Produktentwicklung 2

Hochschule Luzern

Technik & Architektur

Brushless DC Treiber

Konzeptbeschreibung

Autoren:
Yves Studer
Daniel Winz
Ervin Mazlagić

Projektgruppe:
PREN-ET

Horw
15. Juni 2015

Inhaltsverzeichnis

1 Fachgruppe Elektrotechnik

Elektrotechnik-Studierende aus mehreren Teams schliessen sich zusammen, um gemeinsame Probleme anzugehen. Dabei handelt es sich um die benötigte Hard- und Software, um Motoren anzusteuern und gegebenenfalls zu regeln. In diesem Zusammenschluss werden drei Gruppen gebildet, um Lösungen für DC-, Stepper- und Brushless-Motoren auszuarbeiten. Die Idee besteht darin, dass nicht jede Gruppe für dasselbe Problem womöglich denselben Lösungsansatz verfolgt, sondern die Ressourcen kombiniert, Synergien nutzt, um eine bessere Lösung zu erarbeiten. Auf diese Weise kann das teamübergreifende Arbeiten im Rahmen des PREN erlernt und geübt werden. Somit wird der Idee der Interdisziplinarität im erweiterten Sinn Rechnung getragen. Die Gruppen und deren Mitglieder sind in Tabelle 1 aufgeführt.

Team	Mitglied	Github-Name	DC	BLDC	Stepper
27	Daniel Winz	daniw		•	•
32	Yves Studer	ystuder		•	
33	Flavio Kreiliger	Flavinsky	•		•
38	Bettina Wyss	BettyET			•
39	Ervin Mazlagić	ninux	•		

Tabelle 1: Übersicht der PREN-ET Projektgruppen

Für den Austausch und die Ablage von Daten und Unterlagen wird die Plattform Github ausgewählt, da damit via Git[1] versioniert werden kann. Dazu wird auf Github die Organisation PREN-ET gegründet. Diese ist unter `https://github.com/PREN-ET` einsehbar. für die einzelnen Projekte werden Repositorys angelegt. Diese sind in Tabelle 2 aufgeführt.

Repository	Link	Beschreibung
info	`https://github.com/PREN-ET/info`	Allgemeine Informationen zur Organisation von PREN-ET
doc	`https://github.com/PREN-ET/doc`	Dokumentationen
dc	`https://github.com/PREN-ET/dc`	Treiber für Gleichstrommotoren
bldc	`https://github.com/PREN-ET/bldc`	Treiber für Brushless Motoren
stepper	`https://github.com/PREN-ET/stepper`	Treiber für Schrittmotoren
frdm	`https://github.com/PREN-ET/frdm`	Beispiele zur Ansteuerung mittels FRDM-KL25Z

Tabelle 2: Übersicht der PREN-ET Repositorys

2 Brushless Motoransteuerung

2.1 Theorie der Ansteuerung

Brushless-Motoren (BLDC-Motoren) sind Synchron-Drehstrom-Maschinen. Das bedeutet, sie werden mittels eines kontinuierlichen magnetischen Drehfeldes in Bewegung gesetzt. Dabei ist darauf zu achten, dass der Läufer dem Drehfeld synchron folgen kann, daher auch die Namensbezeichnung des Motors. Falls der Läufer dem Drehfeld nicht folgen kann, wird keine Spannung vom Rotor in die Statorwicklung induziert, die der Erregerspannung entgegenwirkt. Daraus folgt, dass ein immenser Strom fliesst, der nur von der Wicklungsimpedanz des Motors begrenzt wird. Das Drehmoment ist abhängig vom Polradwinkel und erreicht sein Maximum bei einem Polradwinkel von 90°. Die Kommutierung wird entweder als Sinuskommutierung oder als Blockkommutierung ausgeführt. Die Sinuskommutierung bildet die Sinusform eines dreiphasigen Netzes nach. Die Sinusform kann auch mit einem Rechtecksignal angenähert werden. Dabei spricht man von einer Blockkommutierung. Diese ist einfacher zu realisieren, aber erreicht nicht das selbe Drehmoment wie eine Sinuskommutierung. In diesem Projekt wird aufgrund der einfacheren Implementierung eine Blockkommutierung realisiert.

[1] Verteiltes Versionskontrollsystem

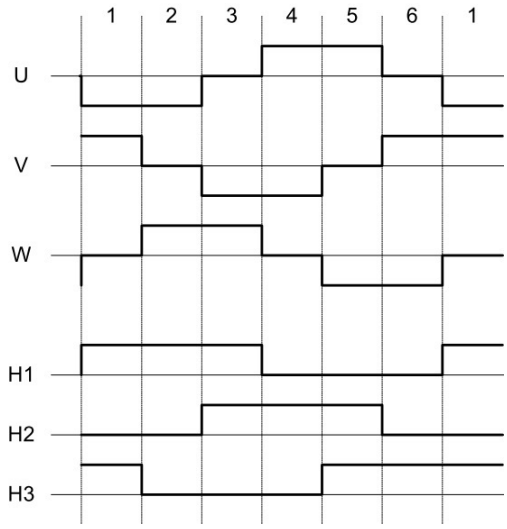

Abbildung 1: Zeitliche Darstellung der Ansteuerung mit Hall-Sensoren (Atmel Corporation, 2013)

Es gibt hauptsächlich drei Methoden, das Drehfeld zu generieren und zu regeln. Die einfachste Methode ist die Zwangskommutierung: Dabei wird ein Drehfeld erzeugt und dem Motor aufgezwungen. Der Läufer muss dem Drehfeld folgen, der maximal zulässige Winkel von 90° zwischen dem Feld und dem Läufer muss eingehalten werden. Wird dieser Winkel überschritten, kommt der Motor zum Stillstand.

Die zweite Methode zur Regelung des Motors verwendet drei Hallsensoren, die im Motor integriert sind. Dies macht den Motor aufwändiger und dementsprechend teurer. Die Regelung mit Hallsensoren ist verhältnismässig einfach, da nach den Signalen die einzelnen Spulen direkt angesteuert werden können. Der Zusammenhang zwischen der Ansteuerung und den Hallsensor-Signalen ist in Abbildung 1 ersichtlich. Dabei stehen U, V und W für die Phasenströme und H_1, H_2 und H_3 für die entsprechenden Signale der Hallsensoren. Aus dieser Darstellung ist ersichtlich, dass wenn ein Hallsensor eine Änderung anzeigt, ein Nulldurchgang im entsprechenden Stromverlauf stattgefunden hat. Dies ist der Zeitpunkt, zu dem die Kommutierung durchgeführt werden muss.

