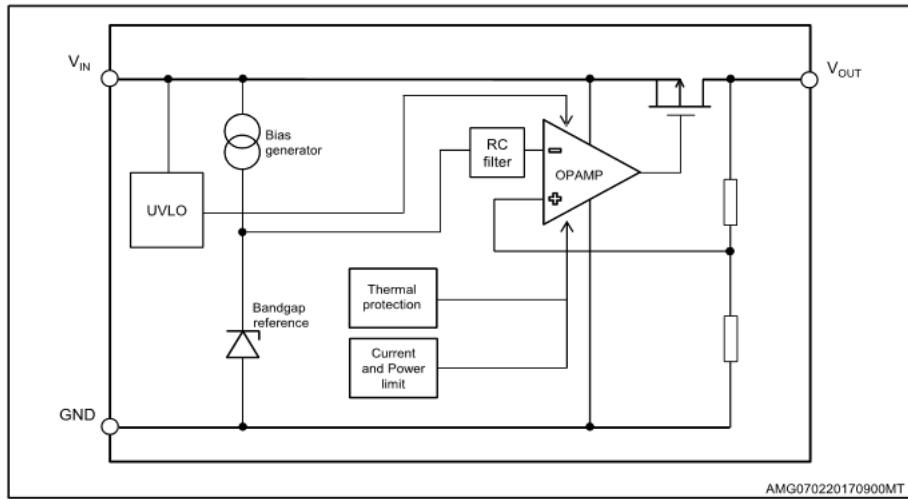


Abbildung 4.6: Blockschaltbild LDO

Der Differenzverstärker vergleicht die Ausgangsspannung mit einer stabilen Referenzquelle aus einer Zenerdiode, sodass die gemessene Spannungsabweichung über das Stellglied ausgeregelt werden kann. Ist die Ausgangsspannung zu niedrig, so wird der Transistor stärker angesteuert bis die geforderte Ausgangsspannung erreicht wird, im umgekehrten Fall wird der Strom über den Transistor reduziert. Der Transistor wird in dieser Schaltung also quasi wie ein veränderlicher Widerstand verwendet, an dem die überflüssige Spannungsdifferenz abfällt und in Wärme umgewandelt wird. Die verwendeten LDOs besitzen außerdem eine Strombegrenzungsschaltung und eine Schutzschaltung, die die Betriebstemperatur überwacht und das Bauteil vor thermischer Überlastung schützt, sowie eine Unterspannungsabschaltung.

Verschaltung auf Platine

Grund für die Wahl dieser Art von Spannungsreglern für die Platine ist ihre kompakte Bauform, ihr günstiger Einkaufspreis und das geringe Rauschen im Vergleich zu Schaltreglern da keine Schaltvorgänge auftreten. Auf der Platine werden LDOs vom Typ LDL1117 von STMicroelectronics verwendet. Für die 3.3 V Versorgung ist dies der LDL1117S33R und für die 5 V der LDL1117S50R LDO. Nach dem Datenblatt [Ido] ist die Eingangskapazität zu $1 \mu\text{F}$ und die Ausgangskapazität zu $4.7 \mu\text{F}$ zu wählen. Diese werden aus Stabilitätsgründen und zur Entkopplung verwendet. Empfohlen wird weiterhin Keramikkondensatoren zu verwenden, welche X5R oder X7R Dielektrika aufweisen. In Abbildung 4.7 ist der Schaltplan der Spannungsversorgung für die Platine zu sehen. An VIN wird die Batteriespannung angelegt, welche mit $1 \mu\text{F}$ gegen GND entkoppelt wird. An VOUT liegen die jeweiligen 3.3 V bzw. 5 V an, welche jeweils über $4.7 \mu\text{F}$ gegen GND geschaltet sind. Zur allgemeinen Spannungsglättung wird eine große Kapazität zwischen der Batteriespannung und GND geschaltet, welche starke Schwankungen beim Durchschalten der H-Brücke verhindern soll.

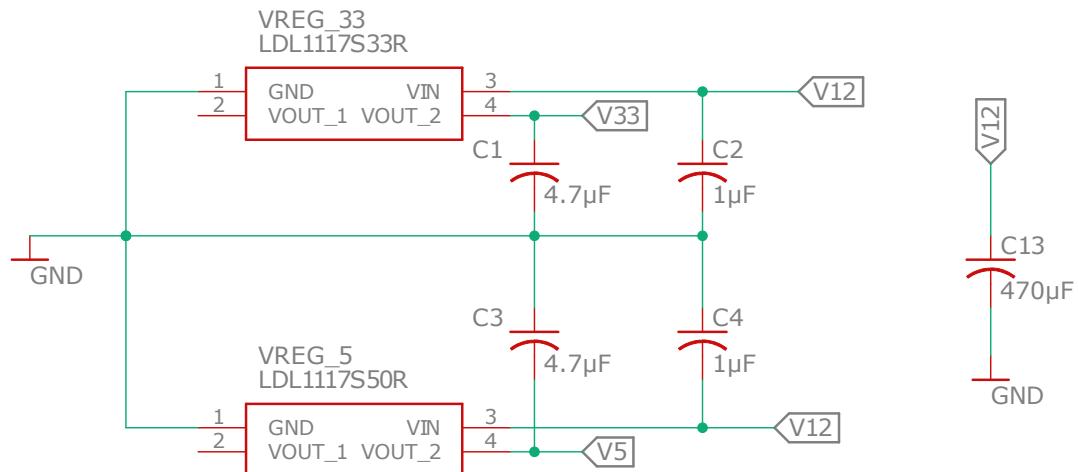


Abbildung 4.7: Schaltplan Spannungsversorgung

4.3 Platinenanschluss

Die Platine wird mit einem 1-776267-1 von TE Connectivity verbunden um notwendige Signalleitungen nach außen zu führen und zusätzlich Spannungsversorgung und Lagesensorik mit der Platine verbinden zu können. In Abbildung 4.8 ist die Pin-Belegung des Steckers zu sehen. Um den Mikrocontroller über SWD programmieren zu können, muss der Stecker die notwendigen Signalleitungen SWDIO, SWCLK, TRACESWO, V33 und GND nach außen führen. Über diese Pins können über einen ST-Link/V2 Programme von einem Computer auf den Mikrocontroller übertragen werden. Weiterhin soll der Mikrocontroller über CAN-Signale mit der MicroAutobox kommunizieren können. Dazu werden die Schnittstellen CANH(CAN-HIGH) und CANL(CAN-LOW) über den Stecker nach außen geführt. Die Steckerpins V5, L_1, L_2 und GND werden für die Lagesensorik benötigt. Pin 4 und Pin 5 sind beide mit der Batterie verbunden und bilden die Spannungsversorgung der gesamten Platine. Aufgrund der potentiell hohen Ströme wird die Versorgung über zwei Pins zugeführt. Über den NRST-Pin lässt sich der Mikrocontroller zurücksetzen.

Als Gegenstück des Steckers existieren drei verschiedene Varianten. Jeweils eine Variante sind Programmier- und Betriebsstecker, die dritte Variante vereint diese beiden Optionen. Der Programmierstecker beinhaltet dabei alle Eingänge der oben genannten Signalleitungen, die zum Flashen des Mikrocontrollers über SWD nötig sind, und wird nur kurzzeitig verwendet. Den Rest der Zeit wird der Betriebsstecker verbunden, welcher die Leitungen zur Spannungsversorgung und Lagesensorik führt. Der kombinierte Stecker ist vor allem in der Entwicklungsphase hilfreich, um nicht ständig den Anschluss wechseln zu müssen.

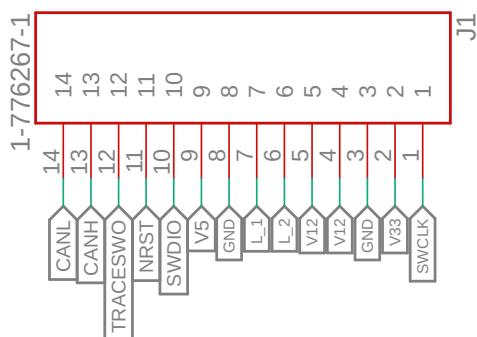


Abbildung 4.8: Anschlusspins Stecker

4.4 H-Brücke

Um den Aktor in beide Richtungen betreiben zu können wird eine elektronische Schaltung benötigt, welche eine Stromrichtungsumkehr ermöglicht. Eine einfache Möglichkeit bietet die sogenannte H-Brückenschaltung, die vereinfacht in

Abbildung 4.9 dargestellt ist. Eine H-Brücke ist eine Vollbrücke bestehend aus vier Schaltern und demnach ein Vierquadrantensteller. Werden die Schalter S_1 und S_4 geschlossen, kommt es zu einem Stromfluss durch den Aktor. Werden hingegen lediglich die Schalter S_2 und S_3 geschlossen, fließt der Strom entgegengesetzt. Ein Kurzschluss entsteht, wenn S_1 und S_3 oder S_2 und S_4 gleichzeitig geschlossen werden. Die Kombination aus Schließung von S_1 und S_2 oder S_3 und S_4 resultiert in einem ungeschlossenen Schaltkreis.

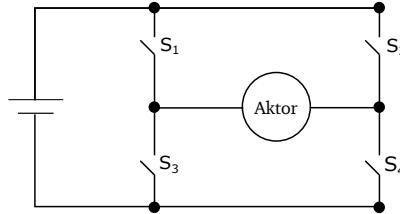


Abbildung 4.9: Vereinfachter Aufbau einer H-Brücke

In realen Systemen werden die Schalter durch Transistoren realisiert. Um Kurzschlüsse zu vermeiden, bietet sich die Schaltung einer H-Brücke über zwei Halbbrücken-ICs an. Jede Halbbrücke besteht dabei aus zwei Transistoren, wobei die integrierte Schaltung verhindert, dass beide gleichzeitige durchschalten. Bisher wurde am Aktorprüfstand eine gekaufte H-Brückenschaltung verwendet, die auf zwei BTS7960-Halbbrücken der Firma Infineon basiert. Diese lieferte in vorangegangenen Arbeiten zufriedenstellende Leistungsergebnisse. Da sie jedoch lediglich einen maximalen Stromfluss von 43 A zulässt, die Schaltung aber in Zukunft auch für einen Aktor mit 50 A benutzt werden soll, kann sie für die zu entwickelnde H-Brücke nicht verwendet werden. Eine Alternative ist die BTN8982-Halbbrücke, welche auch von der Firma Infineon produziert wird und für einen maximalen Strom von 55 A ausgelegt ist. Das Blockdiagramm dieser Halbbrücken ist in Abbildung 4.10 dargestellt.

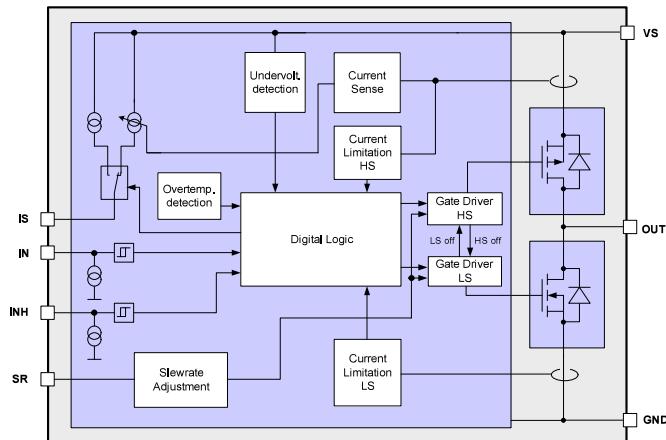


Abbildung 4.10: Blockdiagramm der BTN8982 Halbbrücke

Sie besteht aus einem *p-channel highside MOSFET*, einem *n-channel lowside MOSFET* und einer *Driver IC*, die *logic level inputs* unterstützt, wodurch der Anschluss an einen Microcontroller erleichtert wird. Die BTN8982 ist in einem Temperaturbereich von -40°C bis 150°C einsetzbar und kann mit einer Versorgungsspannung von bis zu 40 V betrieben werden, wobei diese im normalen Betrieb zwischen 8 V bis 18 V liegen sollte. Zusätzlich sind mehrere Schutzfunktionen integriert, beispielsweise ein Überstrom- und ein Übertemperaturschutz. Die BTN8982 besitzt acht Ausgänge, welche in Tabelle 4.3 aufgelistet und im folgenden näher erläutert werden.

Pin Nummer	Bezeichnung	Erläuterung	Anschluss an
1	GND (Ground)	Erdung	Ground MCU
2	IN (Input)	definiert die Schalterstellung (1 = High Switch Mode; 0 = Low Switch Mode)	O-Pin MCU
3	INH (Inhibit)	1: Betriebsmodus, 0: Schlafmodus	O-Pin MCU
4, 8	OUT (Output)	Ausgang der Brückenschaltung	Aktor
5	SR (Slew Rate)	Einstellen der Steigung der Spannungsantwort	Ground über Widerstand
6	IS (Status)	Strommessung & Fehlererkennung	I-Pin MCU
7	VS (Supply)	Stromversorgung	Batterie

Tabelle 4.3: Pinverteilung Halbbrücken

Pin 1 ist der GND-Pin und bestimmt das Bezugsniveau der Versorgungsspannung, welche an Pin 7 angelegt wird. Ob Strom durch den highside MOSFET oder den lowside MOSFET fließt, wird durch Pin 2 (IN) bestimmt. Liegt eine 1 an fließt Strom durch den highside MOSFET, während bei einer 0 Strom durch letzteren fließt. Wird die Halbbrücke über ein PWM Signal geschaltet, geschieht dies über das Signal an diesem Pin. Der dritte Pin (INH) setzt die Halbbrücke in einen Schlafmodus, solange eine 0 anliegt. Somit kann die Halbbrücke nur genutzt werden, wenn eine 1 anliegt. Der Schlafmodus wird als Notaus verwendet, falls die Software einen kritischen Fehler detektiert (vgl. Unterabschnitt 6.2.10). Pin 4 und 8 (OUT) sind die Pins an die die Versorgungsspannung durchgeschaltet wird und stellen damit die direkte Schnittstelle mit dem Aktor da. Pin 5 (SR) bestimmt die Slew Rate des PWM Signals IN-Pin. Der sechste Pin (IS) ist für die Strommessung, sowie für die Meldung von Fehlern zuständig. Die Funktionsweise ist in Abbildung 4.11 dargestellt.

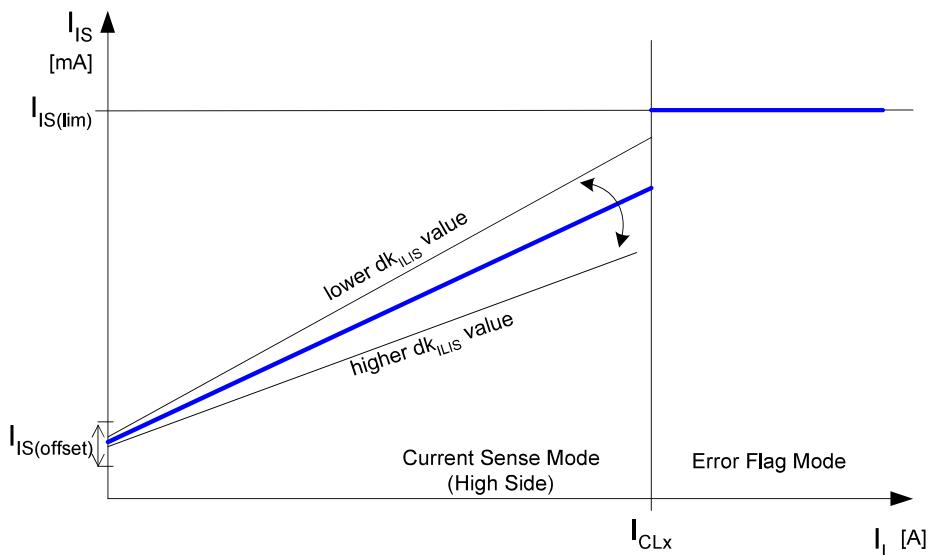


Abbildung 4.11: Funktionsweise des IS-Pins

Im normalen Operationsmodus (Current Sense Mode) liefert eine integrierte Stromquelle einen Strom I_{is} der proportional zum durchgeschalteten Strom I_L ist. Tritt ein Fehler auf wie zum Beispiel Übertemperatur oder Überstrom, schaltet die Halbbrücke in den Error Flag Mode. Hierbei wird die Stromquelle am IS-Pin gewechselt, welche einen konstanten Strom I_{IS} liefert. Dieser beträgt maximal 6.5 mA. Der Proportionalitätsfaktor k ist durch

$$k = \frac{I_{L1} - I_{L2}}{I_{IS}(I_{L2}) - I_{IS}(I_{L1})} \quad (4.1)$$

definiert und beträgt laut Datenblatt typischerweise 19.500. Messungen haben gezeigt, dass k in der geschalteten H-Brücke unterhalb dieses Wertes liegt, sodass er als obere Abschätzung des geflossenen Stroms verwendet werden kann. Die Spannungen an IN- und INH-Pins und somit auch die Funktionalität der H-Brücke wird durch einen hochgeschwindigkeits Leistungsverstärker/Puffer (CD74HCT125) bereitgestellt. Leistungsverstärker sind elektronische Verstärkerschaltungen, die eingesetzt werden, um die Qualität elektrischer Signale zu verbessern [Conrads2014]. Der schematische Schaltplan des verwendeten Leistungsverstärkers und die zugehörige Logiktabelle in Abbildung 4.12 dargestellt. Der CD74HCT125 besitzt vier unabhängige Wege, die getrennt voneinander aktiviert werden können. Diese Aktivierung

geschieht hierbei über ein *Low Level* Spannungssignal am nOE-Pin. Low Level bedeutet, dass die Spannung maximal 1.35 V betragen darf. Aufgrund des Platinendesigns Kapitel 5 lässt sich dies recht problemlos bewerkstelligen, was neben der Hitzebeständigkeit und Schnelligkeit das Hauptargument für diesen Leitungsverstärker ist. Ist ein Weg aktiviert, sorgt ein *High Level* Spannungssignal am Eingang (nA) dafür, dass die Versorgungsspannung (VCC) an den jeweiligen Ausgang (nY) durchgeschaltet wird. In der ausgeführten Schaltung wird am VCC-Pin des CD74HCT125 eine Spannung von 5 V angelegt. Die ersten drei Wege sind direkt mit GND verbunden, wodurch eine dauerhafte Aktivierung realisiert wird. Die ersten beiden Wege sorgen für die Spannung an den IN-Pins der Halbbrücke, während der dritte Weg die INH-Pins versorgt. Die Eingänge sind dabei mit dem Microcontroller und die Ausgänge mit den Pins der Halbbrücke verbunden. Somit wird das PWM-Signal des Microcontrollers, welches eine maximale Spannung von 3.3 V besitzt, auf ein PWM-Signal mit einer maximalen Spannung von 5 V angehoben.

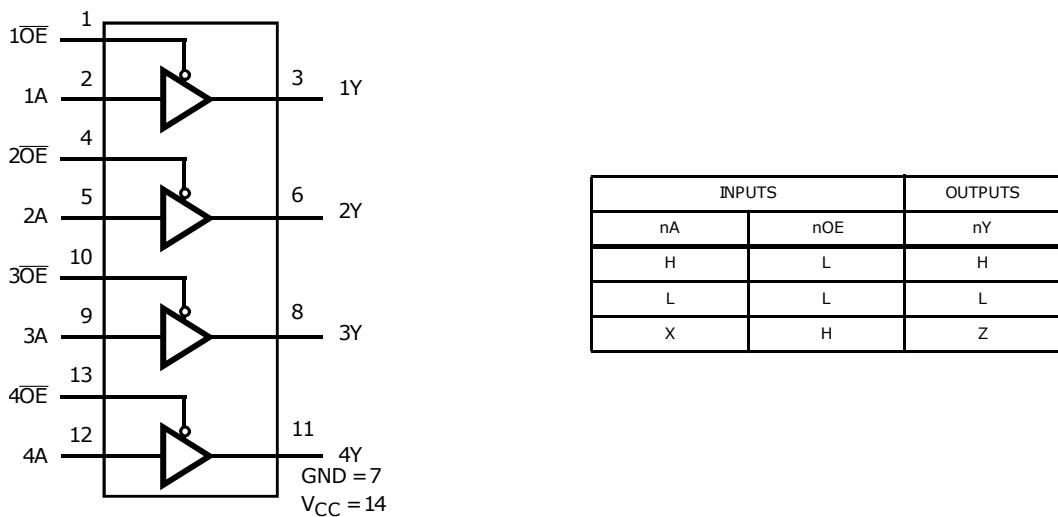


Abbildung 4.12: Schematischer Aufbau und Logiktabelle des CD74HCT125 Leitungsverstärkers

Die H-Brücke kann auch ohne einen Leitungsverstärker direkt mit dem Microcontroller verbunden werden, was zur Folge hätte, dass keine 5 V sondern 3.3 V Spannung an den Pins der H-Brücke anliegen. Abbildung 4.13 zeigt den geflossenen Strom durch den dabei angeschlossenen Aktor mit und ohne Leitungsverstärker bei variierenden PWM-Signalen. Es ist zu sehen, dass die Messung mit Geschalteten Leitungsverstärker deutlich symmetrischer sind, was auf eine bessere bidirektionale Schaltbarkeit schließen lässt. Außerdem sind die maximal durchgeschalteten Ströme größer. Dies lässt vermuten, dass die 3.3 V nicht ausreichen um die Transistoren in den Halbbrücken vollständig schalten zu lassen. Dementsprechend wurde sich bei der endgültigen Schaltung für den Einsatz eines Leitungsverstärkers entschieden.

