



Fachhochschule Nordwestschweiz
Hochschule für Technik

Fachbericht Projekt 4 (EIT) - Team 2

Skatemate

Team 2:

Raphael Brügger
Rafael Eberle
Noah Hütter
Mike Mössner
Selina Reich
Remo Suter

Auftraggeber:

Hans Gysin

Coaches:

Hans Gysin
Matthias Meier
Pascal Schleuniger
Anita Gertiser
Bonnie Domenghino
Pascal Buchschacher

Abstract

The Commute is an electrically supported long board. Because of rising trend of electric mobility, this product is intended for the daily commuter to reach his destination. With an innovative steering device called Magic Glove, the board's slim design and light weight, the Commute provides great usability. The Magic Glove uses an ATmega328 microcontroller to convert the movement of a finger, which is measured by an integrated bending strip, into a digital signal and send it to the long board. To reduce the chance of an accident caused by connecting wires, the steering glove communicates wirelessly over a 2.4GHz module. In addition, this connection is used as a theft protection and also prevents the board from rolling away unintentionally. To control the starting acceleration, the Commute uses the new motor control technology Field Oriented Control or FOC. With the implementation, the board recycles energy gained from braking. The Commute also has a charging circuit which charges the internal Lithium Polymer battery by simply attaching the charging cable. To provide a longer battery life, the charging circuit measures the cell voltages so they stay balanced. The charging state is visualized by three LEDs, charging, ready and fully charged. An AtMega328 microcontroller charges the battery, balances the cells and controls the charging current flowing into the cells. The Magic Glove has been tested by our developer and works as expected. It enables a seamless driving experience without the need of a remote.

Keywords: Electric mobility, Bending Strip, Field Oriented Control, Cell balance, Long board

Inhaltsverzeichnis

1 Einleitung	1
2 Grobkonzept und technische Grundlagen	2
2.1 Grobkonzept	2
2.2 Technische Grundlagen	5
2.2.1 Flex-Sensor	5
2.2.2 Balancing	6
2.2.3 BLDC-Motor	6
2.2.4 FOC	7
2.2.5 Leistungsstufe	10
2.2.6 Funkübertragung	11
3 Hardware	12
3.1 Überblick	12
3.2 Steuerung - Magic Glove	12
3.3 Stromversorgung	13
3.4 Motoransteuerung	18
4 Software	20
4.1 Überblick	20
4.2 Steuerung - Magic Glove	20
4.3 Stromversorgung	22
4.4 Motoransteuerung	26
5 Validierung	34
5.1 Überblick	34
5.2 Brett	34
5.3 Steuerung - Magic Glove	34
5.3.1 Hardware	35
5.3.2 Software	36
5.4 Stromversorgung	37
5.4.1 Hardware	37
5.4.2 Software	39
5.5 Motoransteuerung	40
5.5.1 Hardware	41
5.5.2 Software	42
5.6 Gesamtvalidierung	46
6 Schlusswort	47
Literaturverzeichnis	49
A Anhang	50
A.1 Bedienung	50
A.2 Quellcode	50
A.3 Schemati	50

1 Einleitung

Die elektrische Mobilität ist auf dem Vormarsch. Sie ermöglicht eine geräusch- und emissionsarme Fortbewegung. Dabei werden die technischen Möglichkeiten je länger je vielfältiger, sodass immer mehr und preiswertere elektrische Fortbewegungsmittel angeboten werden. Die Elektromobilität erreicht auch immer mehr den Freizeitbereich. Zur Weiterentwicklung der Elektromobilität soll im Rahmen dieses Projektes ein elektrisches Rollbrett entwickelt werden, das im Freizeitbereich attraktiv ist. Gefordert ist ein innovatives Steuerungskonzept als auch eine effiziente Ansteuerung des Motors. Das Rollbrett als Endprodukt hat zum Ziel, im Freizeitbereich attraktiv zu sein und einen Beitrag an die Entwicklung der Elektromobilität zu leisten. Weitere Anforderungen sind, dass der Antrieb ausgeschaltet werden kann. Wenn niemand auf dem Rollbrett steht, muss dies aus sicherheitstechnischen Gründen sogar automatisch geschehen. Zudem soll das Rollbrett auch ohne elektrischen Antrieb benutzt werden können. Für die Alltagstauglichkeit muss das Rollbrett witterfest sein, so auch die Steuerung selbst.

Die Lösung des Entwicklerteams Skatemate ist das Longboard Commute. Das Konzept dazu wurde speziell für Pendler entwickelt. Obwohl die Motorisierung immer mehr Bereiche durchdringt, sind die Möglichkeiten für Pendler bisher stark eingeschränkt. Diese Lücke soll das Produkt Commute schliessen. Als Kriterien ergibt sich dadurch, dass das Longboard gut tragbar, mobil zu laden und einfach zu steuern ist. Insbesondere soll die Steuerung keine zusätzlichen Fahrkenntnisse benötigen, also auch für Laien tauglich sein.

Deshalb ist das Longboard mit einer praktischen Steuerung ausgestattet und mit einem schlanken Design versehen. Dadurch kann es problemlos überallhin mitgenommen werden, um kürzere Strecken schnell und elegant zu bewältigen. Somit wird dem Pendler ermöglicht, den letzten Abschnitt von dem Bahnhof oder der Busstation zum Ziel elektrisiert zurückzulegen.

Die Steuerung erfolgt über die Bewegung des Zeigefingers (Geschwindigkeit) und über Gewichtsverlagerung (Richtung). Je nach Krümmung des Fingers beschleunigt oder bremst der Motor. Die technische Umsetzung erfolgt über einen Flex-Sensor.

Der Motors wird mithilfe der feldorientierten Regelung (field oriented control FOC) angesteuert. Dadurch kann ein sanftes Anfahren realisiert werden. Zudem gibt die FOC einen interessanten Einblick in eine Regelungstechnik, die auch bei industriellen Motoren angewendet wird.

Damit der Akku nicht für jedes Laden von dem Longboard gelöst werden muss, wird ein eigener, direkt am Deck integrierter Akkulader entwickelt, so dass das Longboard direkt über ein Ladekabel mit einer Steckdose verbunden werden kann. Dazu wird ein Balancing-System entwickelt.

Dieser Bericht stellt die technische Dokumentation des elektrischen Longboards Commute dar. Das Kapitel Grobkonzept und technische Grundlagen zeigt das Grobkonzept des Projektes auf und erklärt die benötigten technischen Grundlagen für das Verständnis des Projektes. Dies beinhaltet die Funktionsweise des Flex-Sensors, des Balancing, der feldorientierten Regelung, der H-Brücke und der Funkübertragung. In den Kapiteln Hardware und Software wird die Ausführung im Detail präsentiert. Das Kapitel Validierung gibt Auskunft darüber, wie das System getestet wurde.

2 Grobkonzept und technische Grundlagen

Dieses Kapitel beginnt mit dem Grobkonzept des Longboards Commute, um einen Überblick über das Elektroskateboard zu geben. Anschliessend werden die notwendigen technischen Grundlagen für die Hard- und Software erklärt.

2.1 Grobkonzept

Das Projekt Commute von Skatamate kann in drei Grundbereiche unterteilt werden: Die innovative Steuerung, die Motorregelung und die Stromversorgung.

In den folgenden Unterkapitel werden diese drei Bereiche und die dazugehörigen Begriffe detaillierter erläutert. Die intuitive Bedienung des Longboards geschieht über eine Fingerbewegung, welche dank des Magic Gloves gemessen werden kann. Der Motor wird mithilfe der feldorientierten Regelung (field oriented control FOC) angesteuert, mit dieser Regelungstechnik können Wechselgrössen elegant gesteuert werden. Die Stromversorgung verfügt über ein eigens dazu konzipiertes Akkuladegerät.

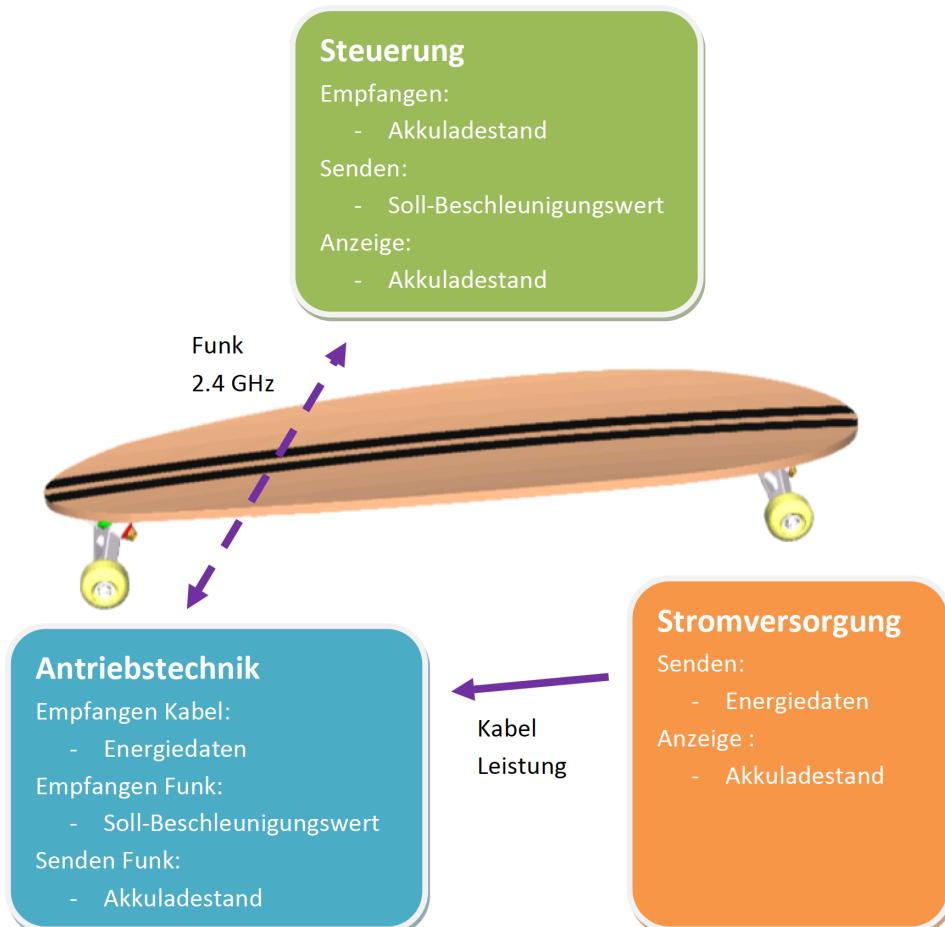


Abbildung 2.1: Blockschaltbild Grobkonzept

Die Abbildung 2.1 stellt die Interaktion zwischen den drei Bereichen dar. Die Antriebstechnik ist über ein Kabel mit der Stromversorgung verbunden, die Inputs der Steuerung erhält sie über ein Funknetz.

Um eine Übersicht über die enthaltenen Komponenten der drei genannten Bereiche zu erhalten, sind sie im Blockschaltbild der Abbildung 2.2 dargestellt.

Für die Steuerung wird die vom Flex-Sensor gemessene Krümmung des Fingers an der Mikrocontroller weitergeleitet. Dieser Verarbeitet die Daten in eine Wunschbeschleunigung und leitet sie mithilfe eines Funkmoduls an die Antriebstechnik weiter. Zudem steuert er die Front-LEDs an. Mit Strom versorgt wird der Mikrocontroller und das Funkmodul mithilfe einer Knopfbatterie.

Der Mikrocontroller der Antriebstechnik erhält die Wunschbeschleunigung wiederum über ein Funkmodul. Mit der FOC werten daraus die benötigten PWM-Spannungen berechnet. Diese werden durch eine Leistungsstufe, bestehend aus den Treiber und FETs, am Motor angelegt. Versorgt werden die Treiber und somit der Motor über die Akku-Stromversorgung.

Zwischen der Akkuspannung und den Treibern ist ein elektrischer Schalter. Dieser wird über den Mikrocontroller der Stromversorgung gesteuert. Der Mikrocontroller steuert auch den Ladevorgang, also das Balancing.

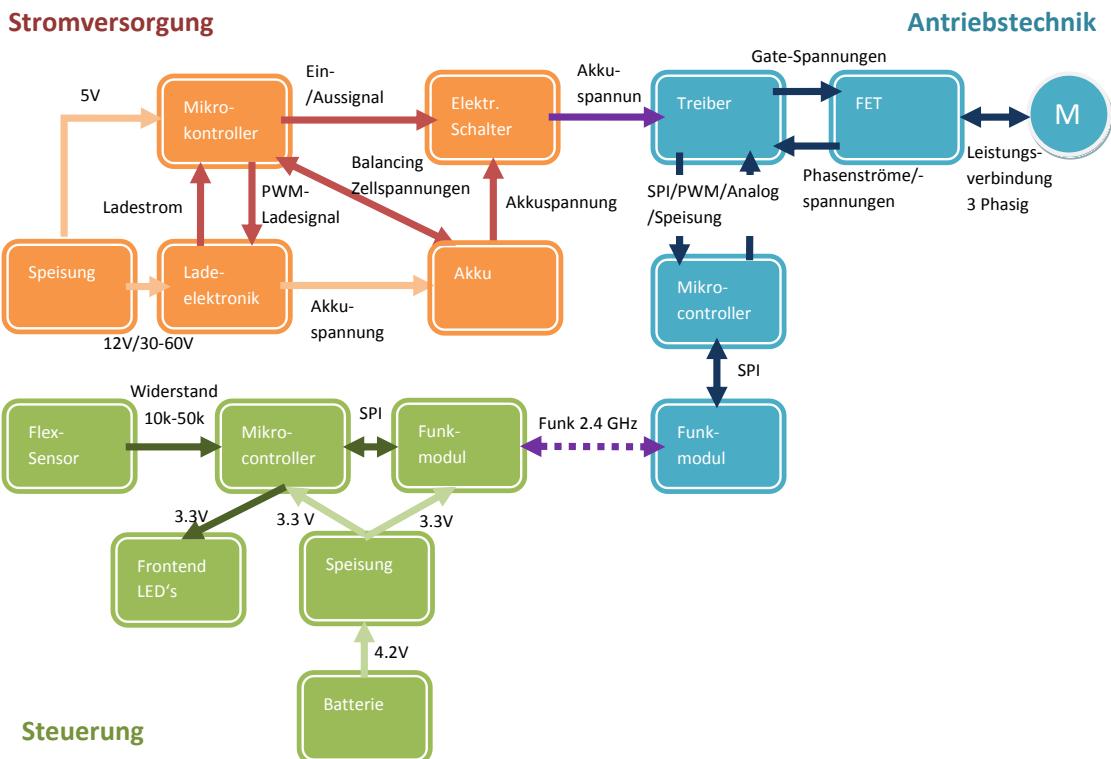


Abbildung 2.2: Detailliertes Blockschaltbild

Steuerung

Es gibt verschiedene Wege, wie ein elektrisches Longboard gesteuert werden kann. Eine Variante ist, Drucksensoren ins Brett einzubauen. Mit einer Gewichtsverlagerung in oder gegen die Fahrtrichtung wird eine Geschwindigkeitsregulation erreicht. Da auf unebenem Gelände das Gewicht laufend ausbalanciert werden muss, müssten diese Bewegungsimpulse kompensiert werden. Ausserdem können lokal begrenzte Sensoren die Bewegungsfreiheit auf dem Longboard einschränken. Aus diesen Gründen wurde diese Variante verworfen. Das Team hat sich stattdessen entschieden, die Geschwindigkeit über eine Fingerbewegung zu steuern. Dazu wird ein Handschuh mit integrierten Sensoren entwickelt. Dieses Wearable enthält einen Flex-Sensor, der die Beugung des Zeigefingers misst. Dieser Messvorgang und der Flex-Sensor werden im Kapitel 2.2.1 beschrieben. Der Mikrocontroller quantifiziert die mit dem Flex-Sensor gemessene Beugung des Fingers. Diese Sollbeschleunigung wird anschliessend an die Motoransteuerung übermittelt. Dies geschieht über ein 2.4GHz-Funknetz, dazu ist ein Funkmodul integriert. Die Stromversorgung erfolgt über eine Li-Ion Knopfbatterie LTR2450. In der Abbildung 2.2 sind die genannten Elemente der Steuerung dargestellt. Das gesamte Konzept ist im Magic Glove vereint.

Antriebstechnik

Als Antrieb ist der OX1 2-10 Motor vorgegeben. Dies ist ein Bürstenloser-Gleichstrommotor (brushless DC motor, BLDC-Motor). Er verfügt über keine Hallsensoren, mit denen die Rotorposition gemessen werden könnte. Ein BLDC-Motor ist wie ein permanentmagnetischer Drehstrom-Synchronmotor (PMSM) aufgebaut. Die technischen Hintergründe des Motors werden im Kapitel 2.2.3 gegeben.