Für die dritte Möglichkeit bildet man einen virtuellen Sternpunkt und detektiert mit Komparatoren die Sternpunktdurchgänge. In der Controller-Logik muss der Zeitunterschied der Kommutierung bis zum Durchschreiten des Sternpunktes gemessen werden. Diese Zeit muss noch einmal abgewartet werden, bevor die Kommutierung durchgeführt wird. Falls der Motor über einen Anschluss für den Sternpunkt verfügt, kann auch dieser verwendet werden. Da die Position des Rotors auf diese Weise beim Stillstand nicht ermittelt werden kann, muss der Motor mit einer Zwangskommutierung gestartet werden.

2.2 Neuer Ansatz

In einem modifizierten Ansatz wird versucht, die Hallsensor-Signale aus den Ansteuerungen des Motors zu gewinnen. Hierzu wird eine Schaltung pro Phase benötigt, um die Nulldurchgänge beim virtuellen Sternpunkt detektieren zu können. Die Abbildung 2 zeigt die Schaltung, mit der dieser Ansatz realisiert werden kann. Mit dem Flip-Flop aus Abbildung 2 kann die PWM aus dem Sensorsignal unterdrückt werden. Diese rekonstruierten Hallsensor-Signale können direkt logisch verknüpft und genutzt werden, um den Motor mit einer Dreiphasen-H-Brücke anzusteuern (Tobias Plüss, 2014). Anhand des zeitlichen Verlaufs, der aus Abbildung 1 zu entnehmen ist und der Ansteuerung einer H-Brücke, ergibt sich die Wahrheitstabelle, die in Tabelle 3 ersichtlich ist. Das Signal U_h symbolisiert den Highside[2]-Transistor der Phase U auf der H-Brücke und die Spalte U_l entspricht dem Lowside-Transistor.

Die Tabelle in Abbildung 3 kann pro Signal zu folgenden logischen Verknüpfung vereinfacht werden.

[2]Transistor, der gegen die Versorgungsspannung schaltet. Typischerweise aufwendiger anzusteuern, da am Gate eine Spannung von einigen Volt über der Versorgungsspannung angelegt werden muss.

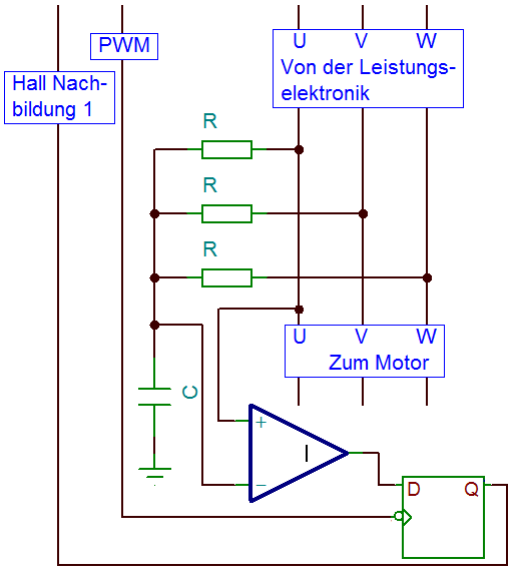

Abbildung 2: Schema des Rekonstruktionsprinzip (Tobias Plüss, 2014)

H_1	H_2	H_3	U_h	U_l	V_h	V_l	W_h	W_l	Illegal
0	0	0	0	0	0	0	0	0	1
0	0	1	0	0	0	1	1	0	0
0	1	0	0	1	1	0	0	0	0
0	1	1	0	1	0	0	1	0	0
1	0	0	1	0	0	0	0	1	0
1	0	1	1	0	0	1	0	0	0
1	1	0	0	0	1	0	0	1	0
1	1	1	0	0	0	0	0	0	1

Tabelle 3: Wahrheitstabelle der Ansteuerung

2.3 Ansteuerungshardware

In Kapitel 3.2 zeigten sich einige Punkte, die für ein eigenes Board für die Brushless-Motoransteuerung sprechen. Auf diesem Board wird ein eigener Controller eingesetzt, der als Kommunikationsschnittstelle SPI verwendet. Die benötigten Pins sind in der Tabelle 4 ersichtlich. Der Controller wird benötigt, um eine Zwangskommutierung vorzunehmen, um den Motor in eine Initialdrehung zu versetzen. Sobald der Motor dreht, kann der neue Ansatz funktionieren. Weiter wird auf dem Controller eine Regelung implementiert, um die Drehzahl des Motors zu regeln. Zusätzlich kann der Winkel zwischen Detektion und Kommutierung angepasst werden, wenn dies notwendig ist.

Die Kommunikation mit dem Controller soll über SPI geschehen. Darüber sollen die Solldrehzahl und gegebenenfalls Parameter eingestellt werden können. Alle weiteren Schritte, wie die benötigte Zwangskommutierung am Anfang und die Regelung, werden eigenständig vom Boardcontroller vorgenommen.

$$U_h = H_1 \wedge \bar{H_2} \quad V_h = H_2 \wedge \bar{H_3} \quad W_h = \bar{H_1} \wedge H_3$$
$$U_l = \bar{H_1} \wedge H_2 \quad V_l = \bar{H_2} \wedge H_3 \quad W_l = H_1 \wedge \bar{H_3}$$

Pin	Funktion
SCK	Bustakt
MISO	Master in Slave out
MOSI	Master out Slave in
CS	Chip select
Int	Interrupt
Reset	Reset

Tabelle 4: Schnittstelle des Brushless-Boards

3 Prinziptest

Das Schema des Gesamtaufbaus des Tests ist in der Abbildung 3 ersichtlich. Die 3-Phasen H-Brücke im oberen grünen Rechteck wird direkt vom FPGA[3] angesteuert. Die Hardware dieser Brücke ermöglicht eine voll galvanisch getrennte Ansteuerung mit 3.3V Logikpegeln. Diese Brücke wurde zur Verfügung gestellt und direkt verwendet. Die Rekonstruktion der Hallsensoren-Signale findet im rot markierten Teil des Aufbaus statt. Dieser Part wird auf einer Laborplatte aufgebaut und gelötet. Die so generierten Signale $U_{Hallsensor}$, $V_{Hallsensor}$, $W_{Hallsensor}$ werden einem FPGA geliefert. Anhand dieser Signale steuert das FPGA die H-Brücken-Transistoren mit den Signalen $U_h, U_l, V_h, V_l, W_h, W_l$. Die im FPGA enthaltene Konfiguration besteht aus simplen AND-Verknüpfungen, die die anliegenden Signale sehr schnell und effizient verarbeiten können. Auf diese Weise ist es möglich, den Motor sehr schnell anzusteuern. In der Abbildung 4 ist der gesamte Aufbau abgebildet. Man beachte die markierten Felder.