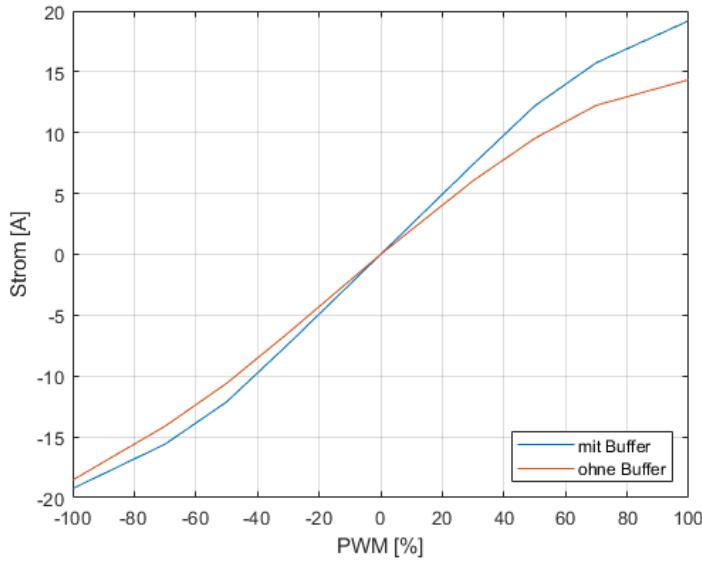


Abbildung 4.13: Strommessung mit und ohne Leitungsverstärker bei unterschiedlichen PWM-Signalen

Der schematische Aufbau der verwendeten H-Brücke ist in Abbildung 4.14 zu sehen und wird im Folgenden genauer erläutert. Bei dem hier vorgestellten Aufbau wurde sich am Datenblatt der BTN8982-Halbbrücken orientiert.

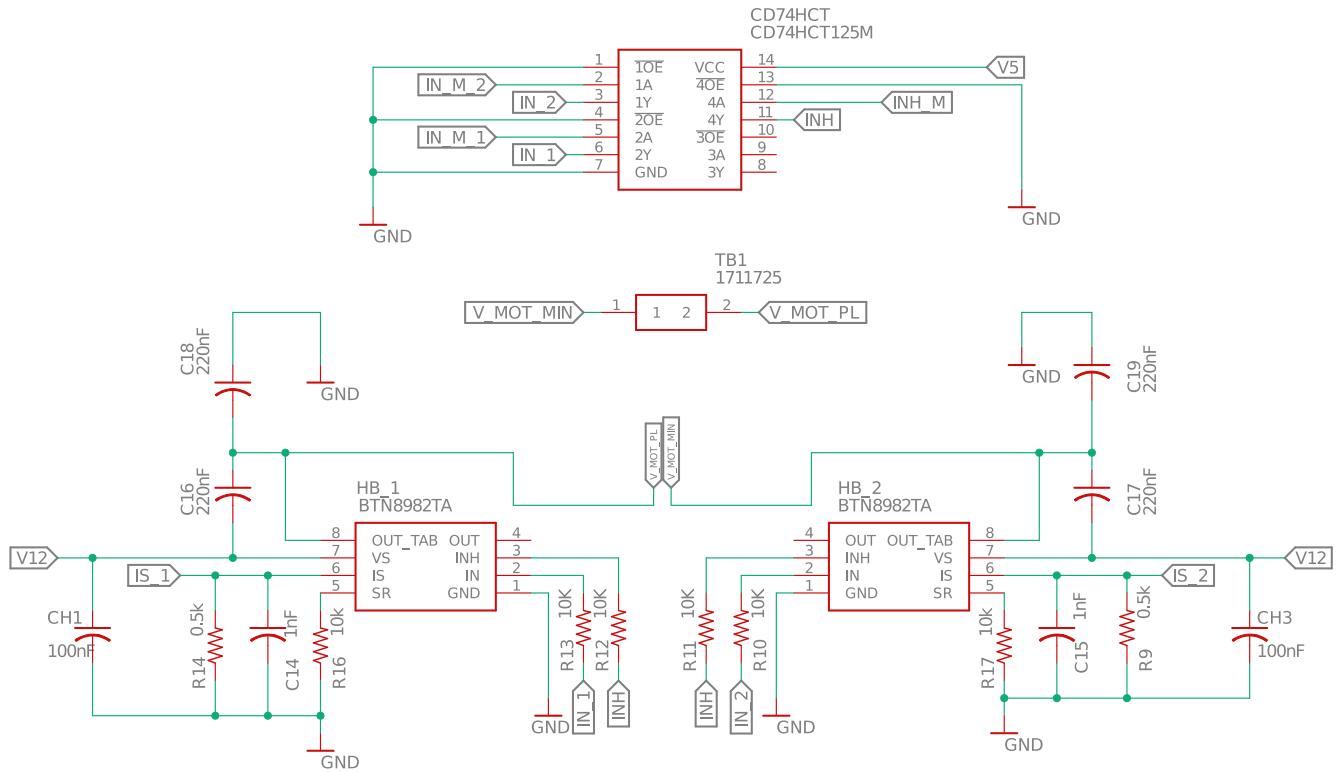


Abbildung 4.14: Schematischer Aufbau der H-Brücke

An beiden Halbbrücken wird am VS-Pin die Versorgungsspannung angelegt. Nahe am jeweiligen VS-Pin ist ein 100 nF Kondensator (CH1, CH3) gegen GND geschaltet, welcher die Schwankungen der Versorgungsspannung, z.B. verursacht durch andere Komponenten, glättet. Ob diese Spannung an die OUT-Pins durchgeschaltet wird, hängt von der Spannung an den IN- und INH-Pins ab. Die INH-Pins setzen die Halbbrücken in einen *Sleepmode*, wenn an ihnen keine Spannung zwischen 3 und 5.3 V angelegt ist. Sofern diese Spannung anliegt, sorgt der gleiche Spannungsbereich am IN-Pin dafür, dass die Versorgungsspannung an die OUT-Pins der Halbbrücke durchgeschaltet wird. Jede Halb-Brücke besitzt zwei solcher OUT-Ausgänge, wobei einer über einen Pin und der andere über eine große Fläche auf der Rückseite realisiert wird.

Letzterer eignet sich besser zum Übertragen großer Ströme, weshalb dieser zur Aktoransteuerung verwendet wird. Die IN- und INH-Pins werden über $10\text{ k}\Omega$ Widerstände (R10,R11,R12,R13) mit den Ausgängen des Leistungsverstärkers verbunden. Diese Widerstände werden zum Schutz der digitalen Eingänge benötigt. Die IS-Pins der Halbbrücke werden direkt mit dem Mikrocontroller verbunden. Parallel sind $0.5\text{ k}\Omega$ (R9,R14), sowie ein 1 nF Glättungskondensator (C14,C15) gegen GND geschaltet. Diese Pins sind für die Sensorik der Halbbrücke verantwortlich. An ihnen ist eine Stromquelle angeschlossen, die einen zum durchgeschalteten Strom proportionalen Strom liefert. Abhängig von den gegen GND geschalteten Widerständen ergibt sich eine Spannung die vom Mikrocontroller gemessen und aus der der durchgeschaltete Strom berechnet werden kann. Da der Strom am IS-Pin maximal 6.5 mA beträgt, resultiert durch einen $0.5\text{ k}\Omega$ eine maximale Spannung von 3.25 V am Mikrocontroller. Ein größerer Widerstand hätte eine größere Spannung zur Folge, wodurch eine Schädigung des Mikrocontrollers nicht mehr ausgeschlossen werden kann. Die Widerstände, die zwischen SR-Pins und GND geschaltet sind (R16,R17), bestimmen die Slewrate des durchgeschalteten Signals. Dieser darf laut Datenblatt zwischen 0Ω und $51\text{ k}\Omega$ liegen. Der Hersteller bietet zudem eine Simulation an, in der verschiedene Slew-Rate-Widerstände getestet werden können. Die Ergebnisse einer solchen Simulation mit den eingestellten Widerständen 0Ω , $10\text{ k}\Omega$ und $51\text{ k}\Omega$ sind anhand des durchgeschalteten Stroms I_{OUT} in Abbildung 4.15 dargestellt. Es ist zu sehen, dass der durchgeschaltete Strom mit zunehmendem Widerstand abnimmt, die Welligkeit jedoch nahezu gleich bleibt. Allerdings gilt es zu beachten, dass die *electromagnetic Interference* (kurz: EMI) für kleinere Slew-Rate-Widerstände ansteigt. Ein Widerstand von $10\text{ k}\Omega$ stellt dabei einen guten Kompromiss dar und liefert auch in realen Messungen sehr gute Ergebnisse.

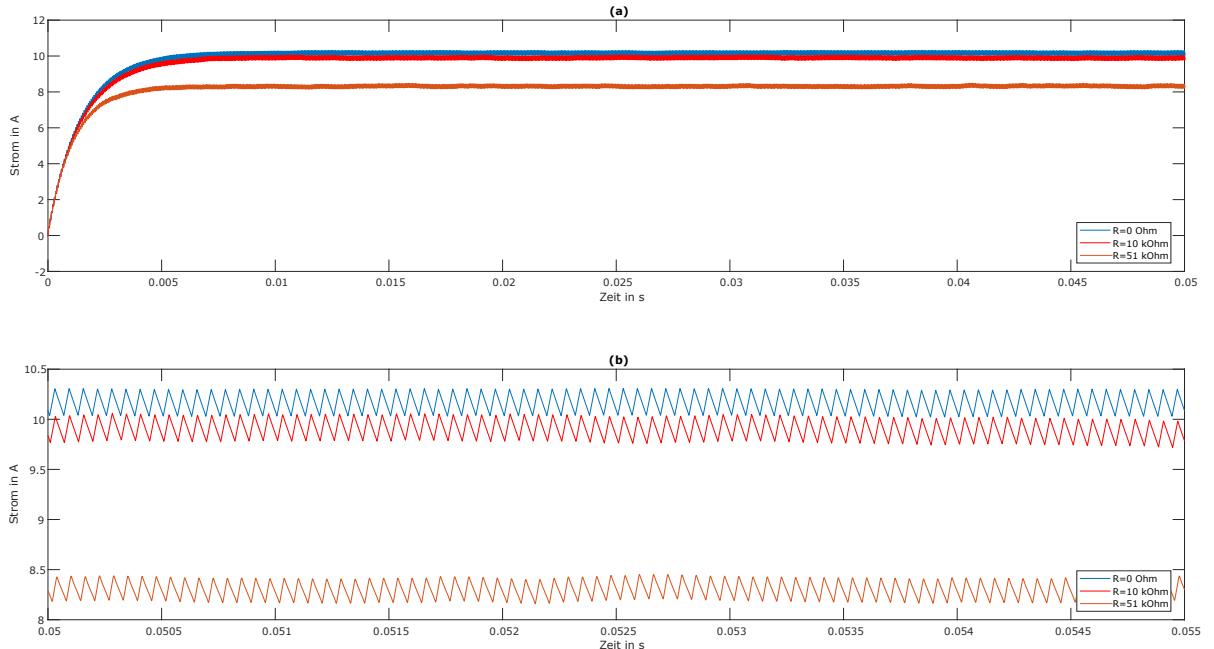


Abbildung 4.15: (a):Simulationsergebnis des durchgeschalteten Stroms aufgetragen über der Zeit,(b): Welligkeit des durchgeschalteten Stroms

Experimente zeigen die besten Ergebnisse bei einem Slew-Rate-Widerstand von $10\text{ k}\Omega$.

4.5 Sensorik

Die Sensorik des Gesamtsystems besteht aus einem Stromsensor, einem Temperatursensor und einem Lagesensor. Während die ersten beiden wichtige Kontroll- und Leistungsgrößen liefern, ist der Lagesensor essentiell für die Schaltvorgänge und damit die Funktionalität des Gesamtsystems. Der Stromsensor ist in den BTN8982 Halbbrücken integriert und ist bereits beschrieben. Die Funktionsweise und Verschaltung des Temperatur- und Lagesensors werden im folgenden erläutert.

4.5.1 Temperatursensor

Als Temperatursensor wird ein B57861S0103A039-Thermistor verwendet. Dieser besteht aus einem Sensor und zwei 350 mm langen Anschlusskabeln. Thermistoren sind elektrische Widerstände, deren Wert temperaturabhängig variiert.

Dabei werden sie in die zwei Gruppen Heiß- und Kaltleiter aufgeteilt [Stiny2015]. Da der B57861S0103A039-Thermistor zur ersten Gruppe gehört, wird im folgenden ausschließlich auf diese eingegangen. Heißleiter besitzen einen negativen Temperaturkoeffizient (NTC), was bedeutet, dass ihr elektrische Widerstand sich mit steigender Temperatur reduziert. Dieser Zusammenhang ist allerdings hochgradig nichtlinear und kann durch

$$R(T) = R_R \cdot e^{B\left(\frac{1}{T} - \frac{1}{T_R}\right)} \quad (4.2)$$

beschrieben werden mit der Referenztemperatur T_R , dem Widerstand R_R bei der Referenztemperatur und dem Beta-Wert B . Als Referenztemperatur wird meistens 25 °C gewählt. Der Beta-Wert ist eine Materialkonstante, die üblicherweise zwischen 1500 K und 6000 K liegt [Stiny2015]. Für den B57861S0103A039-Thermistor beträgt der Widerstand R_R 10 kΩ und der Beta-Wert B 3988 K. Der Betriebstemperatur liegt zwischen –55 °C und 155 °C, was in den typischen Bereich für Heißleiter fällt [Stiny2015].

Eine mögliche Schaltung zur hochgenauen Temperaturmessung mittels Heißleitern ist die Wheatstonsche-Brückenschaltung (Abbildung 4.16) bestehend aus drei ohmschen Widerständen und einem NTC-Thermistor [Stiny2015].

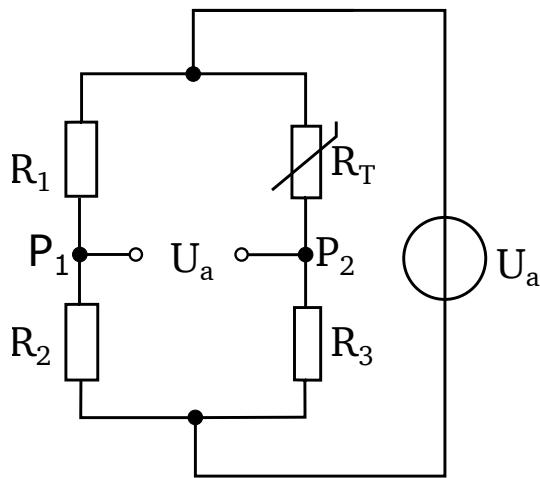


Abbildung 4.16: Wheatstone-Brückenschaltung

Solange die Brücke ausgeglichen ist, liegt keine Spannungsdifferenz zwischen P_1 und P_4 vor. Ändert sich der Widerstand des Thermistors, wird der ausgeglichene Zustand verlassen und es kommt zu einer Spannungsdifferenz, aus der der veränderte Widerstand des Thermistors bestimmt werden kann. Wird Gleichung 4.2 nach der Temperatur T umgestellt, lässt sich daraus die aktuelle Temperatur berechnen. Aus Platzgründen auf der endgültigen Platine und da die hiermit erzielbare Genauigkeit nicht benötigt wird, ist stattdessen ein Spannungsteiler zum Einsatz gekommen. Dieser Spannungsteiler teilt den am Thermistor und an einem festen Widerstand abfallende Spannung und ist in Abbildung 4.17 dargestellt. Aus der Spannung, die am konstanten Widerstand R_c abfällt, kann auf die Spannung geschlossen werden, die am Thermistor R_t abfällt und somit auch auf dessen aktuellen Widerstand. Über Gleichung 4.2 ist dann ein direkter Zusammenhang zur gemessenen Temperatur gegeben. Die Spannungsmessung findet im Mikrocontroller bzw. in dessen Analog-Digital-Convertern (ADC) statt. Dabei sind für eine möglichst exakte Messung mehrere Umstände zu beachten, die nun exemplarisch erörtert werden. Die Überlegungen gelten jedoch auch für die übrigen Spannungsteiler und ADCs. Im STM32F4 kommen *successive approximation analog-to-digital converter* zum Einsatz, die eine Auflösung von bis zu 12 bit liefern [stmref]. Die Messdauer ist dabei konfigurierbar, entspricht jedoch mindestens für jedes Auflösungs-Bit und 3 zusätzliche Takte (12 + 3 Takte) [stmref]. Dieser Takt f_{ADC} entspricht dabei nicht dem Kerntakt, sondern dem *Advanced Peripheral Bus Clock* (APBC), der sich durch konfigurierbare Takt-Teiler vom Kerntakt unterscheidet.

Durch den Aufbau des ADC's bedingt, führen zu große Eingangswiderstände (bzw. Widerstandswerte im Spannungsteiler) zu großen Ungenauigkeiten. Dies ist umso kritischer je kürzer die Messdauer beträgt, da sich die interne Kapazität im ADC bei großen Eingangswiderständen nur langsam aufladen kann. Nach Ablauf der Abtastdauer sollte jedoch die interne Kapazität auf die tatsächlichen Spannung am Eingangspin aufgeladen sein, sonst kommt es zu Abweichungen (siehe dazu auch [ADC], [stm32]). ST stellt für den Mikrocontroller Gleichung 4.3 bereit, durch welche sich der maximale Eingangswiderstand

$$R_{AIN} = \frac{(k - 0,5)}{f_{ADC} C_{ADC} \ln(2^{N+2})} - R_{ADC} \quad (4.3)$$

abhängig der Messdauer berechnen lässt. Dabei entspricht k den konfigurierten Messtakten pro Messung, C_{ADC} der Kapazität des internen ADCs, N der Auflösung und R_{ADC} dem Schaltwiderstand des ADCs. Über k und f_{ADC} lässt sich dabei die

Messdauer beeinflussen und somit die erlaubten Eingangswiderstände am Spannungsteiler beeinflussen. Für eine effektive Messdauer von $2.7 \mu\text{s}$ muss der Eingangswiderstand unter $31.355 \text{ k}\Omega$ liegen, während bei einer Messdauer von nur $1.5 \mu\text{s}$ nur Eingangswiderstände bis 440.6Ω für hohe Genauigkeit zulässig sind. Der Nachteil niedrigere Widerstände im Spannungsteiler liegt jedoch in der Belastung des jeweiligen Sensorausgangs. Dort kann nur ein begrenzter Strom bereit gestellt werden. Daher bietet es sich an, eine Kapazität parallel zum Eingang des ADCs und somit zu C_{ADC} zu schalten, die als Puffer für C_{ADC} dient [ADC]. Hier muss jedoch beachtet werden, dass eine weitere Kapazität zusätzlichen Bauraum, Kosten und Fehlerpotential zur Folge hat. Darüber hinaus werden schnelle Signaländerungen durch die RC-Zeitkonstante zum Auf-/Entladen des Kondensators ggf. herausgefiltert. Für den Thermistor werden keine kritischen Temperatursprünge erwartet, daher wird in diesem Fall eine zusätzliche Kapazität C_c eingesetzt.

Auch durch einen Impedanzwandler, also eine Operationsverstärkerschaltung mit einer Spannungsverstärkung von 1, lässt sich der Eingangswiderstand durch den Spannungsteiler kompensieren. Auch hier ergeben sich jedoch Nachteile durch benötigten Bauraum, anfallende Kosten, gesteigertes Fehlerpotential und zusätzliches Rauschen.

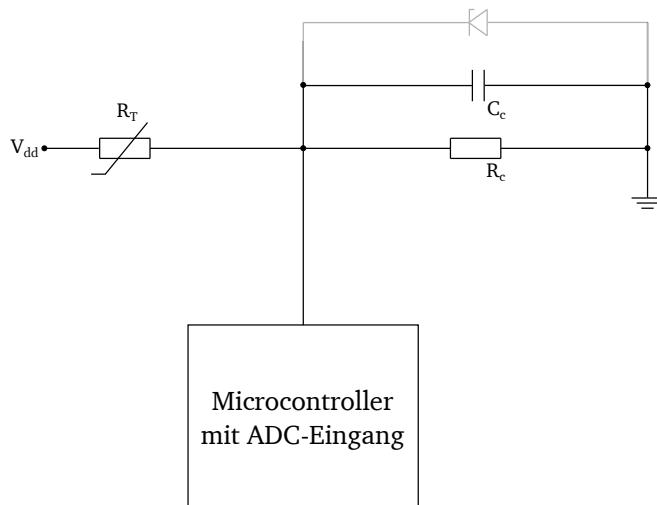


Abbildung 4.17: Schaltung zum Auslesen des Thermistors

Durch den Temperatursensor soll die Spulentemperatur im Tauchspulenaktor gemessen werden, da hier durch eine Überhitzung der Isolation Gefahrenpotential ausgeht. In der Umgebung der Spulen wird jedoch eine große Induktionswirkung erwartet, die die Genauigkeit der Spannungsmessung gefährden oder sogar ganz außerhalb des Messbereichs von 0 V bis 3.3 V liegen und somit die Funktionsfähigkeit des Mikrocontrollers gefährden [stm32]. Eine Gegenmaßnahme stellt die Verseilung der Zuleiterkabel dar. Darüber hinaus kann durch die Schirmung der Kabel eine Schirmdämpfung erreicht werden, was die Induktionswirkung weiter reduziert [Wolfspurger]. Falls trotzdem eine Spannung induziert wird, kann ein Strom im mA-Bereich über interne Dioden im ADC abgeleitet werden [stmref]. Um den Strom zu begrenzen, muss sich ein Widerstand zwischen induzierter Spannung und ADC-Eingang befinden. Eine weitere Methode den Pin vor einer Überspannung zu schützen, stellt eine Zener-Diode dar, die parallel zum Eingang des ADC's geschaltet wird (grau angedeutet). Hierbei ist jedoch darauf zu achten, dass auch im Normalbetrieb ein Kriechstrom durch die Diode fließt, der das Messergebnis verfälschen kann.

Um Probleme dieser Art zu umgehen wurden auch alternative Sensorkonzepte in Betracht gezogen, die nicht über ein ADC ausgewertet werden. Diese haben sich jedoch als zu rechenaufwändig beim Auslesen erwiesen.

4.5.2 Lagesensor

Die Funktionsweise des PLCS-25M Sensors ist in Kapitel 2 bereits erläutert worden. Der Sensor gibt demnach ein Spannungssignal aus, das nach Gleichung 2.2 in den Schaltgabelweg umgerechnet werden kann. Hierbei ist zu beachten, dass der Lagesensor mit 5 V Gleichspannung betrieben wird und demnach der mögliche Ausgangsspannungsbereich zwischen 0 V und 5 V liegt, was außerhalb des gültigen Messbereichs der ADCs liegt. Über einen Spannungsteiler wird die Ausgangsspannung des Lagesensors demnach in den gültigen Messbereich übersetzt. Es gelten dabei bei der Auslegung des Spannungsteilers die gleichen Überlegungen wie in Unterabschnitt 4.5.1. Um darüber hinaus eine Verbesserung der Positionsbestimmung zu erreichen, wird auch der komplementäre Ausgang des Lagesensors gemessen und entsprechend verrechnet. Der dadurch erreichte Genauigkeitsgewinn wird in Kapitel 7 thematisiert.