Zur Ansteuerung gibt es grob unterteilt zwei verschiedene Arten, die Kommutierung und die Feldorientierte Regelung [1]. Die Kommutierung wiederum kann auf zwei verschiedene Arten erfolgen, ungesteuert (Blockkommutierung) oder gesteuert (Sinus-Kommutierung). Diese resultierenden drei Ansteuerungsmöglichkeiten sind in der Tabelle 2.1 zur Übersicht dargestellt und nachfolgend etwas genauer erläutert.

Regelungsart	Vorteile	Nachteile
Blockkommutierung	-einfach	-schlechte Effizienz (eine Spule nicht gebraucht) -Drehmoment-Rippel
Sinus-Kommutierung	-alle drei Spulen benutzt -kein Drehmomentrippel	-kompliziert -Anfahren erschwert
FOC	-dynamisch -effizient -durch Transformation einfacher PI-Regler	-komplex

Tabelle 2.1: Vor- und Nachteile der verschiedenen Regelungsarten

Bei der ungesteuerten Kommutierung (Blockkommutierung) wird der Motor als Schrittmotor genutzt. Dabei folgt die Rotorposition der Steuerung. Diese Variante ist für ein gleichmässig rollendes Longboard ungeeignet, da der Schrittmotor nicht durchgehend, sondern immer nur schrittweise dreht.

Für ein gleichmässiges Drehen muss die Kommutierung also abhängig von der Rotorposition erfolgen (geführte Kommutierung, Sinus-Kommutierung). Dabei reagiert die Steuerung auf die effektive Rotorposition und passt sich dieser an. Am einfachsten wäre dies mit Sensoren. Da der Motor jedoch über keine Sensoren verfügt, muss er über eine sensorlos gesteuerte Kommutierung angesteuert werden. Dies funktioniert jedoch nur ab einer Mindestdrehzahl wirklich gut. Zum Anfahren muss der Motor speziell angesteuert werden.

Bei der Feldorientierte Regelung werden die Spannungen zur Steuerung aktiv der Rotorlage angepasst. Dabei kann zwischen sensorgesteuerter und sensorloser Regelung unterschieden werden. Bei der sensorlosen Regelung muss wiederum zum Anfahren eine zusätzliche Ansteuerung erfolgen. Kann die Anfangsposition ausreichend genau geschätzt werden, läuft der Motor auch bei tiefen Geschwindigkeiten gleichmässig, da er genauer geregelt werden kann. Dies entspricht den Anforderungen für das Commute, da ein sanftes Anfahren sehr wichtig ist. Die FOC als auch die Anfahr-Ansteuerung ist im Kapitel 2.2.4 erklärt.

Ausgeführt wird die FOC auf einem Mikrocontroller. Mittels RF-Modul wird die Sollbeschleunigung empfangen. Eine Halbbrücke mit FET-Treibern (siehe Kapitel 2.2.5) setzt die gewünschte Motorsteuerung um. Zur Übersicht sind diese Elemente im Blockschaltbild Abbildung 2.2 dargestellt.

Stromversorgung

Die Stromversorgung erfolgt über einen 5.2Ah LiPo-Akku, dieser ist vorgegeben. Bei einer durchschnittlichen Leistungsaufnahme von 50W des Motors reicht eine Akkuladung ungefähr für zwei Stunden Fahrt. Auch ein externes Ladegerät ist gestellt, doch um dieses zu nutzen, müsste der Akku für jedes Laden vom Longboard gelöst werden. Das ist umständlich und gefährdet die Wetterfestigkeit und Robustheit des Commute. Deshalb wird ein eigener, integrierter Akkulaader entwickelt, so dass der Akku nicht mehr herausgelöst werden muss. Da die Entladung im Gebrauch sowieso überwacht werden muss, ist der Lader eine Ergänzung und kein alleinstehendes Element.

Kernstück der Stromversorgung ist die Balancerschaltung. Dabei wird der Akku erst mit einem konstanten Strom geladen, bis die Zellspannung erreicht ist. Anschliessend wird mit einer konstanten Spannung geladen, bis der Strom unter 100mA gesunken ist, dann sind die Zellen vollständig geladen. Wie das Balancing genau funktioniert, ist im Kapitel 2.2.2 beschrieben. Anstelle einer eigenen Implementation hätte ein fertiges IC eingekauft werden können. Aus finanziellen Gründen wurde jedoch darauf verzichtet. Wie bereits erwähnt sind die Elemente der Stromversorgung in der Abbildung 2.2 übersichtlich dargestellt.

2.2 Technische Grundlagen

In den folgenden Unterkapiteln werden die technischen Grundlagen zum Verständnis der Hard- und Software erläutert, so dass deren Funktionsweisen verstanden werden können. Dazu werden für die Steuerung der Flex-Sensor und die Funkübertragung, für die Stromversorgung das Balancing und für die Motoransteuerung der BLDC-Motor, die FOC und die Treiber und FETs erklärt.

2.2.1 Flex-Sensor

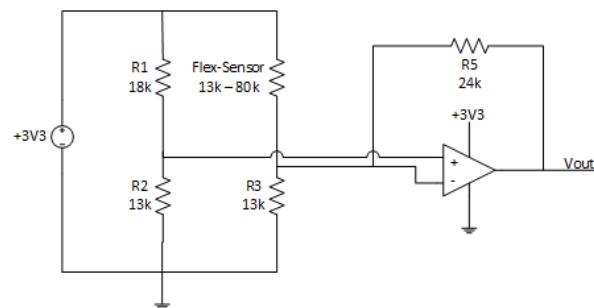
Ein Flex-Sensor eignet sich dazu, als analoges elektrisches Bauteil, eine Biegung zu messen. Im Falle dieses Projekts heisst dies, zu erkennen wie fest der Finger des Fahrers gebogen ist.

Der Sensor ändert je nach Biegung seinen Widerstandswert im Bereich von 13 bis ca. 80k Ω bei der stärksten Verformung. Das Bauteil selbst besteht aus einem flexiblen PCB mit einer darauf befestigten Schicht von metallischen Partikeln. Wird der Sensor nun gebogen, entfernen sich die Partikel voneinander, wodurch sich der Widerstand vergrössert. Biegt man den Sensor zurück in seine Ausgangsposition, verkleinert sich der Widerstandswert wieder.

Um diesen Widerstand messen zu können, wurde eine Verstärkerschaltung mit Messbrücke aufgebaut (Abbildung 2.3). Der Widerstand der als Flex-Sensor angeschrieben ist, verfügt über einen Wertebereich von 13 bis 80 k Ω . Mit den beiden Widerständen R2 und R3 wird die Untergrenze der messbaren Resistanz festgelegt. Mit R1 wird schliesslich die Skalierung der Brücke festgelegt. Die resultierenden Spannungen zwischen R1 und R2 bzw. zwischen R3 und R4 werden schliesslich durch einen Rail-To-Rail Operationsverstärker so fest verstärkt, dass sie den Spannungsbereich von 0 bis 3.3 Volt ausnutzen. Dadurch kann schliesslich der angehängte 10bit A/D-Wandler vollständig ausgesteuert werden.

Flex-Sensor gerade
Widerstand ca. 13k Ω

Flex-Sensor gebogen
Widerstand bis ca. 80k Ω



(a) Funktionsprinzip Flex-Sensor

(b) Messbrücke für Flex-Sensor

Abbildung 2.3: Funktionsprinzip und Ansteuerung für einen Flex-Sensor

2.2.2 Balancing

Das Board wird durch einen sechs Zellen-LiPo-Akku gespeist. Jede einzelne Zelle hat dabei einen eigenen Innenwiderstand, der sich mit dem Alter verändern kann. Beim Ladevorgang entstehen somit unterschiedliche Spannungen, wodurch die Zellen aus der Balance geraten. Damit sich einzelne Zellen nicht überladen, während andere leer bleiben, benötigt es einen Balancing-Vorgang um die Differenzen der Zellen auszugleichen. Dabei werden bei einer zu grossen Spannungsdifferenz alle Zellen auf die Spannung der niedrigsten Zelle entladen. Dazu wird die überschüssige Energie über den Transistor und die Widerstände verheizt.

2.2.3 BLDC-Motor

Für dieses Projekt ist der Motor OX1 2-10 zur Verfügung gestellt worden. Dabei handelt es sich um einen dreiphasigen Brushless-DC-Motor ohne Hallsensoren (Sensoren zur Rotorpositionsermittlung). Die gegebenen technischen Daten sind in der Tabelle 2.2 ersichtlich. Ein BLDC-Motor ist aufgebaut wie eine permanenterregte Synchronmaschine. Der OX1 2-10 ist als Aussenläufermotor gebaut. Die Erregung des Motors erfolgt mit 14 Permanentmagneten, er verfügt somit über sieben Polpaaren. Der Stator besteht aus 12 Spulen, also ist jede Phase vier Mal gewickelt.

In der Abbildung 2.4(a) ist das Feldverhalten dargestellt.

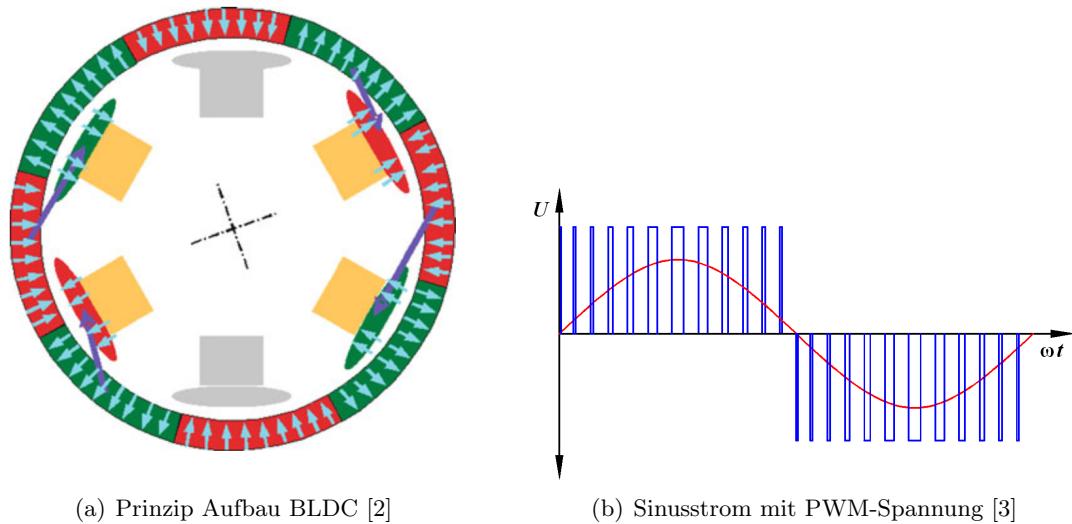


Abbildung 2.4: Aufbau BLDC und PWM-Ansteuerung

Angesteuert wird jede Phase über eine Halbbrücke. Die FOC-Regelungsweise (siehe Kapitel 2.2.4) setzt einen sinusförmigen Strom auf jeder Phase voraus, dies jeweils 120° phasenverschoben. Realisiert wird dieser Strom mittels PWM-Ansteuerung der Halbbrücken. Da der Strom bei einer L-Last das Integral der Spannung ist, lässt sich ein quasi-sinusförmiger Strom generieren. Dargestellt ist dies in der Abbildung 2.4(b).

Leerlaufstrom	1.2A
Maximalstrom	50A
Eingangsspannung	2..10 x 3.6 Lipo (25.2V)
Maximale Leistung	1815W
Maximaler Zug	6700g
Nennstrom	42.5A
Gewicht	460g
Schaft	8mm
Abmessung	$\varnothing 50 \times 65\text{mm}$
Innenwiderstand	0.0361Ω

Tabelle 2.2: Kennzahlen BLDC-Motor

2.2.4 FOC

Für die feldorientierte Regelung werden die Phasenströme des Motors gemessen und anschliessend in ein rotorfestes Bezugssystem transformiert (Clarke- und Park-Transformation). Dazu wird die Rotorlage benötigt, diese wird gemessen oder berechnet (nonlinear observer). Im rotorfesten Bezugssystem wird das Drehmoment und die Flussdichte des Motors mit einem PI- oder PID-Regler geregelt. Anschliessend werden diese Werte zurücktransformiert und für die PWM-Ansteuerung des Motors genutzt(2.2.5).

Clarke- und Park-Transformation

Das allgemeine Prinzip der Transformationen lautet: Ein kompliziertes Problem wird in ein anderes Koordinatensystem transformiert. Das transformierte Problem ist dort viel einfacher zu lösen. Die so leicht gefundene transformierte Lösung wird rücktransformiert, wodurch die Lösung des komplizierten Problems im ursprünglichen Koordinatensystem erhalten wird. Ein mögliches Beispiel dafür ist die Laplace-Transformation von Differentialgleichungen.

Das komplizierte Problem ist hier die Ansteuerung des Motors mittels einer dreiphasigen PWM. Mit der Clarke-Transformation wird in ein statorfestes Koordinatensystem gewechselt, mit der anschliessenden Park-Transformation in ein rotorfestes. Das transformierte Problem ist nun linear und mit einem einfachen PI (oder PID) -Regler lösbar.

Anschliessend wird die gefundene Lösung zurücktransformiert. Dabei kann die komplette Rücktransformation (inverse Park- und inverse Clarke Transformation) ausgeführt und anschliessend der Motor über eine PWM angesteuert werden, oder aber es wird nur bis ins statorfeste Koordinatensystem rücktransformiert (inverse Park Transformation), dann wird der Motor über eine Raumvektor-PWM (SVPWM) angesteuert.

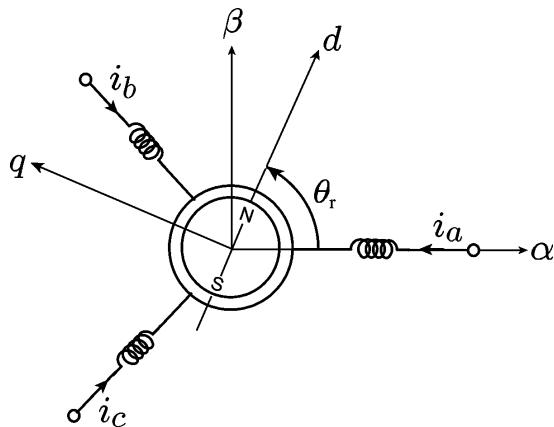


Abbildung 2.5: Schematische Darstellung einer PMSM mit den drei Koordinatensystemen [4]

Clarke-Transformation

Die Clarke-Transformation wechselt die mehr- bzw. dreiphasigen Größen (z.B. Phasenströme) in ein statorfestes Bezugssystem [5]. Das kartesische Koordinatensystem ist in der komplexen Ebene. Die drei Phasenströme I_a , I_b und I_c werden auf zwei komplexwertige Ströme I_α und I_β abgebildet. Dargestellt ist die Transformation in der Abbildung 2.5. Die mathematische Formulierung lautet:

$$\begin{bmatrix} I_\alpha \\ I_\beta \end{bmatrix} = \frac{2}{3} \begin{bmatrix} 1 & -\frac{1}{2} & -\frac{1}{2} \\ 0 & \frac{\sqrt{3}}{2} & -\frac{\sqrt{3}}{2} \end{bmatrix} \begin{bmatrix} I_A \\ I_B \\ I_C \end{bmatrix} \quad (2.1)$$

Die inverse Clarke-Transformation wird entsprechend durch folgende Gleichung beschrieben:

$$\begin{bmatrix} I_A \\ I_B \\ I_C \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ -\frac{1}{2} & \frac{\sqrt{3}}{2} \\ -\frac{1}{2} & -\frac{\sqrt{3}}{2} \end{bmatrix} \begin{bmatrix} I_\alpha \\ I_\beta \end{bmatrix} \quad (2.2)$$

Park-Transformation

Die Park-Transformation transformiert die Größen I_α und I_β in ein rotorfestes Bezugssystem [6]. Dieses neue kartesische Koordinatensystem besteht aus zwei Achsen. Von aussen gesehen (ruhendes Bezugssystem) rotiert es mit der gleichen Winkelgeschwindigkeit wie der Rotor. Eine Achse wird als d-Achse (direct axis) bezeichnet, die andere als q-Achse (quadrature axis). Die d-Achse beschreibt die magnetische Flussdichte des Rotors, die q-Achse das Drehmoment des Rotors.

