

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

Zusammenfassung

Hier könnte Ihr Abstract stehen.



Inhaltsverzeichnis

1 Einleitung	5
1.1 Motivation	5
1.2 Ziele der Arbeit	5
1.3 Anforderungsliste	5
1.4 Aufbau der Arbeit	5
2 Aufbau des Prüfstands und Grundlagen	7
2.1 Grundlagen des Prüfstandes	7
2.1.1 Getriebe allgemein	7
2.1.2 Getriebe des Prüfstands	8
2.1.3 Aktor	8
2.1.4 Aktoransteuerung	9
2.1.5 Autobox	9
2.1.6 Positionsmessung und -regelung	9
2.2 Grundlagen Löten	10
2.3 Mikrocontroller	10
2.4 Eingebettete Systeme und Smart Actuators	11
2.5 Controller Area Network (CAN)	12
2.6 Pulsweitenmodulation	13
3 Vorgehensweise und Testverfahren	15
3.1 Entwicklungsmodell	15
3.1.1 Das V-Modell	15
3.1.2 Konkretes Vorgehen mit inkrementellem Entwicklungsansatz	17
4 Komponentenauswahl und Schaltungsdesign	19
4.1 H-Brücke	19
4.2 Spannungsversorgung	20
5 Platinendesign	23
6 Software	25
7 Analyse und Performance	27
7.1 Beurteilung der Anforderungserfüllung	27
7.2 Kostenaufstellung	27
8 Fazit und Ausblick	29
Abbildungsverzeichnis	31



1 Einleitung

1.1 Motivation

1.2 Ziele der Arbeit

Auch das Institut für Mechatronische Systeme (IMS) der TU Darmstadt nimmt sich dieser Aufgabe an und arbeitet im Rahmen des Projektes Speed4E, dass einen elektrifizierten Antriebsstrang mit Peak-Antriebsdrehzahlen von bis zu $50.000/min$ zum Ziel hat, an einem innovativem Schaltaktor. Im Verlauf vorheriger Arbeiten wurde bereits ein Tauchspulenaktor ausgelegt sowie eine Elektronik für ihn 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. Des Weiteren sollen Sicherheits- und Überwachungsfunktionen für den Aktor entwickelt werden und Statusmeldungen per CAN gesendet werden.

1.3 Anforderungsliste

Um das übergeordnete Ziel weiter zu spezifizieren wurde zunächst eine Anforderungsliste 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 schließlich noch nach Priorität kategorisiert und einer der vier folgenden Anforderungsarten [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.

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, Eingangsspannung (OVP/UVP), Dekalibrierung
FF	Schnittstellen	CAN, 8-12VDC Versorgung (max XA), Programmerschnittstelle (für Updates & Bugfixes)
W	Wartbarkeit	Sicherung wechseln im eingebauten Zustand
BF	kompakte Baugröße	
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 < 5W

Tabelle 1.1: Anforderungsliste

1.4 Aufbau der Arbeit

Nachdem in der oben stehenden Anforderungsliste 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 ADPs sowie wichtige Grundlagen als Basis für die weitere Bearbeitung und zum besseren Verständnis dargelegt. Kapitel 3 stellt das allgemeine methodische Vorgehen für das Entwerfen der eingebetteten Elektronik dar. In Kapitel 4 werden die verschiedenen Komponenten, die für die Elektronik benötigt werden, und ihre Funktionen beschrieben. Es wird erklärt, wie die einzelnen Komponenten miteinander interagieren und wie ihre Schaltung funktioniert. In Kapitel 5 soll schließlich das endgültige Platinendesign und inklusive schematischem Aufbau vorgestellt werden. Darauf folgt in Kapitel 6 die Vorstellung des Programms.

2 Aufbau des Prüfstands und Grundlagen

2.1 Grundlagen des Prüfstandes

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. 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.

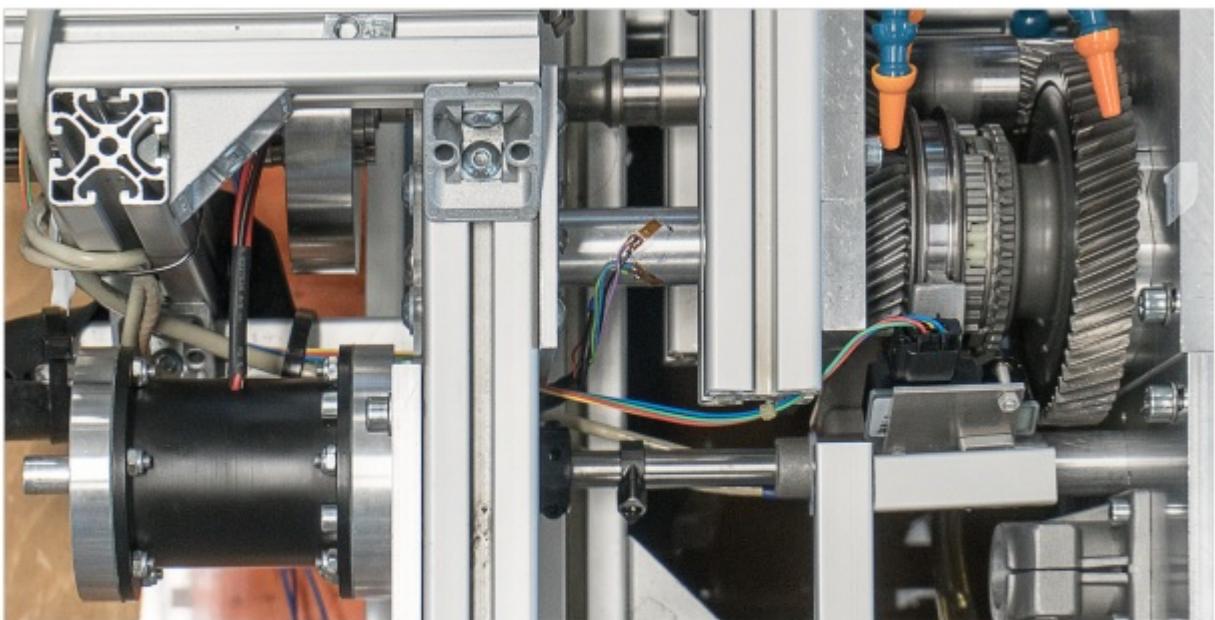


Abbildung 2.1: Prüfstand [adp]

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 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 angetriebenen Schaltgabel in Position gebracht wird.