4.5.3 Sensorik Eingangsspannung

Um Fehlerzustände in der Spannungsversorgung feststellen zu können, ist ein einfacher Spannungsteiler zu entwerfen, anhand dessen die Eingangsspannung in einen für den ADC lesbaren Bereich gebracht wird. Es gelten die gleichen Überlegungen wie in Unterabschnitt 4.5.1.

5 Platinenentwurf

Im Folgenden wird die Gedankengänge und Einflüsse eingegangen, die zum endgültigen Platinendesign geführt haben, eingegangen. Dabei wird neben der reinen Platinenbetrachtung auch die Rahmenbedingungen erläutert.

5.1 Elektronikgehäuse und Anforderungen an die Platinendimensionierung

Die Platine wird in ein Gehäuse eingebettet, welches parallel zu und in Absprache mit dieser Arbeit entwickelt wurde. Die Anforderung, den Aktor durch die entwickelte Elektronik zu einem *Smart Actuator* zu transformieren wird erfüllt, in dem die Platine durch das Gehäuse am Aktorgehäuse verschraubt wird. Abbildung 5.1 zeigt schematisch die Lage der Platine (grün) im Gehäuse. In grau ist der Verbindungsstecker zu sehen, der abgedichtet im Gehäuse liegt. Anschließend wird ein Deckel montiert, wodurch die Elektronik von Schmutzpartikeln und anderen Umwelteinflüssen geschützt wird. Dieser ist aus Übersichtsgründen nicht abgebildet.

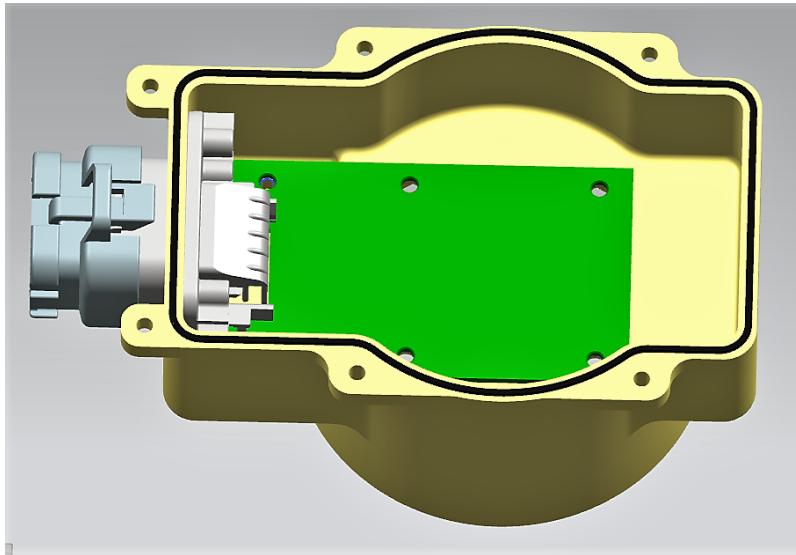


Abbildung 5.1: Derzeitiger Entwurf des Elektronikgehäuses

Durch die Größe des Aktors ergeben sich Höchstmaße an das Platinengehäuse und die Platine selbst. Um ein Einbauen zu garantieren, dürfen die Außenmaße der Platine $88.8 \text{ mm} \cdot 50 \text{ mm}$ nicht überschreiten. In diesem Fall war der Platz ausreichend, um alle nötigen Schaltungen und Funktionen auf der Platine zu verwirklichen. Als Ausweichlösung, falls die ebenen Maße nicht ausgereicht hätten war angedacht, die Elektronik auf zwei Platinen aufzuteilen (z.B. Logik- und Leistungsteil), und sie in zwei Ebenen parallel übereinander anzurordnen. Dies ist weiterhin denkbar, falls die Elektronik erweitert werden soll. Gelötete Teile auf der Platine, welche einen großen Bauraum nach oben benötigen sind vor allem der THT Kondensator mit $1000 \mu\text{F}$ sowie die Wire-to-Board Reihenklemmen. Diese sind bereits auf der linken Seite (Steckerseite) der Platine angeordnet, sodass über der rechten Seite relativ unproblematisch eine zweite Platinenebene eingeführt werden könnte. Da das Gehäuse aufgrund von kostengünstiger und einfacher Fertigung aus Kunststoff geplant und hergestellt wurde, muss eine Lösung gefunden werden um die Platine mit GND zu verbinden, zu kühlen und elektromagnetisch abzuschirmen. Es bietet sich an, eine Aluminiumplatte, die den Maßen der Platine entspricht, zwischen Platine und Gehäuse anzubringen. Dabei wird auf Aluminium zurückgegriffen, da es sich um ein kostengünstiges, leitfähiges Metall handelt, welches außerdem eine hohe Wärmeleitfähigkeit von $204 \frac{\text{W}}{\text{mK}}$ besitzt [wuerth]. Durch das Loch in der Unterseite des Gehäuses, welches in Abbildung 5.2 zu erkennen ist, liegt die Aluminiumplatte, in dieser Darstellung grün eingefärbt, am Aktorgehäuse an. Das hat den Vorteil, dass die Platine Wärme an die Aluminiumplatte über Konduktion abgeben kann, welche wiederum ihre Wärme an das Aktorgehäuse aus Aluminium abgibt. Diese Wärmeübertragung soll durch Wärmeleitpaste beziehungsweise -kleber unterstützt werden. Diese Lösung erspart weitere externe Kühlkörper. Diese Methode hat die Einschränkung, dass nur die aluminiumfreie Seite der Platine bestückt werden darf, ansonsten müssen die betroffenen Bereiche in der Aluminiumplatte ausgespart werden. Da die Platine mit THT-Bauteilen teils auch auf der Unterseite verlötet werden musste, wurden die nötigen Aussparungen durch Bohrlöcher realisiert.

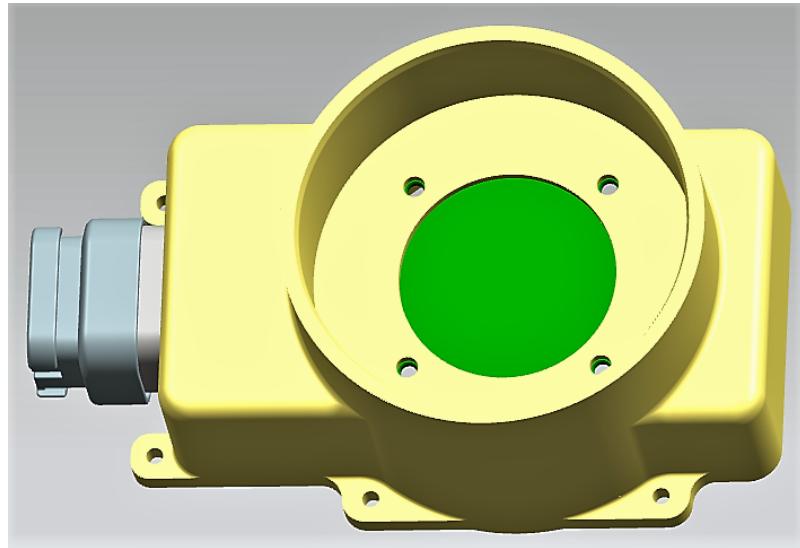


Abbildung 5.2: Rückseite des Elektronikgehäuses zur Darstellung der Aktoranbindung

Die vier Schrauben, mit denen Platine, Aluminiumplatte und Platinengehäuse am Aktorgehäuse verschraubt werden, sorgen zusätzlich für einen optimalen Wärmeübergang von Platine bis zum Aktor. Diese vier Schraubverbindungen plus zwei weitere Schraubverbindungen, welche Platine, Aluminiumplatte und Platinengehäuse verbinden, ermöglichen außerdem eine großflächige GND-Verbindung über die Aluminiumplatte. Die Auflageflächen der Schrauben sind jeweils auf GND-Potential gesetzt, sodass Schrauben und Platte auch dieses Potential annehmen. Eine weiterer wichtiger Aspekt den es bei Platinengehäusen zu beachten gilt, ist die notwendige Abschirmung gegen Eintreten sowie Abstrahlung elektromagnetischer Strahlung. Das Kunststoffgehäuse ist dabei weniger geeignet als die Aluminiumplatte, die aufgrund der elektrischen Leitfähigkeit eine gute Basisschirmung besitzt [bopla]. Für Kunststoffgehäuse wird im Allgemeinen eine Beschichtung aus Aluminium oder Kupfer empfohlen [Gwinner2006].

5.2 Komponentenplatzierung auf der Platine

Beim Layout-Entwurf müssen mehrere Aspekte berücksichtigt werden: Die Platinenmaße und die Platzierung der GND-Schrauben müssen eingehalten werden und die Lokalität der Bauteile sollte gut durchdacht sein. Beispielsweise sollten die Halbbrücken nahe an der Spannungsversorgung platziert werden, um lange leistungsführende Leitungen zu verhindern. Weiterhin müssen die Leitungen der sensiblen Sensorik gegen elektromagnetische Rückwirkung geschützt werden. Beim engültigen Platinenentwurf, welcher in Abbildung 5.3 zu sehen ist, ist die Lokalität der Leistungs-Bauteile, also der Halbbrücken mit den Bezeichnungen HB_1 und HB_2, deutlich von der Platzierung der Logikebene getrennt [emcdes]. Die Halbbrücken befinden sich auf der rechten Hälfte der Platine, nahe des Steckers, wodurch lange Leistungsleitungen vermieden werden können. Der Klemmblock (*terminal block*) als Anschlussklemme für den Tauchspulenaktor ist ebenfalls nahe der Halbbrückenausgänge platziert, sodass auch die Ausgangsleitungen geringe Leitungslängen aufweisen. In Abbildung 5.3 ist dieser Block mit der Bezeichnung TB_1 gekennzeichnet. Der Kondensator zum Stabilisieren der Batteriespannung ist als C13 gekennzeichnet und ebenfalls nahe der Spannungsversorgung platziert. Aus Platzgründen mussten auch die Klemmblöcke für Temperatur- und externe Strommessung (J2 und J3) auf der rechten Platinenhälfte platziert werden. Da sich die leistungsführenden Leitungen jedoch auf die Mitte der Platine konzentrieren stellt dies kein Problem dar. In der linken Platinenhälfte befindet sich der Logikteil, welche den Mikrocontroller (STM32), den CAN-Transceiver (CAN_T), den externen Taktgeber (QUARZ) und den Leitungsverstärker (CD74HCT) umfasst. Neben diesen Bauteilen befindet sich ebenfalls die Spannungsregler für 3.3 V und 5 V, sodass bei der Spannungsversorgung für die ICs keine Probleme durch elektromagnetische Rückwirkungen auftreten und die Versorgungsleitungen zu den Bauteilen kurz gehalten werden kann. Bei der Platzierung der Bauteile spielt ebenfalls die Ausrichtung des Mikrocontrollers eine wichtige Rolle, sodass die benötigten Pins des Mikrocontrollers möglichst geringe Entferungen zum jeweiligen Bauteil aufweisen. Dadurch kann ein Kreuzen der Signalleitungen vermieden und die Länge reduziert werden [emcdes]. So befinden sich beispielsweise die Ausgangspins CAN1_RX und CAN1_TX des Mikrocontrollers in Abbildung 5.3 auf der oberen Seite des STM32 und die Leitungen zum CAN-Transceiver (CAN_T) können deshalb leicht verlegt werden. Die Leitungen für die H-Brücke sind auf der rechten Seite des Mikrocontrollers platziert, sodass sich diese leicht zum Leistungstreiber (CD74HCT) legen lassen. Die Pins für den externen Quarz befinden sich an der unteren Seite des STM32 und somit nahe des Schwingquarzes. Gerade bei diesem Bauteil ist die Entfernung zum Mikrocontroller möglichst gering zu halten und eventuelle Leitungskapazitäten zu verhindern [stmquarz]. Ebenfalls sollte der Taktgeber möglichst

weit von den Halbbrücken entfernt platziert sein um Störungen durch Hochfrequenzsignalen vorzubeugen [stmquarz]. Die Entkopplungskondensatoren der ICs sollten möglichst nahe an den Versorgungseingängen platziert werden, sodass beispielsweise die Entkopplungskondensatoren C10 und C11 nahe des VDD Pins des Mikrocontrollers (STM32) liegen [emcdes].

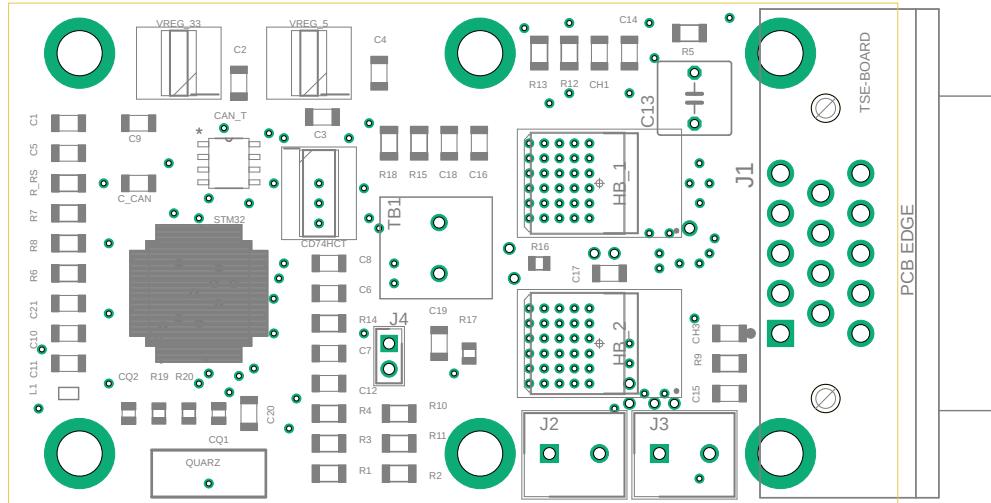


Abbildung 5.3: Platzierungsgrafik der Bauteile

5.3 Dimensionierung der Leiterbahnen

Um die Leiterbahnbreite für die Halbbrücken berechnen zu können, müssen die besonderen Gegebenheiten eines Schaltvorgangs beachtet werden. Es handelt sich um Kurzzeitbelastungen im zeitlichen Rahmen von maximal 100 ms mit einer Stromstärke von maximal 60 A. Aufgrund der Kurzzeitbelastung können die Leitungsquerschnitte deutlich geringer gewählt werden, als die Strombelastbarkeit nach der Norm IPC 2221 es für Leiterbahnen unter Dauerlast zulässt. In einem stromdurchflossenen elektrischen Leiter entsteht Verlustleistung gemäß

$$P = I^2 R$$

da Energie aufgrund des Leitungswiderstandes

$$R = \rho \frac{l}{A} (1 + \alpha_r (T(t) - T_r))$$

dissipiert. Dieser Leitungswiderstand hängt sowohl von den materialabhängigen Konstanten ρ (spezifischer Widerstand) und α_r (Temperaturkoeffizient zur Referenztemperatur T_r), als auch von der Leiterlänge l , dem Leiterquerschnitt A und der Temperaturänderung $T(t)$ zur Referenztemperatur T_r ab. Aufgrund der Erwärmung der Leiterbahn durch die Verlustleistung ist die Temperaturänderung zeitabhängig. Nach dem Stromwärmegegesetz gilt für die Ableitung der Wärmeenergie nach der Zeit

$$\frac{\partial Q_W}{\partial t} = P = I^2 \rho \frac{l}{A} (1 + \alpha_r (T(t) - T_r)). \quad (5.1)$$

Die Änderung der Wärmeenergie führt zu einer Änderung der Temperatur welche sich mittels Wärmekapazität C beschreiben lässt

$$Q_W = (T(t) - T_r) C_W, \quad (5.2)$$

wobei sich die Wärmekapazität aus dem Produkt der materialabhängigen spezifischen Wärmekapazität c und der Masse m des Körpers zusammensetzt

$$C_W = cm. \quad (5.3)$$

Daraus folgt für die zeitliche Ableitung von Gleichung 5.2 die Differentialgleichung

$$\frac{\partial Q_W}{\partial t} = \frac{\partial ((T(t) - T_r) C_W)}{\partial t} = \frac{\partial T(t)}{\partial t} C_W. \quad (5.4)$$

Durch Einsetzen von Gleichung 5.1 in Gleichung 5.4 und der Notationsvereinfachung

$$\xi = I^2 \rho \frac{l}{A} (\alpha_r T_r - 1)$$

ergibt sich die Differentialgleichung erster Ordnung

$$T(t) I^2 \rho \frac{l}{A} \alpha_r - \frac{\partial T(t)}{\partial t} C = \xi. \quad (5.5)$$

Die Differentialgleichung lässt sich mittels Exponentialansatz und der Anfangsbedingung $T(0) = T_r$ lösen, sodass sich eine Funktion der Temperatur in Abhängigkeit der Zeit ergibt

$$T(t) = \left(T_r - \frac{\xi}{I^2 \rho \frac{l}{A} \alpha_r} \right) e^{\frac{I^2 \rho \frac{l}{A} \alpha_r}{C_W} t} + \frac{\xi}{I^2 \rho \frac{l}{A} \alpha_r}$$

woraus durch Einsetzen von ξ , C_W und $m = \varrho l A$ folgt:

$$T(t) = \left(T_r - \frac{\alpha_r T_r - 1}{\alpha_r} \right) e^{\frac{I^2 \rho \alpha_r}{A^2 \varrho c} t} + \frac{\alpha_r T_r - 1}{\alpha_r} \quad (5.6)$$

Für den Anwendungsfall der Platinenauslegung wird jedoch der minimale Leiterquerschnitt bei maximal zulässiger Temperaturänderung benötigt, sodass die Gleichung umgestellt nach dem Leiterquerschnitt A

$$A = I \sqrt{\frac{\rho \alpha_r t}{c \varrho \ln\left(\frac{T(t) - \frac{\alpha_r T_r - 1}{\alpha_r}}{T_r - \frac{\alpha_r T_r - 1}{\alpha_r}}\right)}} \quad (5.7)$$

lautet. Mit Gleichung 5.7 lässt sich für eine bestimmte Belastungsdauer t_i und der maximal zulässigen Endtemperatur der Platine $T(t_i)$ für einen beliebigen Stromfluss I berechnen. Die Herleitung dieser Formel basiert auf der Annahme, dass die Leiterbahn keinen thermischen Austausch mit ihrer Umwelt erfährt und sich somit theoretisch ohne Begrenzung erhitzt kann. Nach dem Energieerhaltungssatz wird die Leiterbahn durch ihren Umgebungsraum und ihre Umgebungstemperatur begrenzt. Unter der Annahme, dass die Umgebungstemperatur durch Nutzung des Aktorgehäuses als Kühlkörper deutlich geringer ist als die maximal zulässige Leiterplattentemperatur ist die Leitungstemperaturabschätzung durch den Wärmeaustausch geringer als in Gleichung 5.6 berechnet. Der Leitungsquerschnitt nach Gleichung 5.7 ist somit überdimensioniert und kann gut als minimale Abschätzung genutzt werden.

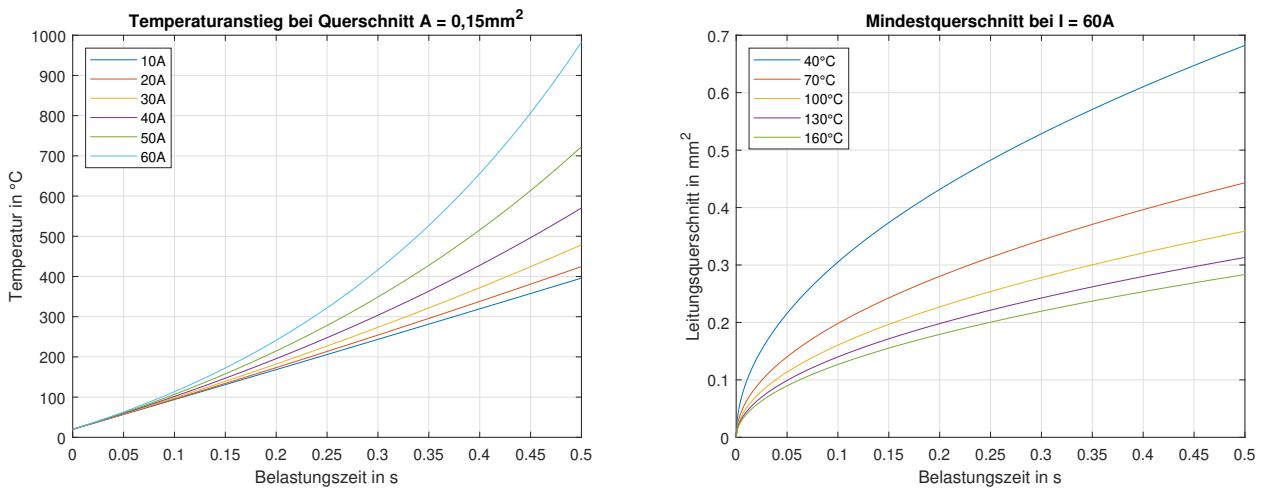


Abbildung 5.4: Dimensionierungsauswirkungen Temperatur und Leitungsquerschnitt

In Abbildung 5.4 ist in der linken Abbildung der Temperaturanstieg über die Belastungsdauer bei verschiedenem Stromfluss und konstantem Querschnitt zu sehen. Es lässt sich erkennen, dass die Temperatur stärker über die Zeit ansteigt, je höher der Stromfluss über die Leiterbahn ist. Weiterhin ist der Steigungsverlauf exponentiell. In der rechten Abbildung ist der minimale Leitungsquerschnitt über die Belastungszeit bei konstantem Strom und verschiedenen zulässigen Maximaltemperaturen zu sehen. Der Verlauf des Leitungsquerschnittsanstiegs ist logarithmisch und bei niedrigeren zulässigen Leitungsmaximaltemperaturen steigt der benötigte minimale Leitungsquerschnitt.