Ausgehend vom statorfesten Bezugssystem benötigt es nur noch eine Rotation, um die Größen in das rotorfeste Bezugssystem zu transformieren [7]. Mathematisch gesehen geschieht die Transformation also über eine Drehmatrix, dargestellt ist die Transformation in der Gleichung 2.3.

$$\begin{bmatrix} I_d \\ I_q \end{bmatrix} = \begin{bmatrix} \cos(\theta) & \sin(\theta) \\ -\sin(\theta) & \cos(\theta) \end{bmatrix} \begin{bmatrix} I_\alpha \\ I_\beta \end{bmatrix} \quad (2.3)$$

Die inverse Park-Transformation wird entsprechend durch folgende Gleichung beschrieben:

$$\begin{bmatrix} I_\alpha \\ I_\beta \end{bmatrix} = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix} \begin{bmatrix} I_d \\ I_q \end{bmatrix} \quad (2.4)$$

d/q-Transformation

Die Clarke- und Park-Transformation kann auch in einem Schritt durchgeführt werden, je nach Quelle wird diese dann auch Park-Transformation oder d/q-Transformation genannt [7]. Die mathematische Formulierung lautet:

$$\begin{bmatrix} I_d \\ I_q \end{bmatrix} = \frac{2}{3} \begin{bmatrix} \cos(\theta) & \cos(\theta - \frac{2\pi}{3}) & \cos(\theta - \frac{4\pi}{3}) \\ -\sin(\theta) & -\sin(\theta - \frac{2\pi}{3}) & -\sin(\theta - \frac{4\pi}{3}) \end{bmatrix} \begin{bmatrix} I_A \\ I_B \\ I_C \end{bmatrix} \quad (2.5)$$

Die inverse d/q-Transformation lautet:

$$\begin{bmatrix} I_A \\ I_B \\ I_C \end{bmatrix} = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \cos(\theta - \frac{2\pi}{3}) & -\sin(\theta - \frac{2\pi}{3}) \\ \cos(\theta - \frac{4\pi}{3}) & -\sin(\theta - \frac{4\pi}{3}) \end{bmatrix} \begin{bmatrix} I_d \\ I_q \end{bmatrix} \quad (2.6)$$

PI-Regler

Der PI-Regler ist ein Regler mit einem Proportional- (P) und einem Integral- (I) Glied. Dies sind elementare Glieder der Regelungstechnik. Das P-Glied entspricht einer Multiplikation des Eingangs mit einer zeitunabhängigen Konstanten, das P-Glied einer Integration des Eingangs. Die Regelung mit einem PI-Regler geschieht im rotorfesten Bezugssystem. Gegenstand der Regelung sind die Flussdichte des Motors (d-Komponente) und das Drehmoment (q-Komponente). Bei einem permanenterregten Motor ist diese Regelung vereinfacht. In der Abb.2.5 ist ersichtlich, dass die d-Achse in Richtung des Permanentmagneten zeigt. Das grösste resultierende Moment wird erreicht, wenn die resultierende Flussdichte im Stator um 90° zur zeitlich konstanten Flussdichte des Rotors verschoben ist. Dies zeigt also genau in Richtung der q-Achse. Die resultierende Flussdichte liegt also auf der q-Achse und hat deshalb keinen d-Anteil. Der Sollwert der d-Komponente wird also null. Das Drehmoment kann somit direkt über die q-Komponente geregelt werden.

Nonlinear Observer

Die Park-Transformation benötigt die momentane Lage des Rotors. Da der vorgegebene Motor nicht über integrierte Sensoren (wie beispielsweise Hall-Sonden) verfügt, die Ansteuerung also sensorlos erfolgt, muss der Winkel der Rotorlage auf einem anderen Weg herausgefunden werden. Dazu gibt es verschiedene Varianten. Im Team wurde entschieden, die Winkelschätzung über einen nichtlinearen Beobachter, nonlinear observer genannt, vorzunehmen. Dazu wird ein mathematisches Modell des Motors erstellt. Zudem werden die Phasenströme gemessen. Mit diesen Informationen wird eine Simulation des Motors durchgeführt, und aufgrund der aktuellen Parameter wird die Position ermittelt. Dies funktioniert, weil von den Phasenströmen auf die Lage des Motors geschlossen werden kann. Beschrieben wird dieses Verfahren in [4]. Grundsätzlich wird in diesem Verfahren folgende Zustandsvariable definiert:

$$x = Li_{\alpha\beta} + \psi_m \begin{bmatrix} \cos\theta \\ \sin\theta \end{bmatrix} \quad (2.7)$$

Der nichtlineare Beobachter wird nun um die in (2.7) definierte Zustandsvariable erstellt:

$$\dot{\hat{x}} = y + \frac{\lambda}{2}\eta(\hat{x})[\psi_m^2 - ||\eta(\hat{x})||^2] \quad (2.8)$$

Mit der Vektor Funktion

$$\eta(x) = x - Li_{\alpha\beta} \quad (2.9)$$

Durch Vergleich mit (2.7) beträgt die euklidische Norm

$$||\eta(x)||^2 = \psi_m^2 \quad (2.10)$$

Durch Ableiten von (2.7) und Einsetzen der Motorengleichung (2.11) erhält man eine messbare Grösse:

$$L\dot{i}_{\alpha\beta} = -R_s i_{\alpha\beta} + \omega\psi_m \begin{bmatrix} \sin\theta \\ -\cos\theta \end{bmatrix} + v_{\alpha\beta} \quad (2.11)$$

$$\dot{x} = L\dot{i}_{\alpha\beta} - \omega\psi_m \begin{bmatrix} \sin\theta \\ -\cos\theta \end{bmatrix} = -R_s i_{\alpha\beta} + v_{\alpha\beta} \quad (2.12)$$

L	Motorinduktivität
R_s	Motor Innenwiderstand
ψ_m	Motorkonstante
$v_{\alpha\beta}$	Phasenspannungen Clarke-transformiert
$i_{\alpha\beta}$	Phasenströme Clarke-transformiert
λ	Verstärkung des Beobachters

Grundsätzlich werden nun periodisch die Ströme gemessen, clarke-transformiert und der Observer ausgeführt. Der Observer braucht dafür sehr exakte Kennzahlen des Motors wie Motorkonstante, Seriewiderstand und Induktivität. So kann der Winkel $\hat{\theta}$ geschätzt werden. Die Berechnungen sind auf dem gewählten Mikrocontroller dank FPU schnell ausgeführt.

2.2.5 Leistungsstufe

Wie in Abbildung 2.4 auf Seite 7 zu sehen ist, muss am Motor eine sinusförmige Spannung anliegen. Die FOC-Berechnungen aus Kapitel 2.2.4 ergeben zeitdiskrete Spannungswerte. Diese werden mit der Raumvektor-PWM Methode in PWM Signale umgewandelt. Die Signale werden

mit der Leistungsstufe ausgegeben. Die Leistungsstufe besteht aus drei Halbbrücken mit je zwei N-Kanal MOSFETs (Abbildung 2.6).

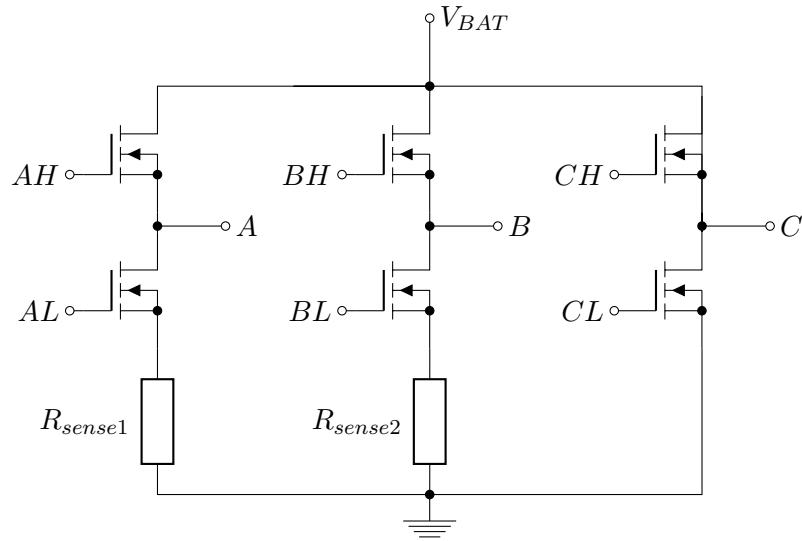


Abbildung 2.6: Leistungsstufe zur Motoransteuerung

Mit den sogenannten High-Side-MOSFETs kann jede Phase einzeln auf die Batteriespannung geschaltet werden. Die Low-Side-FETs arbeiten symmetrisch dazu und können die Phase auf Batteriemasse schalten. Durch die berechneten PWM kann so ein dreiphasiger Sinus approximiert werden. Damit nicht gleichzeitig die High- und Low-Side-FETs einer Phase eingeschaltet sind, was zu einem Kurzschluss führen würde, ist im Treiber IC eine Totzeit-Schaltung eingebaut. Wird der eine FET ausgeschaltet, sind für eine kurze Zeit beide ausgeschaltet, bevor der nächste eingeschaltet werden kann. Dieses Verfahren wird Deadtime genannt.

2.2.6 Funkübertragung

Die Funkübertragung wird mit einem RFM75-Modul ermöglicht. Dieses kommuniziert auf 2.4GHz in einem extrem einfachen proprietären Protokoll. Für dieses Projekt wurde der erste von 83 möglichen Datenkanälen genutzt. Die genaue Übertragungsfrequenz ist somit 2,401GHz. Die folgende Tabelle wurde sinngemäss aus dem Datenblatt des Funkmoduls übernommen und zeigt den Aufbau der übermittelten Pakete, wie sie für dieses Projekt aus den Einstellungen des Moduls gesendet werden.

Inhalt	Grösse
Preamble	1 Bit
Adresse	3 Byte
Payload Länge	6 Bit
Paket-ID	2 Bit
NO ACK	1 Bit
Daten	1 Byte
CRC	1 Byte

Tabelle 2.3: Paketaufbau des Funkmoduls

Das Modul sendet im Stromsparmodus mit einer Datenrate von maximal 1Msps was für eine angestrebte Übertragungsrate von 100 8-Bit-Werten pro Sekunde ausreicht.

3 Hardware

3.1 Überblick

Die Hardware besteht grob zusammengefasst aus den folgenden Komponenten:

- Dem Longboard, bestehend aus dem Deck, den Achsen, dem Motor und der dazugehörigen Riemenübersetzung.
- Der Stromversorgung, die mit den Akkus dessen Kontrollschatungen, Absicherungen und Ladeschaltung für ein zuverlässiges Handling der Energie sorgt.
- Der Motorsteuerung, welche die Ansteuerung des BLDC-Motors über drei Halbbrücken, einem Mikrocontroller, Strommesswiderständen und Kommunikationsschnittstellen integriert.
- Der Steuerung, einem Ein-Finger-Handschuh-ähnlichen-Konstrukt, das die Biegung des Zeigefingers mittels eines Flex-Sensors detektiert, mit einem Mikroprozessor auswertet und über Funk dem Motorcontroller mitteilt. Ebenfalls integriert sind LEDs zur Akku-Ladestand- und Bedienanzeige sowie ein Taster zur Kalibration.

Auf die einzelnen Punkte wird in den folgenden Kapiteln noch detaillierter eingegangen. Hier sind noch kurz die Verbindungen untereinander beschrieben:

Die Kommunikation zwischen Motorsteuerung und der Bedienung, dem Magic Glove, basiert auf einer Funkverbindung mit einer Frequenz von 2.4GHz. Übermittelt wird der Wert der Biegung des Flex-Sensors in der einen Richtung, sowie der Akku-Ladestand in der anderen. Auf die Details wird in den Kapiteln der Software eingegangen.

3.2 Steuerung - Magic Glove

Der Magic Glove ist ein Skatehandschuh, der um einen Flex-Sensor erweitert ist. Somit schützt dieser bei einem Sturz nicht nur die Hände, sondern kann auch die Bewegung des Zeigefingers analysieren. Wird der Zeigefinger gestreckt, wie es beim Sturz aus Reflex passiert, wird das Board gebremst. Ballt man die Hand zur Faust, beschleunigt das Board. Da mit dem Flex-Sensor nur sehr kleine Widerstandsänderungen erzeugt werden, wird mit der sogenannten Wheatstone-Brücke, welche in der Abbildung 3.1 zu sehen ist, gemessen.

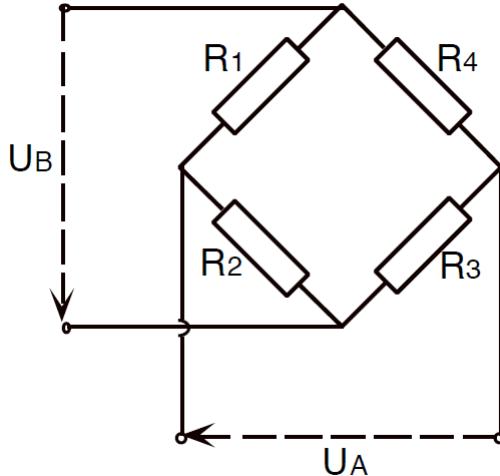


Abbildung 3.1: Wheatstone-Brücke

Wird eine Brückenspeisespannung U_B angelegt, kann mit der Formel 3.1 U_A berechnet werden [8].

$$U_A = \frac{R1 \cdot R3 - R2 \cdot R4}{(R1 + R2) \cdot (R3 + R4)} \cdot U_B \quad (3.1)$$

Ist der Flex-Sensor dabei in Ruhelage, misst man eine Spannung von 0V. Aus dem Verhältnis von U_A zu U_B erhält man die relative Dehnung S des Flex-Sensors [8].

$$S = \frac{U_A}{U_B} \quad (3.2)$$

U_A wird mit dem 10-Bit Analog-Digital-Wandler auf dem Mikrocontroller gemessen und mithilfe der fixen Spannung U_B wird die relative Dehnung S berechnet.

3.3 Stromversorgung

Die Stromversorgung, montiert bei der nicht angetriebenen Achse, liefert die Energie über zwei Leitungen aus Kupferband zur Motorsteuerung. Das Kupferband ist unter dem Grip-Tape versteckt und besteht aus zwei Schichten des Bandes. Eine auf dem Brett, eine auf dem Grip-Tape, jeweils nach ca. 15cm aufgetrennt, damit durch die Federung des Boards das Band nicht zerrißt. Die untere und die obere Schicht sind verschoben angeklebt, sodass sie einander immer überdecken, und ein problemloser Stromfluss garantiert wird. Die Strombelastbarkeit des 38mm breiten und 35µm dicken Kupferbandes ist mit der Formel für Leiterbahnen berechnet. Nach

$$I = 9.6 \cdot A^{0.68} \cdot \Delta T^{0.43} \quad (3.3)$$

mit dem Strom I in Ampère, der Fläche A in mm^2 , und der Temperaturdifferenz ΔT in Kelvin, ergibt sich ein Strom von $I=60A$ bei einem Temperaturdelta von lediglich 35° [9]. Da dieser Deltawert noch etwas höher werden darf, und so breite Leiter eher bessere Kühlverhalten aufweisen, wurde diese Näherung als ausreichend erachtet.

An die Stromversorgung werden einige Anforderungen gestellt. So muss sie für die Motorsteuerung grosse Ströme mit bis zu 50A liefern können. Gleichzeitig muss die Spannung der Zellen kontinuierlich überprüft werden, um ein Tiefentladen zu verhindern. Zudem wird eine Standby-Killer Schaltung implementiert und dessen Funktion erklärt.

Um das Board einfach laden zu können ohne jedes Mal den Akku ausbauen zu müssen, hat sich das Team entschieden, eine Akku-Ladeschaltung einzubauen. Diese muss die Zellen balancieren und sie vor Überspannung bzw. Überladung schützen. Im Folgenden wird das Balancing, die Ladeelektronik, die Selbsterhaltung, der verwendete Mikrocontroller und die Zellmessung genauer erläutert.

Balancing

Das Balancing, wie im Kapitel 2.2.2 erklärt, ist eine wichtige Funktion damit der Akku nicht zerstört wird. Das Ganze soll möglichst einfach aufgebaut und mit dem Mikrocontroller ansteuerbar sein. Wie in Abbildung 3.2 ersichtlich, wurde dies mit einer P-Kanal MOSFET-Schaltung ermöglicht. Der Widerstand am Drain des P-Kanal-MOSFET reguliert den Strom mit dem die einzelnen Zellen ausgeglichen werden. Damit die Schaltung mit einem Mikrocontroller gesteuert werden kann, braucht es einen N-Kanal-MOSFET, der das Gate des P-Kanal MOSFET auf Ground zieht. Mit dieser einfachen Schaltung ist es möglich, die sechs Zellen des Akku zu balancieren.

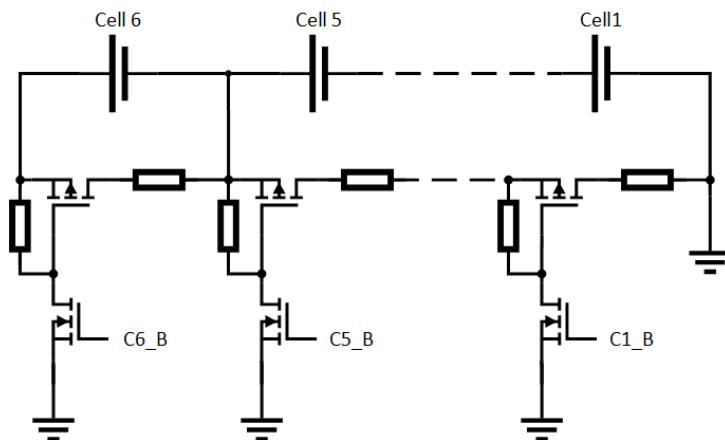


Abbildung 3.2: Balancing

Ladeelektronik

Damit der Akku fachgerecht geladen wird, braucht es eine Ladeelektronik, die den LiPo-Akku nach dem CCCV-Verfahren (constant current, constant voltage) lädt. Es gab zwei Möglichkeiten, diese aufzubauen. Die eine Möglichkeit wäre ein teures IC gewesen, welches auf dem Markt verfügbar ist. Die andere Möglichkeit ist eine günstige Buck-Converter Schaltung, mit der man nach dem oberen Verfahren laden kann. Ein solcher Buck-Converter ist in diesem Produkt eingesetzt.