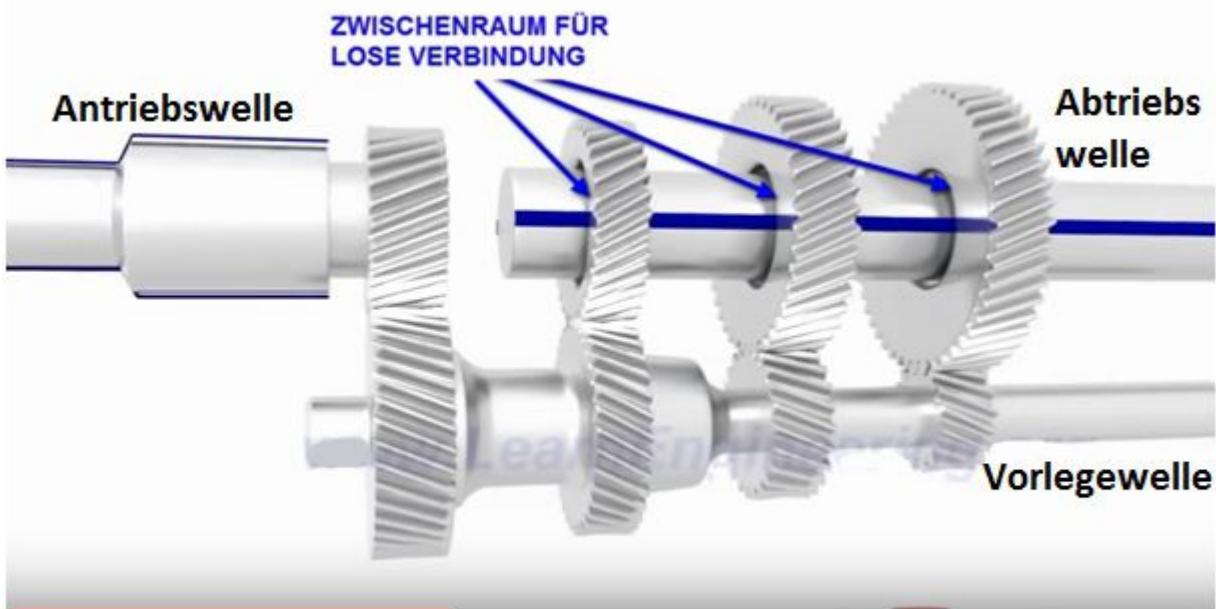


Abbildung 2.2: Fahrzeuggetriebe

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 Klauekupplung erfolgt. Somit können sowohl Schaltaktoriksysteme mit als auch 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.

Sein 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

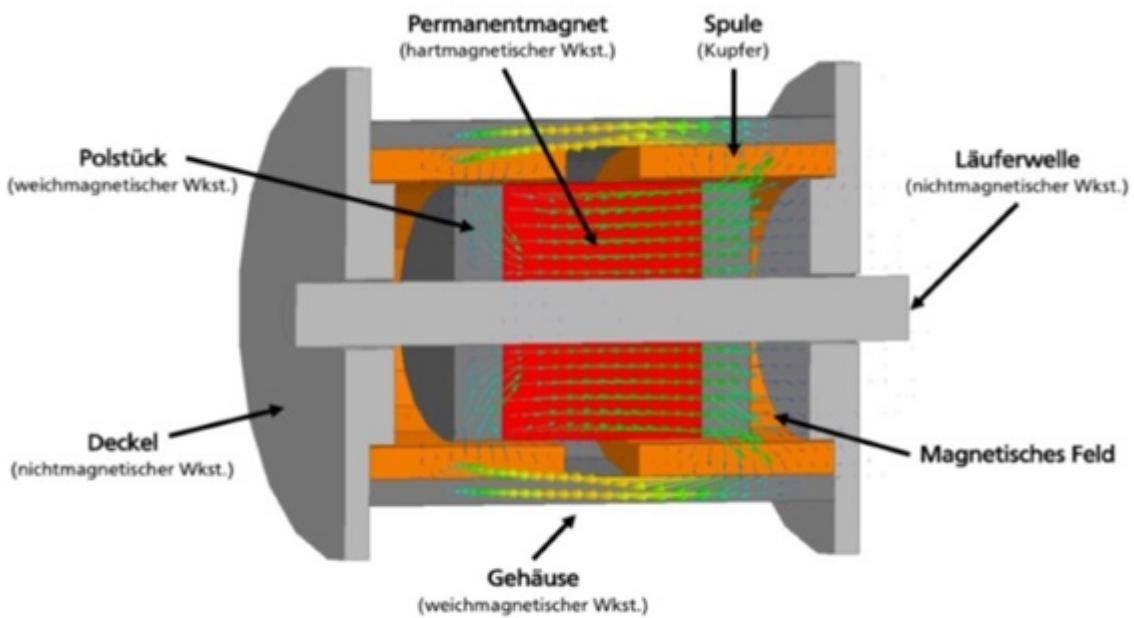


Abbildung 2.3: Querschnitt Tauchspulenaktor [Hahn2018]

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.