5.3.1 Anwendung auf realer Platine

Die Kenndaten für die minimale Dimensionierung und die Kennwerte des Platinenmaterials sind in Tabelle 5.1 beschrieben. Als Maximaltemperatur wurde dabei der Wert 80 °C gewählt. Der FR4-Kern der Platine ist nach Angaben des Herstellers für eine Temperatur bis zu 135 °C ausgelegt. Die lineare Temperaturabhängigkeit des Kupfers gilt annähernd im Bereich von -50...150 °C [Stiny2015]. Weiterhin ist die Anforderung an die Bauelemente, dass sie einen Temperaturbereich von mindestens -40...105 °C abdecken. Um sich innerhalb dieser Bereiche zu bewegen und bei Temperaturausschlägen einen Sicherheitsfaktor zu berücksichtigen, sind die Leiterbahnen für die Maximaltemperatur von 80 °C auszulegen. Nach Tabelle 5.1 ist für eine Kupferschichtdicke von 70 µm eine Leiterbahnbreite von mindestens 2.61 mm zu wählen. Beim Platinendesign ist diese aus Sicherheitsgründen zu 3 mm gewählt, was einem maximalem Stromfluss von etwa 69.1 A entspricht. Die Dimensionierung betrifft die Spannungsversorgungsleiterbahnen von der Batterie zu den Halbbrücken und die Leiterbahnen von den Halbbrücken zum Aktorklemmblock. Die restlichen Leiterbahnen auf der Platine sind für Dauerlast nach IPC 2221 auszulegen. Im Anhang findet sich ein Tabellenauszug nach IPC 2221 für die Strombelastbarkeit und eine Tabelle der Leistungsdaten der Platinenelemente, nach der die Leiterbahnen dimensioniert sind.

Kenngroße	Zeichen	Wert	Einheit
Temperaturkoeffizient Cu (20°C)	α_{20}	$3,93 \cdot 10^{-3}$	K ⁻¹
Spez. Widerstand Cu	ρ	$1,721 \cdot 10^{-2}$	Ω·mm ² /m
Spez. Wärmekapazität Cu	c	385	J/kg·K
Dichte Cu	ϱ	8,92	g/cm ³
Referenztemperatur	T_{20}	20	°C
Maximalstrom	I_{max}	60	A
Belastungsdauer	t_i	0,1	s
Maximaltemperatur	$T(t_i)$	80	°C
Kupferschicht Dicke	d	$70 \cdot 10^{-6}$	m
resultierender Querschnitt	A	0,1824	mm ²
resultierende Leiterbahnbreite	b	2,6059	mm

Tabelle 5.1: Kenndaten Platine mit Kupferschicht

5.4 Routen der Platine

Bei der letztendlichen Verlegung (*routing*) der Leiterbahnen gibt es mehrere Aspekte zu beachten, die einerseits Einfluss auf die Elektromagnetische Verträglichkeit (EMV), andererseits auch auf die Komplexität des Routingvorgangs haben.

5.4.1 Design-Regeln und EMV

Das Verlegen langer Leitungen sollte nach Möglichkeit vermieden werden um potentielle Störgrößen wie elektromagnetische Induktion und kapazitives Leitungsverhalten zu verringern [haendschke]. Wie bereits in Abschnitt Abschnitt 5.2 erwähnt, sollten deshalb eine gewisse Lokalität der Bauteile eingehalten werden. Um die Leiterbahnen verlegen zu können sollten die Leiterplattenebenen (*Layer*) abwechselnd verschiedene präferierte Verlegerichtungen aufweisen, sodass die Signale leicht an jeden Punkt der Platine verlegt werden können. In Abbildung 5.5 ist schematisch der Aufbau einer zwei Layer Platine dargestellt. Im Falle einer gleichen Verlegerichtung kann es auftreten, dass sich die Signalleitungen auf keinem der vorhandenen Layer kreuzen können und somit unnötig lange Leitungen verlegt werden müssen [haendschke]. Signale können zwischen den Platinen-Layern über Durchkontaktierungen (*vias*) wechseln. Durchkontakteierungen sollten jedoch so selten wie möglich genutzt werden, da diese induktives Leitungsverhalten bewirken. In der Abbildung ist diese metallische Durchkontaktierung zwischen Layer 1 und Layer 2 exemplarisch dargestellt. Eine Signalleitung in Layer 1 kann, wenn sie mit der Metallfläche der Via verbunden ist, in Layer zwei weiter verlegt werden.

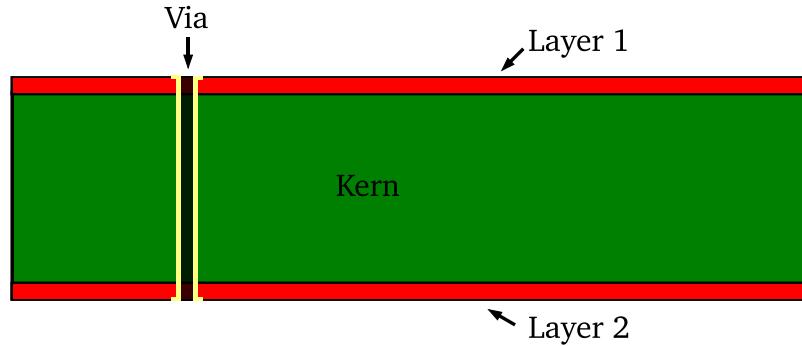


Abbildung 5.5: Darstellung einer zwei Layer Platine inklusive Via

Prinzipiell können Platten mit deutlich mehr Layer-Ebenen entworfen werden, zwei Layer Platten werden jedoch bei industriellen Endprodukten aufgrund ihrer geringen Kosten präferiert [emcdes]. Im endgültigen Platinendesign wurde sich deshalb auch für eine zwei Layer Platine entschieden. Beim Verlegen der Leiterbahnen sollte weiterhin darauf geachtet werden, dass möglichst keine Winkel über 45 Grad auftreten, da diese unerwünschte Reflexionen hervorrufen können [emcdes]. Bei wichtigen Signalleitungen sollte darauf geachtet werden, dass der Abstand zu anderen Leitungen großtmöglich ist, um eine Rückwirkung der Leiter aufeinander zu minimieren. Zur Verbesserung der Schirmung der Signalleitungen gegeneinander wird häufig eine GND-Leitung zwischen den Signalleitungen verlegt, welche die Rückwirkung verringert [Franz2012]. Wenn diese Grundregeln der Leiterbahnplanung bekannt sind, gibt es nach [emcdes] drei Schritte, die zu einer fertigen Platine führen:

- Komponentenauswahl und Platzierung (siehe Kapitel 4 und Abschnitt 5.2)
- Entwurf des Versorgung- und Entkopplungskonzepts
- Verlegen der Signalleitungen

Nachdem in Abschnitt 5.2 die Bauteilplatzierung erläutert wurde, ist in Abbildung 5.6 der erste Entwurf der Logikebene der Platine mit Versorgungskonzept und verschalteten Abblockkondensatoren zu sehen. Die Versorgungsleitungen werden dabei vor den Signalleitungen verlegt, da diese je nach Leistungsführung in ihrer Breite angepasst werden müssen. Die Logikebene beschränkt sich in dem Entwurf auf den STM32 und einen CAN-Transceiver mit Pin-Ausgängen zur Ansteuerung der H-Brücke. Im dritten Schritt werden dann die Signalleitungen verlegt. Die Abbildung ist nur exemplarisch erstellt und wurde nie in Auftrag gegeben. Sie soll lediglich eine Nachvollziehbarkeit der Herangehensweise bewirken.

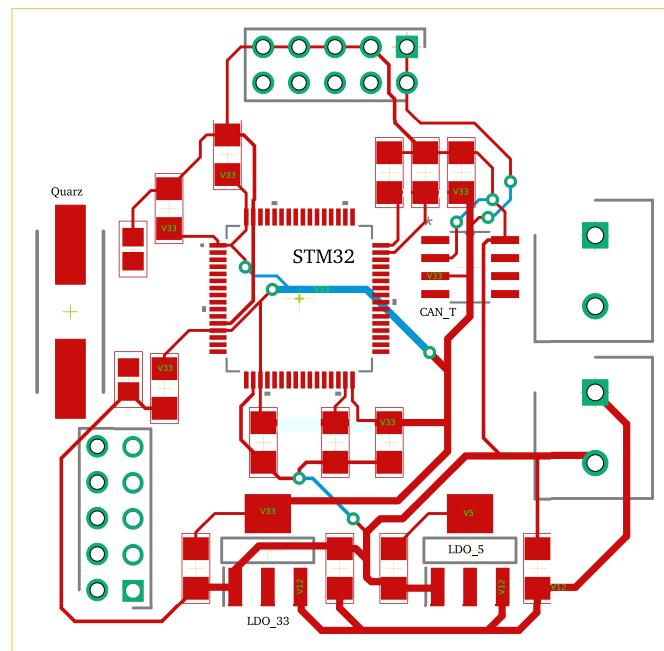


Abbildung 5.6: Logikboard zur Endplatine inklusive Versorgungsleitungen

Üblicherweise werden zur Verbesserung der EMV nach dem Verlegen der Leiterbahnen die freien Platinenflächen zur Abschirmung auf GND Niveau gesetzt, welche als Teilmassenflächen bezeichnet werden. In der einschlägigen Literatur wird darauf hingewiesen, dass diese mit Vorsicht behandelt werden müssen, da Resonanz- und Antenneneffekte auftreten können. Um diesen Effekten vorzubeugen müssen die Masseflächen so gut wie möglich miteinander verbunden werden, sodass diese bei richtigem Anschluss die Schirmungsfunktion übernehmen [Franz2012]. Bei einem Layout mit 4 oder mehr Layern wird empfohlen vollständige GND-Layer und Versorgungslayer einzuführen, sodass eine optimale Abschirmung erfolgen kann [Franz2012]. Ein weiterer Effekt der im Bezug auf GND-Flächen betrachtet werden sollte ist das so genannte GND-Bouncing. Dieser Effekt tritt auf, wenn man keine gute Verbindung zwischen den GND-Flächen hat und somit unterschiedliche GND-Potentiale entstehen. Verstärkt wird dieser Effekt von Switching-ICs, also aktiven Bauteilen mit Schaltcharakteristiken, welche teilweise beim gleichzeitigen Umschalten von 1 nach 0 einen Zeitpunkt aufweisen, zu dem VCC und GND kurzgeschlossen sind und sich somit das GND Potential kurz anhebt [gndbnc]. Aufgrund dieses Effektes sollte die GND-Fläche so gut wie möglich verbunden sein um das GND Potential gleichmäßig zu halten. Problematisch ist dabei auch nicht zwingend wenn es kein geschlossenes GND-Layer gibt, solange die GND-Verbindungen der Ebenen sich ausreichend überlappen und über Durchkontaktierungen gut verbunden sind [Franz2012]. Zur Abschirmung der aktiven Bauelemente gegen Welligkeiten der Spannungsversorgung werden wie in Kapitel 4 erwähnt Abblock kondensatoren verwendet. Dabei sollten möglichst SMD Bauteile genutzt werden um entstehende parasitäre Induktivitäten zu verringern [emcdes]. Als Abschirmung gegen von außen auftretende elektromagnetische Felder sei die Schirmwirkung von Kühlkörpern oder Gehäuseteilen aus Metall hervorgehoben [Franz2012].

5.4.2 Leiterbahnverlegung des Entwicklungsboards

Die Verlegung der Leiterbahnen auf dem endgültigen Entwicklungsboard geschieht nach den zuvor definierten Designregeln, wobei aufgrund des geringen Platzes häufig Kompromisse eingegangen werden müssen. Das Entwicklungsboard weist 2 Layer auf, welche der Anordnung aus Abbildung 5.5 entsprechen und im Folgenden über die Bezeichnung Top-Layer (obere Leiterebene) und Bottom-Layer (untere Leiterebene) gekennzeichnet werden. Grundanforderungen an die Platine ist die Positionierung der GND-Löcher, durch welche die Platine mittels Schrauben über das Aktorgehäuse mit GND verbunden ist. In Abbildung 5.7 und Abbildung 5.8 ist das fertige Top- und Bottom-Layer der Platine zu sehen. Problematisch für das Einhalten der Designregeln ist hauptsächlich die Platzierung der Halbbrücken und der mittleren Schraubenlöcher, welche den Platz für die Signalleitungen von dem Mikrocontroller in Richtung Platinenstecker stark begrenzen. Die präferierte Verlegerichtung auf dem Top-Layer ist vertikal gewählt um die jeweiligen ICs über kurze Wege ohne große Leitungsverluste erreichen zu können. Dabei liegen CAN-Tansciever und Leitungsverstärker oberhalb und der externe Schwingquarz unterhalb des Mikrocontrollers, was optimal für die präferierte Leitungsrichtung ist. Problematisch ist die Leitungsverlegung in Richtung der Halbbrücken und des Steckers, da der Spalt zwischen Schraubloch und Halbbrücke in der oberen Platinenhälfte durch die Versorgungsspannung belegt ist.

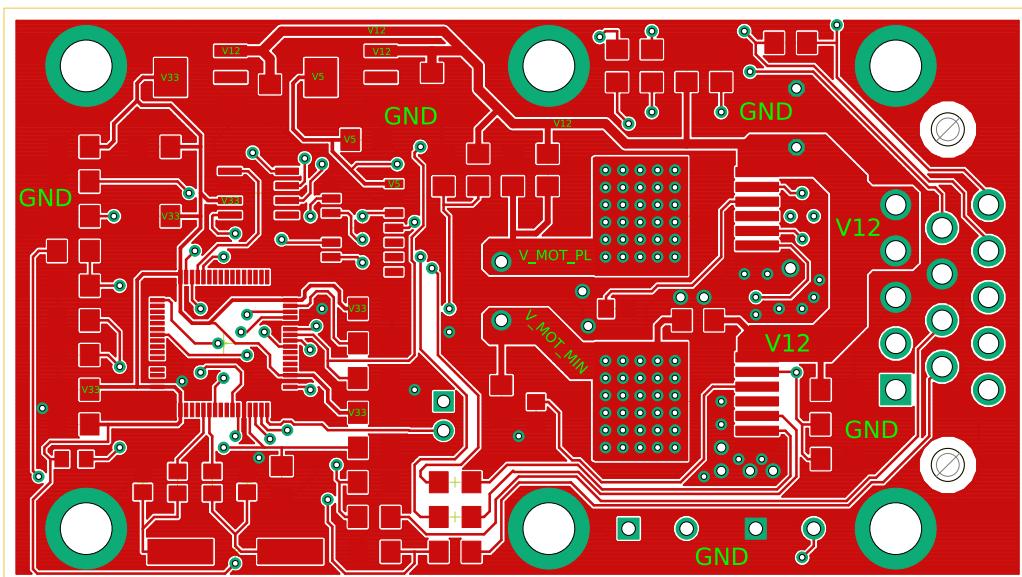


Abbildung 5.7: Darstellung des Top-Layers

Die präferierte Verlegerichtung des Bottom-Layers ist horizontal ausgerichtet, sodass diese Leiterebene hauptsächlich zum Erreichen der Halbbrücken und des Steckers genutzt wird. Bei Betrachtung von Abbildung 5.8 fällt jedoch auf, dass der Platz für die Leitungen sehr begrenzt ist und somit ein Teil der horizontalen Leiterbahnen auf das Top-Layer verlegt werden musste. Eine weitere wichtige Aufgabe beim Verlegen der Leiterbahnen ist die Reduktion elektromagnetischer Rückwirkung, wobei sich an die Methoden aus Unterabschnitt 5.4.1 gehalten wird. Potentielle Gefahrenquelle der Rückstrahlung sind die Halbbrücken, da diese beim Schalten mit PWM-Signal Stromimpulse in hohen Frequenzen durchschalten können. Wichtige Signalleitungen wie die der Sensorik sind also möglichst weit von den Flächen mit der Bezeichnung V_MOT_PL und V_MOT_MIN (Halbbrücken Ausgänge) zu verlegen. Aufgrund des Platzmangels lassen sich die Signalleitungen der Sensorik nicht vollständig aus dem Einflussgebiet der Halbbrücken entfernen, sind jedoch von diesen so weit wie möglich entfernt verlegt. Bei genauerer Betrachtung der Layer fällt auf, dass jedoch trotzdem Signalleitungen nahe des Einflussgebietes der Halbbrücken verlaufen. Diese Leiterbahnen sind hauptsächlich die Leiterbahnen der SWD-Verbindung, also der Programmierschnittstelle der Platine. Daher stellt dies kein Problem dar, da die Platine nicht im laufenden Aktorbetrieb programmiert wird und somit die Lokalität im Einflussgebiet der Halbbrücken kein Hindernis darstellt. Die Leitungen zur Ansteuerung der Halbbrücken HB1_IN und HB2_IN sollten ebenfalls nicht im direkten Einflussgebiet liegen, damit die Halbbrücken vom Mikrocontroller noch problemlos angesteuert werden können. Als Konzept zur Abschirmung elektromagnetischer Rückwirkung wurde bereits der Einsatz von Abblockkondensatoren erwähnt, der eine wesentliche Rolle spielt. Neben diesen sind die freien Boardflächen auf beiden Layern auf GND-Potential gesetzt um damit ebenfalls Schirmungseffekte erzielen zu können. Diese sind zwar auf beiden Layern nicht optimal zusammenhängend, es wurde aber darauf geachtet über Durchkontaktierungen eine stabile gemeinsame GND-Fläche zu erhalten, welche gegen GND-Bouncing geschützt ist. Die GND-Flächen werden ebenfalls über die Aluminiumplatte und den Aktor auf ein gleichmäßiges Potential gezogen. Neben den EMV-Maßnahmen sollte auch die Wärmeentwicklung der Bauteile berücksichtigt werden. Aufgrund des hohen Stromflusses stellen die Halbbrücken besondere Anforderungen an die Kühlung. Insbesondere in Transistorschaltvorgängen innerhalb der Halbbrücke während denen ein Teil der Versorgungsspannung an den Transistoren abfällt und somit nach $P = U \cdot I$ erhöhte Wärmeenergie entsteht. Um die Halbbrücken unter ihrer kritischen Temperatur halten zu können, muss die entstehende Wärme abgeführt werden. Dies kann beispielsweise durch Durchkontaktierungen (*Thermische Vias*) unter den kritischen Bauteilen erreicht werden. Über die Thermischen Vias vergrößert sich die zusammenhängende Metallfläche auf der sich die Wärme ausbreiten kann in das Bottom-Layer, und kann somit gut über den anliegenden Aluminium-Körper abgeführt werden.

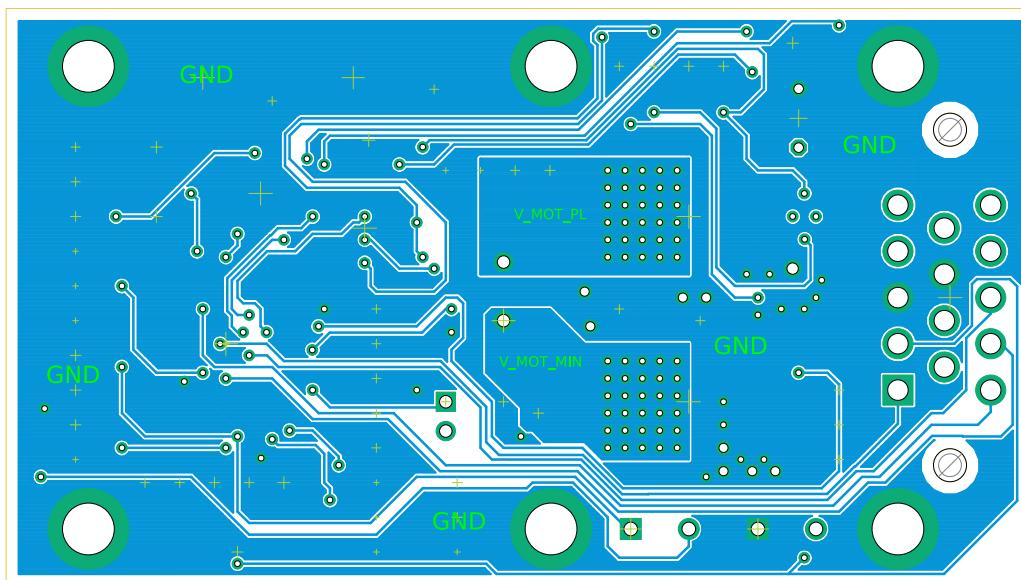


Abbildung 5.8: Darstellung des Bottom-Layers

5.5 Fertigung

In Abbildung 5.9 und Abbildung 5.10 ist die Platine nach dem Fertigungsprozess zu sehen. Als Kenndaten für die Fertigung wurde eine Leiterbahndicke von $70\ \mu\text{m}$ festgelegt. Als Kern wird ein FR4 Verbundwerkstoff mit einer Dicke von $1.5\ \text{mm}$ verwendet. Gut zu sehen sind auf den fertigen Leiterplatten auch die thermischen Vias, welche in das Lötpad von HB_1 und HB_2 eingelassen sind.