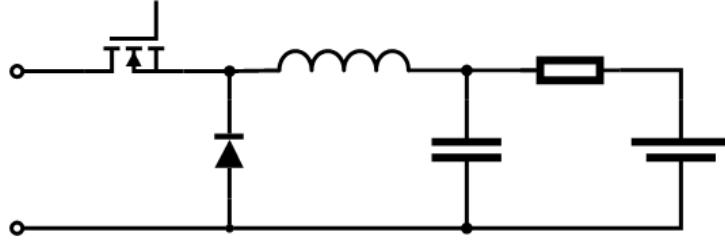


Abbildung 3.3: Buck-Converter

Das Schema des Buck-Converters in Abbildung 3.3 zeigt, dass dieser einfach als auch schnell aufgebaut ist. Der Widerstand ist ein Shunt-Widerstand, an dem der Strom gemessen wird. Anhand dieses Stromes wird der PWM, welcher über einen High-Side-Treiber am MOSFET anliegt, durch den Mikrocontroller geregelt. Die Taktfrequenz des Clocks liegt bei 31.3kHz und ist mit der Formel (3.4) aus dem Datenblatt des ATmega berechnet [10].

$$f_{PWM} = \frac{f_{\mu C}}{N \cdot 256} \quad (3.4)$$

Mit dieser Taktfrequenz kann man nun die Grösse der Spule definieren. Dies wird mit folgender Formel (3.5) berechnet.

$$L = \frac{RF \cdot (V_{IN} - V_{OUT}) \cdot \frac{V_{OUT}}{V_{IN}}}{f_s \cdot I_{OUT}} \quad (3.5)$$

Berechnet wurde eine Spule von rund $120\mu\text{H}$. Eingesetzt ist eine Coilcraft-Spule mit dem Wert von $100\mu\text{H}$. Da die Batterie wie ein grosser geladener Kondensator wirkt, sind keine all zu grossen Ripple am Ausgang des Buck-Converter zu erwarten.

Selbsterhaltung

Wenn der LiPo-Akku voll geladen ist oder die Fahrt für mehr als 20min unterbrochen wird, sollte der Mikrocontroller sich selbstständig herunterfahren und die gesamte Stromversorgung möglichst keinen Strom brauchen. Dafür wurde eine Selbsterhaltungsschaltung entwickelt (Abbildung 3.4). Sobald beim Mikrocontroller einer von beiden Zustände (langer Stillstand oder voll geladen) eintrifft, wird der N-Kanal-MOSFET ausgeschaltet. Somit wird die Speisung auf dem gesamten Print unterbrochen. Dadurch wird die Batterie bei einer längeren Lagerung vor dem Zerstören geschützt.

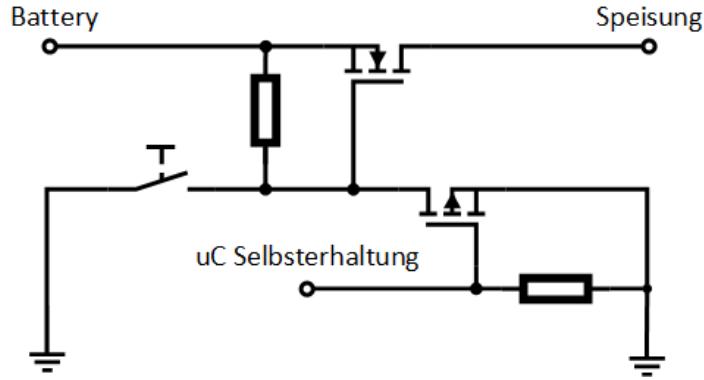


Abbildung 3.4: Selbsterhaltung

Mikrocontroller

Der Mikrocontroller muss verschiedene Anforderungen erfüllen. In der folgenden Tabelle 3.1 werden die General-Purpose-Input-Output-Anforderungen (GPIO-Anforderungen) an den Mikrocontroller aufgelistet.

Beschreibung	Anforderung	Input/Output	Anzahl
Überwachung der Zellen	Analog	Input	6
Überwachen des Ladestroms	Analog	Input	1
Überwachen der Eingangsspannung	Analog	Input	1
Entladen der Zellen (Balanceing)	Digital	Output	6
Anzeige des Ladezustands (LED)	Digital	Output	3
Schalten des Ausgang Stroms zum Motor	Digital	Output	1
Selbsthaltung des Mikrocontrollers	Digital	Output	1
Regelung des Ladestroms	PWM	Output	1
Kommunikation SPI Schnittstelle MISO/MOSI/SCK	Serial	In/Output	3

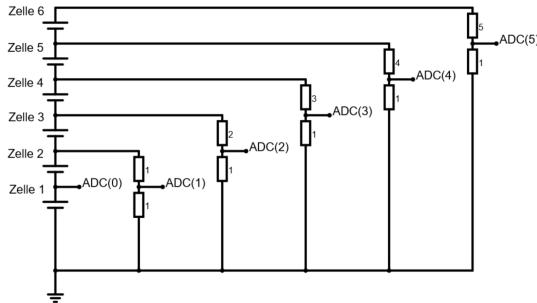
Tabelle 3.1: GPIO Anforderungen für den Mikrocontroller

Für die Regelung der Stromversorgung wird der Mikrocontroller ATmega328P-AU verwendet.

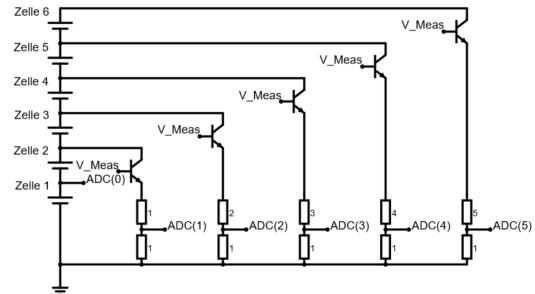
Zellmessung

Für das bessere Verständnis der Zellmess-Schaltung wird die Entwicklung geschildert.
Das Ladegerät besteht aus jeweils sechs Zellen, die in Serie geschaltet sind, um dem Motor genügend Spannung zur Verfügung zu stellen. Mit einer Zellenspannung von 4,15V pro Zelle liegt am obersten Punkt (Anode der Zelle 6) eine Spannung von 24,9V an. Da die Spannung an den Eingängen des Mikrocontrollers nicht seine Speisespannung übersteigen sollte, mussten diese Spannungen verringert werden. Für den verwendeten Mikrocontroller beträgt die maximale Spannung am Eingang 5V.

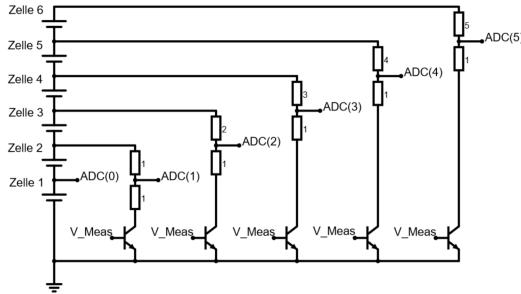
Die einfachste Variante ist, mit einem Spannungsteiler die Spannungen so zu skaliert, dass die Maximalspannung nicht überschritten wird. Diese Grundschaltung ist in Abb.3.5(a) zu sehen.



(a) Zellmessung Grundschaltung



(b) Zellmessung mit Transistoren über den Widerständen



(c) Zellmessung mit Transistoren über den ADC-Eingängen

Abbildung 3.5: Drei Varianten der Zellmessung

Der Nachteil dieser Schaltung ist, dass die Zellen durch die Spannungsteiler kontinuierlich entladen werden. Somit können bei einem längeren Nichtgebrauch die Akkuzellen entladen oder sogar tiefentladen sein.

Um dies zu verhindern, wurde die Schaltung mit Transistoren erweitert, sie ist in der Abb.3.5(b) zu sehen. Diese Transistoren unterbrechen den Entladestromkreis über den Widerständen. Dazu werden die Transistoren übersteuert, um sie als Schalter zu verwenden und den durch die Sättigungsspannung U_{CE} entstehende Fehler zu verkleinern. Dieser kann jedoch je nach Sättigungsspannung der Transistoren und aufgrund der unterschiedlichen Bauarten nur schwer korrigiert werden. Der zweite grosse Nachteil (und gleichzeitig das Killerkriterium für diese Schaltung) ist, dass durch das Unterbrechen des Spannungsteiler-Stromkreises an den ADC-Ausgängen wieder die Zellenspannung anliegt. Dadurch würden sich die Zellen über die Ableitdiode am Eingang des Mikrocontrollers entladen und diese Diode voraussichtlich zerstören.

Dies könnte verhindert werden, indem der Stromkreis oberhalb der ADC-Eingänge unterbrochen wird. Dies ist in der Schaltung der Abb.3.5(c) gezeichnet. Dadurch wären die Eingänge vor Überspannung geschützt. Aber auch diese Schaltung enthält diverse Nachteile. So fliesst beim geschlossenen Zustand der Basis-Emitter-Strom durch den Spannungsteiler. Dies verfälscht die Messresultate. Zusätzlich muss zwischen Basis und Emitter eine Schaltspannung von mindestens 0.7V anliegen. Dazu müsste die Schaltspannung 0.7V über der Zellspannung liegen, was nur mit grossem Aufwand erreichbar wäre. Das Team entschied, die erste Methode zu wählen, da sie die genauesten Messungen liefert und die Entladung der Zellen bei einer durchschnittlichen Benutzung des Boards nicht problematisch ist, also keine Tiefentladung auftritt.

3.4 Motoransteuerung

Die Motoransteuerung ist bei der angetriebenen Achse montiert. Die Funktion dieser Steuereinheit wird im folgenden im Überblick kurz erläutert und danach wird auf die einzelnen Bestandteile eingegangen. Bei diesem Board kommt die Drehfeldanregung mit FOC zum Einsatz, was bedeutet, dass an den drei Phasen je ein jeweils um 120° phasenverschobener Sinusstrom angelegt wird. Dies erzeugt ein rotierendes Drehfeld, welches den aus Permanentmagneten bestehenden Erregerkreis auf dem Rotor mit einem Drehmoment versieht. Dieser grundsätzliche Ablauf wird hardwaretechnisch so umgesetzt: In einem Mikrocontroller, wird pro Motorphase ein PWM-Signal erzeugt, das integriert und gemittelt ein Sinussignal ergibt. Über einen intelligenten und schnellen Treiber werden dann die in einer H-Brücke angeordneten N-Kanal-MOSFET Transistoren angesteuert, welche dieses PWM-Signal mit grosser Leistung an die Klemmen geben. Da jede Phase im Motor eine Spule ist, und der Strom durch eine Spule das Integral der Spannung ist, ergibt sich so der quasi-sinusförmige Stromverlauf. An zwei Phasen werden die Ströme über Shunt-Messwiderstände gemessen und dem Mikrocontroller über einen A/D-Wandler mitgeteilt. Dies ist dann die Grundlage für die integrierte Regelung.

Mikrocontroller

Verwendet wird der Mikrocontroller STM32F303, weil er mit seinen 8MHz Taktfrequenz genügend schnell ist und zudem im Team bereits positive Erfahrungen damit gemacht wurden. Zudem unterstützt er die PWM-Funktion und hat genügend I/O-Ports. An ihm wird über ein SPI Interface noch das HopeRF RFM75-Modul angeschlossen, das mit der Steuerung, dem Magic Glove, über eine 2.4GHz-Funkverbindung kommuniziert.

Typ	Pins	Ports	Beschreibung
Analog	11,14-16,26-27	A, B, C	Analoge Eingänge zur Messung der Phasenströme und -Spannungen sowie der Versorgungsspannung.
Digital	54-56,37-38	B, C, D	Standard digitale Steuersignale zum und vom Treiber-IC. Statussignale für Fehlermeldungen (FAULT, OCTW, PWRGD) und Steuersignale (EN_GATE, DCCAL).
SPI	50-53	A, C	SPI Bus zum Treiber-IC für Einstellungen und Statusmeldungen.
PWM	34-36, 41-43	A, B	PWM Signale für die Halbbrücken. Diese gehen an das Treiber-IC, welches die FETs ansteuert.
Digital	8,10	C	Steuersignale IRQ und Chip-Enable für das RFM-Modul.
SPI	51-53, 9	C	SPI Bus zum RFM-Modul. Die Datenleitungen SDI und SDO sowie die Taktleitung SCK werden mit dem Treiber IC geteilt. Es unterscheiden sich nur die Chip-Select-Signale.
UART	29,30	B	Serielle Schnittstelle zu Debug-Zwecken. Wird nur in der Entwicklung verwendet.
SWD	7, 46,49	A	Programmierschnittstelle
USB	44,45	A	USB Schnittstelle vorgesehen, wird jedoch nicht verwendet.

Tabelle 3.2: Ein-/Ausgänge Mikrocontroller

Endstufe (Treiber und H-Brücken)

Für die Ansteuerung der Transistoren wird der DRV8301 Gate-Treiber von Texas Instruments verwendet. Dieser Treiber beinhaltet einen Gate Driver mit 1.7A Peak Gatestrom, Current Shunt-Differenzverstärker, Über- und Unterspannungsschutz sowie Übertemperaturschutz und Überstromschutz. Ebenfalls ist ein Buck Converter für die Spannungsversorgung eingebaut. Die Transistoren sind IRFS7534 N-Kanal-Power-MOSFETs von Infineon, die 60V und 164A permanent aushalten. Sie eignen sich besonders für schnelles Schalten von grossen Lasten, also genau dieses Einsatzgebiet. Dies vor allem wegen des tiefen R_{DSon} (typ. 1.6mΩ) und der geringen Eigenkapazität. Für die Ansteuerung Transistoren wird der DRV8301 Gate-Treiber von Texas Instruments verwendet. Dieser Treiber beinhaltet einen Gate Driver mit 1.7A Peak Gatestrom, Current Shunt Differenzverstärker, Über- und Unterschutzspannung sowie Übertemperaturschutz und Überstromschutz. Ebenfalls ist ein Buck Converter für die Spannungsversorgung eingebaut.

Messschaltungen

In der Ansteuerung von zwei Phasen sind 1mΩ Shunt-Widerstände zur Strommessung eingebaut. Die Spannung über den Widerständen wird im Gate-Treiber um den Faktor 10 verstärkt und mit einem Offset von $\frac{V_{logic}}{2}$ versehen. Der Mikrocontroller erhält dadurch von dem Treiber eine Spannung zwischen 0V und V_{logic} . Damit wird der dritte Phasenstrom berechnet. Dieser wird zur Regelung verwendet.

4 Software

4.1 Überblick

Die Software besteht aus drei unterschiedlichen Teilen mit separaten Teilaufgaben (Steuerung, Stromversorgung und Motoransteuerung). Zum einen die Batteriezellenbalancierung und -ladung. Andererseits die eigentliche Motoransteuerung, welche per Drahtlos-Modul mit dem Magic Glove(Human Interface) verbunden ist. Das reibungslose Zusammenspiel dieser drei Softwarekomponenten ist äusserst kritisch, da jegliche Softwarefehler extrem gefährlich für den Piloten sein können.

4.2 Steuerung - Magic Glove

Die Software des Magic Glove ist um eine einfache Bedienung zu garantieren, sehr simpel aufgebaut. Sie misst in periodischen Abständen von 10ms die Beugung des Fingers des Piloten und übermittelt diesen Wert via eingebautem Drahtlos-Modul zur Motorensteuerung, welche die Motordrehzahl entsprechend anpasst. Ausserdem kann durch Betätigen des Buttons auf der Platine der Batteriestand des Longboards auf den acht angebrachten LEDs dargestellt werden. Die Steuerung merkt automatisch, wann die Verbindung zum Board verloren ist, stellt diese aber wieder her, sobald das Board wieder in Reichweite des Magic Gloves ist.

Entwicklungsumgebung

Compiler	AVR GCC
IDE für Debugging	ATMEL Studio
Programmer	AVR Dragon

Tabelle 4.1: Verwendete SW-Entwicklertools

Modulübersicht

Die Programmfunktionalität des Magic Glove ist in verschiedene C-Module untergebracht. Eine Liste der implementierten Module ist in der Tabelle 3.2 untergebracht.

Name	Beschreibung
<code>rfm7x.c</code>	Enthält die Methoden und gewisse Konstanten für die Benutzung des RFM75-Moduls.
<code>spi.c</code>	Enthält die Logik für die SPI-Schnittstelle des Mikrocontrollers.
<code>main.c</code>	Enthält den eigentlichen Programmablauf der Steuerung. Beschrieben in 4.2

Tabelle 4.2: Magic Glove: C-Module

Headerübersicht

Wichtige Einstellungen und Parameter sind zur einfacheren Wartung in Headerdateien untergebracht. Eine Liste der wichtigsten Headerdateien ist in Tabelle 4.3 dargestellt.