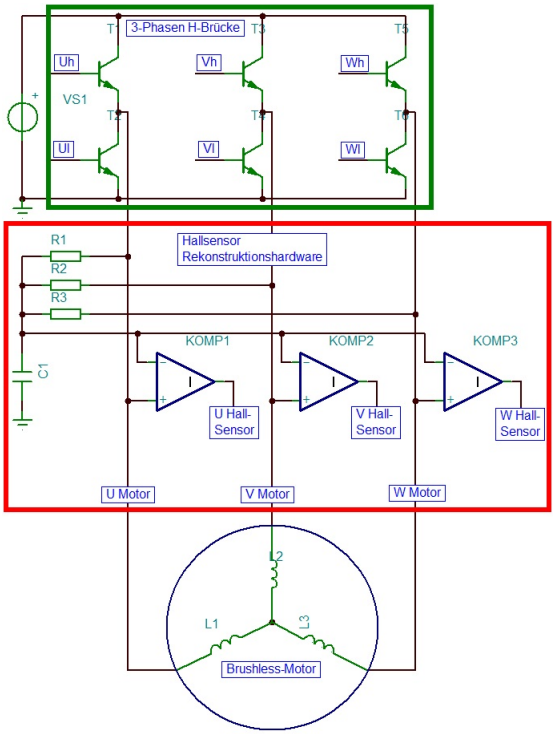

Abbildung 3: Schema des Brushless-Versuchsaufbaus

Am linken unteren Rand ist der Motor befestigt. In der Mitte des Bildes ist die Hardware zur Rekonstruktion der Hallsensoren-Signale. Die generierten Signale werden dem FPGA in der unteren rechten Ecke zugeführt. Diese Signale werden logisch verknüpft und danach die sechs Signale generiert, um die H-Brücke in der rechten oberen Hälfte anzusteuern. Die H-Brücken wiederum treiben den Motor an. Die im FPGA enthaltene Logik basiert auf der Wahrheitstabelle, die in Tabelle 3 abgebildet ist.

3.1 Messmittel

Gerät	Typ	Nummer
Speisegerät	Rohde & Schwarz NGSM 32/10	Inv.-Nr. 009
Oszilloskop	Agilent MSO6052A	Inv.-Nr. 48; S/N: MY44001903
Mainframe	Hameg HM8001-2	SN: 059520046
Speisegerät	Hameg HM8040-3	SN: 015405014
Pulsgenerator	Hameg HM8035	Inv.-Nr. 44

Tabelle 5: Messmittel des Versuchsaufbaus

[3] **F**ield-**P**rogrammable **G**ate **A**rray

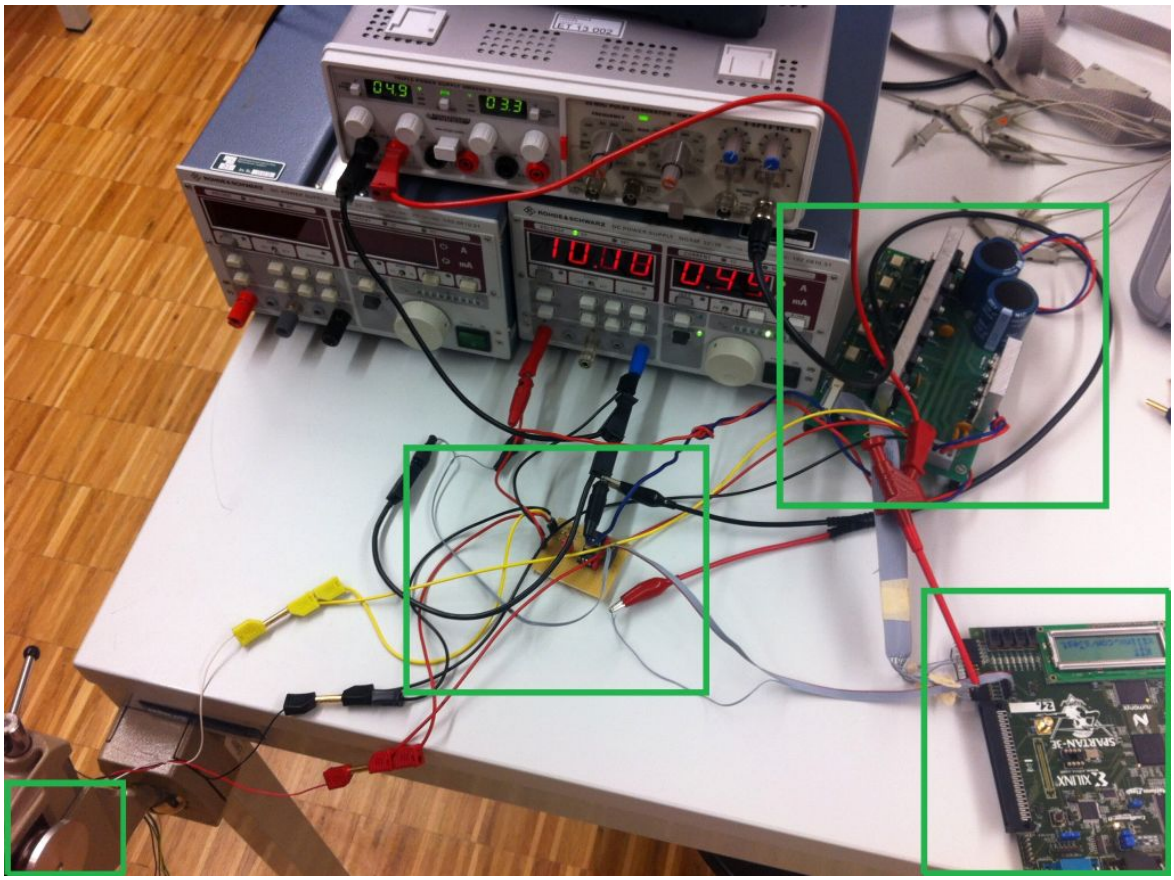

Abbildung 4: Testaufbau

3.2 Resultat

Mit dem beschriebenen Aufbau konnte ein BLDC-Motor erfolgreich angesteuert werden. Wie in Abbildung 4 am linken unteren Rand zu erkennen ist, ist an der Motorwelle eine Aluminiumplatte montiert. Mit dieser und einem Magneten konnte der Motor mittels einer Wirbelstrombremse belastet werden. Auf diese Weise konnte rund 120W elektrische Leistung umgesetzt werden. Dabei stellte sich heraus, dass die PWM nachgeregelt werden muss, wenn eine Last getrieben wird. Weiter bietet der Aufbau, so wie er getestet wurde, keine Möglichkeit den Motor ohne äussere Manipulation zu starten.