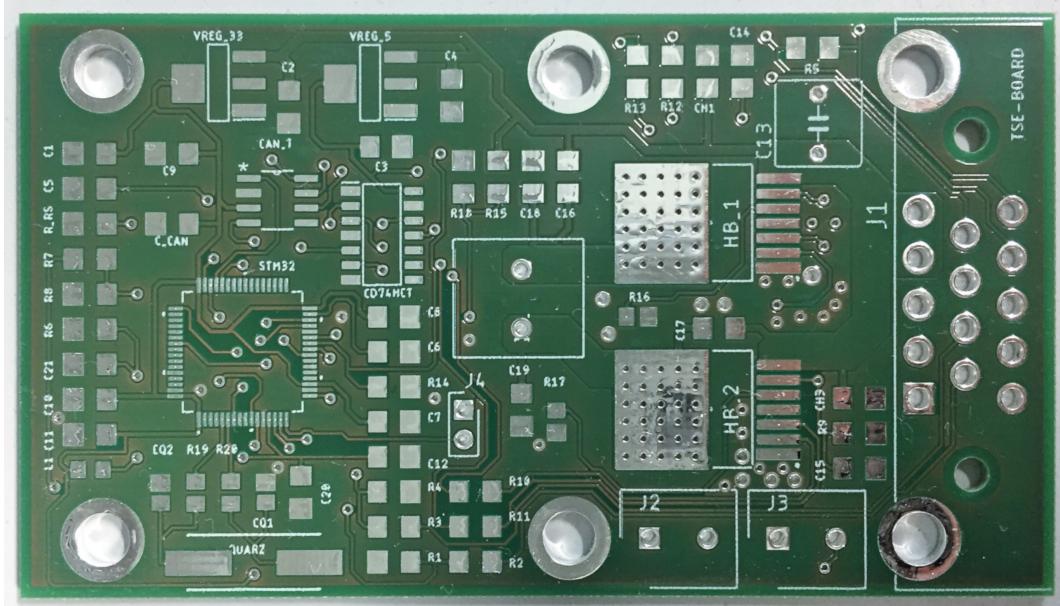


Abbildung 5.9: Top-Layer nach dem Fertigungsprozess

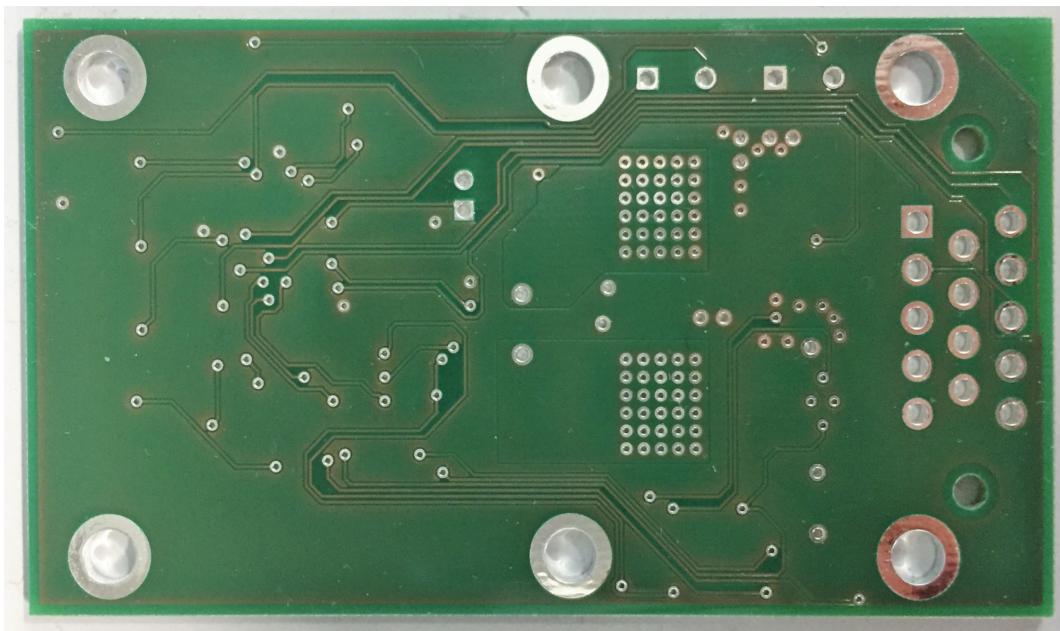


Abbildung 5.10: Bottom-Layer nach dem Fertigungsprozess



6 Software

Dieses Kapitel behandelt das zum Einsatz kommende Softwaresystem in Hinblick auf die verwendete Toolchain, die Programmstruktur als solche und Optimierungspotentialen. Dadurch soll dem Leser ein Überblick geschaffen werden, sodass dieser den Programmablauf nachvollziehen kann und selbstständig Weiterentwicklungen durchführen kann.

6.1 Toolchain

Unter Toolchain werden die Werkzeuge bzw. Programme und ihr Zusammenwirken bezeichnet, die zur Programmierung verwendet werden.

Für die vorliegende Aufgabenstellung ist ein Programm notwendig, das auf einem Mikrocontroller des Typs STM32F4 lauffähig ist. Zur Entwicklung des Programms kommt ein innovativer Ansatz zum Einsatz, der einfach verständlich und schnell erlernbar ist. Er basiert auf *Matlab Simulink* und ermöglicht die Programmierung des Mikrocontrollers auf Blockbasis. Um hardwarespezifische Blöcke bspw. zum Auslesen der ADCs zur Verfügung zu haben, wird ein Simulink Blockset namens *Wajung* eingesetzt **??**. Um das Blockschaltbild für den STM32 ausführbar zu machen, wird neben Simulink selbst auch der *Simulink Coder* und der *Embedded Coder* für Matlab benötigt. Durch diese Toolboxes ist es möglich aus einem Simulink Modell C-Code zu generieren **??**. Der *Embedded Coder* führt dabei zusätzlich Optimierungen durch, die eine effiziente Ausführung auf einem eingebettetem System ermöglichen **??**. Auch dieser C-Code ist nicht direkt ausführbar auf einem ARM-basierten Prozessor, wie es der STM32F4 ist. Es existieren jedoch geeignete Übersetzungstools (Compiler), die den C-Code in ARM-Assembler übersetzen und direkt lauffähig machen können. Im *Wajung Blockset* werden dafür die *GNU Tools for ARM embedded processors* mitgeliefert. Der kompilierte Code muss im letzten Schritt auf den Mikrocontroller übertragen werden, auch *flashen* oder *programmieren* genannt. Die Firma ST liefert dafür ein Programm namens *STLink Utility*, welches neben des Programmierzorgangs über die definierte Programmierschnittstelle SWD auch Debugging-Fähigkeiten liefert, womit Fehlerzustände lokalisiert und beseitigt werden können. Einen groben Überblick über den Prozess vom Simulink Modell bis zur Ausführung auf der Zielhardware wird in Abbildung 6.1 gegeben.

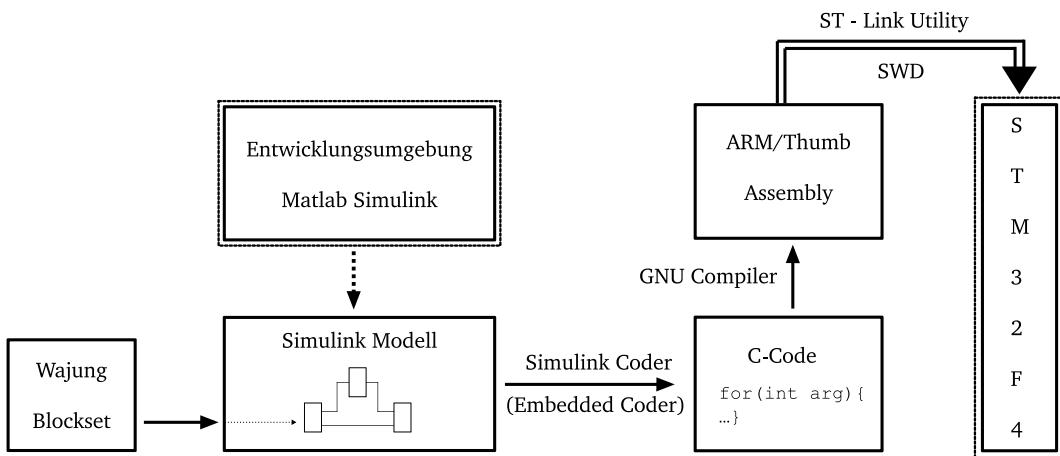


Abbildung 6.1: Vereinfachte Darstellung der eingesetzten Toolchain

6.2 Vorstellung des Hauptprogramms

Um einen Einblick in das Hauptprogramm zu bieten, das auf dem finalen *smart-actuator* zum Einsatz kommt, wird zunächst geklärt welche Systemarchitektur zum Einsatz kommt. Insbesondere im Gebiet des *automated driving* sind viele Architekturansätze bekannt, durch welche komplexe Software strukturierter, robuster und zuverlässiger werden soll **[automated]**. Weit verbreitet ist die 1983 entwickelte Architektur von Rasmussen, die zwischen wissensbasiertem, regelbasiertem und reflexbasiertem Verhalten unterscheidet. Unter reflexbasiertem Verhalten werden Aktionen verstanden, die rein reaktiv Sensordaten in Aktor-Aktivitäten umsetzen, ohne dabei komplexen Regeln zu folgen. Regelbasiertes Verhalten auf der anderen Seite ist bestimmt durch feste vordefinierte Regeln, die auf bestimmte Situationen angewendet

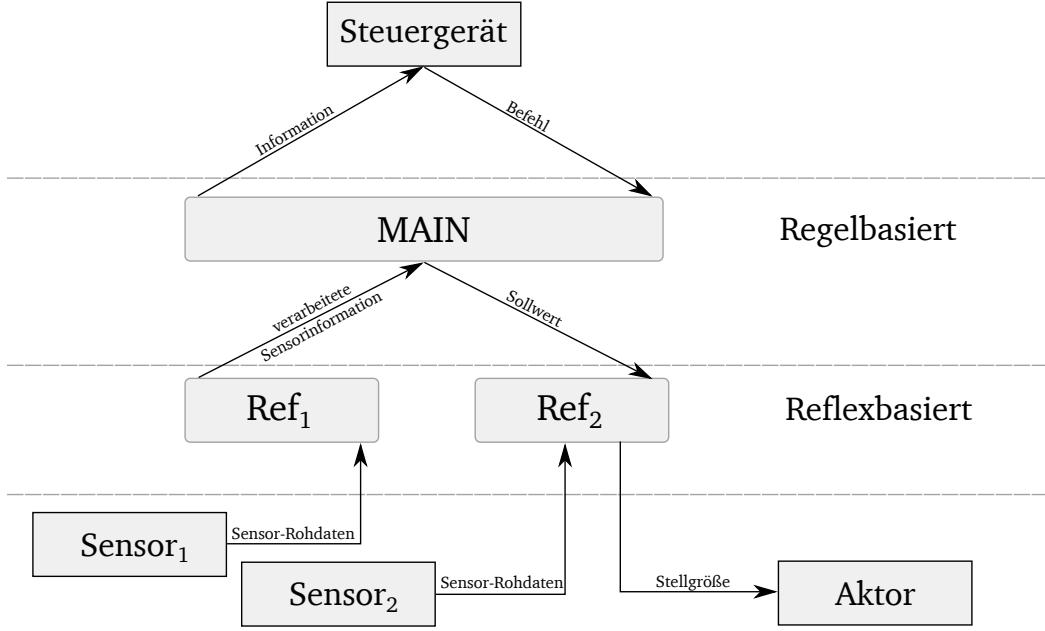


Abbildung 6.2: Drei-Ebenenarchitektur nach Rasmussen (vereinfacht)

Abkürzung	Erläuterung
POS	Schaltgabelpositionsermittlung
CAN	CAN - Schnittstelle
CAL	Kalibrierung des Lagesensors
ERR	Fehlererkennung
MAIN	Hauptzustandsautomat
SEN	Sensorik
CONS	Regler zum Schalten
CONH	Regler zum Positionshalten
PWM	Auswahlglied für PWM-Signal
MOT	Motortreiber

Tabelle 6.1: Erläuterung der Abkürzungen im Softwaresystem

werden. Falls eine Situation jedoch nicht vorhersehbar ist bzw. keine Regeln dafür hinterlegt sind, muss durch wissensbasiertes Verhalten gehandelt werden [automated]. Dieses Verhalten kann beispielsweise durch neuronale Netze erreicht werden, die jedoch in dieser Arbeit keine Anwendung finden.

Wichtig sind daher die unteren beiden Ebenen, wobei nur die reflexbasierte Ebene Zugriff auf Sensorik und Aktorik hat. Auf übergeordneter regelbasierten Ebene werden dann vorverarbeitete Werte ausgewertet und anhand fester Regeln Handlungsanweisungen an die untere Ebene erzeugt. In Abbildung 6.2 ist die Architektur nochmal vereinfacht dargestellt. Eine regelbasierte Hauptkomponente (MAIN) entscheidet je nach anliegenden vorverarbeiteten Sensorinformationen und Steuerbefehlen des übergeordneten Steuergeräts, welches Verhalten auf Reflexebene erwünscht ist. Es wird beispielsweise eine Sollgröße für einen Regler (Ref_2) vorgegeben, die schnellstmöglich ausgeregelt wird.

Konkret wird die Hauptkomponente (MAIN) durch einen Zustandsautomaten realisiert, der mehrere Ein- und Ausgänge zu anderen Komponenten bietet. Die Außendarstellung ist in Abbildung 6.4 dargestellt. Dabei sind links Eingänge von Sensorik- und CAN-Komponenten angeordnet und rechts Ausgänge. Die erste Buchstabenfolge beschreibt dabei jeweils von welchem Block das Signal ursprünglich generiert wird. Die Abkürzungen sind in Tabelle 6.1 erläutert.

6.2.1 Gesamtsystem

In Abbildung 6.3 ist das gesamte Softwaresystem dargestellt. Hierbei ist die Sensorik hellblau, die Regelung hellgrün, die Aktorik dunkelgrün, die CAN-Kommunikation gelb und die Fehlererkennung orange dargestellt. Alle Komponenten agieren reflexbasiert direkt auf die Steuerungssignale des mittig angeordneten Hauptzustandsautomaten. Eine Ausnahme bildet die Kalibrierung, die ihrerseits durch einen Zustandsautomaten realisiert ist, wobei dieser durch den Hauptzustandsautomaten aktiviert und gestoppt wird. Im Folgenden wird der Hauptzustandsautomat und die reflexbasierten

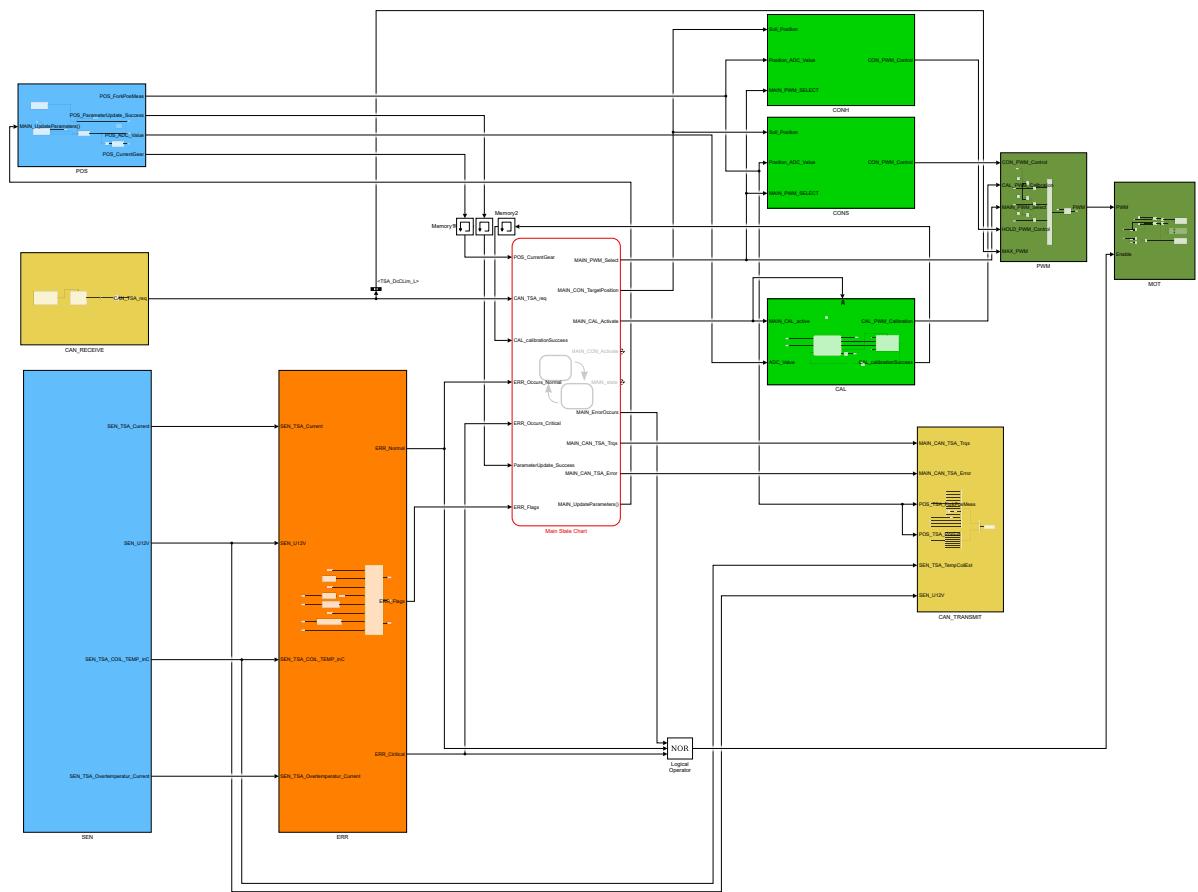


Abbildung 6.3: Darstellung des Gesamtsystems

Komponenten einzeln erläutert.

6.2.2 Hauptzustandsautomat (MAIN)

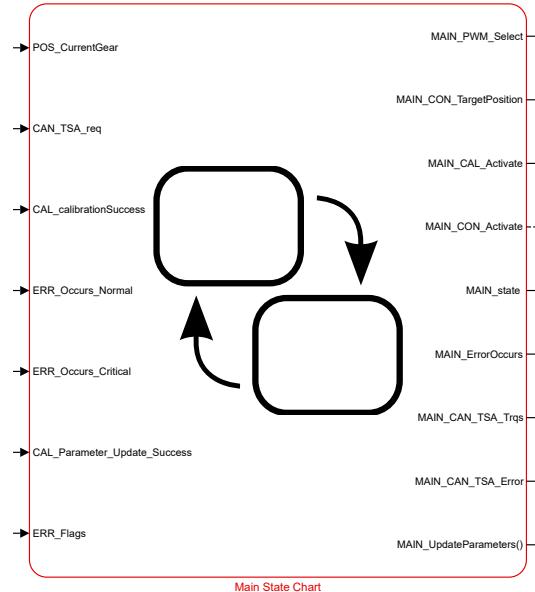


Abbildung 6.4: Hauptzustandsautomat (MAIN), Außenansicht

Der Hauptzustandsautomat ist die Kernkomponente der Software und beeinflusst jede Komponente des Systems. Als Eingangssignal wird dem Hauptzustandsautomaten

- der aktuell eingelegte Gang (*POS_CurrentGear*),
- die letzte empfangene CAN-Nachricht (*CAN_TSA_req*),
- das Statussignal einer laufenden/abgeschlossenen Kalibrierung (*CAL_calibrationSuccess*),
- das Fehlersignal bei *normalen Fehlern* (*ERR_Occurs_Normal*),
- das Fehlersignal bei *kritischen Fehlern* (*ERR_Occurs_Critical*),
- das Statussignal beim Laden der Kalibrierparameter (*ParameterUpdate_Success*) und
- eine Liste an Fehler Bits (*ERR_Flags*)

zugeführt. Als Ausgangssignal gibt der Hauptzustandsautomat

- die aktuelle Quelle für PWM Generierung (*MAIN_PWM_Select*),
- den Wunschgang,
- das Aktivierungssignal der Kalibrierung,
- den aktuellen Zustand des Automaten,
- ein Fehlersignal bei internem Fehler im Automaten,
- die *Trqs*-CAN-Nachricht (siehe ??),
- die *Error*-CAN-Nachricht (siehe ??) und
- den Auslöser zum Auslesen der Kalibrierparameter

aus. Über *MAIN_PWM_Select* wird nicht nur die Quelle ausgewählt, die den Motortreiber ansteuert, sondern ebenfalls welche Teilsysteme aktiv sind. Somit wird sichergestellt, dass zu jedem Zeitpunkt ausschließlich ein einziger Regelalgorithmus oder Kalibrieralgorithmus ausgeführt wird (siehe dazu auch Abschnitt 6.3).

aus. Über *MAIN_PWM_Select* wird nicht nur die Quelle ausgewählt, die den Motortreiber ansteuert, sondern ebenfalls welche Teilsysteme aktiv sind. Somit wird sichergestellt, dass zu jedem Zeitpunkt ausschließlich ein einziger Regelalgorithmus oder Kalibrieralgorithmus ausgeführt wird (siehe dazu auch Abschnitt 6.3).

Intern agiert der Zustandsautomat in vier streng separierten Betriebsmodi. Nach jedem Reset (bspw. nach dem Anlegen der Versorgungsspannung) befindet sich der Automat zunächst im Initialisierungsschritt. Hier werden zunächst für die Ausführung relevante Variablen initialisiert. Daran anschließend werden die Kalibrierparameter aus dem nichtflüchtigen Speicher geladen. Das Auslesen und Zwischenspeichern der Werte findet dabei in *POS* statt und wird durch das Setzen eines entsprechenden Triggersignals angestoßen. Durch die Kalibrierwerte ist es nun möglich die aktuelle Schaltgabelposition zu bestimmen, wobei zunächst im Betriebsmodus **Inaktiv** verblieben wird. Durch einen entsprechenden CAN-Befehl kann der Übergang in den Betriebsmodus **Regulär** erfolgen. Hierin ist es möglich Schaltbefehle auszuführen oder eine **Kalibrierung** zu beginnen. Es wird unterschieden zwischen *Schalte in Gang X* und *Bleibe in Gang X*, wobei jeweils entweder der Regler zum Schalten oder der Regler zur Positionserhaltung aktiv ist. Für Letzteren existiert noch keine zufriedenstellende Realisierung. Aus allen Betriebsmodi wird durch das Auftreten eines Fehlers (angezeigt durch die entsprechenden Eingangssignale am Automaten) direkt in den Betriebsmodus **Fehler** übergegangen. In diesem Betriebsmodus werden keine Stellgrößen am Motortreiber zugelassen und alle Regler sind deaktiviert. Erst durch das Senden des entsprechenden CAN-Befehls wird wieder in den Betriebsmodus **Inaktiv** geschaltet.

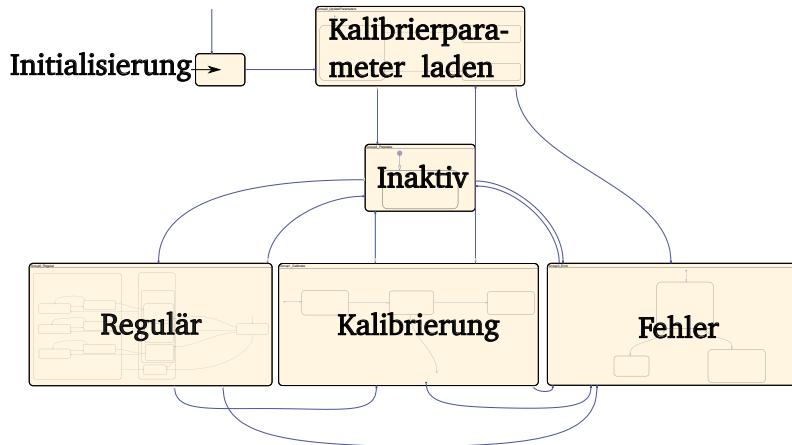


Abbildung 6.5: Hauptzustandsautomat (MAIN), vereinfachter Aufbau

6.2.3 Positionsermittlung (POS)

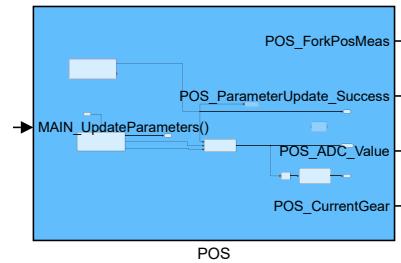


Abbildung 6.6: Schaltgabelpositionsermittlung

Die Positionsermittlung der Schaltgabel genießt eine Sonderstellung unter der Sensorik. Sie stellt die Regelgröße dar und ist daher von essentieller Bedeutung für die Regelung. Dementsprechend wird die Positionsregelung getrennt von der restlichen Sensorik behandelt. Die Sensorik ist im SEN-Subsystem anzutreffen (siehe Unterabschnitt 6.2.5). Eingangsseitig kann durch den Hauptzustandsautomaten angestoßen werden, dass die Kalibrierdaten für die Lagesensorik aus dem nichtflüchtigen Speicher gelesen werden. Ausgangsseitig wird

- die momentane Gabelposition in Millimetern,
- der Zustand des Lesezugriffs auf die Kalibrierdaten,
- die Sensor Rohdaten und
- der aktuell eingelegte Gang

bereit gestellt.