Name	Beschreibung
<code>rfm7x.h</code>	Enthält die Funktionsdefinitionen und spezifischen Einstellungen für das RFM75-Modul.
<code>rfm7x_config.h</code>	Enthält die Konfigurationseinstellungen welche bei der Initialisierung auf das RFM75-Modul geladen werden.
<code>spi.h</code>	Speichert die Einstellungen der SPI für den verwendeten ATmega-Chip.
<code>asf.h</code>	Automatisch generierte Header-Datei des Atmel Software Frameworks.

Tabelle 4.3: Magic Glove: Header-Dateien

Libraries

Die Ansteuerung des RFM75-Modules wurde mittels einer existierenden Library implementiert. Alle verwendeten externen Libraries sind in der Tabelle 4.2 aufgelistet.

Name	Beschreibung	
<code>RFM7x</code>	Diese Library ermöglicht unter anderem die Ansteuerung des RFM75-Moduls und ermöglicht eine problemlose Ansteuerung desselben. Die Library ist unter der MIT-Lizenz verfügbar und somit frei weiterzuverwenden.	https://github.com/jnk0le/RFM7x-lib

Tabelle 4.4: Magic Glove: Übersicht über externe Libraries

Programmablauf

Der Code auf dem ATmega328PB-AU sieht folgendermassen aus:

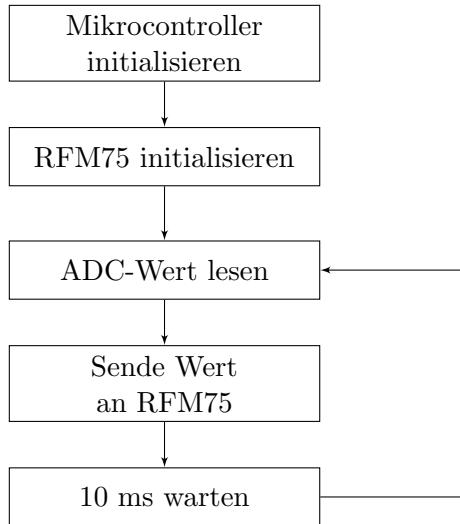


Abbildung 4.1: Ablauf des Magic Glove C-Code

4.3 Stromversorgung

Mit der Software zum Battery Management werden wichtige Punkte geregelt, um ein langes Leben der Akkuzellen zu gewährleisten. Darunter gehören sowohl Über- als auch Unter-Spannungsschutz sowie Ausbalancieren der einzelnen Zellenspannungen beim Laden.

Entwicklungsumgebung

Die verwendeten Entwicklertools sind in der Tabelle 4.5 zusammengefasst.

Compiler	AVR GCC
IDE für Debugging	ATMEL Studio
Programmer	AVR Dragon

Tabelle 4.5: Verwendete SW-Entwicklertools

Programmablauf

Der aktuelle Ladevorgang wird über drei verschiedenen LEDs angezeigt. Dabei zeigen diese den Constant Current (CC), Constant Voltage (CV) und den Vollen Zustand (off) an. Diese können als Laden (CC, LED1) Betriebsbereit (CV, LED2) und Vollständig geladen (off, LED3) interpretiert werden.

Das folgende Zustandsdiagramm (Abbildung 4.2) zeigt den Ablauf der Software auf dem Mikrocontroller.

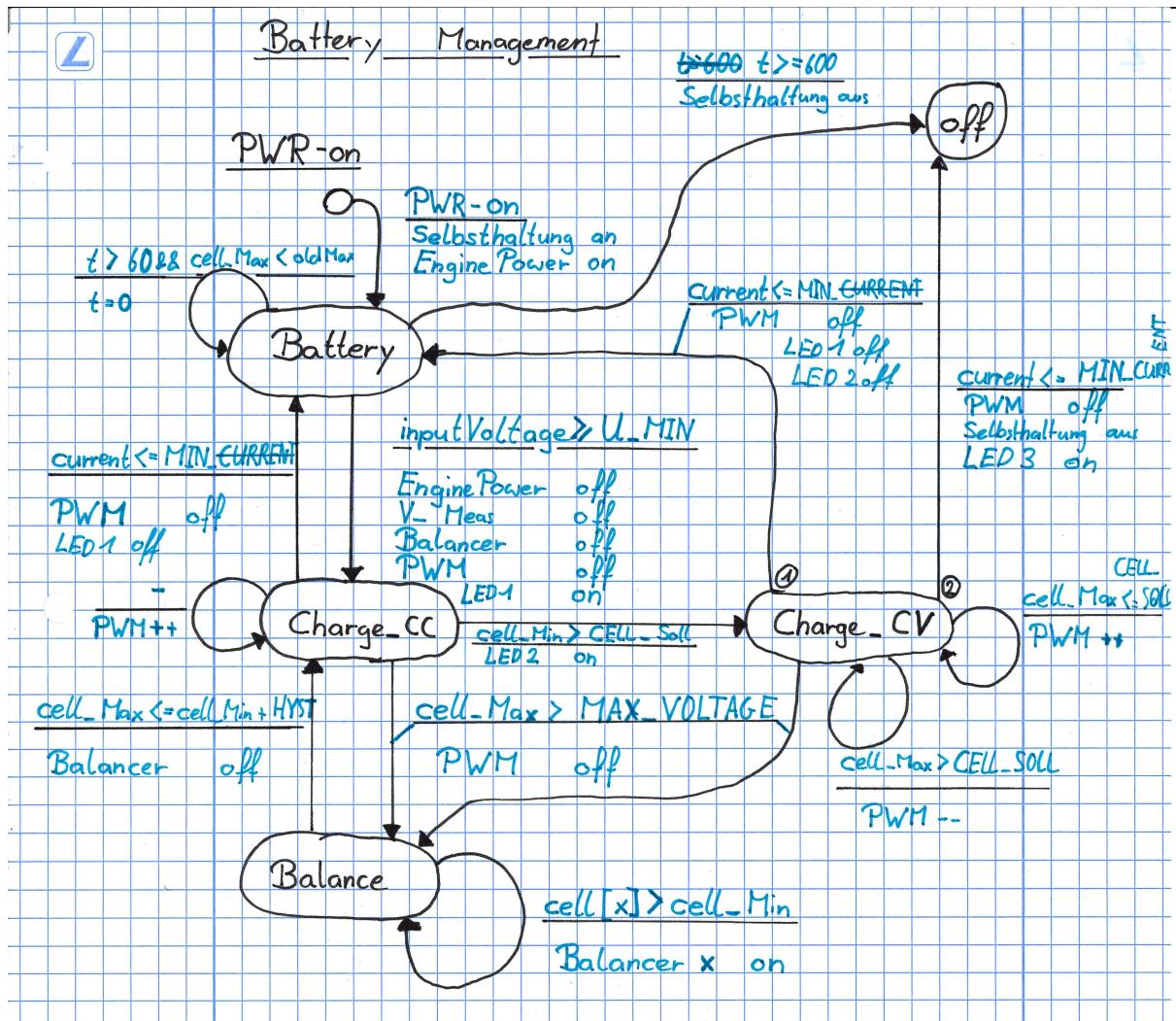


Abbildung 4.2: State Diagramm des Batterie-Management

Wie im Kapitel 3.3 beschrieben, hält sich der Mikrocontroller mit einer Selbsthaltung in Betrieb. Diese wird jedoch einerseits mit dem Ein-Taster als auch mit der Eingangsspannung des Ladegerätes überbrückt. Wird einer von diesen aktiviert, fährt der Mikrocontroller hoch und betätigt die Selbsthaltung.

Beim Start wird als Erstes der Zustand **Battery** geladen. Wenn das Ladekabel nicht angeschlossen ist, wird die Schaltung den Motor mit der benötigten Spannung von den Akkuzellen versorgen. Falls das Board über zehn Minuten nicht gebraucht wird oder sich der Akku der unteren Spannungsgrenze nähert, schaltet sich das Board aus. Sobald eine Ladespannung des Ladegeräts angelegt wird, wechselt der Mikrocontroller in den Zustand **Charge_CC**.

In diesem Zustand wird der Strom zunehmend auf die maximalen 5A hochgefahren und dort konstant gehalten. Dies geschieht durch den PWM-Ausgang der auf den Schaltregler führt. Je mehr Strom in die Zellen fließt, desto höher wird deren Spannung. Sobald eine Zelle den Maximalwert von 4.3V überschreitet, wird in den Zustand **Balance** gewechselt. Dort werden alle Zellen auf den Wert der niedrigsten Zelle entladen. Die beiden Zustände werden solange wiederholt, bis sich die Spannung der niedrigsten Zelle über der Soll Spannung befindet. Danach

wird in den nächsten Zustand **Charge_CV** gewechselt.

Im Zustand **Charge_CV** wird mit der Regelung des Zuführstromes versucht, die Zellenspannung weiterhin auf dem Sollwert von 4.15V zu halten. Dieser Strom nimmt mit der immer weiter fortschreitenden Aufladung der Zellen kontinuierlich ab, bis schliesslich ein unterer Grenzwert von 100mA erreicht wird. An diesem Punkt gilt der Akku als voll geladen und wechselt in den Zustand **Off**. Anders als vielleicht zuerst angenommen bleibt das Board dank der Überbrückung der Selbsthaltung aktiv und zeigt durch die drei leuchtenden LEDs an, dass der Akku voll aufgeladen ist. Sobald die Spannung des Ladegeräts abfällt, fällt auch die Selbsthaltung ab. Somit ist das Board komplett ausgeschaltet.

Ein weiterer Schwerpunkt der Software ist die Berechnung der Spannungen der einzelnen Zellen. Die Zellen sind seriell miteinander verbunden. Somit addieren sich die Spannungen an den Ausgängen jeder Zelle bis auf 24,9V auf der sechsten Zelle. Da der AD-Wandler des Mikrocontrollers nur zwischen 0 und 5 Volt messen kann, müssen die Ausgänge der Zellen mit einem Spannungsteiler mit dem Faktor $\frac{1}{n}$ herunter skaliert werden. Ab der zweiten Spannung sind die Ausgänge jedoch abhängig von den vorherigen Zellen. Um einen korrekten Wert zu erhalten, muss die Differenz inklusive den richtigen Skalierungsfaktoren berechnet werden. Mithilfe dieser Daten kann man für die Spannung der einzelnen Zellen $Zelle[n]$ folgende Formel herleiten:

$$Zelle[n] = V_{ADC}[n] - V_{ADC}[n - 1] \quad (4.1)$$

Die Zellspannungen werden in jedem Durchlauf gemessen und in einem Array abgespeichert. Aus diesen Werten werden die minimale und maximale Spannung berechnet, welche für die verschiedenen Statuswechsel in der Statemachine verwendet werden.

Peripherie

Um den gesamten Mikrocontroller zu entlasten, wurden möglichst viele Anwendungen in interne Peripherien ausgelagert.

Peripherie	Port	Verwendung
ADC	PD0-PD5	Auf diesen ADC-Eingängen werden die Zellspannungen des Akkus eingelesen. Dabei wird dies bei jedem Durchlauf aufgerufen. Sobald die Werte nicht mehr überein stimmen, wird ein Balancing ausgelöst. Somit wird sicher gestellt, dass der Akku nicht zerstört wird.
ADC	ADC6	An diesem ADC-Eingang wird der Strom gemessen, mit welchem der Akku gerade geladen wird. Damit kann der Duty-Cycle des PWMs richtig eingestellt werden.
ADC	ADC7	Auf diesem ADC-Eingang wird die Ladespannung gemessen. Diese wird gebraucht um den Duty-Cycle zu bestimmen, mit welchem der PWM des Ladereglers getaktet werden soll. Weiter wird überprüft ob mindestens eine Spannung von 30V anliegt.
Timer0	PD6	Der Timer wird ausschließlich für den PWM verwendet. Der PWM ist auf rund 31kHz eingestellt, was dem Maximum entspricht. Mit diesem PWM wird der High-Side Driver gesteuert und somit die gesamte Ladeschaltung.
Timer1	-	Dieser Timer wird für die Zeitmessung verwendet. Dabei werden die Sekunden gezählt. Dies wird dazu gebraucht, um die Inaktivität des Boardes festzustellen. Nach einer Zeit von 20min wird das gesamte Board ausgeschaltet.
USART	PD0 - PD2	Wenn man an diesem Ausgang des Mikrocontrollers ein TTL-Kabel anschliesst, können bestimmte Werte wie zum Beispiel die Akkuspannungen und der Duty-Cycle ausgelesen werden.

Tabelle 4.6: Verwendete Peripherie des Mikrocontrollers

4.4 Motoransteuerung

Die Motoransteuerung regelt das Drehmoment des BLDC-Motors durch Implementation des FOC-Algorithmus 2.2.4. Im folgenden Kapitel wird beschrieben welche Entwicklungsumgebung zum Einsatz kommt, wie der Code aufgebaut ist und wie der grobe Ablauf des Programms umgesetzt ist.

Plattform

Zum Einsatz kommt ein STM32F303 Mikrocontroller. Er besitzt eine ARM Cortex-M4 CPU mit FPU und DSP Instruktionen. Als RTOS wird ChibiOS verwendet. Es stellt grundlegende Kernel Funktionalitäten zur Verfügung wie Schedulers und Threads sowie einen HAL. Die komplette Firmware ist in C geschrieben. Die vom Code generierte Dokumentation kann unter <https://noah95.github.io/skatemate-sw/> aufgerufen werden.

Entwicklungsumgebung

Compiler	arm-none-eabi-gcc 5.4.1 20160919
Editor	Sublime Text 3
IDE für Debugging	Eclipse CDT
Debugger Probe	openocd 0.10.0 via ST-Linkv2

Tabelle 4.7: Verwendete SW-Entwicklertools

Peripherie

Um dem Mikrocontroller Rechenleistung abzunehmen werden möglichst viele Aufgaben an die eingebauten Peripheriegeräte ausgelastet. Einige Aufgaben wie das Einlesen von Spannungen kann außerdem nur von speziellen Peripheriegeräten übernommen werden. In der Tabelle 4.8 sind alle verwendeten Peripherien aufgelistet.

Peripherie	IRQ	DMA	DMA IRQ	Verwendung
TIM1	-	-	-	<p>Zur Generation der PWM-Signale wird der Timer TIM1 verwendet. Er ist konfiguriert in Center Aligned PWM Mode. So zählt der Timer zu einem definierten Wert hoch und wieder runter.</p> <p>Die PWM Kanäle schalten, sobald deren Komparator-Wert dem Zählerwert entsprechen. So sind alle PWM-Ausgänge in der Mitte des Pulses zentriert. Alle Kanäle sind komplementär, so wird der Low-Side-FET ausgeschaltet, wenn der High-Side-FET eingeschaltet ist und umgekehrt. Kanäle eins bis drei werden für die drei Phasen verwendet und der Kanal vier zur Triggerung des ADC.</p> <p>Der Duty-Cycle des vierten Kanals ist auf den kleinsten Wert konfiguriert, so werden die Signale genau in der Mitte des PWM Pulses eingelesen. So wird das TIM1-TRGO-Signal generiert.</p>
ADC1	-	1 CH1	-	Der ADC1 wandelt die drei Phasenspannungen sowie die Versorgungsspannung. Er wird getriggert von dem TIM1 TRGO Event. Die vier Kanäle werden sequentiell konvertiert und mittels DMA im RAM abgespeichert.
ADC3	FC	5 CH5	-	Der ADC3 wandelt die zwei Phasenströme, sowie die interne Temperatur und Referenzspannung. Er wird getriggert von dem TIM1 TRGO Event. Die vier Kanäle werden sequentiell konvertiert und mittels DMA im RAM abgespeichert. Der End of Sequence Interrupt ist aktiviert. Er wird getriggert, sobald alle Kanäle gewandelt sind.
TIM15	-	-	-	Der TIM15 wird für Zeitmessungen zu Debugzwecken verwendet. Er zählt im 72MHz Takt.
TIM3	-	-	-	Der TIM3 wird für einen Input Capture verwendet. So kann das Signal eines Modellbau RC-Empfängers eingelesen werden. Dies wird nur zu Debugzwecken verwendet. Der TIM3 wird vom ChibiOS HAL ICU Treiber verwendet.
USART3	-	-	-	Per USART3 wird mit dem Computer zu Debugzwecken kommuniziert. Die ChibiOS Shell läuft über diese Schnittstelle. Zudem werden Daten zur Visualisierung Charakter-basiert über USART3 übertragen.
SPI3	-	-	-	SPI3 wird verwendet zur Kommunikation mit dem Treiber IC und dem RFM-Modul. Das Treiber IC wird nur zur Initialisierung beschrieben.
USB1	-	-	-	USB1 wird initialisiert, aber nur optional verwendet für eine virtuelle serielle Schnittstelle. So kann die Funktion von USART3 ohne USB-Seriell-Wandler verwendet werden.