Diese beiden Tatsachen sprechen dafür, dass das Prinzip grundsätzlich funktioniert. Für die Realisierung würde sich ein eigenes Board anbieten, auf dem ein eigener Controller die Regelung und die Zwangskommutierung beim Starten des Motors übernimmt.

4 Hardware

4.1 Übersicht

Der Aufbau der Ansteuerung der BLDC-Motoren ist als Blockschaltbild in Abbildung 5 ersichtlich.

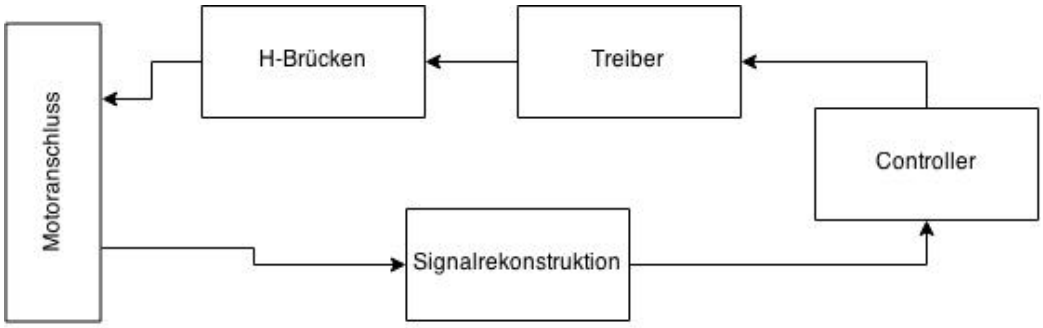

Abbildung 5: Blockschaltbild des BLDC-Boards

In der Abbildung 6 ist die Realisierung der Blöcke ersichtlich. Dabei ist der erste Kasten der Anschluss des Motors, der zweite die H-Brücken, der dritte der Treiber, der vierte die Signalrekonstruktion und der fünfte Teil ist der Controller, auf dem die Firmware läuft. Das ganze Schema des Boards ist im Anhang A angefügt.

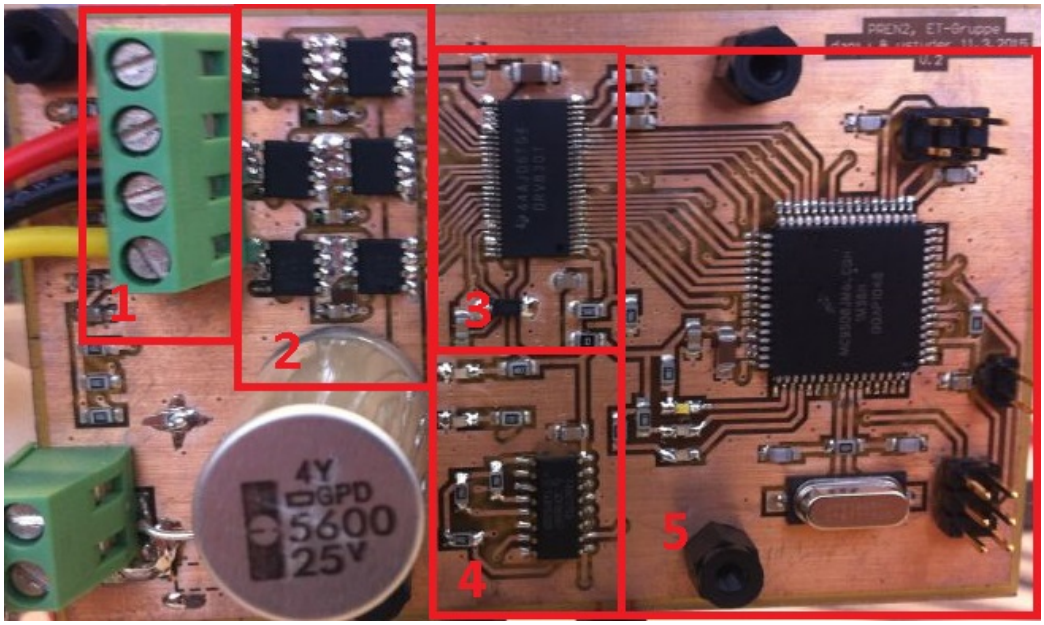

Abbildung 6: Blockschaltbild des BLDC-Boards

4.2 Kommutierung

Der Motor wird mit drei Halbbrücken kommutiert. Für die Ansteuerung der MOSFET[4] wird ein Predriver DRV8301 von Texas Instruments eingesetzt. Die Halbbrücken werden einzeln vom Controller geschalten. Damit nur ein PWM-Signal erzeugt werden muss, sind die Ansteuerleitungen mittels Widerständen mit dem PWM-Signal verbunden. (siehe Abbildung 7) Sind die Ansteuerungen vom Controller nicht aktiv getrieben, liegt die PWM an. Die Konfiguration der Predrivers erfolgt über SPI.