6.2.4 CAN-Kommunikation (CAN)

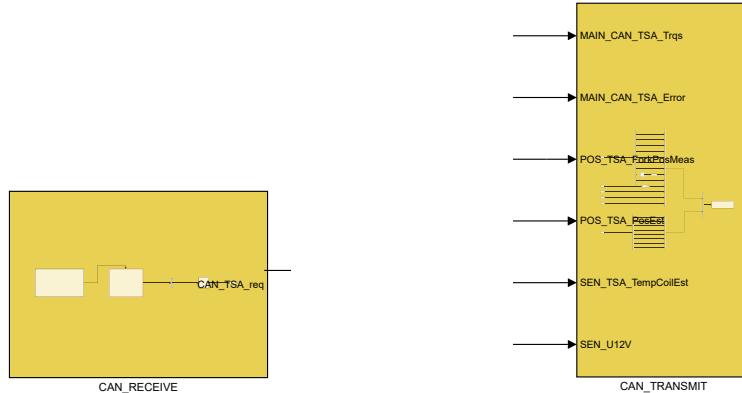


Abbildung 6.7: CAN-Kommunikation

Die CAN-Kommunikation wird durch eine Empfangskomponente (*receive*) und eine Sendekomponente (*transmit*) realisiert. Die Empfangskomponente funktioniert Interrupt-basiert und wird somit immer aktiv, wenn eine neue Nachricht empfangen wird. Über einen Rate-Transmission-Block wird diese Empfangsfrequenz an die Arbeitsfrequenz des Hauptzustandsautomaten angepasst. Als Ausgang steht dem Automaten ein Bus-Vektor mit dem Nachrichteninhalt zur Verfügung. Die Sendekomponente basiert auf *Non-Blocking*-Basis. Dadurch können während des Sendeorgangs nebenläufig auch noch andere Aufgaben abgearbeitet werden. Die einzelnen Eingangswerte werden vor dem Senden zu größeren Nachrichten zusammengefügt.

6.2.5 Sensorik (SEN)

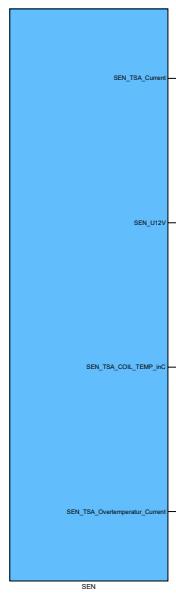


Abbildung 6.8: Sensorik

Die Sensorik umfasst eine Strommessung durch den Aktor, eine Spannungsmessung am Eingang, eine Spulentemperaturmessung und eine Detektion gegen Übertemperatur/Überstrom an den Halbbrücken. Die Sensorwerte stehen zunächst als ADC-Rohwerte für die weitere Verarbeitung bereit. Diese werden je nach Sensortyp skaliert oder wie im Falle des externen Thermistors durch komplexe Formeln in den realen Messwert konvertiert (vgl. Kapitel 4). An den Ausgängen stehen anschließend die Messgrößen in gewünschter Einheit bereit. Der Lagesensor wird nach Unterabschnitt 6.2.3 implementiert.

6.2.6 Fehlererkennung (ERR)

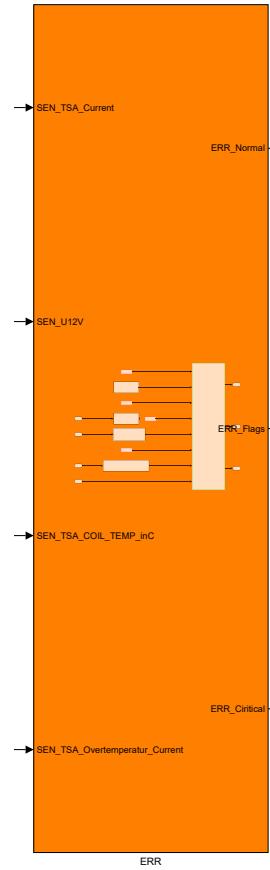


Abbildung 6.9: Fehlererkennung

Durch die Fehlererkennung werden die gemessenen Überwachungsgrößen mit Schwellwerten verglichen und ggf. ein Fehlersignal zum Hauptzustandsautomaten weitergeleitet. Dabei muss berücksichtigt werden, dass Messfehler erkannt und ggf. gefiltert werden und nicht zu falsch positiv Ergebnissen führen. Auf der anderen Seite sollten Fehler schnellstmöglich erkannt werden und die falsch negativ - Rate minimal sein. Neben dem Hauptzustandsautomaten besitzt die Komponente auch noch eine Schnittstelle zum Motortreiber, um den Stromfluss im Aktor schnellstmöglich zu unterbinden. Hierfür wird das in Unterabschnitt 6.2.10 genauer erklärte *enable*-Signale genutzt.

6.2.7 Kalibrierung (CAL)

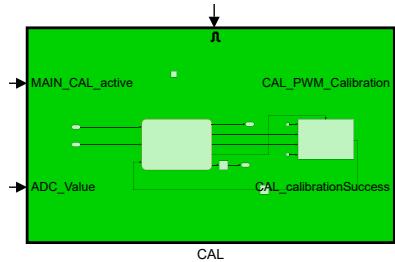


Abbildung 6.10: Kalibrierung (Lagesensor)

Die Kalibrierung dient dazu, den Spannungswerten des Lagesensors konkrete Positionen der Schaltgabel zuordnen zu können. Das Kalibrierungsverfahren basiert auf dem Ansatz von [VorgaengerADP]. Es werden dabei die Anschlagspositionen durch große Stellgrößen angefahren und der Sensor an diesen Werten ausgelesen. Nach mehrmaliger Ausführung

und anschließender Mittlung werden die Kalibrierungswerte nichtflüchtig abgespeichert. Im Falle eines Spannungsverlustes stehen die Kalibrierungsdaten nach wie vor bereit. Die Komponente besitzt Schnittstellen zur Lagesensorik sowie zum Hauptzustandsautomaten, von welchem aus die Kalibrierung eingeleitet wird.

6.2.8 Regelung (CONS/CONH)

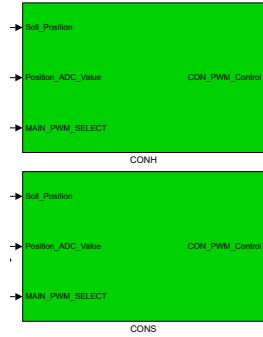


Abbildung 6.11: Regelung

Für die Regelung der Schaltgabelposition nach Unterabschnitt 2.1.6 werden den Reglern die momentane Gabelposition zugeführt. Darüber hinaus wird über den Hauptzustandsautomaten die Sollposition vorgegeben. Über den Soll-Ist-Vergleich und die Anwendung eines Regelgesetzes wird schließlich eine PWM als Stellgröße ermittelt. Eine Regelfrequenz von 10 kHz hat sich als ausreichend schnell herausgestellt und kann durch den Mikrocontroller geleistet werden. Beim *Schaltregler* kommt ein PID-Regler mit Störgrößenkompensation zum Einsatz, der in [VorgaengerADP] erarbeitet wurde. Eine praktikable Lösung für einen *Halteregler* konnte nicht ermittelt werden. Der jeweilige Regler ist nur aktiv, falls er durch den *MAIN_PWM_SELECT* des Hauptzustandsautomaten auch explizit angefragt ist.

6.2.9 Auswahlglied für PWM-Signal (PWM)

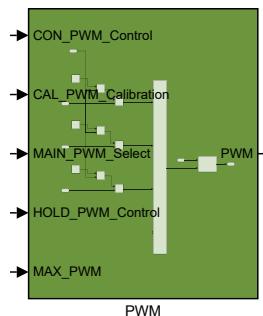


Abbildung 6.12: Auswahlglied für PWM-Signal

Über das Auswahlglied für das PWM-Signal wird selektiert, welche der möglichen PWM-Signale zum Motortreiber durchgeschaltet wird. Die Auswahl wird aufgrund des *MAIN_PWM_SELECT*-Eingangs getroffen. Darüber hinaus kann die maximale PWM durch entsprechende CAN-Signale begrenzt werden.

6.2.10 Motortreiber (MOT)

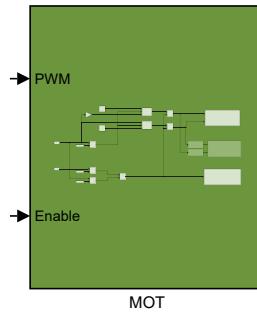


Abbildung 6.13: Motortreiber

Der Motortreiber erwartet ein vorgegebenes PWM-Signal, bzw. dessen Puls/Pause-Verhältnis. Über das Vorzeichen wird bestimmt, welche der beiden H-Brücken gepulst angesteuert wird und welche konstant auf Erdpotential gesetzt wird. Über ein explizites *nable*-Signal kann der Motortreiber bspw. im Falle einer Notausschaltung deaktiviert werden. Hierfür wird der Motortreiber über einen separaten Pin in einen *sleep-mode* gesetzt (vgl. Abschnitt 4.4). Zwei Vorgabewerte werden demnach in Signale umgewandelt, die zur H-Brücke weitergeführt werden.

6.3 Codegenerierung und Performanceoptimierung

Der generierte und kompilierte Code wird auf dem Mikroprozessor Cortex M4 ausgeführt, der auf dem STM32F4 eingesetzt wird. Dieser Mikroprozessor zeichnet sich durch eine vergleichsweise hohe Rechenleistung aus und verfügt darüber hinaus über eine *Floating Point Unit* (FPU). Somit ist er in der Lage Gleitkommazahlen zu verrechnen. Bei maximaler Taktfrequenz ist die Rechenleistung begrenzt auf 225 DMIPS. Der Prozessor kann daher nicht beliebig viele Aufgaben in beliebiger Arbeitsfrequenz abarbeiten. Für weiterführende Informationen sei auf das Datenblatt [stm32] verwiesen. Um zu verstehen, wie die begrenzten Rechenressourcen optimal ausgelastet werden können, ist es von Vorteil den Codegenerierungsprozess zu kennen. Über Wajung und die Simulink Toolboxes wird Code nach dem sogenannten *Multirate Single Tasking*- Prinzip generiert. Somit ist es möglich, mehrere Aufgaben mit unterschiedlichen Arbeitsfrequenzen abzuarbeiten. Über den sogenannten *SysTick Timer* wird in der Frequenz der höchstenfrequenten Aufgabe ein Interrupt ausgelöst. Ein Interrupt bezeichnet eine Unterbrechung des regulären Programmablaufs, wobei nach der Abarbeitung der sogenannten Interrupt Service Routine wieder der reguläre Programmablauf an der letzten Stelle fortgeführt wird. Innerhalb des *SysTick Timer*-Interrupts wird angestoßen, dass diejenigen Aufgaben ausgeführt werden, die zum entsprechenden Zeitpunkt ausgeführt werden sollen. Beispielsweise wird eine Aufgabe, die mit 100Hz ausgeführt werden soll, durch jeden zehnten *SysTick Timer*-Interrupt mit 1 kHz angestoßen. Eine Besonderheit fallen Aufgaben zu, die als laufzeitkritisch eingestuft werden, denn diese Aufgaben werden direkt innerhalb der Interrupt Service Routine abgearbeitet und haben somit Priorität vor den übrigen Aufgaben. Ein Beispiel einer solchen Aufgabe ist der geschlossene Regelkreis zur Reglung der Gabelposition. Um den Ablauf des Programms nicht zu gefährden, muss darauf geachtet werden, dass diese Aufgaben nicht länger dauern als bis zum Auslösen des nächsten *SysTick Timer*-Interrupts. Zusätzlich muss noch genügend Zeit bleiben, um die übrigen Aufgaben auszuführen. [wajung]

Um dies zu ermöglichen wird eine Reihe von Maßnahmen ergriffen, die im Folgenden kurz vorgestellt werden. Zunächst sind ausschließlich Teile des Systems aktiv, die zum aktuellen Zeitpunkt auch benötigt werden. Alle Funktionalitäten zur Kalibrierung, Halteregelung und Speicherung nichtflüchtiger Daten sind beispielsweise während eines Schaltvorgangs deaktiviert. So wird verhindert, dass wartende inaktive Prozesse Rechenzeit beanspruchen. Als weitere Maßnahme wird die Arbeitsfrequenz für verschiedene streng voneinander geteilte Bereiche evaluiert, sodass jeder Bereich nur so oft wie nötig aufgerufen wird. Auch durch die Wahl entsprechender Variablentypen kann die Gesamtperformance optimiert werden. Viele Mikrocontroller bieten keine FPU und können demnach hardwarebeschleunigt keine Gleitkommaoperationen durchführen (vgl. STM32F0, STM32F1). Daher werden in solchen Fällen statt Gleitkommazahlen Festkommazahlen oder eine Skalierung auf Ganzzahlen verwendet. Da der STM32F4 jedoch über eine FPU verfügt, ist der Leistungsgewinn in dieser Hinsicht begrenzt. Wichtig ist jedoch zu berücksichtigen, dass der STM32 einen 32-bit Prozessor besitzt und die Arithmetische Logik Einheit (ALU) nur Zahlen im 32-bit Bereich direkt verrechnen kann. Auf 64-bit Zahlen wird daher im Unterschied zu modernen 64-bit-Architekturen verzichtet, auch wenn sie einen höheren Za

hlenbereich bzw. eine höhere Genauigkeit bei Gleitkommazahlen besitzen [stm32]. Neben des Programms als solches kann auch der Mikrocontroller in seiner Leistungsfähigkeit maximiert werden. Naheliegend ist dabei die Kern-Taktfrequenz so weit wie möglich zu erhöhen, damit in gleicher Zeit mehr Befehle abgearbeitet werden können. Dies

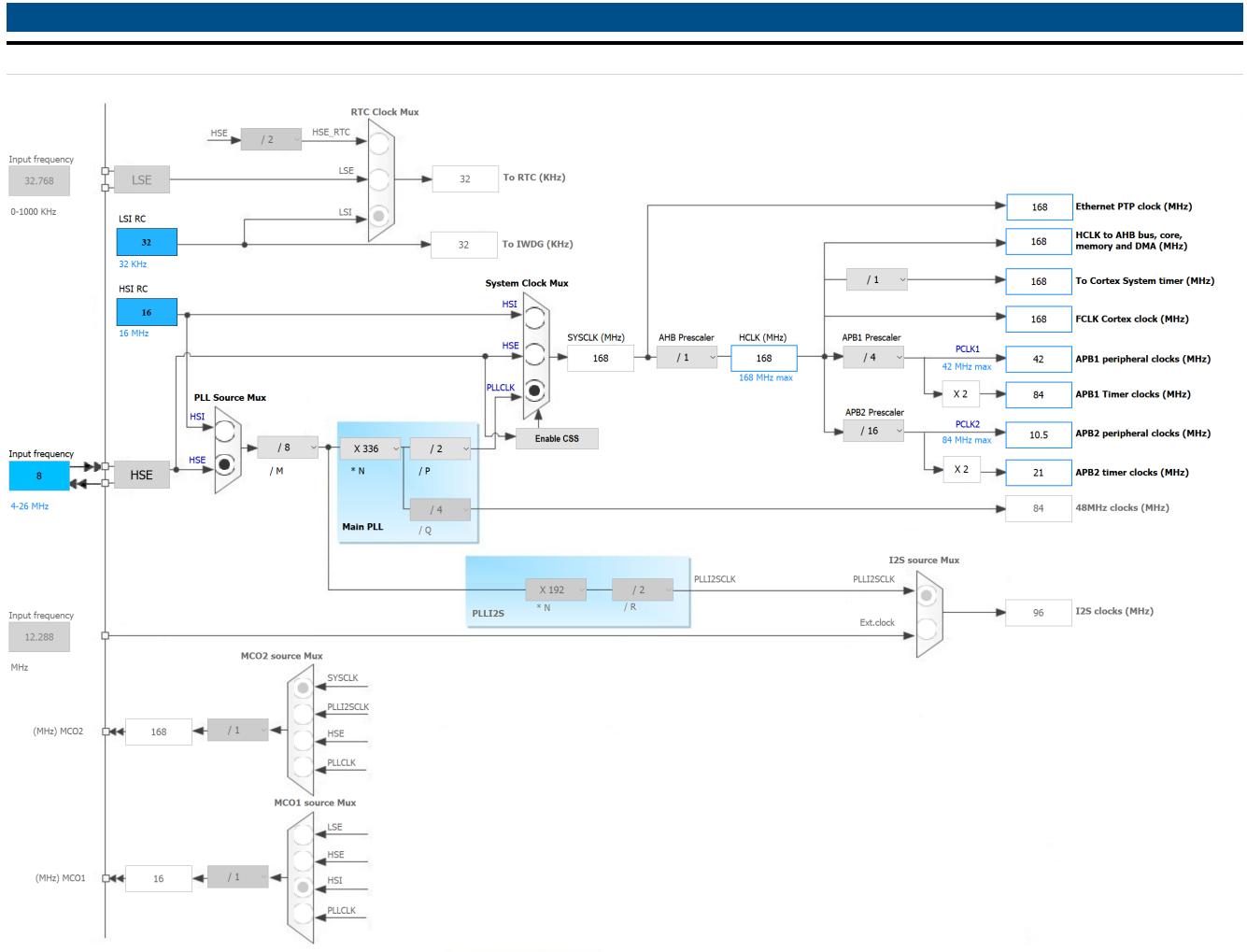


Abbildung 6.14: Taktbaum generiert in STM32 CubeMX

setzt voraus, dass eine ausreichend hohe Versorgungsspannung angelegt wird. Ein Zusammenhang zwischen maximaler Taktfrequenz und dafür benötigter Versorgungsspannung ist in (Wird noch eingefügt) gegeben. Ansonsten sind Instabilitäten nicht auszuschließen, was den Betrieb des Schaltaktors gefährden würde. Der Kerntakt (FCLK), über welchen die Rechengeschwindigkeit beeinflusst wird, leitet sich aus dem sogenannten Clocktree aus Abbildung 6.14 ab. In diesem können auch zahlreiche weitere Taktraten eingestellt werden wie der für die ADCs relevante APB Peripheral Clock. Es stehen verschiedene Taktgeber zur Verfügung, aus denen die Takte abgeleitet werden können. Im vorliegenden Fall wird ein externer Schwungquarz an High Speed External (HSE) verwendet. [stmref]

6.4 Einleitung in benutzerseitige CAN Schnittstelle

Für den späteren Anwender ist vornehmlich die Benutzerschnittstelle über CAN relevant. Hier wird zunächst zwischen drei Nachrichten unterschieden, die innerhalb des CAN-Netzwerkes fortlaufend ausgetauscht werden. Über die Nachricht *TSA_req*, die in Tabelle 6.2 beschrieben ist, kann ein übergeordnetes Steuergerät Befehle an den Aktor chicken. Diese werden dann abhängig des aktuellen Zustands entsprechend umgesetzt. Parallel werden vom Aktor permanent Nachrichten an das Steuergerät gesendet, die Informationen über den aktuellen Zustands liefern. Dabei werden die Informationen auf zwei Nachrichten aufgeteilt, genauereres ist Tabelle 6.3 und Tabelle 6.4 zu entnehmen. Das Gesamtkonzept ist in Abbildung 6.15 dargestellt.