Tabelle 4.8: Verwendete Peripherie des Mikrocontrollers

Programmablauf

Der Grossteil der Software läuft in der Interruptroutine des AD-Wandlers, sodass der Regler mit einer konstanten Frequenz berechnet wird. Dabei laufen parallel Threads für Steuerung und Debugging. Folgend werden die in Tabelle 4.9 aufgelisteten Routinen beschrieben.

Name	Thread/ISR	Verwendung
ADC ISR	ISR	FOC Routine und Motor Regelung
main	Thread	Einlesen der Magic Glove-Daten
mcfoc main	Thread	Statusmaschine für die Motorregelung mit Anfahren und Schutzfunktionen
mcfoc secondary	Thread	Zur Aufzeichnung von Regeldaten. Nur zu Debugzwecken
shell	Thread	Bedient die Konsole. Nur zu Debugzwecken. Eingebaute ChibiOS-Funktion

Tabelle 4.9: Thread und ISR Übersicht

ISR ADC

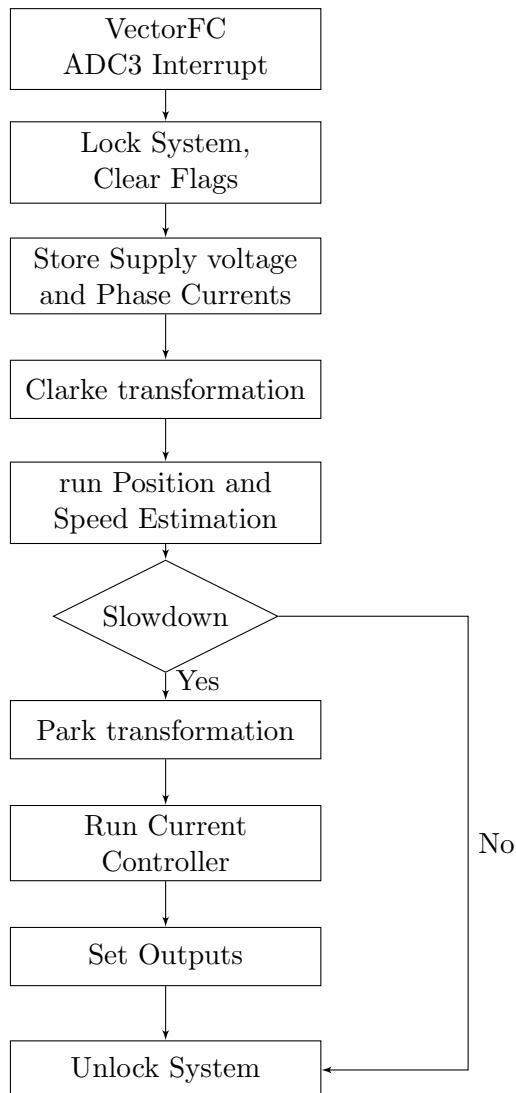


Abbildung 4.3: Ablauf der ADC Interrupt routine

Vor den Berechnungen wird das Betriebssystem mittels `chSysLockFromISR` in den I-Locked State geschaltet. So können keine anderen Interrupts die Berechnungen unterbrechen und I-Class-API Funktionen sind verfügbar. Die konvertierten Signale werden vom DMA-Buffer in Spannungswerte, respektive Stromwerte umgerechnet und abgespeichert. Die Ströme werden Clarke-transformiert, sodass der Positions- und Geschwindigkeits Beobachter aufgerufen werden kann. Der Stromregler wird langsamer ausgeführt als die Beobachter. Mit dem Define `FOC_CURRENT_CONTROLLER_SLOWDOWN` kann dieser Faktor eingestellt werden.

Vor dem Stromregler werden die Messdaten park-transformiert. Der Stromregler wird aufgerufen und danach die berechneten Duty-Cycle im Timer gesetzt.

Am Ende der ISR wird das System aus dem I-Locked State entsperrt.

Thread main

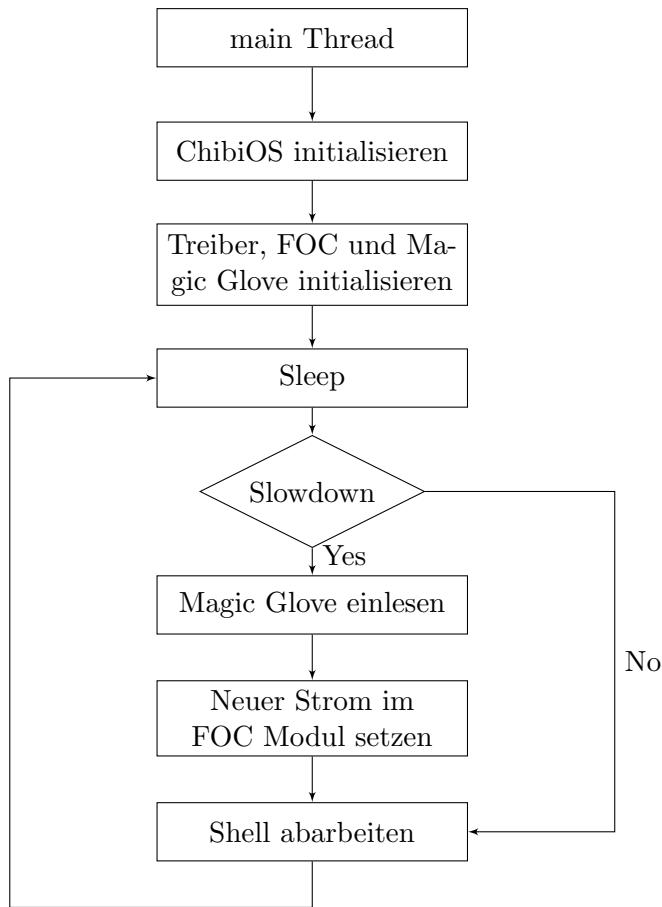


Abbildung 4.4: Ablauf des main Threads

Der Main-Thread wird nach der Initialisierung des Clocks und der Konfiguration der GPIOs gestartet. Nach der Initialisierung der restlichen Module startet die Endlosschleife. Da die Shell schneller abgearbeitet werden muss, wird das Einlesen des Magic Gloves verlangsamt und bei jeder Iteration die Shell abgearbeitet. Nach dem Einlesen der Steuerdaten werden diese an das `mc_foc` Modul weitergeleitet.

Thread mcfoc main

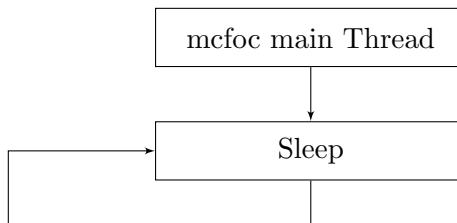


Abbildung 4.5: Ablauf des main Threads

Zur Zeit der Verfassung dieser Dokumentation war das Anfahren noch nicht implementiert. Dieser Thread ist vorgesehen für die Ansteuermethode des Motors zu entscheiden: Ob eine spezielle Anfahrmethode benötigt ist, der Motor bereits dreht oder gebremst werden soll. Nach einem

Stillstand muss angefahren werden, bei Sollwertsprüngen eventuell verlangsamt angesteuert und beim Bremsen den Maximalstrom limitieren werden.

Thread mcfoc secondary

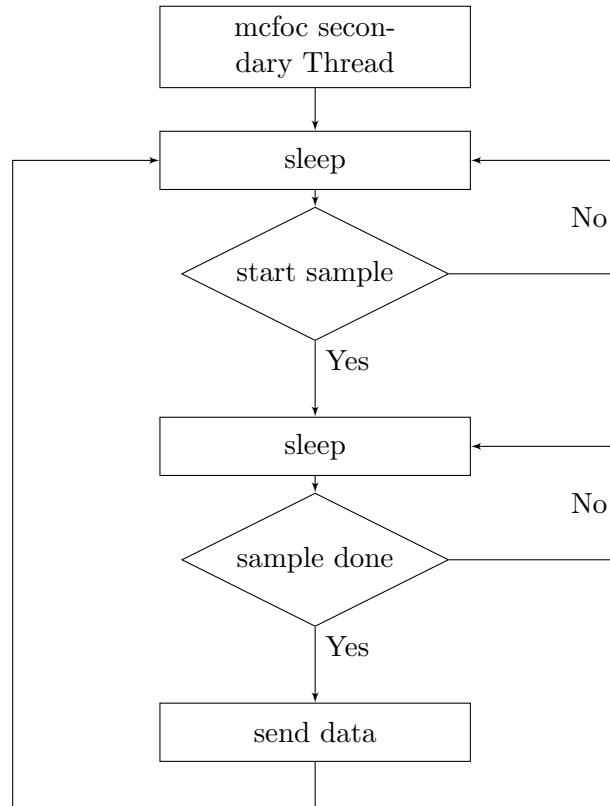


Abbildung 4.6: Ablauf des main Threads

Der sekundäre `mc_foc` Thread ist zuständig für das Datenaufzeichnen und -übertragen. Mit dem Befehl `sample` wird der Aufzeichnungsvorgang gestartet. In den Interruptroutinen werden die zu aufzeichnenden Daten im RAM gespeichert und sobald, der Puffer voll ist, sendet der Thread diese Daten via Shell Stream an den Computer.

Shell Thread

Der Shell Thread ist eine ChibiOS-eigene Funktion. Er liest die Daten von der seriellen Schnittstelle oder dem USB-Treiber und verarbeitet sie.

Modulübersicht

Zur Übersichtlichkeit und Wartbarkeit ist die Software in Module unterteilt. Ein Modul besteht aus einer C-Quelldatei und optional aus einer Headerdatei. Die Module sind nach ihren Funktionen unterteilt und wie in Tabelle 4.10 aufgelistet.

Name	Beschreibung
<code>main</code>	Im main-Modul befindet sich die main routine, welche nach der reset-Funktion aufgerufen wird. Darin wird das Betriebssystem und alle anderen Module initialisiert. Im main-Thread werden die Daten des Magic Gloves empfangen und an das mc_foc-Modul gesendet.
<code>usbcdc</code>	Startet die USB-Schnittstelle und deklariert alle Shell-Funktionen.
<code>usbcfg</code>	In diesem Modul werden die USB Endpunkte erstellt und behandelt. Es ist Teil vom ChibiOS und nicht selbst verfasst.
<code>util</code>	Das util-Modul stellt verschiedene Module zur Verfügung. Wichtig für die Motoransteuerung sind die darin enthaltenen Trigonometriemethoden wie schnelle arcus-Tangens-Berechnungen.
<code>drv8301</code>	Die Initialisierung des FET-Treiber-ICs geschieht in diesem Modul. Es beinhaltet auch alle SPI-Registerdefinitionen. Die default-Werte sind in der init-Funktion enthalten.
<code>mc_foc</code>	In diesem Modul geschieht die komplette Motorsteuerung. In der init-Funktion werden Peripherie-Module initialisiert, Standardwerte geladen, Eingänge kalibriert und die Kontrol-Threads gestartet. Die ADC-Interrupt-Routinen befinden sich am Ende dieses Moduls.
<code>utelemetry</code>	Dieses Modul ermöglicht die Datenübertragung an einen Computer mit der geeigneten Empfängersoftware. So können Variablen zeitabhängig beobachtet werden. In der letzten Version wird dies jedoch nichtmehr verwendet.

Tabelle 4.10: Modulübersicht

Headerübersicht

Wichtige Einstellungen und Parameter sind zur einfacheren Wartung in Headerdateien untergebracht. Eine Liste der wichtigsten Headerdateien ist in Tabelle 4.11 dargestellt.

Name	Beschreibung
<code>defs.h</code>	Globale Definitionen wie Thread-Namen und Stackgrösse, Clockdefinitionen und Debug-Macros.
<code>chconf.h</code>	Einstellungen für den ChibiOS-Kernel.
<code>halconf.h</code>	Welche Module der ChibiOS-HAL-Treiber verwenden soll.
<code>mcuconf.h</code>	Welche Module für den ChibiOS-HAL-Treiber verwendet werden, verschiedene Clock-Einstellungen.
<code>stm32f30x.h</code>	Nur ein Wrapper für die STM32F30x standard peripheral library.
<code>stm32f30x_conf.h</code>	STM32F30x standard peripheral Konfiguration.
<code>board.h</code>	Definiert die IO-Lines, welche an den Mikrocontroller angeschlossen sind und deren Initialisierungswerte.

Tabelle 4.11: Headerübersicht

Bibliotheken

Einige Programmfunctionen wurden aus existierenden Bibliotheken verwendet. Alle verwendeten externen Libraries sind in der Tabelle 4.12 aufgelistet.

Name	Beschreibung
CMSIS	Der Cortex-Mikrocontroller-Software-Interface-Standard CMSIS stellt eine DSP-Library zur Verfügung, mit der effizienter Gebrauch der DSP-Instruktionen und der FPU gemacht werden kann. Trigonometrische Funktionen und Transformationen werden durch CMSIS realisiert.
ChibiOS	Realtime Betriebssystem: www.chibios.org
STM32F30x std periph library	Der HAL von ChibiOS ist nicht genug flexibel für die spezifische Anwendung der Timer und ADC. Die Standard-Peripheral-Library von ST stellt eine einfache, prozessorgebundene Hardware-Abstraktion zur Verfügung.

Tabelle 4.12: Übersicht über alle verwendeten externen Libraries

5 Validierung

5.1 Überblick

Ziel der Validierungsphase ist das Testen des fertigen Longboards, unter anderem anhand von Testpersonen. Der Weg bis dahin muss aber ebenfalls kontrolliert und geprüft sein. Zur Sicherstellung der Funktionalität werden die Komponenten einzeln als auch zusammen getestet. Jeder Hardware- als auch Software-Bestandteil – Brett, Steuerung, Stromversorgung und Motoransteuerung – hat also sein eigenes Testkonzept.

5.2 Brett

Das Brett selbst, gebaut aus gepressten Birkenholzplatten, dessen Bearbeitung und die darauf befindliche Stromleiter und Gehäuse wird einem Stabilitätstest unterzogen, wo geprüft wird, ob die Komponenten ein Ausreizen der Flexibilität des Brettes vertragen. Die Verbindungen werden durchgemessen. Es wurde keine Widerstandserhöhung festgestellt. Die Versuchsbedingung ist in der Tabelle 5.1 aufgeführt.

Getestete Belastung	100kg
Getestete Flexibilität	Mitte durchbiegen bis zum Boden

Tabelle 5.1: Belastung Brett

Leider ist das Brett angebrochen, nachdem die Bohrungen für die Montage der Gehäuse vorgenommen wurden.

5.3 Steuerung - Magic Glove

Für die Validierung des Magic Glove wurde eine Hand 3D-gedruckt und ein separater Empfänger mit einem Arduinoboard gebaut (siehe Abbildung 5.1), welcher die empfangenen Daten über eine serielle Verbindung an einen Computer sendet. Die Hand wird nun von Minima zu Maxima gebeugt und die gemessenen Werte aufgezeichnet. Somit kann garantiert werden, dass der maximale Bewegungsradius aufgelöst wird und die Daten korrekt gesendet werden.

Für den Test der Schaltung wird der Magic Glove mit der internen Batterie gespeist. Somit kann gewährleistet werden, dass sich die Schaltung auch im Einsatz nicht anders verhalten wird.

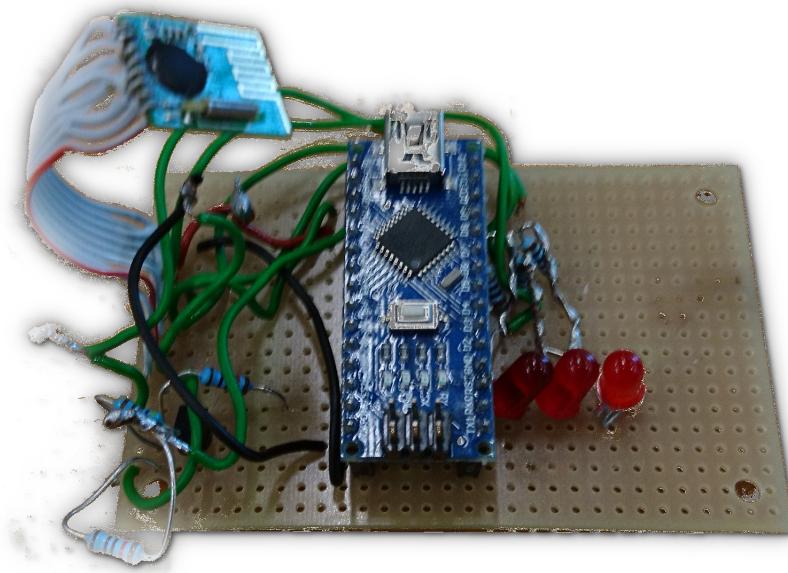


Abbildung 5.1: Empfänger zur Validierung des Magic Glove

In diesem Kapitel finden sich folgende Validierungen:

Hardware

- Validierung Batteriemanagement
- Aussteuerung Analogteil

Software

- Validierung ADC-Wandler
- Übertragungsgeschwindigkeit RFM75

5.3.1 Hardware

Validierung Batteriemanagement

Zum Testen der Ladeschaltung auf dem Magic Glove wird die verwendete Li-Ionen-Zelle mit einem Lastwiderstand auf 20% entladen und dann eine Spannung von 5 Volt an den Schaltungsteil angelegt. Nach 6 Stunden wird die Spannung des Akkumulators gemessen. Zusammengefasst sind die Messungen in der Tabelle 5.2.