2.1.4 Aktoransteuerung

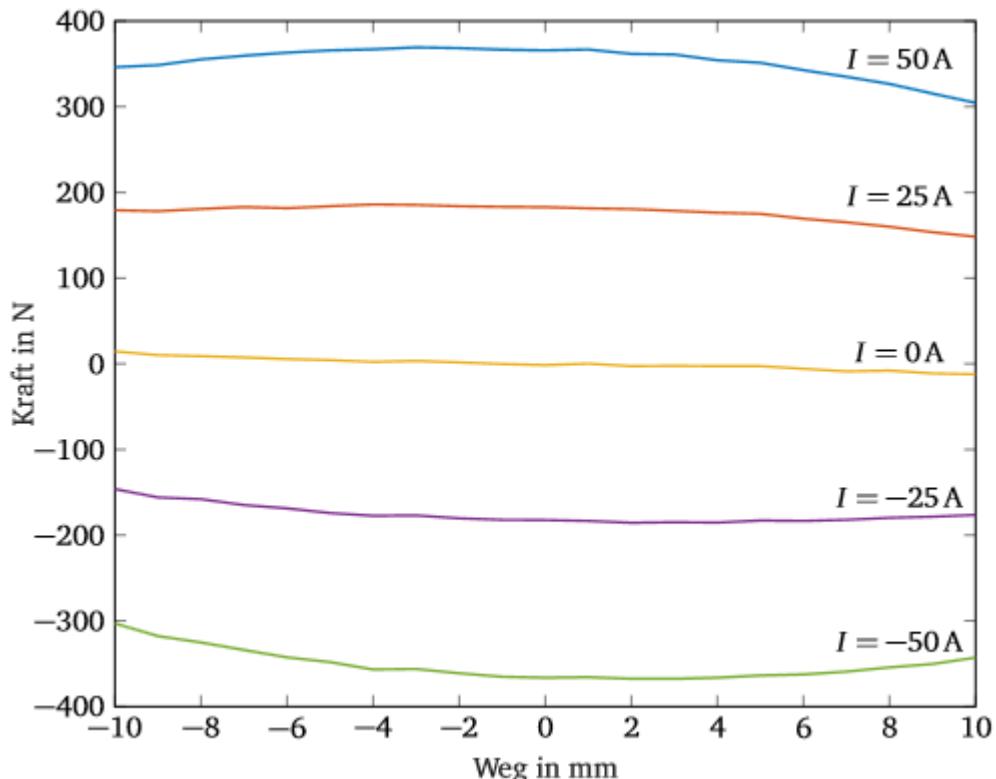
Zur Aktoransteuerung wird ein Arduino IBT2 Motortreiber verwendet, die Stromversorgung erfolgt über ein Manson SBC-2130 Battery Charger im Power Supply Mode. Der Motortreiber besteht aus zwei BTS7960 MOSFETs, die je nach Vorgabe des pulsdauermodulierten Signals (PWM-Frequenz) eine Spannung von bis zu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

2.1.5 Autobox

2.1.6 Positionsmessung und -regelung

Die Position der Läuferstange wird über einen PLCD-25M Sensor gemessen, dessen berührungslose PLCD (permanent-magnetic linear contactless displacement) Technologie die magnetische Sättigung nutzt. 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 nun die Position der Sättigungszone entlang der Sensorachse über eine induzierte Spannung bestimmt werden.

Der Sensor wird an die MicroAutoBox zwecks Spannungsversorgung sowie zur Durchführung der Kalibrierung angeschlossen. Die Kalibrierung wurde in [messtechnik] entwickelt und erfolgt bisher bei jedem Start, damit es bei unbeabsichtigten Verstellungen des Sensors nicht zu verfälschten Messergebnissen kommt. Zu Beginn des Einfahrvorgangs 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 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.



Aktor.png

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

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

Die bisherige Regelung der Schaltgabelposition wurde in einem vorangegangenem Advanced Design Project ausgetragen. 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 $K_P = 2,6\%V/mm$, $K_I = 20\%V/smm$ und $K_D = 1,6\%Vs/mm$ 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 Metalllegierung unter Einfluss von Wärme [I\ieC {\\"o}ten]. Die Metalllegierung 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: es wird unterschieden in Weichlöten (bis 450°C), Hartlöten (450°C - 900°C) und Hochtemperaturlöten (über 900°C). Im Rahmen dieses Projektes wurde das Verfahren Weichlöten mit Lötkolben angewendet, um die elektrischen Bauteile auf der Platine anzubringen. Dazu konnten die vom Institut bereitgestellten Lötstationen und -materialien verwendet werden.

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, an den in der Regel ein Quarz angeschlossen wird, sowie die

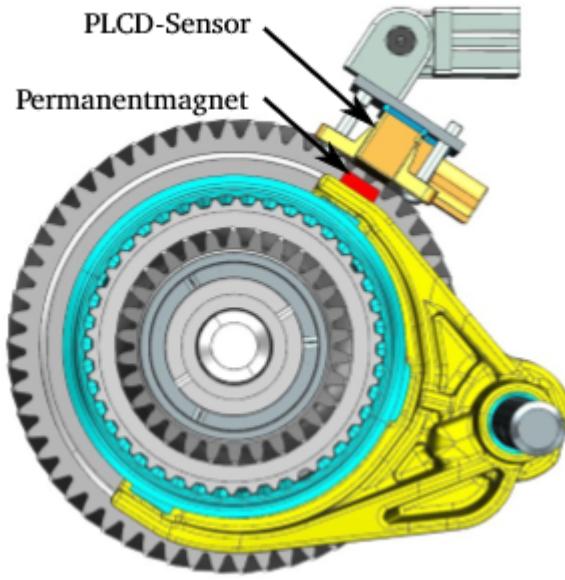


Abbildung 2.5: Einbauposition PLCD Sensor an der Schaltgabel [adp]

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 digital oder analog sein, wobei analoge Signale mithilfe eines Analog/Digital Wandlers (AD-Wandler) zuerst in digital Signale umgewandelt werden müssen, damit der Mikrocontroller sie verarbeiten kann. Eine Spezialform von Eingangskanälen 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 Taktgenerator gibt die Arbeitsfrequenz an, also wie schnell der CPU arbeiten soll. Der Arbeitsspeicher (*RAM: random access memory*) speichert temporär Daten, die aber spätestens nach dem Entfernen der Betriebsspannung gelöscht werden. 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. Der Timer hilft dabei, Anzahlen von Ereignissen zu zählen oder Zeitabstände zu messen, indem er Spannungswechsel an einem Eingangskanal zählt.

Die genaue Ausführung des Chips kann je nach Aufgabentyp variieren, sodass eine Vielzahl an verschiedenen Mikrocontrollern erhältlich ist. Diese unterscheiden sich meistens 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 Roboter, 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 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*) ist dafür da, die 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, der Aktor enthält dann die eingebettete Elektronik. Smart Actuators reduzieren beziehungsweise ersetzen die benötigte Interaktion mit einem Menschen oder externen Rechner und können somit effektiver und schneller