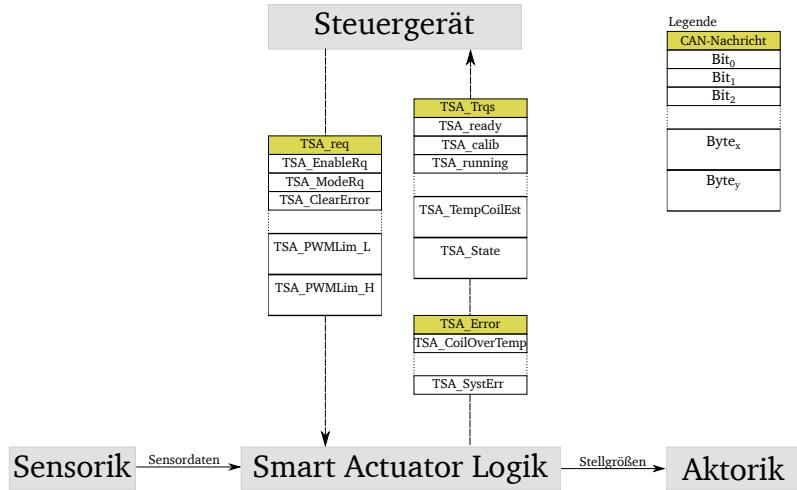


Abbildung 6.15: Gesamtkonzept der CAN-Kommunikation

Bezeichnung	Datentyp	Erläuterung	Zusatz
TSA_EnableRq	Bit	Aktivierungssignal des Aktors	Durch Setzen dieses Bits wird von Betriebsmodus Inaktiv zu Regulär gewechselt
TSA_ModeRq	Bit	Wahl des gewünschten Betriebsmodus Regulär oder Kalibrierung	0: Regulär, 1: Kalibrierung
TSA_ClearError	Bit	Fehler Quittierung	Durch Setzen wird von Betriebsmodus Fehler zu Inaktiv gewechselt, falls der Fehler behoben wurde
TSA_ShiftFirst	Bit	Schaltbefehl in ersten Gang	nur möglich in Betriebsmodus Regulär
TSA_ShiftSecond	Bit	Schaltbefehl in zweiten Gang	nur möglich in Betriebsmodus Regulär
TSA_ShiftNeutral	Bit	Schaltbefehl in neutrale Position	nur möglich in Betriebsmodus Regulär
TSA_PWMLim_L	unsigned Byte	Obere Grenze für PWM - Puls/Pausen-Verhältniss	Low Byte
TSA_PWMLim_H	unsigned Byte	Obere Grenze für PWM - Puls/Pausen-Verhältniss	High Byte

Tabelle 6.2: Erläuterung der CAN-Nachricht TSA_req

Bezeichnung	Datentyp	Erläuterung	Zusatz
TSA_ready	Bit	Aktor ist betriebsbereit	Aktor befindet sich weder im Fehler- noch im Initialisierungszustand
TSA_calib	Bit	Laufender Kalibriervorgang	-
TSA_running	Bit	Warten auf Befehle	Aktor befindet sich in Betriebsmodus Regulär
TSA_shifting	Bit	Laufender Schaltvorgang	Es können keine weiteren Schaltbefehle entgegen genommen werden
TSA_firstGearEng	Bit	Erster Gang eingelegt	Gangposition erkannt und in gültigem Toleranzbereich
TSA_secondGearEng	Bit	Zweiter Gang eingelegt	Gangposition erkannt und in gültigem Toleranzbereich
TSA_neutralGearEng	Bit	Neutrale Gangposition eingelegt	Gangposition erkannt und in gültigem Toleranzbereich
TSA_Heartbeat	Bit	Lebenszeichen im Sekundentakt	Gibt Auskunft über funktionierenden Programmablauf
TSA_Error	Bit	Fehlerzustand	Aktor in Betriebsmodus Fehler , Informationen liefert <i>TSA_Error</i>
TSA_ForkPosMeas	unsigned Byte	Aktuelle Schaltgabelposition	Gemessen in Millimetern
TSA_DcInput	unsigned Byte	Aktuelle Eingangsspannung	Gemessen in V
TSA_TempCoilEst	unsigned Byte	Aktuell gemessene Spulentemperatur	Gemessen in Grad Celsius, um +100 Offsetverschoben
TSA_State	unsigned Byte	Momentaner Zustand in Hauptzustandsautomaten	ID des Zustands des Hauptzustandsautomaten (zur Fehleranalyse)

Tabelle 6.3: Erläuterung der CAN-Nachricht *TSA_Trqs*

Bezeichnung	Datentyp	Erläuterung	Zusatz
TSA_CoilOverTemp	Bit	Übertemperatur in Spulenwicklungen festgestellt	Schwellwert konfigurierbar über <i>ERR_limCoilTemp</i>
TSA_HbridgeOvertemp	Bit	Übertemperatur in H-Brücke festgestellt	Schwellwert konfigurierbar über <i>ERR_limHTemp</i>
TSA_CalibErr	Bit	Fehlerhafte Kalibrierung	Timeout überschritten, ungültige Kalibrierergebnisse oder fehlerhafte Langzeitspeicherung
TSA_dcVErr	Bit	Eingangsspannungsbereich unter- oder überschritten	Schwellwert konfigurierbar über <i>ERR_dcLimMin</i> und <i>ERR_dcLimMax</i>
TSA_SensErr	Bit	Fehlerhafte Sensormesswerte	Sensorwerte außerhalb des erwarteten Bereichs oder inkonsistent
TSA_ShiftErr	Bit	Fehler bei Schaltvorgang	Timeout überschritten oder unerwarteter schlechter Schaltverlauf
TSA_ShortCirc	Bit	Überstromerkennung in Aktor	Schwellwert konfigurierbar über <i>ERR_limCurrent</i>
TSA_SystErr	Bit	Ein Systemfehler ist aufgetreten	Softwaresystem befindet sich in ungeplantem Zustand und muss zurückgesetzt werden

Tabelle 6.4: Erläuterung der CAN-Nachricht *TSA_Error*



7 Analyse und Performance

7.1 Produktvergleich mit Anforderungsliste

Um das Endresultat im Kontext der zuvor definierten Anforderungen analysieren zu können, wird sich auf die Anforderungsliste aus Kapitel 1 berufen, welche aus Übersichtsgründen in Tabelle 7.1 noch einmal abgebildet ist.

Im Rahmen der Arbeit wurde eine zuvor eingerichtete **CAN-Schnittstelle** zum Senden von CAN-Nachrichten genutzt. Dazu werden die Signale in die nach Abschnitt 6.4 definierten Befehle unterschieden, sodass sich eine sinnvolle und benutzerfreundliche Menge an übermittelbaren Nachrichten ergibt. Es ist sowohl möglich die benötigten Schaltbefehle zu übermitteln, als auch Statusmeldungen über Übertemperatur, Überspannung etc. zu empfangen. Damit ist die Festforderung erfüllt.

Um eine **nichtflüchtige Kalibrierung** zu erhalten, werden die Daten der durchgeföhrten Kalibrierung in den Flash-Speicher abgelegt. Aufgrund der Speichertechnologie ist dieser nichtflüchtig, womit die Kalibrierungsdaten auch bei Unterbrechung der Versorgungsspannung des Mikrocontrollers erhalten bleiben. Bei erneuter Kalibrierung werden die alten Kalibrierungsdaten überschrieben.

In Kapitel 7 sind die Plots zu der Regelung des Tauchspulenaktors zu finden, anhand derer eine Abschätzung der **Schaltzeiten** gewonnen werden kann. In den Plots ist zu sehen, dass zwischen erstem Stellsignal und Einlegen des Ganges etwa 45 ms vergehen. Nach einer gemittelten Abschätzung beträgt die Latenz zwischen Senden des Signals und erstem Stellsignal etwa 40 ms. Diese Latenz beinhaltet ebenfalls die Verzögerung zwischen Auswahl des Schaltvorgangs im Graphical User Interface *Control Desk* auf dem Computer und dem Aussenden der Schaltnachricht durch die MicroAutoBox. Damit ist die Bereichsforderung von Schaltzeiten < 100 ms auch für die höchste Abschätzung von 85 ms erfüllt.

Bei der **selbstständigen Fehlererkennung** wurden im Zusammenspiel zwischen Kapitel 4 und Kapitel 6 Erkennungsstrategien für Überstrom, Temperaturüberschreitung und Fehler in der Eingangsspannung implementiert. Überstrom lässt sich über den IS Pin der H-Brücke detektieren, dessen Funktion in Abschnitt 4.4 genauer erklärt wird. Zur Überprüfung einer Temperaturüberschreitung wird ein Temperatursensor im Aktorgehäuse verbaut, der in Unterabschnitt 4.5.1 beschrieben ist. Die Überprüfung der Eingangsspannung erfolgt nach Unterabschnitt 4.5.3. Die Aufgabe der Dekalibrierungserkennung ließ sich im Rahmen des jetzigen Versuchsstandes nicht realisieren, könnte aber nach einer Erweiterung des Prüfstandes möglich sein. Im Falle eines betriebenen Motors ließe sich detektieren, ob dieser im Leerlauf betrieben wird, was der Getriebestellung im Neutralgang entspricht. Wird ein Lastmoment detektiert was sich von dem Referenzlastmoment unterscheidet, ließe sich ein Kalibrierungsfehler erkennen, da in diesem Fall ein Gang eingelegt ist. Das Referenzlastmoment entspricht dem Leerlaufmoment. Diese Überlegung zur Detektion eines Kalibrierungsfehlers ist zu prüfen, womit diese Festforderung nur in Teilen erfüllt wurde.

Als Anforderungen an die **Schnittstellen** sind CAN-Kommunikation, Versorgungsspannungen von 8 – 16 VDC und der Aufbau einer Programmierschnittstelle genannt. Diese Schnittstellen sind alle erfolgreich implementiert worden, was in Kapitel 4 beschrieben ist. Als Programmierschnittstelle wurde sich für eine SWD-Verbindung via ST-Link/V2 entschieden, sodass Updates und Bugfixes problemlos über den Stecker möglich sind. Zusätzlich wurde die Möglichkeit einer UART Kommunikation offen gehalten, sodass diese ebenfalls für Debugging benutzt werden kann. Diese Festforderung ist in allen Punkten erfüllt.

Der Wunsch über **Wartbarkeit** wurde erfüllt. Die Sicherungen laufen über einen externen Sicherungsblock, welcher im späteren dem Sicherungsblock des Fahrzeugs entsprechen soll. Die Platine ist bei derzeitigem Elektronikgehäusestand entnehmbar und somit austauschbar. Bei Softwareproblemen muss dies jedoch nicht erledigt werden, da die Programmierschnittstelle über den Stecker nach außen geführt wird.

Mit einer Baugröße von 88,8x50 mm weist die Platine eine kompakte Baugröße auf und ist somit für den Einbau in einem Smart Actuator geeignet. Im Vergleich mit dem zuvor verwendeten Motortreiber Arduino IBT2 Motortreiber, welcher eine Platinenfläche von 50x50 mm besitzt, weist die Platine weniger als die doppelte Fläche auf, obwohl Logik, CAN-Kommunikation, Spannungsregler, Sensorik und EMV-Maßnahmen ebenfalls darauf platziert sind. Damit ist diese Bereichsforderung erfüllt.

Wirkungsgrad analysieren: Wird morgen erledigt

Die **Temperaturbeständigkeit** wird in Kapitel 4 behandelt und die Auswahl der Bauteile nach diesem Kriterium berücksichtigt. Um dieser Anforderung gerecht zu werden, ist die Mehrheit der Bauelemente nach dem AEC-Q Standard ausgewählt worden. Alle verwendeten aktiven und passiven Bauteile entsprechen so den Forderungen an einen Betriebstemperaturbereich von -40...105 °C.

In der Regelung des Tauchspulenaktors tritt **Überschwingen** auf, was in Abschnitt 7.4 dokumentiert ist. Als kritischer

Wert für einen unbeabsichtigter Gangwechsel ist ein Überschwingen von 1 mm festgesetzt. Die Messergebnisse weisen Überschwingweiten von 0.4 mm auf, daher ist ein unbeabsichtigter Gangwechsel ausgeschlossen. Die Bereichsforderung ist damit erfüllt.

Die **Standby-Leistungsaufnahme** beträgt in etwa 1.38 W, da zur Versorgung des Mikrocontrollers und der restlichen Bauteile ein Standby-Strom von etwa 100mA fließt.

Nach Unterabschnitt 7.4.4 ist zu erkennen, dass über Gegenbestromung der Spule eine Gegenkraft zur Massenträgheit (Lorentzkraft) aufgebaut wird. Darüber wird die **Schaltgabelkraft** am Anschlag minimiert.

Relevanz	Anforderung	Erläuterung	Erfüllt?
FF	Benutzerfreundliche Kommunikation durch CAN Schnittstelle	Empfang von Befehlen, Senden von Statusmeldungen	✓
FF	Nichtflüchtige Kalibrierung	Eine Kalibrierung ist nur einmalig und zur Rekalibrierung notwendig	✓
BF	Schaltzeit	< 100 ms (Latenz zwischen Senden des Befehls und vollständig ausgeführtem Gangwechsel)	✓
FF	Selbstständige Fehlererkennung	Überstrom, Temperatur, Eingangsspannungsreich, Dekalibrierung	(✓)
FF	Schnittstellen	CAN, 8-16VDC Versorgung, Programmierschnittstelle (für Updates & Bugfixes)	✓
W	Wartbarkeit	Sicherung wechseln im eingebauten Zustand	✓
BF	kompakte Baugröße	88,8x50 mm	✓
BF	Effizienz (gemittelt über einen Schaltvorgang)	elektrischer Wirkungsgrad > 90 %	
FF	Temperaturbeständigkeit	bis 105°C	✓
BF	Aktorüberschwingen	Toleriert, solange kein unbeabsichtigter Gangwechsel	✓
W	Schaltgabelkraft am Anschlag	möglichst gering	✓
FF	Standby	Standbyleistungsaufnahme < 2W	✓

Tabelle 7.1: Anforderungsliste

7.2 Kostenaufstellung

Die Materialkosten für die fertige Platine inklusive aller Bauteile belaufen sich bei einer einzigen Platine auf 80 Euro. Bei einer Stückzahl von 100 kann der Preis bereits auf 33,5 Euro gesenkt werden, während die Bauteilkosten bei einer Fertigung von 1000 Platinen nochmal auf knapp unter 25 Euro sinken. Diesen großen Preisunterschied verursacht vor allem die unbestückte Platine selbst, die bei Bestellung von einer einzigen 38,5 Euro kostet und bei einer Bestellung von 1000 Stück nur noch 0,82 Euro. Tabelle 7.2 zeigt die verwendeten Bauteile und deren Anzahl sowie den kumulierte Preis pro Bauteilart (Anzahl des Bauteils multipliziert mit dem Einzelpreis) für jeweils eine Fertigung von einer Platine, von 100 Platinen und von 1000 Platinen. Die Preise stammen dabei von den Anbietern, bei denen die Komponenten jeweils eingekauft wurden.

Die gesamte Preisaufstellung für die Stückzahlen eins, 100 und 1000 inklusive der einzelnen Widerstände und Kondensatoren, sowie die Händlerlinks zu allen Bauteilen befindet sich im Anhang. In nachfolgenden Abbildungen wird die Verteilung der Kosten auf die verschiedenen Bauteilgruppen dargestellt. Die Bauteile wurden unterteilt in die Platine, die passiven Bauteile (Kondensatoren, Widerstände und Schwingquarz), die integrierten Halbleiterchips (Mikrocontroller, Spannungsregler, CAN Transceiver, Leitungstreiber, Halbbrücken) sowie die Stecker, die die Schnittstellen nach außen darstellen.

Es ist zu erkennen, dass die Platine bei geringen Stückzahlen ungefähr die Hälfte der Kosten ausmacht, während sie bei hohen Stückzahlen fast gar nicht mehr ins Gewicht fällt. Bei hohen Stückzahlen sind der größte Kostenfaktor die ICs. Ein Preisvergleich mit direkter Konkurrenz fällt schwer, da es wenig äquivalente Produkte auf dem Markt gibt, jedoch können die beiden vorherig am Prüfstand verwendeten Motortreiber zum Vergleich gezogen werden, die nur einen Teil der Platinenfunktionen abdecken. Der Kaufpreis des Motorcontrollers MDC1460 des Herstellers RoboteQ liegt bei circa 260 Euro (bei über vierfachem Bauraum), der Kaufpreis des Arduino IBT_2 Motortreibers liegt bei circa 15 Euro. Zu bemerken ist, dass sich die Kosten der Platine auch mit einem mit einzurechnenden Gewinnaufschlag innerhalb des Preisrahmens der beiden Motortreiber bewegen, obwohl sie deutlich mehr Funktionen hat.

Anzahl	Bauteil	Bauteilpreis	Bauteilpreis 100+	Bauteilpreis 1000+
1	Platine	38.5 Euro	2.39 Euro	0.82 Euro
1	Mikrocontroller	10.66 Euro	7.79 Euro	6.58 Euro
2	Halbbrücken	6.56 Euro	5.96 Euro	5.04 Euro
1	Leitungstreiber	0.617 Euro	0.348 Euro	0.252 Euro
1	CAN Transceiver	2.82 Euro	2.28 Euro	1.51 Euro
1	Voltage Regulator 3,3V	0.337 Euro	0.162 Euro	0.103 Euro
1	Voltage Regulator 5V	0.337 Euro	0.162 Euro	0.103 Euro
1	Klemmblock	1.16 Euro	1.07 Euro	0.912 Euro
2	Klemmblock	0.742 Euro	0.618 Euro	0.526 Euro
1	Steckverbinder	0.0442 Euro	0,0442 Euro	0,0387 Euro
1	AMPSEAL Automotive Steckverbinder	7.09 Euro	6.22 Euro	5.23 Euro
1	Quarz	0.605 Euro	0.355 Euro	0.384 Euro
21	Widerstände	4.88 Euro	3.046 Euro	1.6754 Euro
21	Kondensatoren	5.907 Euro	3.0648 Euro	1.501 Euro
Gesamtpreis		80,26 Euro	33,51 Euro	24,68 Euro

Tabelle 7.2: Preisliste

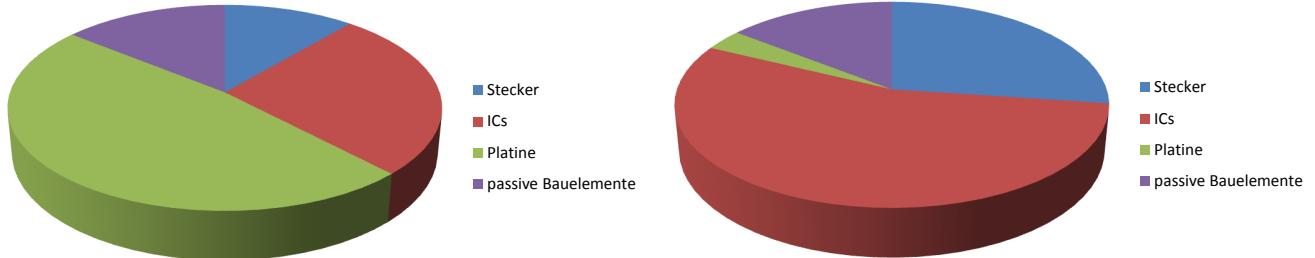


Abbildung 7.1: Aufteilung der Kosten für die Stückzahlen 1 (links) und 1000 (rechts)

7.3 Platinendesign-Analyse

Die Platine erfüllt für den derzeitigen Aktor die Aufgaben über Positionsregelung bei Schaltvorgängen, Überwachung der Umgebung (Temperatur, Eingangsspannung, Ausgangstrom) und die Kommunikation mittels CAN-Nachrichten. Bisher sind keine bedeutsamen Probleme mit EMV aufgetreten, sodass die Sensorik auch bei Ansteuerung der Halbbrücken funktioniert und die CAN-Kommunikation erhalten bleibt. Die Anforderung an eine kompakte Baugröße wird mit 88,8x50 mm erfüllt. Eine einfache Handhabung in der Peripherie wird über den Anschlussstecker erreicht. Im Falle von Softwareupdates lässt sich der Betriebstecker leicht entfernen und die Updates über den Programmierstecker flashen, was in Abschnitt 4.3 genauer behandelt wird. Sollten im Betrieb nach Einführen des neuen Aktors trotz der Schirmungsmaßnahmen Probleme mit EMV auftreten, so wird empfohlen das Platinendesign auf ein 4-Layer Design zu erweitern, um wie in Kapitel 5 beschrieben, zusätzliche GND-Planes einzufügen zu können. Weiterhin bleiben die LDOs der 3.3 V und 5 V Schiene zu beobachten, da diese beim Anschluss der Autobatterie relativ viel Wärme abgeben müssen um auf die jeweiligen Spannungsebenen zu regeln. Derzeit sind keine Probleme damit aufgetreten jedoch kann das Verhalten im Dauerbetrieb noch nicht vollständig abgeschätzt werden. Falls keine Probleme auftreten wird empfohlen diese auch bei Einführung eines 4-Layer Designs beizubehalten, da die LDOs konstante Spannungen ohne switching Charakteristiken liefern. Bei Problemen könnte ein Buck-Spannungsregler statt des 5 V LDOs eingeführt werden, welcher von Batteriespannung auf die 5 V regelt. Für die 3 V könnte weiterhin ein LDO verwendet werden, welcher als VCC die 5 V des Buck-Spannungsreglers nutzt. Damit wäre die Stabilität der 3 V Versorgung für die ADC-Genauigkeit gesichert und das Problem entstehender Wärmeverluste reduziert. Allerdings könnte in dieser Verschaltung die Genauigkeit des Lagesensors nicht auf dem aktuellen Niveau garantiert werden, da dieser an die 5 V angeschlossen ist und der Buck-Spannungsregler weniger konstante Spannungen ausgibt als ein LDO.

7.4 Regelergebnisse

In Abbildung 7.2 und Abbildung 7.3 sind die exemplarisch gemessenen Positionsverläufe, Regelabweichungen und Stellgrößen für das Schalten von Gang 2 zu Neutral und umgekehrt aufgetragen. Das Schalten von und nach Gang 1 wird nicht betrachtet, da der verbaute Synchronring die Regelung bei einer Drehzahl $n = 0$ zu stark behindert.

Die Messungen wurden hinreichend oft durchgeführt, sodass eine Reproduzierbarkeit der Ergebnisse sichergestellt werden konnte. Die Sollwerte sind jeweils mit roten Linien gekennzeichnet und betragen für den zweiten Gang 10 mm und Neutral 0 mm. Für beide wurde ein Toleranzband festgelegt, dass 10 % der Sprunghöhe zwischen den Gängen entspricht. Der verwendete PID-Regler mit Störgrößenkompensation wurde in Unterabschnitt 2.1.6 beschrieben.

7.4.1 Schalten in Gang 2

Für das Schalten in Gang 2 ist in Abbildung 7.2 ein Überschwingen von $380 \mu\text{m}$ zu beobachten. Dies kann allerdings auf die lose Befestigung des Lagesensors im Prüfstand zurückgeführt werden. Der Schaltvorgang bringt den ganzen Prüfstand kurzzeitig zum Schwingen, sodass der Lagesensor eine zusätzliche relative Bewegung zur Läuferstange erfährt. Die Sollposition wird erstmals nach 36 ms erreicht und verlässt das Toleranzband nach einer Dauer von 54 ms nicht mehr. In dem Verlauf der Stellgröße zeigt sich, dass für den Schaltvorgang das maximale PWM-Signal bei 63 % liegt und somit ein großer Anteil der möglichen Stromdurchschaltung gar nicht ausgenutzt wird. Die Stellgröße bleibt nahezu konstant, bis ca. 2 mm vor dem Erreichen der Sollposition. Anschließend fällt sie kurzzeitig ins Negative, bevor sie bei 0 % einpendelt.

7.4.2 Schalten in Neutral

Bei diesem Schaltvorgang zeigt Abbildung 7.3 ein erstes Überschwingen von $288 \mu\text{m}$ über den Sollwert. Auch dieses Überschwingen kann durch die Bewegung des Sensors erklärt werden. Die Sollwertvorgabe wird nach 39 ms zum ersten Mal erreicht und dessen Toleranzband wird nach 50 ms nicht mehr verlassen. Die Stellgröße springt zu Beginn des Schaltvorgangs auf ein PWM-Signal von -72% , was anschließend betragsmäßig bis zu einer Regelabweichung von 1.4 mm auf einen Wert von -28% abnimmt. An dieser Stelle springt das PWM-Signal in den niedrigen positiven Bereich, bevor es bei 0 % einpendelt.