[4]**M**etal **O**xide **S**emiconductor **F**ield **E**ffect **T**ransistor

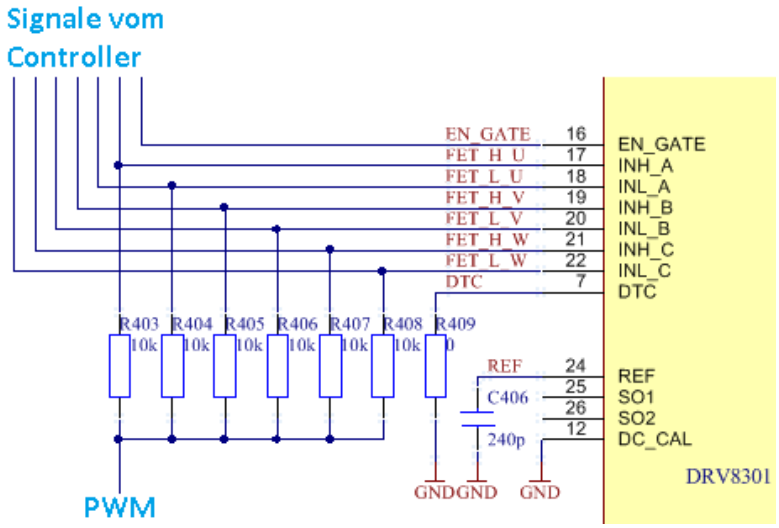

Abbildung 7: Schema-Ausschnitt der PWM-Verteilung

4.3 Phasendetektion

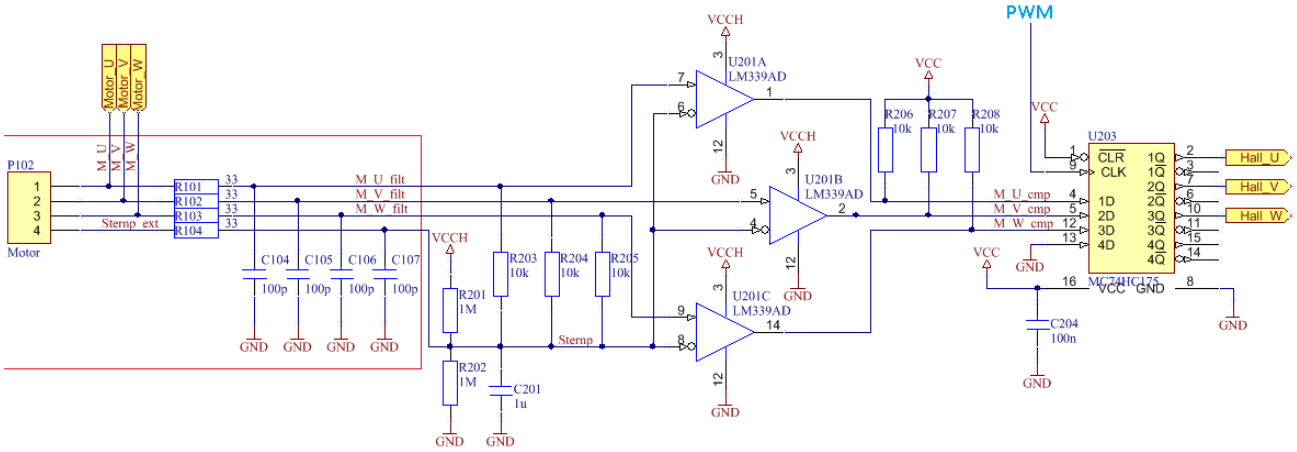

Abbildung 8: Schema-Ausschnitt der Phasendetektion

Die Phasendetektion besteht aus Sternpunktnachbildung (R201 - R205 in Abbildung 8), Komparator (U201) und Synchronisation (U203). Die Sternpunktnachbildung erzeugt einen virtuellen Sternpunkt mit Hilfe eines Widerstandsnetzwerkes und eines Tiefpassfilters. Besitzt der verwendete Motor einen herausgeführten Sternpunkt, so kann auch dieser verwendet werden. Die einzelnen Phasenspannungen werden mit diesem virtuellen Sternpunkt verglichen. Mit einem Flipflop wird das Signal mit der PWM synchronisiert.

4.4 Controller

Als Controller kommt ein HC9S08JM60 der Firma Freescale zum Einsatz. Der Takt wird mittels Quarz mit 12MHz erzeugt. Um den aktuellen Betriebsszustand anzuzeigen, stehen drei LED zur Verfügung.

4.5 Speisung

Die Eingangsspannung wird mittels eines optional bestückbaren Filters von Murata gefiltert. Der Predriver wird direkt mit dieser gefilterten Speisung betrieben. Mit dem im Predriver integrierten Step-Down-Converter wird eine Spannung von 3.3V erzeugt. Für Varianten mit hoher Betriebsspannung (>30V) wird der Komparator der Phasendetektion über einen diskret aufgebauten Spannungsregler mit einer Spannung von 30V versorgt.

5 Firmware

5.1 Übersicht

Die Firmware des Boards erfüllt mehrere Aufgaben. Während dem Anfahren des Motors realisiert sie die Zwangskommutierung, im Nennbetrieb regelt sie die Drehzahl des Motors. Sobald der Motor in Betrieb ist, wir die Kommutierung mittes Interrupts hardwaremässig realisiert.

5.2 Takt

Der Referenztakt wird durch einen Quarz mit einer Frequenz von 12MHz erzeugt. Dieser Takt dient als Referenztakt für die PLL[5] des HS9S08JM60. Dazu wird er durch acht geteilt um im erlaubten Bereich[6] für die PLL zu liegen. In der PLL wird der Takt mit 32 multipliziert. Dies ergibt einen CPU Takt von 48MHz. Da der Bustakt der Hälfte des CPUtakts entspricht, weist dieser eine Frequenz von 24MHz auf. Der Externe Takt (16MHz) wird zusätzlich als externer Referenztakt zur Verfügung gestellt und vom RTC verwendet. (siehe auch Abschnitt 5.3 RTC)

5.3 RTC

Um regelmässig abzuarbeitende Aufgaben zu steuern, wird ein entsprechender Takt benötigt. Dafür wird der RTC[7] verwendet. Dieser verwendet als Takt den externen Referenztakt mit einer Frequenz von 16MHz. Dieser wird mit dem Prescaler auf eine Frequenz von 16kHz geteilt. Über das Modulo Register kann eine Periodendauer im Bereich 62.5µs ... 16ms eingestellt werden. Es wird zunächst eine Periodendauer von 1ms verwendet. In der ISR[8] wird ein Flag gesetzt, welches in der Hauptschlaufe abgefragt wird.

5.4 PWM

Mit der PWM werden die Ausgangsstufen angesteuert. Über das Puls - Pausenverhältnis wird die Leistung eingestellt. Damit diese Ansteuerung jedoch nicht hörbar wird, muss der Motor mit einer PWM Frequenz oberhalb des hörbaren Frequenzbereichs des Menschen angesteuert werden. Es wird eine Frequenz von 24kHz verwendet. Für das Erzeugen der PWM wird der Timer TPM2 verwendet.