Schaltgabelposition.png

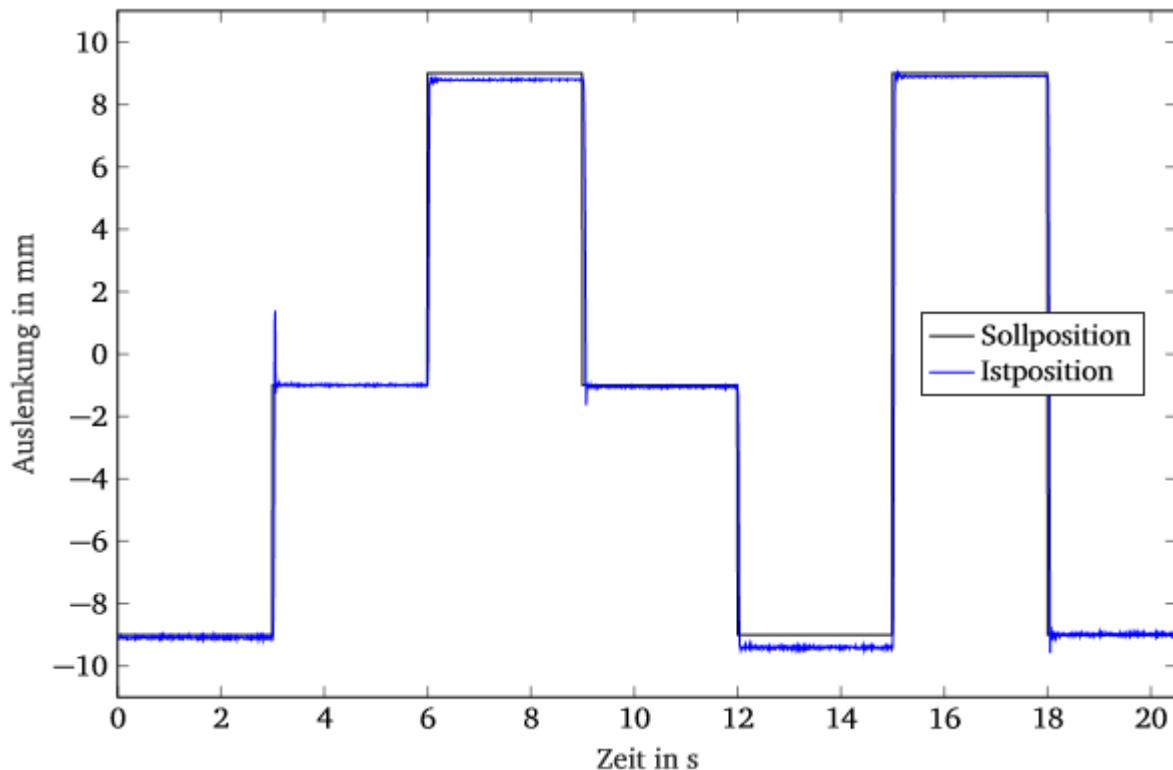


Abbildung 2.6: Sprungantwort Schaltgabelposition [adp]

ler arbeiten. Außerdem kann oft viel Bauraum eingespart werden, es gibt weniger Komponenten und deutlich weniger Verkabelung, womit der Einbau in ein Gesamtsystem leichter ist. Hervorzuheben sind weiterhin die verringerten Kosten, die Überwachung und Diagnose des eigenen Zustands sowie die verbesserte Präzision. Einbußen müssen Smart Actuators allerdings in Sachen Flexibilität, da sie oft für eine spezielle Aufgabe ausgelegt sind und ihr Verhalten stark vorgegeben ist [smartaktor].

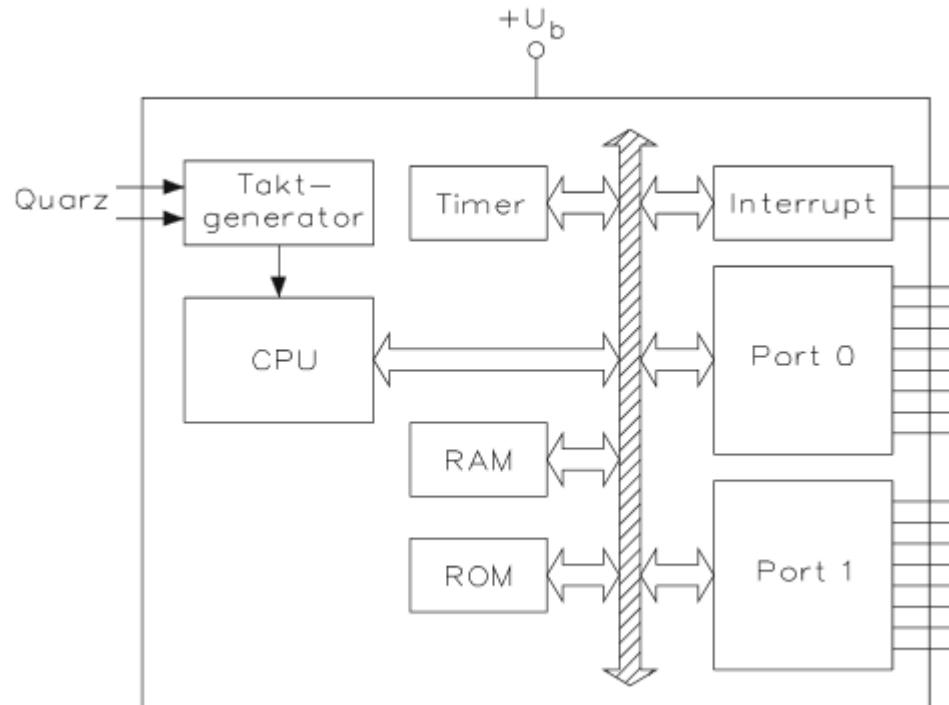
2.5 Controller Area Network (CAN)

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 (reressive) 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 zugewiesen [Lawrenz2010]. 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

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

berechnet werden.



Mikrocontroller.pdf

Abbildung 2.7: Blockschaltbild eines typischen Mikrocontrollers [Bernstein2015]

Im Folgenden wird speziell auf die in dieser Arbeit eingerichtete CAN-Kommunikation und den Aufbau der CAN-Nachrichten 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 genutzt 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 :

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

Das Tastverhältnis kann logischerweise 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 $p = 0,75$, das heißt 75% der Betriebsspannung werden an den angeschlossenen Verbraucher weitergegeben.