7.4.3 Diskussion der Regelergebnisse

Die beiden Schaltvorgänge erreichen ihren Sollwert und die zugehörige stationäre Genauigkeit in unter 50 ms, was die Anforderungen von einer Schaltzeit $\leq 100 \text{ ms}$ klar erfüllt. Für die stationäre Genauigkeit ist die korrekte Positionierung des Lagesensors von entscheidender Bedeutung. Diese Position bleibt allerdings nicht über alle Schaltvorgänge gleich, aufgrund der losen Halterung und den Erschütterungen des Prüfstands. Abhilfe könnte hier eine besserer Fixierung schaffen, wodurch die Regelergebnisse weiter verbessert werden könnten. Aber auch unter Annahme, dass die Überschwingungen nicht auf den Lagesensor zurückzuführen sind, stellen sie aufgrund ihrer geringen Größe ($\leq 400 \mu\text{m}$) kein Problem dar. Erst ab einem Überschwingen von $\geq 1 \text{ mm}$ wäre in der Neutralstellung die Gefahr des unabsichtlichen Schaltens in einen anderen Gang gegeben.

In den Verläufen der Stellgrößen ist zu erkennen, dass der Aktor ab ca. $1 - 2 \text{ mm}$ nicht mehr beschleunigt, sondern leicht abgebremst wird.

Mit der erreichten stationären Genauigkeit sind die Bedingungen für eine erfolgreiche Implementierung eines Haltereglers erfüllt. Dabei wird der Positionsregler durch einen anderen Regler abgelöst, sobald die Position das Toleranzband für eine bestimmte Zeit nicht mehr verlässt. Die Aufgabe des neuen Reglers besteht dann lediglich darin die Position zu halten, bis ein neuer Schaltbefehl erfolgt. Hauptproblem für diesen Regler ist der Haftgleiteffekt, welcher auf der Tatsache beruht, dass der Haftreibungskoeffizient größer als der Gleitreibungskoeffizient ist [Bowden2001].

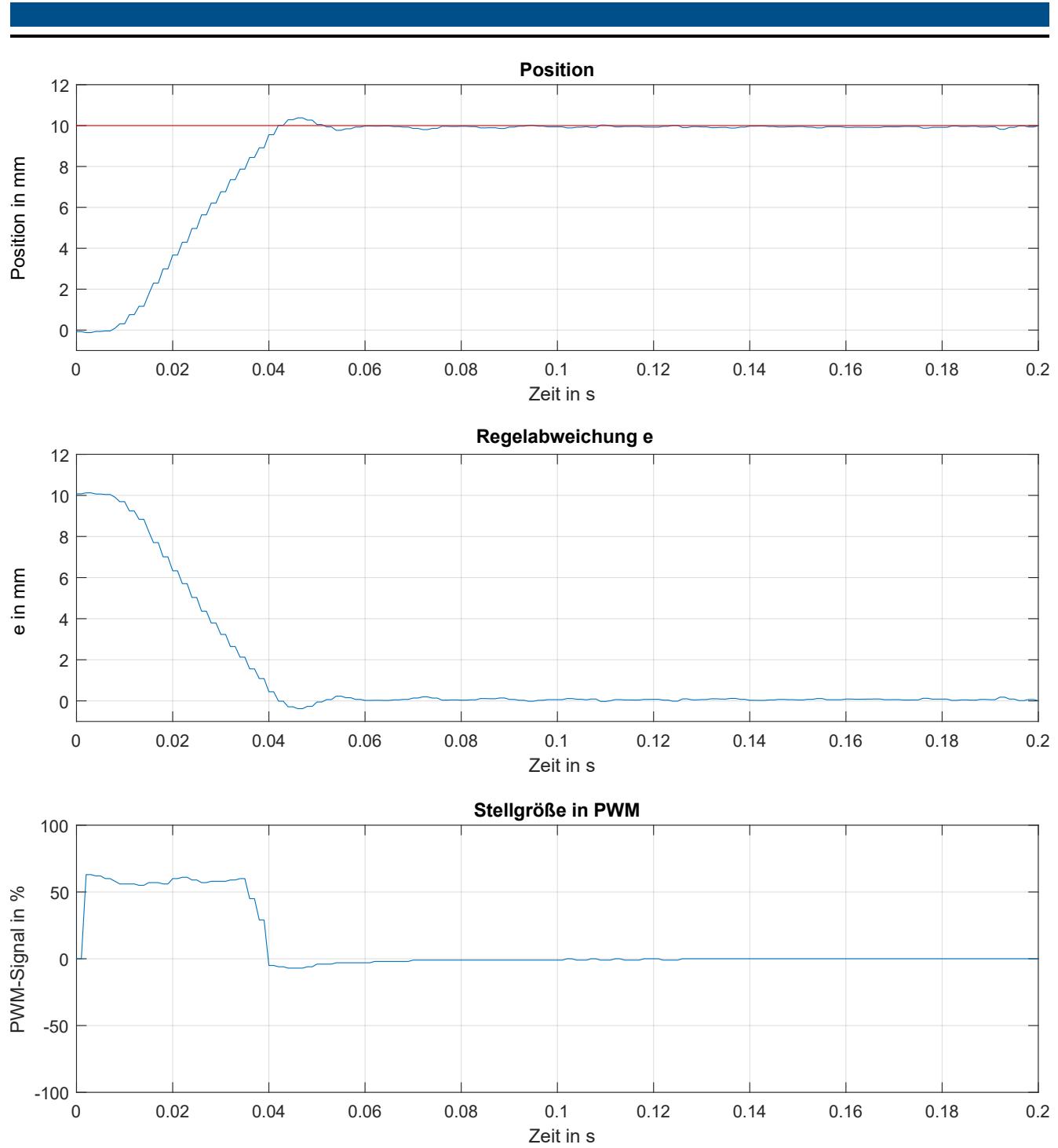


Abbildung 7.2: Position, Regelabweichung und Stellgröße für das Schalten in Gang 2

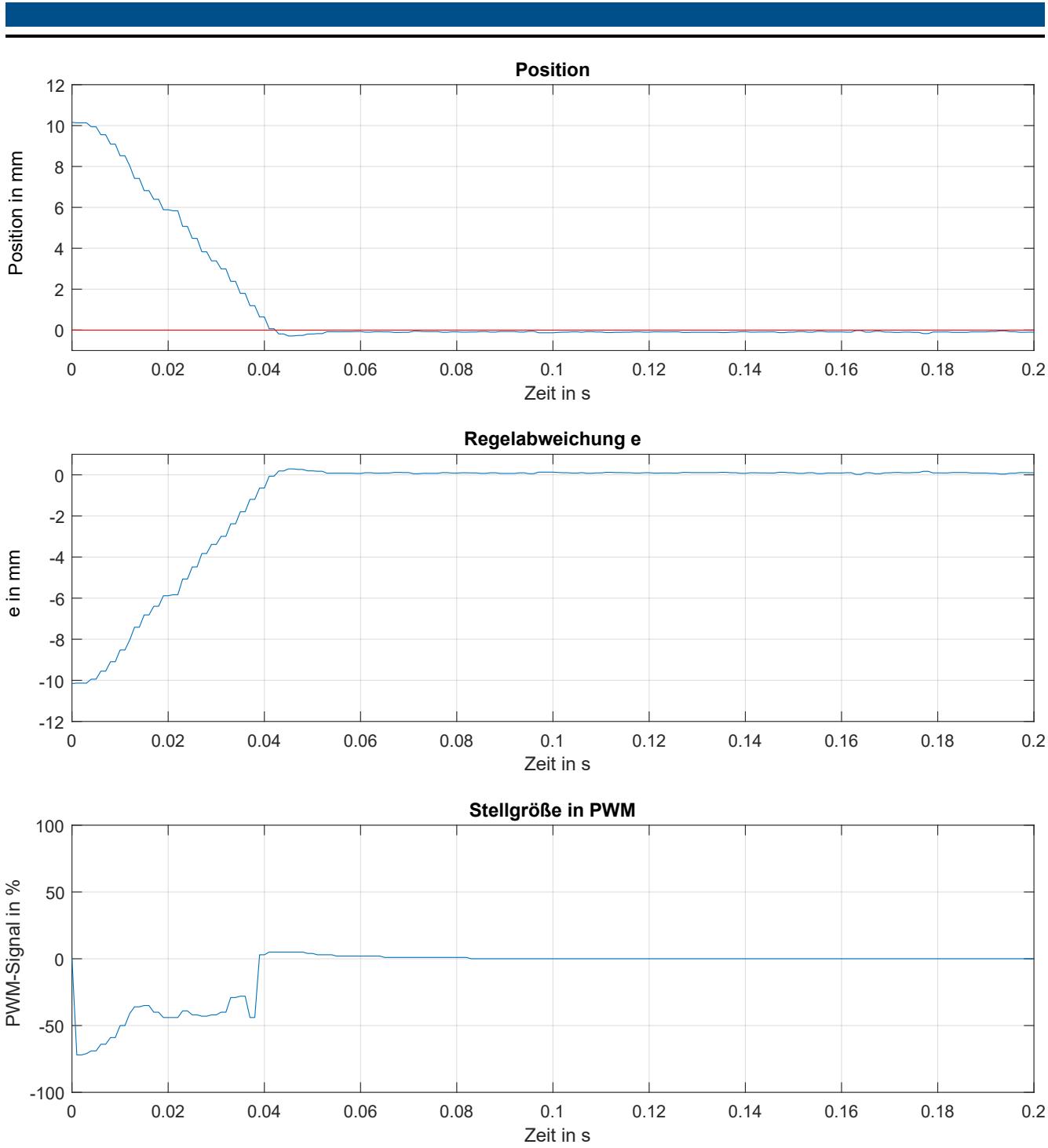


Abbildung 7.3: Position, Regelabweichung und Stellgröße für das Schalten in Neutral

7.4.4 Schaltgabelkraft am Anschlag

In Abbildung 7.4 ist ein Schaltvorgang von Neutral in Gang 2 und die zugehörige Strommessung des Aktors aufgetragen. Zu sehen ist, dass beim Erreichen der Sollgröße zum Zeitpunkt $t = 0.1316$ s nur etwa 0.1 A im Aktor fließt. Das bedeutet, dass die Lorentzkraft der Spule auf den Läufer gemäß Gleichung 2.1 keine nennenswerte Kraft auf den Aktor ausübt. Somit wirkt auf den Läufer zu diesem Zeitpunkt fast ausschließlich die Kraft aufgrund seiner Massenträgheit. Nach dem Übertreten des Läufers der Sollposition reagiert die Regelung durch Gegenbestromung, sodass die Massenträgheit durch die Lorentzkraft gehemmt wird und der Läufer in Richtung Sollposition gebracht wird. Die Anschlagskraft wird somit minimiert.

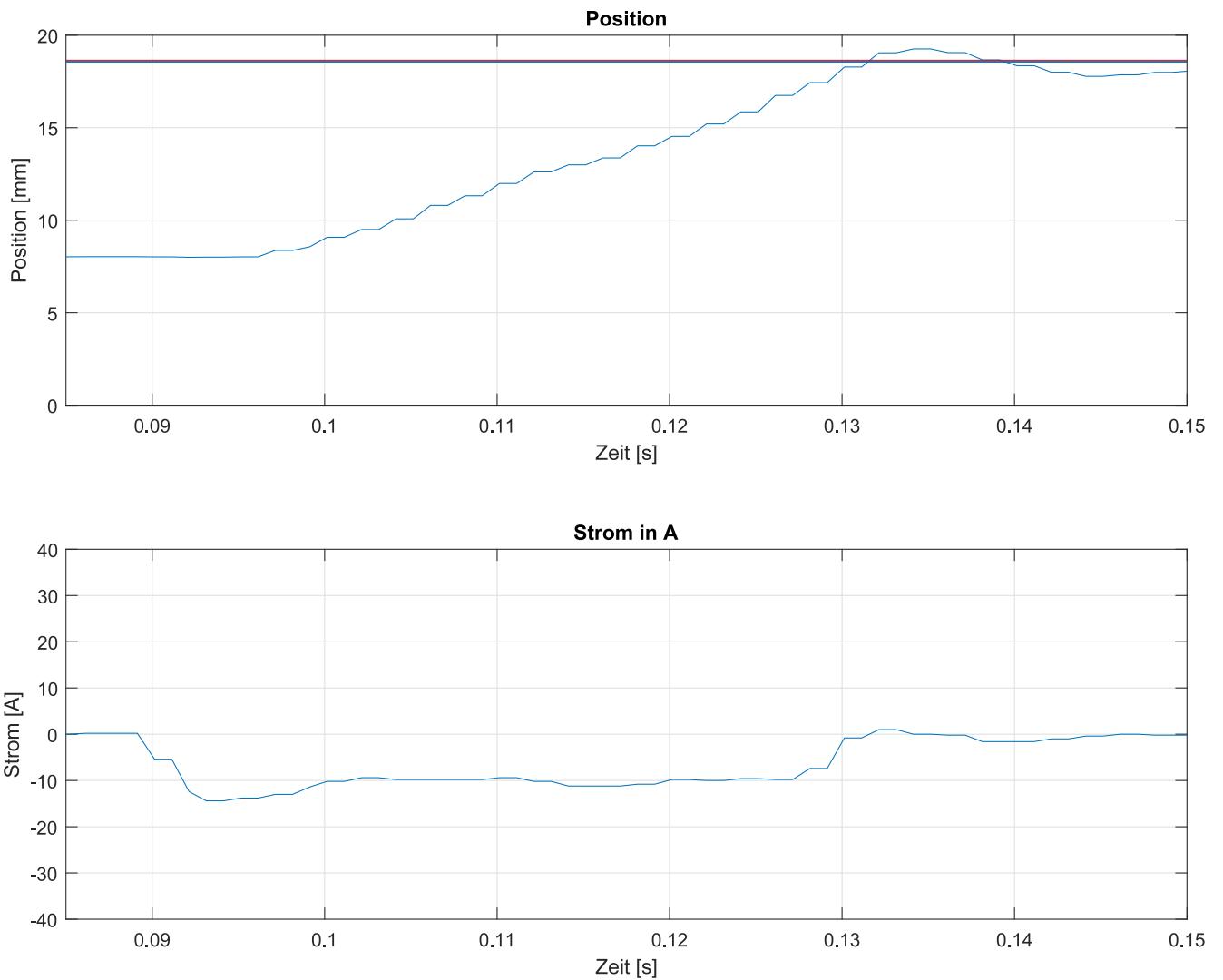


Abbildung 7.4: Strommessung im Schaltvorgang von Neutral in Gang 2



8 Fazit und Ausblick

Ziel dieser Arbeit war es, eine eingebettete Elektronik für den linearen Schaltaktor am Prüfstand zu entwickeln und zu testen. Im ersten Schritt wurden dazu die bereits bestehende Hard- und Software der bisher am Prüfstand angebrachten Elektronik analysiert, die Anforderungen an das Endprodukt festgelegt und auf Grundlage dessen schrittweise eigene Schaltungen auf Testboards kreiert und untersucht. Hierfür wurde die Elektronik vor allem in Aktoransteuerung, Recheneinheit und Sensorik unterteilt, welche später auf der Endplatine wieder zusammengeführt wurden. Nachdem die H-Brücke, geschaltet über den Mikrocontroller, erfolgreich den Aktor in Betrieb setzen konnte und die Datenübertragung von Lage- und Temperatursensor mit dem Mikrocontroller sichergestellt werden konnte, wurden anschließend alle Subsysteme kombiniert, sodass ein endgültiger Prototyp entstand, welcher alle Performanceanforderungen erfüllen sollte. Dieser wurde mit Hilfe der parallel erarbeiteten Programmierung in Matlab Simulink durch Verwendung des Waijung Blocksets über den Mikrocontroller gesteuert und die Funktionen überprüft. Auch die Regelung der Schaltgabelposition wurde überarbeitet und optimiert. Anhand des funktionierenden Prototypen wurde daraufhin die Platine nach einer Einarbeitung in Platinendesign geplant und entworfen. Die industriell gefertigte Platine konnte nun im institutseigenen Lötlabor mit allen Komponenten bestückt werden und im Anschluss am Prüfstand angeschlossen und getestet werden. Nachdem der Mikrocontroller einmal *geflasht* wird, steht die Funktionalität bei anliegender Versorgungsspannung bereit. Die CAN-Schnittstelle der Platine stellt eine Echtzeitkommunikation mit der MicroAutoBox her, sodass der Aktor über die Benutzeroberfläche ControlDesk am PC gesteuert werden kann. Die H-Brückenschaltung ermöglicht das Ansteuern des Aktors und somit das Schalten des Getriebes in Normalstellung oder einen der beiden Gänge. Der Lagesensor erfassst währenddessen die Position der Schaltgabel, die über die nichtflüchtige Kalibrierung genau berechnet werden kann. Die implementierte Regelung sorgt für die möglichst genaue Einhaltung der Sollvorgabe der Schaltgabelposition. In der vorliegenden Arbeit konnte eine funktionierende eingebettete Elektronik für den linearen Tauchspulenaktor am Prüfstand des IMS entwickelt werden. Besonders hervorzuheben ist dabei die Kompaktheit der Platine, die direkt am Aktor angebracht wird, gegenüber dem vorherigen Zustand. Schnittstellen sowie Versorgung sind über einen einzigen Steckerausgang verbunden, der wahlweise mit dem passenden Gegenstück zur Programmierung oder zum Betrieb gekoppelt werden kann. Abbildung 8.1 zeigt übersichtlich das Ergebnis der Arbeit mit der eingebetteten Elektronik aufgeteilt in ihre Subsysteme, sowie den Schnittstellen zum Aktor, Lagesensor sowie zur MicroAutoBox, welche in der Platinenschaltung integriert wurden.

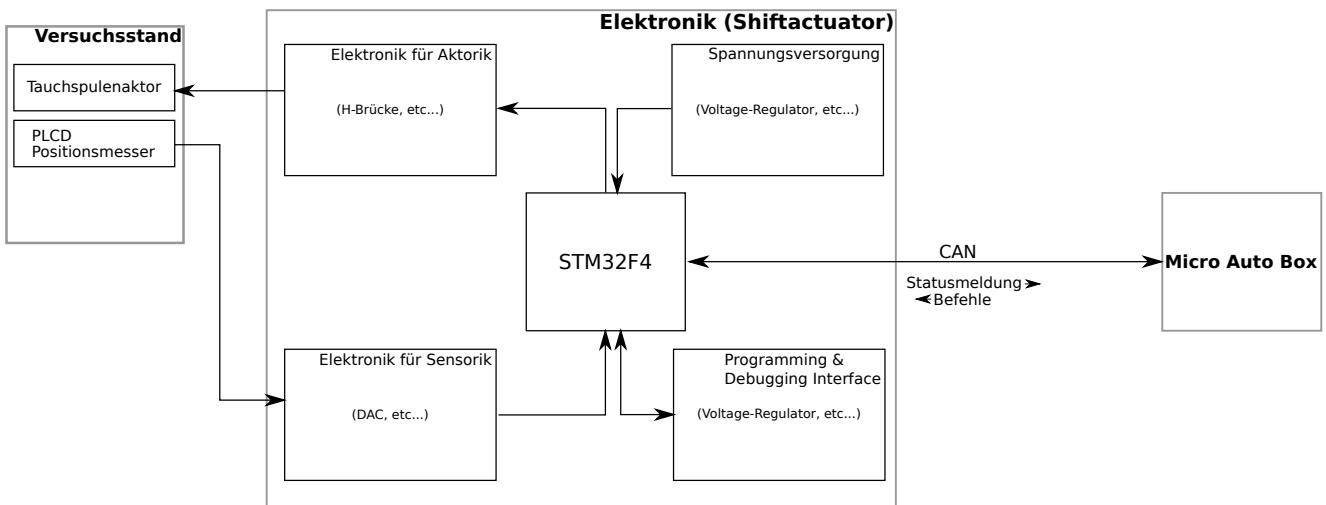


Abbildung 8.1: Ergebnis der eingebetteten Elektronik und Schnittstellen

Die gestellten Anforderungen wurden weitestgehend erfüllt (vgl. Abschnitt 7.1), sodass von einem erfolgreichen und zufriedenstellenden Produktdesign ausgegangen werden kann. Eine Weiterentwicklung der eingebetteten Elektronik kann aufgrund der flexiblen Systemintegrierbarkeit und des leicht zu bedienenden Programms in weiteren Arbeiten erfolgen.

8.1 Ausblick

Der Schaltaktorikprüfstand des IMS wird stetig weiterentwickelt. So sind auch im Anschluss an diese Projektarbeit weitere Schritte zur Optimierung des Gesamtsystems denkbar und geplant. Zunächst soll der bisher angebrachte Aktor durch einen im Wirkprinzip äquivalenten Aktor mit einem Strom von ca. 50 A bei maximal anliegender Spannung anstatt den jetzigen ca. 19 A bei Maximalspannung. Diese geplante Neuerung wurde auch in der Komponentenauswahl während dieser Arbeit berücksichtigt, weshalb die Funktionalität der eingebetteten Elektronik auch weiterhin gegeben sein sollte. Weiterhin geplant ist es, die MicroAutoBox als CAN-Kommunikationspartner langfristig auszutauschen. Das Mitschwingen des kompletten Prüfstands während den großen Kraftübertragungen eines Schaltvorgangs stellt ein Problem dar, welches die Messungenauigkeiten bedingt. Sinnvoll wäre es, die Halterung des PLCD-Sensors zur Erfassung der Schaltgabelposition stabiler zu gestalten, um eine optimale Positionsmessung und somit bessere Regelergebnisse zu erreichen. Bisher ist der Sensor leicht beweglich befestigt und kann sich somit schnell verstellen. Eine weitere Lösung für dieses Problem ist der Einsatz einer sensorlosen Positionserfassung, wie sie bereits in einer vorangegangenen Arbeit [adp] theoretisch entwickelt wurde. Damals konnte die Umsetzung noch nicht erfolgen, da das Messverfahren nur auf der MicroAutoBox implementiert war, dessen Abtastrate des AD-Wandlers nicht hoch und die Anstiegsrate der digitalen Ausgänge zur Erzeugung einer PWM-Frequenz nicht schnell genug waren. Der STM32F405RGT7 mit einer ADC Abtastrate von bis zu 6Ms/s im Interleave Mode und einer Ansprechzeit von 125 ns ist laut den gestellten Anforderungen aber ausreichend, um das entwickelte Messverfahren durchzuführen.

Wie in Unterabschnitt 7.4.3 analysiert wäre die Implementierung eines Haltereglers Gegenstand zukünftiger Forschung. Über diesen ließen sich eine Optimierung der Ganghaltung ermöglichen.