5.5 Kommutierungsverzögerung / Zeitmessung

Um den exakten Kommutierungszeitpunkt einstellen zu können und um die Zeit zwischen zwei Kommutierungen zu messen, wird ein weiterer Timer benötigt. Dafür wird TPM1 benutzt. Als Taktquelle für den Timer wird der Bustakt mit einer Frequenz von 24MHz verwendet. Dieser wird mit dem maximal möglichen Prescaler von 128 geteilt. Dies ergibt eine Frequenz von 187.5kHz und eine Auflösung von 5.33µs. Damit ist eine maximale Messdauer von 349.5ms möglich.

5.6 Kommutierung

Die Kommutierung wird mit dem Timer TPM1 realisiert. Bei der Zwangskommutierung wird der Kanal 0 verwendet. Dieser liest aus globalen Registern die Zeitdauer für die Kommutierung aus. In der ISR wird eine Zustandsmaschine ausgeführt. Diese steuert die drei Phasen des Motors entsprechend dem aktuellen Zustand an. Bei Autokommutierung wird die Zustandsmaschine basierend auf den Signalen aus der Phasendetektion (siehe Abschnitt 4.3) ausgeführt. Dazu kommen die Kanäle 3 bis 5 zum Einsatz, welche als Input Capture konfiguriert sind. So kann parallel zur Kommutierung die Drehzahl des Motors gemessen werden.

[5] phase-locked loop, eine elektronische Schaltung, die die Phasenlage und Frequenz beeinflussen kann
[6] Die Frequenz des Eingangssignals der PLL muss im Bereich 1 ... 2MHz liegen. (Freescale Semiconductor, 2009, p. 195)
[7] **R**eal **T**ime **Counter**
[8] **I**nterrupt **S**ervice **Routine**

5.7 Kommunikation zum Host

Zur Interaktion mit dem BLDC-Board wird die SPI1-Schnittstelle des µC verwendet. Dabei ist das SPI-Interface im 8 Bit Mode mit LSB-First konfiguriert. Zusätzlich zu dieser Schnittstelle ist eine IRQ-Leitung vorhanden, mit der das BLDC-Board den Host triggern kann, um auf ein Problem hinzuweisen. Die Steckerbelegung ist in Tabelle 6 ersichtlich.

Pin	Name
1	GND
2	MISO
3	CS
4	MOSI
5	CLK
6	IRQ

Tabelle 6: Steckerbelegung der SPI-Schnittstelle

Die Kommunikation zwischen BLDC-Board und Host funktioniert über ein Protokoll zur Interaktion. Die Spezifikation dieses Protokolls ist in der Tabelle 7 ersichtlich. Das obere Nibble des CMD's[9] enthält den Befehl und das untere Nibble die Anzahl Argumente, die zum CMD gehören. Wenn das untere Nibble `0xF` ist, wird die Länge der Übertragung im nächsten Byte signalisiert.

Name	Wert	Beschreibung	Parameter
Dummy	`0x00`	Byte das benötigt wird, um zu clocken für die Übertragung von Argumenten	
Start	`0x10`	Startet den Motor	
Stop	`0x20`	Stoppt den Motor	
setRPM	`0x32`	16 Bit Zahl um die Drehzahl einzustellen	1. Byte = High-Byte 2. Byte = Low-Byte
setVoltage	`0x42`	U_{GS} der FET's.	1. Byte = Spannungswert-High-Byte 2. Byte = Spannungswert-Low-Byte
setCurrent	`0x51`	Wert der Strombegrenzung. Der Stromwert ergibt sich nach der Formel Current = Wert $\cdot$ 10	Registerwert = $\frac{\text{Sollwert in [mA]}}{10}$
getStatus	`0x64`	Gibt der Board-Status zurück	1. Byte = Motor-Status 2. Byte = Fehler-Code 3. Byte = RPM-High-Byte 4. Byte = RPM-Low-Byte
areYouAlive	`0x71`	Damit kann die Kommunikation und das BLDC-Board testen	Das BLDC-Board gibt `0x55` zurück
setPwm	`0x81`	Damit kann die PWM des Motors eingestllt werden	PWM-Wert im Bereich 1-100 %
startMessung Param	`0xC3`	Messung parametrisiert starten	1. Byte = Pulsdauer 2. Byte = RPM-High-Byte 3. Byte = RPM-Low-Byte
startMessung	`0xD0`	Messung mit einem Schritt starten	
getMessung	`0xEF`	gibt die gespeicherte Messung zurück	1. Byte = Länge 2. - n. Byte = Daten der Messung

Tabelle 7: Kommunikationsprotokoll

5.8 Regler

In der Software ist ein PID-Regler mit Vorsteuerung implementiert. Dies ist im Blockschema in der Abbildung 9 abgebildet. Je nach Wahl der Parameter P, I, D und $F_{\text{Vorsteuerung}}$, können nur einzelne Teile des Reglers

[9] Command-Byte, ein spezielles Byte, das zur Signalisation von Befehlen verwendet wird

verwendet werden. Auf diese Weise kann zum Beispiel ein PI-Regler realisiert werden. Als Feedback wird die Zeit, die für die Kommutierung bereits besprochen wurde, verwendet. Damit kurzfristige Unregelmässigkeiten gedämpft werden, wird das Feedback durch ein FIR-Filter[10] gefiltert. Der Ausgang dieses Filters wird mittels eines Korrekturfaktor in RPM umgerechnet. Dieser Wert wird mit der Sollgrösse verrechnet und dem Regler zugeführt. Die Stellgrösse am Ausgang des Reglers wird begrenzt. Eine Begrenzung der Stellgrösse wird mit einer LED angezeigt.