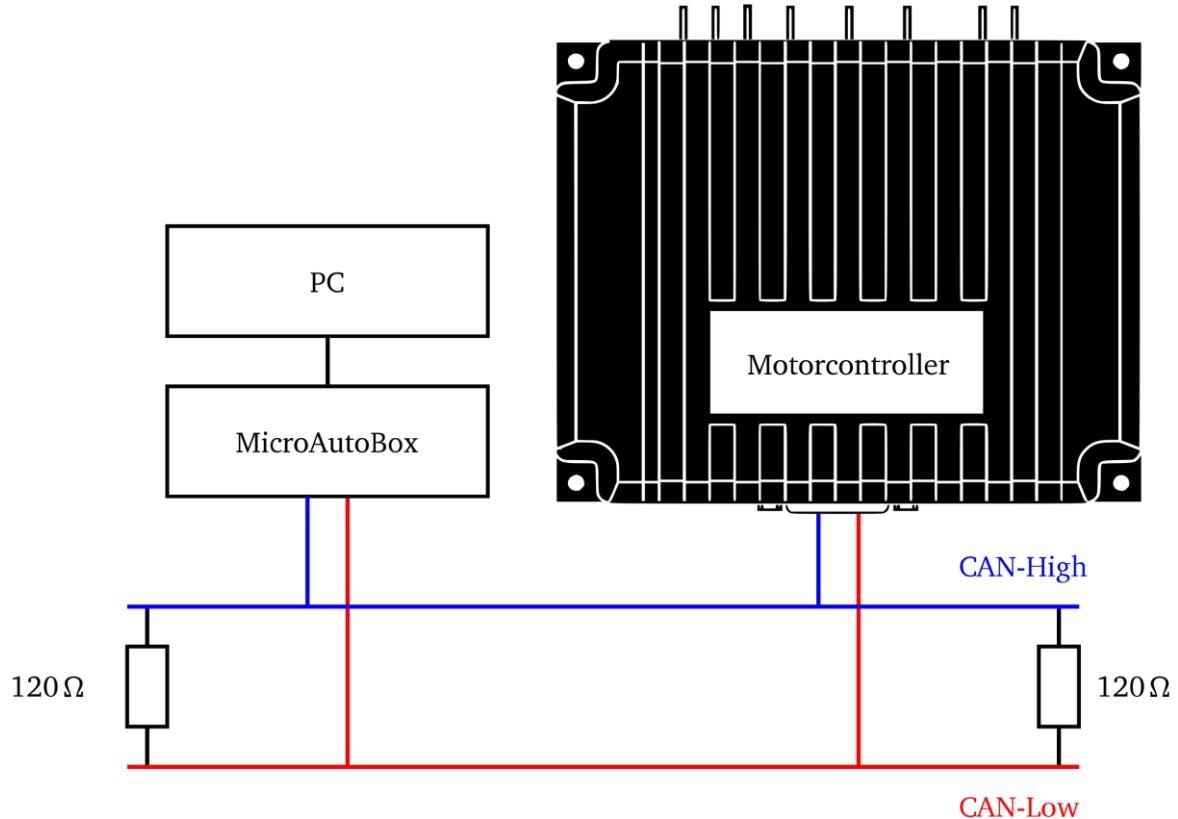


Abbildung 2.8: Aufbau CAN-Bus [manual]

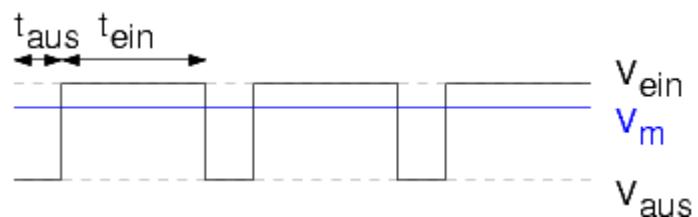


Abbildung 2.9: beispielhaftes PWM Signal

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 und
- Bereitstellung der Smart-Funktionalitäten

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 angewendete Modell entspricht dem Modell nach [Boehm 79]. 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 Tauchspulenaktuator 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 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*

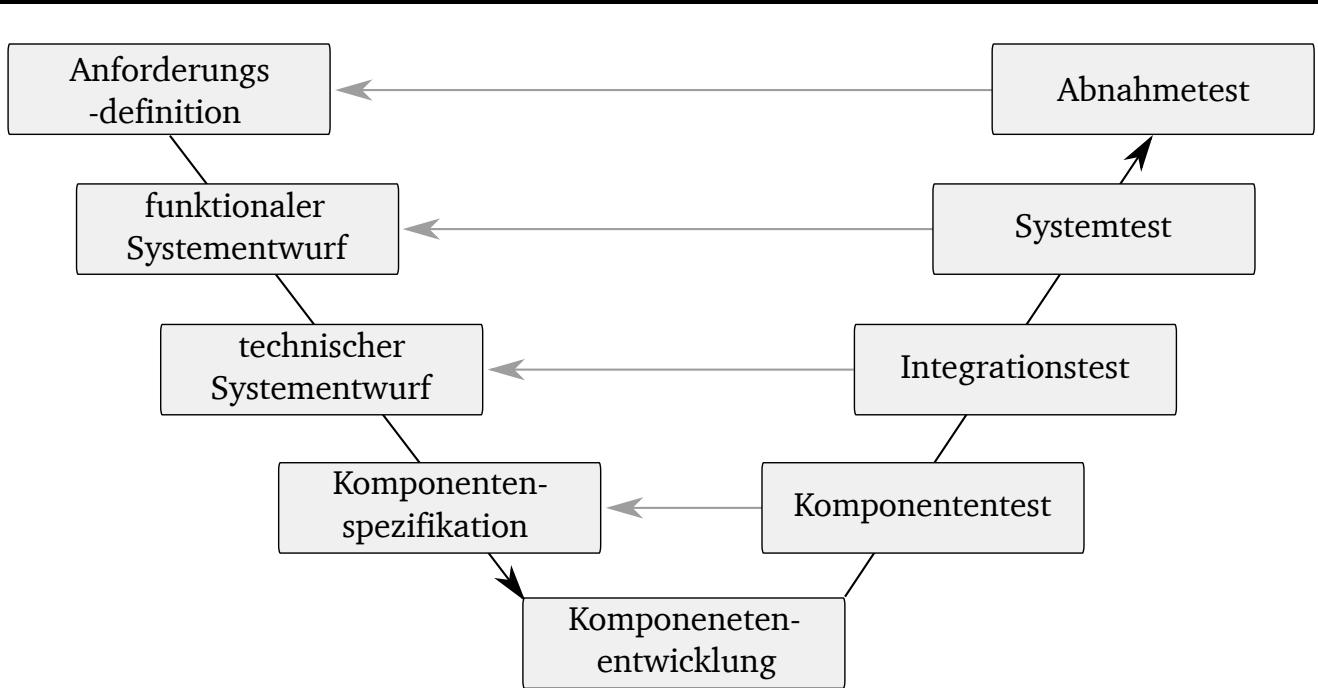


Abbildung 3.1: V-Modell nach [Boehm 79]

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 durch die H-Brücke der Motor voll in "negative" Bewegungsrichtung bestromt bzw. beschleunigt wird. Ob die Elektronik als solche den Erwartungen bzw. der Spezifikation entspricht, kann durch Strom-/Spannungs- sowie Temperaturmessungen verifiziert werden.