Gemessene Akkuzelle (Entladen)	3.72 V
Gemessene Spannung nach 6h	4.15V

Tabelle 5.2: Messung Übertragungsgeschwindigkeit RFM75 Resultate

Aussteuerung Analogteil

Um die Aussteuerung des Operationsverstärkers im Analogteil zu messen, wird einerseits mit dem Oszilloskop die Spannung am ADC-Eingang gemessen und andererseits der gemessene Wert via UART-Kommunikation an den Computer gesendet. Die maximale Aussteuerung, die

mit dem Operationsverstärker erreicht werden konnte, beträgt 3.03V. Dies entspricht einer Aussteuerung von 91.7% am ADC-Eingang, falls dessen Referenz auf genau 3.3V ist. Zusammengefasst sind die Messungen in der Tabelle 5.3.

Gemessene min. Spannung	21.7mV
Gemessene max. Spannung	3.03V

Tabelle 5.3: ADC-Messung Magic Glove

5.3.2 Software

Validierung ADC-Wandler

Für die Validierung der Software wurde ein separater Empfänger mit einem Arduino Nano gebaut. Dieser kann über USB angesteuert werden und sendet die gemessenen Werte über UART an den angeschlossenen Computer. Nach der Aussteuerung wird der Flex-Sensor an mehrere vordefinierte Positionen bewegt. Zusammengefasst sind die Messungen in der Tabelle 5.4.

Kleinster nummerischer ADC-Wert	7
Höchster nummerischer ADC-Wert	256

Tabelle 5.4: ADC-Messung Magic Glove

Übertragungsgeschwindigkeit RFM75

Für diesen Test wird auf dem Empfänger-PCB eine LED an- beziehungsweise ausgeschaltet. Dieser Puls wird dann wiederum mit dem Oszilloskop gemessen. Ist die daraus entstehende Höchstfrequenz höher als die gewünschte Übertragungsgeschwindigkeit, kann problemlos die erforderliche Datenmenge gesendet werden. Zusammengefasst ist dieser Test in der Tabelle 5.5.

Gewünschte Sendefrequenz	100 Samples/Sekunde
Gemessene Zeit zwischen Paketen	3.58ms
Resultierende Empfangsfrequenz	ca. 279 Samples/Sekunde

Tabelle 5.5: Messung Übertragungsgeschwindigkeit RFM75 Resultate

5.4 Stromversorgung

Damit die Ergebnisse reproduzierbar sind, wird die Stromversorgung auf einem Prüfstand getestet. In der ersten Validierungsphase wird der Print und die Software ohne Akku getestet, welcher dann in der zweiten Validierungsphase angeschlossen wird. **Achtung:** Da es sich beim Akku um einen LiPo-Akku handelt, welcher einen Spitzenstrom von bis zu 500A liefert, soll der Print zuerst auf allfällige Kurzschlüsse geprüft werden.

In diesem Kapitel finden sich folgende Validierungen:

Hardware

- Funktion der FETs
- Messung der Spannungen
- Ausmessung des High-Side-Driver

Software

- Einlesen Spannung und Ströme
- PWM-Regelung
- Balancing Steuerung

5.4.1 Hardware

FETs

Als erstes werden die FETs des Balancing der Reihe nach eingeschaltet. Dabei wird jeweils eine Spannung von 4.2V zwischen Zellanschluss 6 und 5, Zellanschluss 5 und 4 usw. angeschlossen. Sobald die richtige Spannung angeschlossen ist und derjenige FET durchgeschaltet ist, soll einen Strom von 100mA fliessen. In der Tabelle 5.6 sind die Messungen zusammengefasst.

Messobjekt	Strom
Balance Zelle 1	103.1mA
Balance Zelle 2	102.7mA
Balance Zelle 3	103.5mA
Balance Zelle 4	103.4mA
Balance Zelle 5	102.9mA
Balance Zelle 6	102.8mA

Tabelle 5.6: Balancing Strom je Zelle

Weiter werden die FETs getestet, welche die Spannung zum Motorcontrollboard steuern. Dabei wird ein Leistungswiderstand auf Masse geschaltet. Das Ergebnis ist in der Tabelle 5.7 ersichtlich.

Versorgungsspannung	25V
Lastwiderstand	2.5Ω

Tabelle 5.7: Messbedingungen FETs zu Motorcontroll

Nun soll ein Strom von rund 10A fliessen. Dabei sollen die FETs nur leicht bzw. gar nicht warm werden. Die Messbedingung ist in der Tabelle 5.8 festgehalten.

Strom	10.02A
-------	--------

Tabelle 5.8: Strom durch Motorcontrol-FET

Der FET wurde bei diesem Strom kaum merklich warm.

Der FET, welcher fürs Laden zuständig ist, wird bei der Ausmessung des High-Side-Driver getestet.

Spannungsmessung

Nun werden die einzelnen Spannungsteiler ausgemessen. Dabei werden am Balancing-Anschluss jeweils die einzelnen Zellen angeschlossen. Weiter wird die Ladespannung angeschlossen. Nun soll an den Spannungsteiler gemessen werden. Der Akku soll voll geladen sein und die Spannungsteiler dürfen die Mikrocontroller-Speisung nicht überschreiten. Die Messbedingungen sind in der Tabelle 5.9 aufgelistet.

Versorgungsspannung Batterieseitig	25V
Versorgungsspannung Ladeseitig	30V
Versorgungsspannung Mikrocontroller	5V
Versorgungsspannung High-Side-Driver	15V

Tabelle 5.9: Messbedingung Spannungsmessungen

Um zu Prüfen, ob die Spannungsregelung funktioniert, wird auch diese gemessen. Die Messergebnisse finden sich in der Tabelle 5.10.

Messobjekt	Gemessene Spannung	Skalierungsfaktor
Versorgungsspannung Mikrocontroller	5.016V	-
Versorgungsspannung High-Side-Driver	15.65V	-
Zelle 1	4.132V/4.132V	1
Zelle 2	8.268V/4.077V	0.493
Zelle 3	12.34V/4.064V	0.329
Zelle 4	16.45V/4.096V	0.249
Zelle 5	20.85V/4.153V	0.202
Zelle 6	24.72V/4.120V	0.162
Spannungsteiler Ladeseitig	30.01V/2.132V	0.071

Tabelle 5.10: Gemessene Spannung

Die Skalierungsfaktoren lagen ziemlich genau an unserem gewünschten Ergebnis. Somit konnten die berechneten Werte für die Widerstände übernommen werden.

High-Side-Driver

Damit der High-Side-Driver getestet werden kann, soll die Batterie entfernt und einen Leistungs-widerstand von rund 5Ω angeschlossen werden. Der High-Side-Driver wird dabei extern mit 15V gespeist. Als erstes wird am Mikrocontroller ein PWM-Signal mit 50% Duty-Cycle ausgegeben. Dabei soll am Ausgang des High-Side-Driver dasselbe Signal mit 15V Spannungsspitze anliegen. Nun wird eine externe Ladespannung angeschlossen. Diese liegt für Testzwecke bei 15V. Dabei soll die Strombegrenzung auf 3A eingestellt sein. Nun wird der Duty-Cycle ständig erhöht.

Dabei wird der Strom beim Leistungswiderstand gemessen und mit nachfolgender Tabelle 5.11 abgeglichen.

Duty-Cycle: 0%	0A
Duty-Cycle: 20%	0.6A
Duty-Cycle: 40%	1.2A
Duty-Cycle: 60%	1.8A
Duty-Cycle: 80%	2.4A

Tabelle 5.11: Ladestrom bei verschiedenen Duty-Cycle

Einen Duty-Cycle von 100% wird nicht geprüft, da der High-Side-Driver nicht dazu ausgelegt ist, einen FET dauerhaft durch zusteuern.

Dies konnte aufgrund eines defekten High-Side-Drivers nicht evaluiert werden.

5.4.2 Software

Um diese Ergebnisse zu überprüfen, muss am UART des Mikrocontroller ein TTL-Kabel angeschlossen werden.

ADC

Um die einzelnen Zellen auszulesen, kann die Batterie sowie eine Ladespannung von mindestens 30V angeschlossen werden. Dabei werden die einzelnen Spannungen jede Sekunde einmal über den UART Port ausgegeben. Dabei soll die Spannung mit dem Multimeter gemessen und überprüft werden. Um den Strom zu messen, wird die Messung gleich wie in Kapitel 5.4.1 aufgebaut. Die gemessenen Spannungen sind in der Tabelle 5.12 aufgeführt.

ADC-Eingang	Spannung Multimeter	Spannung ADC	Differenz
Zelle 1	4.132V	4.116V	0.016V
Zelle 2	4.134V	4.121V	0.013V
Zelle 3	4.065V	4.028V	0.037V
Zelle 4	4.117V	4.106V	0.011V
Zelle 5	4.132V	4.131V	0.001V
Zelle 6	4.131V	4.111V	0.020V
Ladespannung	2.113V	2.105V	0.008

Tabelle 5.12: gemessene Spannungen

All diese Werte liegen innerhalb der Toleranz des ADCs des Mikrocontrollers und sind somit ausreichend genau.

PWM-Regelung

Um das PWM-Signal zu überprüfen, wird Softwareseitig ein vordefinierter Strom eingestellt. Dabei wird anstatt des Akkus ein veränderbarer Leistungswiderstand am Ausgang angeschlossen. Nun wird der Widerstand verändert und überprüft, ob die Software richtig nachregelt. Die Vorgabe für des PWM-Signals ist in der Tabelle 5.13 festgehalten. Das Ergebnis ist in der Abbildung 5.2 grafisch dargestellt.

Teststrom	1A
f_{PWM}	32kHz

Tabelle 5.13: Vorgabe PWM

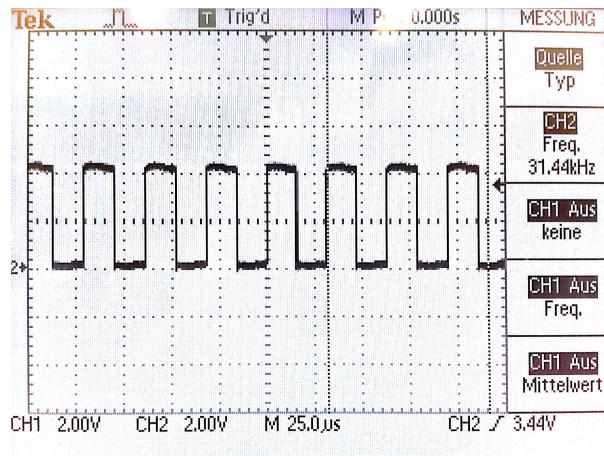


Abbildung 5.2: PWM Mikrocontroller bei 50% Duty-Cycle

Balancing-Regelung

Für diese Validierung wird jeweils eine Spannung an den Zellen-Anschlüssen angeschlossen. Dabei wird überprüft, ob die Software bei den jeweiligen Zellen die FETs durchsteuert, um die gleiche Spannung wie bei den anderen Zellen zu erreichen.

5.5 Motoransteuerung

Um reproduzierbare Ergebnisse zu erzielen, wird die Motoransteuerung auf einem Prüfstand getestet. Dazu wird der Motor ohne Last befestigt und die Schaltung von einem Netzteil gespeist. Weiter wird die Motoransteuerung in einzelnen Blöcken validiert. Dabei wird unterschieden in Hardware und Software.

Hardware

- Funktion der FETs
- Spannungsmessung mit Spannungsteiler

Software

- Einlesen der Spannungen und Ströme
- SVPWM (Raumvektormodulation)
- Positions- und Geschwindigkeitsbeobachter
- D und Q Stromregler

5.5.1 Hardware

FETs

Die FETs werden der Reihe nach von der Software eingeschaltet. Zum Test der FETs wird am Ausgang ein Lastwiderstand auf Masse geschaltet. So kann die Spannung am Ausgang gemessen und aufgezeichnet werden. Zusätzlich wird die Gatespannung gemessen. In der Tabelle 5.14 sind die Messbedingungen ersichtlich.

Versorgungsspannung	15V
Lastwiderstand	$1k\Omega$
Gatestrom Treiber	1.7A

Tabelle 5.14: Messbedingungen FETs

Zur Gunsten der Übersichtlichkeit wird nur die Messung eines FETs (Abb. 5.3)dargestellt, die zugehörigen Einstellungen sind in der Tabelle 5.15 aufgeführt.

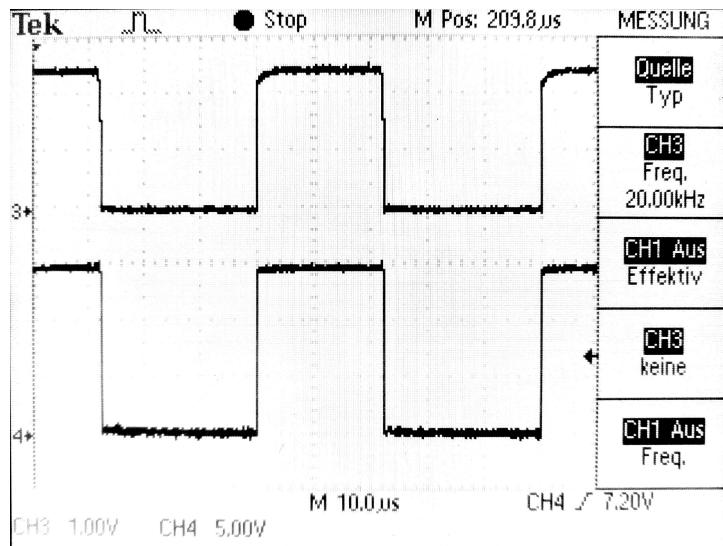


Abbildung 5.3: Einschalten High-Side-FET

Kanal 3	High FET A Gate	10x Abschwächung
Kanal 4	Spannung Phase A	1x Abschwächung

Tabelle 5.15: Scopoeinstellungen

Der Kanal 3 zeigt die Steuerspannung am Gate des FETs. Sie wird vom Treiber IC erzeugt und beträgt im eingeschalteten Zustand 24V, das sind rund 7V Gate-Source-Spannung. Dies sichert ein schnelles Durchschalten des FETs. Erkennbar ist dies an der steilen Flanke am Ausgang der Halbbrücke, zu sehen auf Kanal 4.

Spannungsmessung

Ist der Mikrocontroller im Resetzustand, sind alle FETs ausgeschaltet. So kann eine Spannung an die Ausgänge der Halbbrücke gegeben und am Ausgang der Spannungsteiler die Spannung ge-

messen werden. Diese darf bei einem bestimmten Eingangsspannungsbereich die Mikrocontroller-Speisung nicht überschreiten. Die Messbedingungen sind in der Tabelle 5.16, die Ergebnisse in der Tabelle 5.17 zusammengefasst.

Versorgungsspannungsbereich	0 bis 20V
Mikrocontroller-Speisung	3.3V
Spannungsteiler Faktor	0.0534

Tabelle 5.16: Messbedingungen Spannungsmessung

Spannung an Phase	Spannung gemessen am Mikrocontroller	Faktor
5V	0.268V	0.0536
10V	0.536V	0.0536
15V	0.804V	0.0536
20V	1.072V	0.0536

Tabelle 5.17: Spannungsmessung Spannungsteiler

Die Spannungen, gemessen nach den Spannungsteiler, entsprechen exakt dem angelegten Wert mal dem Teilungsfaktor. Dies ist erstaunlich exakt und völlig innerhalb der Widerstandstoleranzen.

5.5.2 Software

Einlesen der Spannungen

Über die Shell des Mikrocontrollers werden alle gemessenen Spannungen periodisch ausgegeben. Um die Spannungen zu messen, werden die FETs ausgeschaltet und eine Spannung an den Ausgängen angelegt. Um die Strommessung zu validieren, werden die Low-Side-FETs eingeschaltet und ein Strom an den Ausgängen eingespeist. So kann der gesamte Signalpfad der Messungen validiert werden. Die Messparameter sind in der Tabelle 5.18 zusammengefasst, die Ergebnisse finden sich in der Tabelle 5.19 und 5.20.

Testspannung	0 bis 20V
Teststrom	0 bis 2A

Tabelle 5.18: Messbedingungen Spannungsmessung Software

Spannung an Phase	Spannung gemessen vom Mikrocontroller
5V	3.7V
10V	8.5V
15V	13.3V
20V	18.0V

Tabelle 5.19: Spannungsmessung Software

Die Spannungswerte weichen stark von den Sollwerten ab. Dies ist aber nicht weiter tragisch, da viel mehr der Spannungsunterschied von Bedeutung ist. Zudem wird beim FOC Verfahren nur die Versorgungsspannung gemessen.