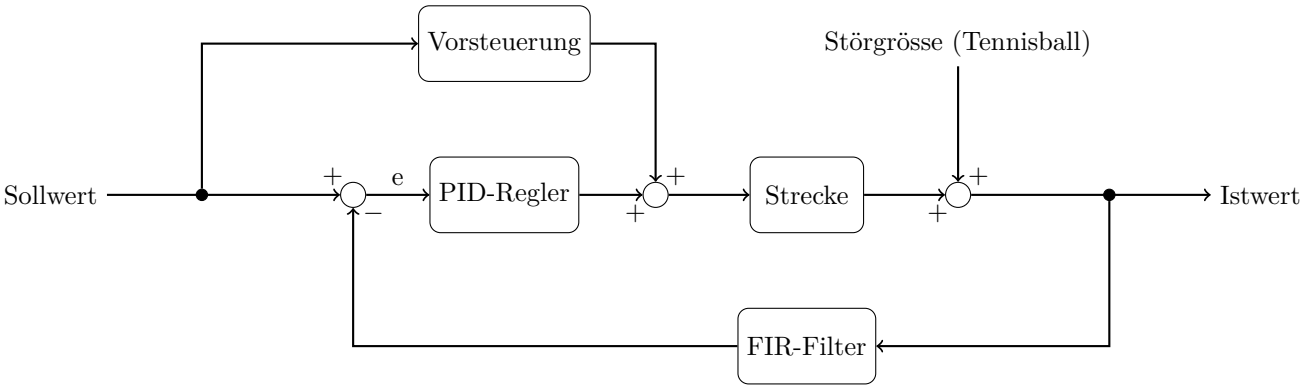

Abbildung 9: Blockschema der Regelung

[10] **F**inite **I**mpulse **R**esponse **F**ilter, Ein Filter mit garantierter endlicher Antwort auf einen Impuls

6 Fallback

Ist der Einsatz des vorgesehenen BLDC-Treibers nicht möglich, so muss eine alternative Ansteuerung erfolgen. Eine solche kann mit einer handelsüblichen Steuerung aus dem Modellbau erfolgen. Eine solche BLDC-Steuerung ist per PWM angesteuert, wobei die im Modellbau üblichen Signale gelten, wie in der Abbildung 10 dargestellt.

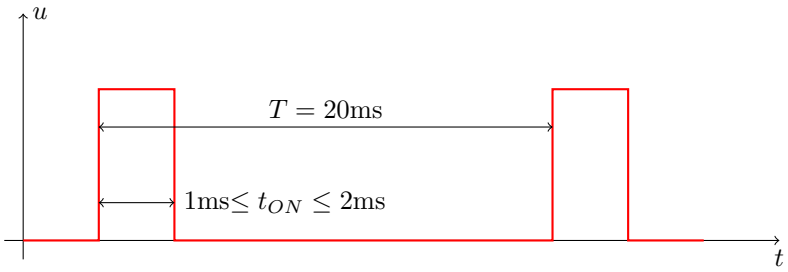

Abbildung 10: Signalverlauf eines typischen Modellbau-PWM Signals

Der Einsatz von Modellbausteuerungen für BLDC-Motoren erfordert ein Feedback der Drehzahl, da diese lediglich eine Steuerung darstellen. Die Drehzahlregelung muss über eine externe Einheit erfolgen, beispielsweise einen Mikrocontroller. Solche BLDC-Steuerungen werden im Modellbau typischerweise als *Regler* bezeichnet und sind auch für hohe Leistungen durchaus preiswert.

6.1 Konzeptbeschreibung

Um eine Regelung der Drehzahl des BLDC-Motors zu ermöglichen, bedarf es eines Feebacks, welches die Drehzahl wiedergibt. Dies ist mit einem Hall-Effekt-Schalter zu realisieren. Dieser reagiert auf die Magnetfelder, welche durch Magnete auf dem Rotationskörper gegeben sind. Aus solch einem Aufbau resultiert ein Feedback, welches mit Impulsen einen Segmentdurchlauf des Rotationskörpers wiedergibt, wie in Abbildung 11 dargestellt. Dieses Feedback wird mittels eines Mikrocontrollers ausgewertet und regelt damit den Input der Steuerung mit

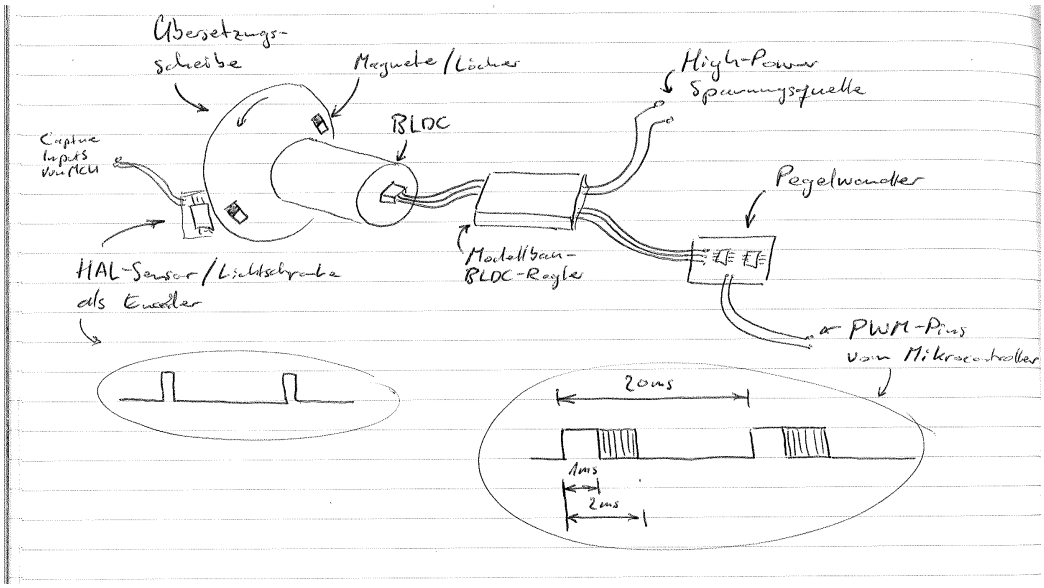

Abbildung 11: Erste Skizze des Fallback-Konzepts

dem PWM-Signal, beziehungsweise der Impulsdauer. Das Einlesen einer Flanke, die Zeitmessung bis zur nächsten Flanke und die Stellung eines PWM-Signals, sind Tasks, welche übliche Mikrocontroller direkt durch ihre Peripherie-Module ausführen können. Dies ermöglicht eine einfache Adaption in ein bestehendes Modell, denn es werden lediglich zwei Timer-IO für diesen Fallback verwendet. Je nach Mikrocontroller ist ein Pegelwandler für die PWM-Signale notwendig.