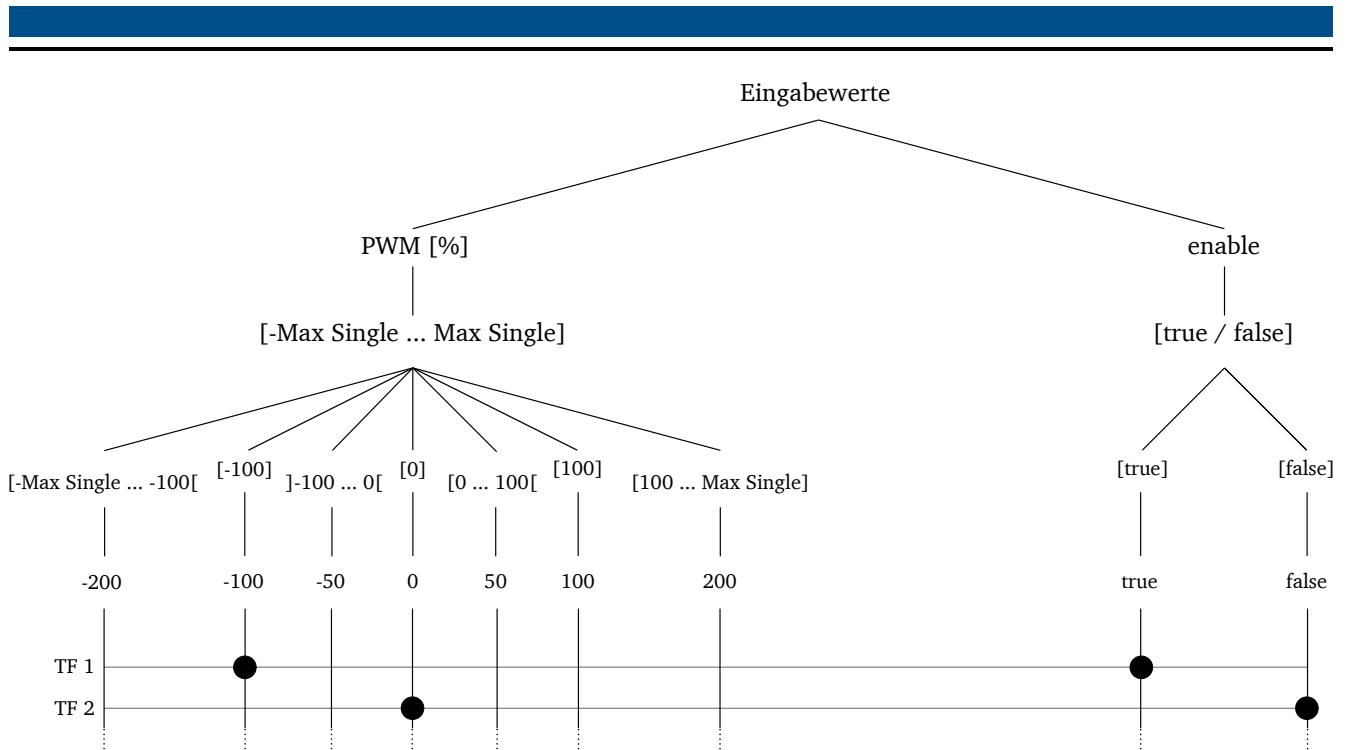


Abbildung 3.2: Klassifikationsbaum am Beispiel der H-Brücke

Für *reine Softwarekomponenten* hingegen werden Testfälle auf eine geeigneter Weise generiert. Bei Softwarekomponenten auf Basis eines Zustandsautomaten werden Testfälle aus einer zustandsbasierten Testfallgenerierung gewonnen. Dabei wird als Testziel vorausgesetzt, dass alle Entscheidungen (Knoten) 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

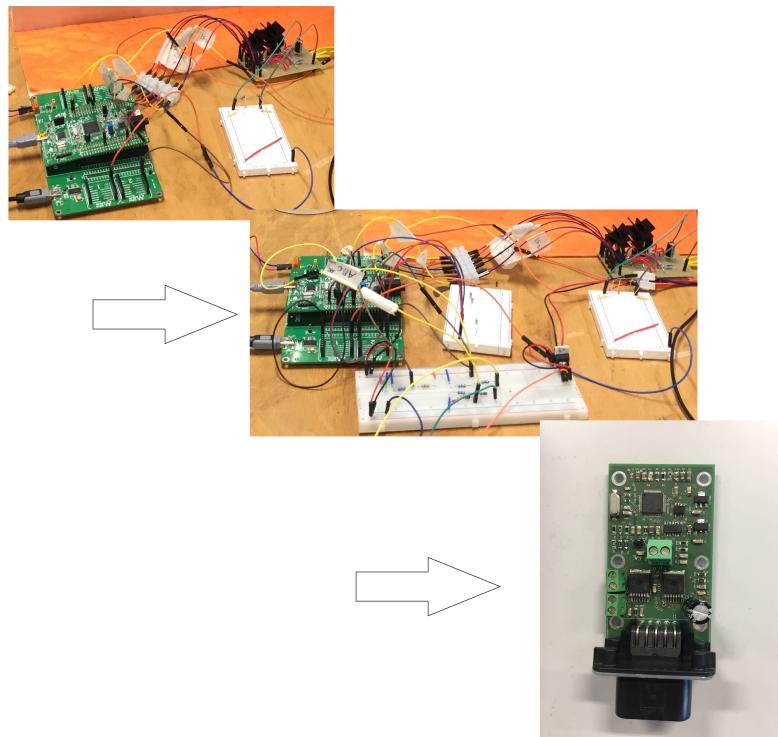


Abbildung 3.3: Bildliche Darstellung des Prototypings (Iterationsprozesses)

nach Anforderungsspezifikation durchzuführen. Dafür muss unter Anderem die Aktorik ansteuerbar sein und die Sensorknik 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 Komponentenauswahl und Schaltungsdesign