7 Encoder & Drehzahlgeber

Die vorgesehenen Motorfunktionen verlangen lediglich beim Brushlessmotor nach einem Feedback über die Rotation des Motors, da der Schrittmotor definiert und fein granuliert betrieben wird. Der Gleichstrommotor stellt keinerlei Ansprüche, weder an die Drehzahl, noch an die Position.
Encoder sind relativ teuer und der Einsatz des Brushlessmotors verlangt lediglich nach einem Feedback zur Rotation beziehungsweise Winkelgeschwindigkeit. Die absolute oder relative Position ist für die Anwendung nicht von Bedeutung. Somit lässt sich ein einfaches Feedback vorsehen, für die Regelung der Drehzahl mit optischen oder magnetischen Elementen. Als optisches Messinstrument kann eine Lichtschranke mit Reflexionsstreifen

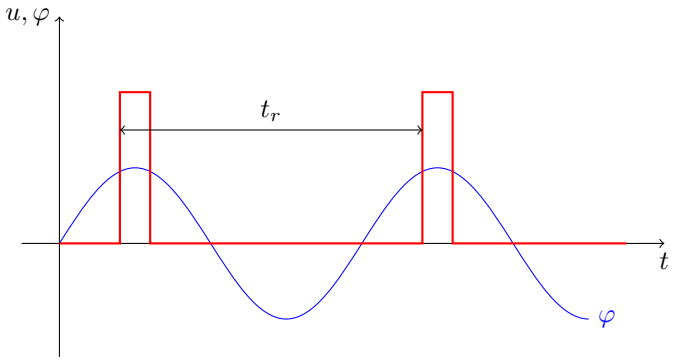

Abbildung 12: Vereinfachtes Puls-Feedback eines Hall-Effekt-Schalters

oder Löchern eingesetzt werden. Diese verlangen nur nach einer geringfügigen Modifikation des rotierenden Körpers und sind relativ günstig. Optische Messtechnik hat den Nachteil, das Störungen relativ leicht in die Messung einfliessen können, was fatale Folgen für die Regelung hat. Magnetische Messinstrumente sind gegenüber Störungen deutlich resistenter, da hierfür starke Magnetfelder benötigt werden, welche so nicht einfach auftreten. Der Einsatz einer solchen Messtechnik verlangt jedoch nach einer Modifikation der Mechanik, da Magnete in den rotierenden Körper eingebaut werden müssen. Dies birgt ein gewisses Risiko für mechanische Unwucht des Rotationskörpers.

7.1 Magnetischer Drehzahlgeber

Um einen eigenen magnetischen Drehzahlgeber zu erstellen wird ein sogenannter Hall-Effekt-Schalter eingesetzt. Dieser reagiert mit seinem Ausgang auf ein auftretendes Magnetfeld. Das Gegenstück zum Hall-Effekt-Schalter ist ein Magnet, welcher in das rotierende Objekt eingebaut wird. Aus mechanischen Gründen, wie etwa der Unwucht, werden typischerweise 2 Magnete oder ein Vielfaches davon in den rotierenden Körper eingebaut.
Bei der Rotation des Körpers entstehen durch das Passieren der Magnete am Hall-Effekt-Schalter Impulse. Aus diesen Impulsen lässt sich mit einer Zeitmessung direkt die Drehzahl bestimmen. Die Abbildung 12 illustriert das Prinzip anhand eines Beispiels mit einem Magneten am Rotationskörper. Ein solches Verfahren lohnt sich bei schnellen Winkelgeschwindigkeiten und ist für diesen Anwendungsfall sehr effizient. Zugehörige Hall-Effekt-Schalter lassen sich einfach montieren und sind gegen Störungen sehr robust. Ein mögliches Modell für einen Hall-Effekt-Schalter ist der AH180N. Dieser bietet einen Open-Drain Ausgang, welcher somit logische Pegel liefert (siehe Abbildung 13). Interessant ist diese Art von Drehzahl-Geber insbesondere durch ihren geringen Preis, denn solche Hall-Effekt-Schalter, wie der AH180N, befinden sich im Preissegment von unter einem Franken.

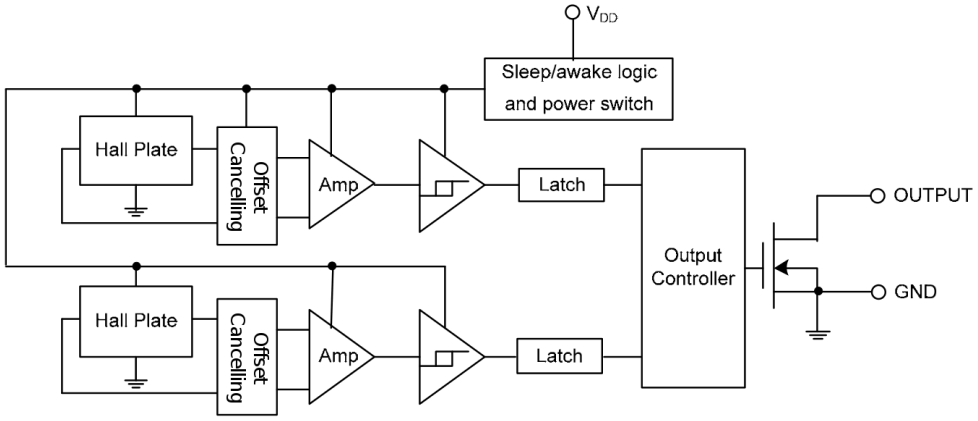

Abbildung 13: Funktionelles Blockschaltbild des Hall-Effekt-Schalters AH180N (Diodes Incorporated, 2012)

Literatur- und Quellenverzeichnis

Atmel Corporation. (2013). *Atmel AVR443: Sensor-based Control of Three Phase Brushless DC Motor* (App. Note). 1600 Technology Drive, San Jose, CA 95110 USA: Autor. (2596C-AVR-07/2013)

Diodes Incorporated. (2012). *AH180N, MICROPOWER OMNIPOLAR HALL-EFFECT SWITCH* (Datasheet). 4949 Hedgcoxe Road, Suite 200, Plano, TX 75024 USA: Autor.

Freescale Semiconductor. (2009). *MC9S08JM60 Series Data Sheet* (Datasheet). 2100 East Elliot Road, Tempe, Arizona 85284: Autor. (Rev. 3)

Tobias Plüss. (2014, November). Hochschule Luzern, Technik und Architektur. (In einem Gespräch)

Abbildungsverzeichnis

Tabellenverzeichnis

A Schema des BLDC-Boards