4.1 H-Brücke

Zur Ansteuerung des Aktors wird eine H-Brücke verwendet. Als Halbbrücken dienen zwei BTN8982 (Datenblatt in Anhang !!!) mit jeweils einem p-channel highside MOSFET und einem n-channel lowside MOSFET mit bereits integrierten Schutzmechanismen wie Abschaltung bei zu geringer Spannung oder Übertemperatur beinhalten. Die Halbbrücken sind für Ströme bis zu 55 Ampere sowie Spannungen bis zu 40 Volt ausgelegt und sind temperaturbeständig bis 150° Celsius. Das Blockdiagramm dieser Halbbrücken ist in nachstehender Abbildung dargestellt. In dem Blockdiagramm sind bereits

Halbbruecken.jpg

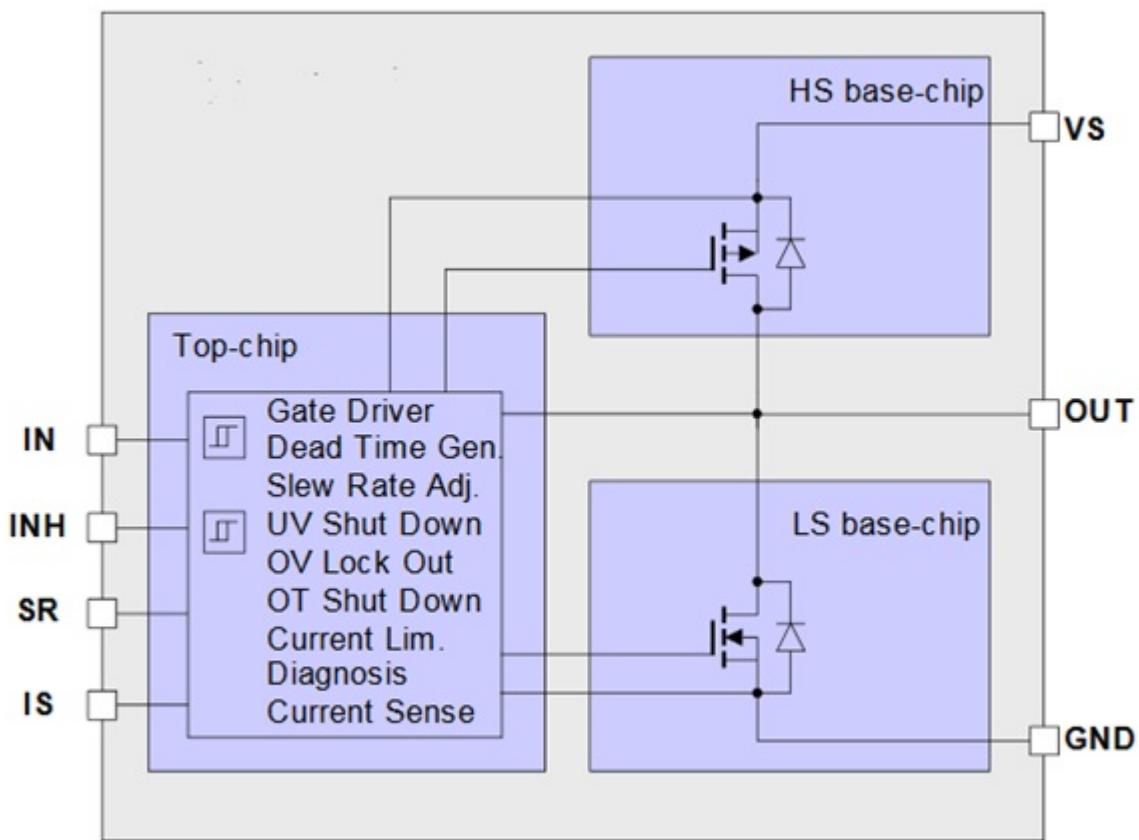


Abbildung 4.1: Blockdiagramm Halbbrücken

die PINs der Halbbrücken zu erkennen. Diese sollen nun in Tabelle 4.1 nochmal aufgezählt und ihre Funktionen erklärt werden.

Der Status Pin liefert dabei für eine Halbbrücke im High Switch Mode eine zum fließenden Versorgungsstrom proportionale Spannung, für eine Halbbrücke im Low Switch Mode keine Spannung oder Strom und im Fehlbetrieb eine konstante unabhängige Spannung.

Die Brückenschaltung orientiert sich an der Schaltung des zuvor verwendeten Motortreibers sowie an der vorgeschaerten Schaltung aus dem Datenblatt der Halbbrücken und erlaubt, dass sich der Motor je nach Durchschalten in beide Richtungen drehen kann. Im ersten Entwicklungsschritt wurde die geplante Schaltung auf einem Steckbrett getestet und anstatt des Aktors zwei LEDs in die Brückenschaltung eingebaut. Die Schaltung wurde an die jeweiligen Pins des Mikrocontrollers angeschlossen und testweise mit einer 8 Volt Batterie versorgt. Mit einem Testskript wurde die Funktionalität

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	kurzgeschlossen
6	IS (Status)	Strommessung & Fehlererkennung	I-Pin MCU
7	VS (Supply)	Stromversorgung	Batterie

Tabelle 4.1: Pinverteilung Halbbrücken

nachgewiesen und erste PWM-Signale zur Einstellung unterschiedlicher Helligkeit der LEDs gesendet. Für den ersten allgemeinen Prototypen wurde die geplante Schaltung auf einer Lochrasterplatine verwirklicht, die mit THT- Bauteilen (*through hole technology*) bestückt wurde, welche anschließend verlötet wurden. Beim Test dieses Prototypen wurden einige Fehlerquellen erkannt, die durch die neuen Erkenntnisse behoben werden konnten. Dazu gehörte BLABLA

4.2 Spannungsversorgung

Die komplette Elektronik wird auch weiterhin mit dem bisher verwendeten Manson SBC-2130 Battery Charger versorgt. Dieser stellt eine konstante Spannung von 13,8 Volt. Da die verschiedenen Komponenten jedoch Versorgerspannungen von 3,3 Volt und 5 Volt benötigen, muss die Schaltung durch einen Spannungsregler erweitert werden. Zusätzlich werden dadurch Schwankungen in der Eingangsspannung geglättet. 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 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.

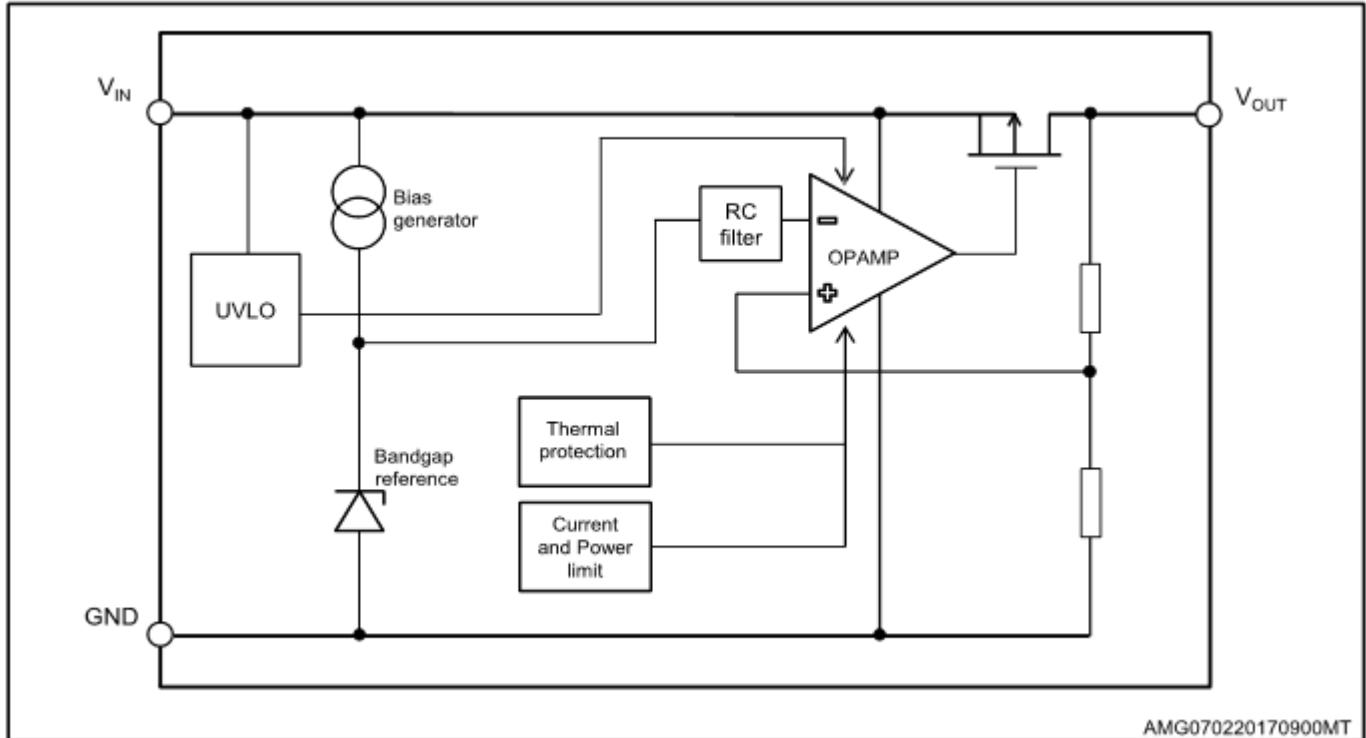


Abbildung 4.2: Blockschaltbild LDO

Der Differenzverstärker vergleicht die Ausgangsspannung mit einer stabilen Referenzquelle aus einer Zenerdiode, so dass diese 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. Die verwendeten LDOs besitzen außerdem eine Strombegrenzungsschaltung und eine Schutzschaltung, die die Betriebstemperatur überwacht und das Bauteil vor thermischer Überlastung schützt, sowie eine Unterspannungsabschaltung.



5 Platinendesign



6 Software



7 Analyse und Performance

7.1 Beurteilung der Anforderungserfüllung

7.2 Kostenaufstellung

Die Kosten 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 7.1 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.

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 7.1: Preisliste

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 bei geringen Stückzahlen die Platine 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.

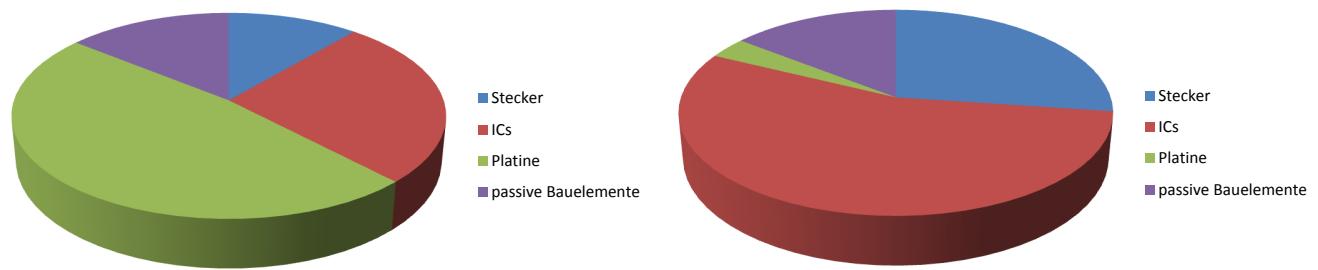


Abbildung 7.1: Aufteilung der Kosten für die Stückzahlen 1 (links) und 1000 (rechts)

8 Fazit und Ausblick



Abbildungsverzeichnis

2.1	Prüfstand [adp]	7
2.2	Fahrzeuggetriebe	8
2.3	Querschnitt Tauchspulenaktor [Hahn2018]	9
2.4	Kraft-Weg-Strom Kennlinien des Tauchspulenaktors [adp]	10
2.5	Einbauposition PLCD Sensor an der Schaltgabel [adp]	11
2.6	Sprungantwort Schaltgabelposition [adp]	12
2.7	Blockschaltbild eines typischen Mikrocontrollers [Bernstein2015]	13
2.8	Aufbau CAN-Bus [manual]	14
2.9	beispielhaftes PWM Signal	14
3.1	V-Modell nach [Boehm 79]	16
3.2	Klassifikationsbaum am Beispiel der H-Brücke	17
3.3	Bildliche Darstellung des Prototypings (Iterationsprozesses)	18
4.1	Blockdiagramm Halbbrücken	19
4.2	Blockschaltbild LDO	20
7.1	Aufteilung der Kosten für die Stückzahlen 1 (links) und 1000 (rechts)	28