Strom	Gemessen vom Mikrocontroller
0.5A	0.587A
1A	1.007A
1.5A	1.511A
2A	1.930A

Tabelle 5.20: Strommessung Software

Auch bei den Stromwerten ist der relative Unterschied wichtiger als der Absolutwert.

SVPWM Raumvektormodulation

Für diese Validierung wird eine sinusförmige Spannung mit konstanter Frequenz und Amplitude berechnet und der SVPWM-Routine übergeben. Die Ein- und Ausgabedaten werden für eine Zeitdauer aufgezeichnet und dann an den Computer übertragen, wo sie dargestellt werden.

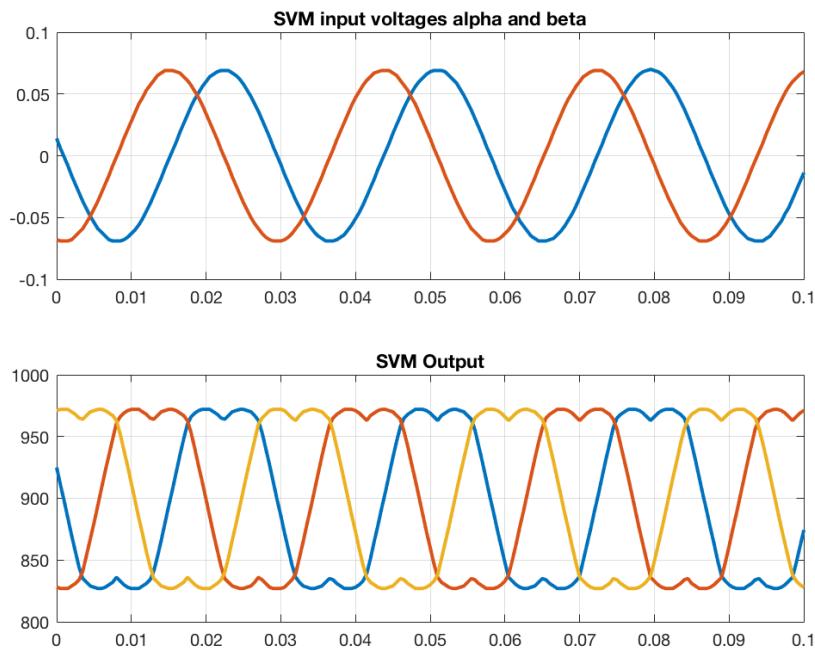


Abbildung 5.4: Validierung SVPWM Raumvektormodulation

Im unteren Plot der Abbildung 5.4 ist sehr gut der dreiphasige Sinus mit dritter harmonischer Welle zu sehen. Diese Werte entsprechen den Duty-Cycle der MOSFETs.

Positions- und Geschwindigkeitsbeobachter

Nun wird der Motor zwangskommutiert. Das heisst, dass wieder eine sinusförmige Spannung mit konstanter Frequenz und Amplitude berechnet und auf die Halbbrücke geführt wird. Der Motor dreht nun mit einer konstanter Drehzahl. Die Ausgangswerte der Positions- und Geschwindigkeitsbeobachter werden erneut aufgezeichnet und mit dem Computer ausgewertet. In der Tabelle 5.21 sind die Messbedingungen aufgelistet.

$v_{d, set}$	0.0
$v_{q, set}$	0.07
f_{set}	35.0 Hz
Versorgungsspannung	15V

Tabelle 5.21: Messbedingungen Spannungsmessung Software

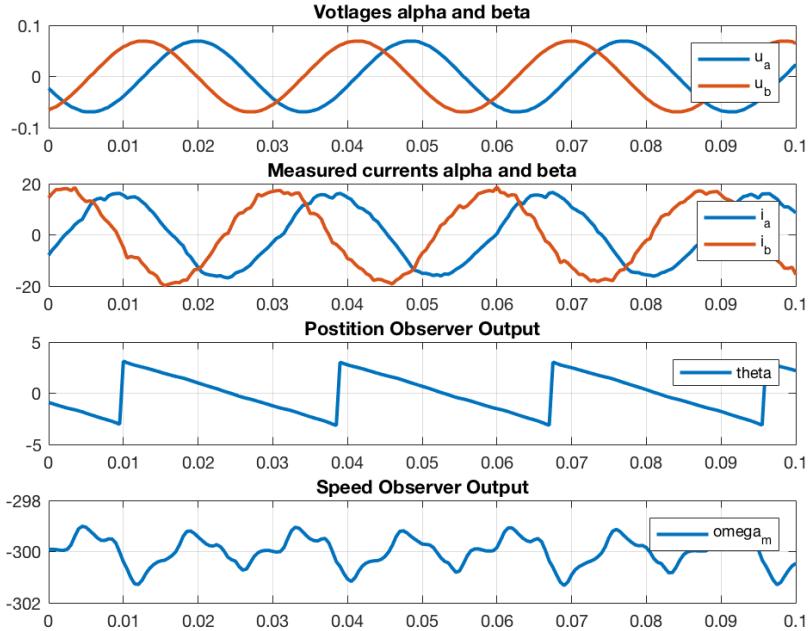


Abbildung 5.5: Validierung Positions- und Geschwindigkeitsbeobachter

Abbildung 5.5 zeigt in den ersten beiden Plots die Eingabewerte des Positionsbeobachters. Die gemessenen Ströme sind ungefiltert dargestellt und nur von kleinem Rauschen behaftet. Die Positionsschätzung θ hat ebenfalls nur einen kleinen Rippel und kann gut für die benötigten Transformationen verwendet werden. Was in diesem Versuch nicht validiert werden kann, ist der Phasenversatz zwischen wahrem und geschätztem Winkel. Dazu wären unter Anderem ein Drehgeber am Motor und eine Datenauswertung nötig, welche zeitkritisch die berechneten und gemessenen Größen aufzeichnen kann.

Die Schätzung der Drehzahl schwingt um den Mittelwert von 300rpm, was gemäss Formel 5.1 genau dem eingestellten Wert entspricht:

$$\omega_m[\text{rpm}] = \frac{60 \cdot \omega_e}{p} = \frac{60 \cdot 35\text{Hz}}{7} = 300\text{rpm} \quad (5.1)$$

Die Welligkeit wird vor dem Verwendung für den Stromregler mit einem Tiefpass gefiltert.

D und Q Stromregler

Der letzte Validierungsschritt für die Motorsteuerung auf dem Prüfstand ist die Überprüfung der D und Q Stromregler. Wie bei Validierungsschritt 5.5.2 werden die Daten der Regler auf

dem Mikrocontroller zwischengespeichert und anschliessend auf dem Computer dargestellt. Bei dieser Validierung wird der Motor im Closed-Loop-Modus betrieben. Es wird softwaremässig ein Sollwertsprung ausgeführt. Die Messbedingungen sind in der Tabelle 5.22 dargestellt.

$i_{d, \text{set}}$	0
$i_{q, \text{set}}$	3 auf 5
Versorgungsspannung	21V

Tabelle 5.22: Messbedingungen D und Q Stromregler

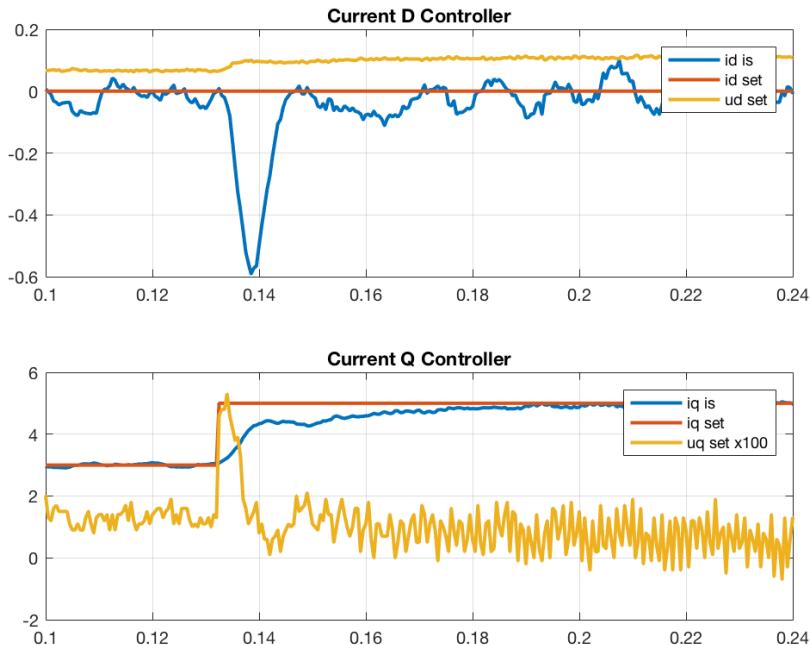


Abbildung 5.6: Validierung D und Q Stromregler

Abbildung 5.6 zeigt die Sprungantwort der beiden Stromregler. Beide sind nicht perfekt und könnten noch optimiert werden. Da der Motor jedoch bei hohen Drehzahlen noch nicht dreht, wurde auf eine ausführliche Optimierung verzichtet. Was jedoch bestätigt werden kann ist, dass der Regler eine schnelle Sprungantwort hat und dank dem I-Anteil keine bleibende Regelabweichung vorhanden ist.

Fazit

Die meisten Komponenten der Motorsteuerung funktionieren. Der Motor dreht jedoch nur bei langsamen Drehzahlen (kleiner 1000rpm) doch mit hohem Drehmoment. Der Fehler liegt in der Implementierung des Stromreglers. Vermutlich ist es eine Sache der richtigen Parameter. Da das Umsetzen sehr zeitaufwändig ist, wurde auf den Feinschliff verzichtet.

5.6 Gesamtvalidierung

Nach eingehendem Testen der einzelnen Komponenten wird das Gesamtprodukt getestet. Dabei wird primär die Lauffähigkeit des Longboards und die Zusammenarbeit der einzelnen Komponenten untereinander untersucht. Insbesondere werden die im Pflichtenheft festgelegten Kriterien überprüft.

Funktioniert und validiert	Nicht funktionsfähig
<ul style="list-style-type: none">• Stereung Magic Glove• Stromversorgung: Hardware, Verbindung zur Motoransteuerung• Motoransteuerung: Hardware, Positionsbeobachter, SVPWM	<ul style="list-style-type: none">• Stromversorgung: Laderegelung und Balancing• Motoransteuerung: Motorregelung

Tabelle 5.23: Zusammenfassung der funktionierenden Komponenten

Tabelle 5.23 fasst zusammen, welche Komponenten funktionieren und welche nicht. Die Steuerung mittels Magic Glove ist voll funktionsfähig und getestet. Um die Stromversorgung fertigzustellen, muss die Regelung des Ladevorgangs in der Software implementiert werden, sowie auch die Implementierung des Balancing. Die Hardware funktioniert und wurde validiert. Dasselbe ist der Fall bei der Motoransteuerung: Die Hardware wurde validiert und funktioniert einwandfrei. Das Einlesen der Ströme sowie das Schätzen der Rotorposition wurde ebenfalls validiert. Der Algorithmus zur Regelung des Motors ist auch implementiert, doch war keine Zeit mehr vorhanden für das Einstellen der Parameter. Der Motor dreht langsam und mit viel Drehmoment, doch bei hohen Drehzahlen blockiert er.

Alltagstauglichkeit

Wie bereits angedeutet, soll das fertige Board anhand von Testpersonen unterschiedlicher Skate-Erfahrung getestet werden. Die sanfte Anfahrmöglichkeit, die Manövrierfähigkeit und das allgemeine Wohlbefinden des Skaters sind die entscheidenden Kriterien. Testpersonen werden zufällig ausgewählt und die Bewertungen erfolgen nach subjektiver Einschätzung. Da das Longboard nicht fahrtüchtig ist, kann die Alltagstauglichkeit nicht getestet werden.

6 Schlusswort

Während dieses intensiven, lehrreichen Projektes wurde ein elektronisches Skateboard entwickelt. Dabei wurde insbesondere eine innovative Steuerung erwartet. Diese Erwartung wurde erfüllt dank des Magic Gloves. Damit lässt sich die Geschwindigkeit des Skateboardes mit nur einem Finger durch Beugung und Streckung regulieren. Die Daten des dazu benötigten Flex-Sensors werden über Radiofunk zur Motoransteuerung geschickt. Die Motoransteuerung setzt diese Wunschbeschleunigungen um dank einer sensorlosen Feldorientierten Regelung(FOC). Diese Ansteuerung ermöglicht ein besonders sanftes Anfahren und dynamische Reaktionen auf Laständerungen. Versorgt wird das System dank eines LiPo-Akkus. Dieser soll zum Laden nicht vom Skateboard gelöst werden müssen, deshalb haben wir einen eigenen, im Skateboard integrierten Akkulader umgesetzt. Der Magic Glove verfügt über eine eigene Knopfbatterie.

Die Umsetzung des Magic Gloves gelang ziemlich zufriedenstellend. Der Magic Glove kommuniziert kabellos mit der Motoransteuerung über das 2.4GHz-Funknetz. Die Messung der gewünschten Beschleunigung erfolgt mithilfe des Flex-Sensors. Diese beiden Bereiche funktionieren. Die Genauigkeit der Messung müsste noch in Feldversuchen, welche aufgrund der nicht funktionierenden Motoransteuerung nicht durchgeführt werden konnten, optimiert werden. Das Steuerungskonzept wurde als innovative Idee bestätigt.

Die Ansteuerung des Motors bereitete einige Schwierigkeiten. Die sensorlose FOC konnte zwar softwaretechnisch prinzipiell gut umgesetzt werden, bei der effektiven Ansteuerung des Motors tauchten jedoch unvorhergesehene Probleme auf. Es stellte sich heraus, dass die Ansteuerung schwieriger als gedacht ist. Ein Problem sind dabei die grösstenteils unbekannten Motorparameter, die für die Einstellung des PID-Reglers benötigt werden. Wir konnten leider keine zufriedenstellende Lösung finden. Läuft der Motor zwangskommutiert, funktioniert die Ansteuerung. Geregelt, d.h. im closed loop, gibt es noch zu viele Probleme, was bedeutet, dass die Regelstruktur nicht zufriedenstellend umgesetzt ist. Es konnte sichergestellt werden, dass der Observer der Software läuft. Auch die Hardware konnte erfolgreich getestet und bestätigt werden.

Die Testversuche haben gezeigt, dass das Konzept selbst funktionieren würde, im Vergleich zu anderen Umsetzungen kann der Motor mit der FOC-Regelung auch bei tiefer Drehzahl gut betrieben werden.

Das selbtkonzipierte Akkuladegerät funktioniert neben der grundlegenden Laderegelung (erst laden mit konstantem Strom, anschliessend mit konstanter Spannung) über ein Balancing-System. Dies sorgt dafür, dass die einzelnen Zellen gleichmäßig geladen werden. Der Prototyp funktioniert leider noch nicht wie gewünscht. Die Hardware funktioniert grösstenteils, das Balancing und das Laden konnte in Testläufen bestätigt werden. Die Software funktioniert nicht zufriedenstellend, die Regelung der PWM ist nicht fertig implementiert. Die Unterbereiche der Software sind zwar einzeln ansteuerbar, funktionieren jedoch im Zusammenspiel noch nicht 100%.

Als weiterführende Arbeit könnte die Motoransteuerung verbessert und zur Funktionstüchtigkeit gebracht werden. Dazu könnte unser Konzept als Grundlage genutzt werden. Weiter könnte für den Akkulader die Software funktionstüchtig gemacht werden (Implementierungsfehler ausmerzen, Regler fertig implementieren). Anschliessend müsste der gesamte Akkulader einem Test unterzogen werden. Der Magic Glove könnte kompakter gestaltet werden, so dass der Tragkomfort gesteigert werden kann. Die Messwerte könnten besser gefiltert werden. Zur feineren Steuerung bräuchte es einige Optimierungen in der Regelung.

Insgesamt startete das Projekt sehr dynamisch und wurde mit viel Einsatz durchgezogen. Die Probleme in der Motoransteuerung sind enttäuschend, insbesondere da keine klare Ursache gefunden werden konnte und das Konzept nicht der Fehler zu sein scheint. Dadurch kann das Skateboard nicht als Ganzes getestet und ausprobiert werden, was doch das grosse Ziel war, auf das hingearbeitet wurde. Nichtsdestotrotz wurden viele Unterziele zufriedenstellend Erreicht.

Literaturverzeichnis

- [1] Wikipedia. Bürstenloser gleichstrommotor. [Online]. Available: https://de.wikipedia.org/wiki/B%C3%BCrstenloser_Gleichstrommotor#Kommutierung
- [2] G. Babiels, *Elektrische Antriebe in der Fahrzeugtechnik: Lehr- und Arbeitsbuch*, 3rd ed. Springer Vieweg, 2014.
- [3] image50.png (png-grafik, 1197 auf 645 pixel). [Online]. Available: <https://www.intechopen.com/source/html/17059/media/image50.png>
- [4] J. L. et al, “Sensorless control of surface-mount permanent-magnet synchronous motors based on a nonlinear observer,” *IEEE TRANSACTIONS ON POWER ELECTRONICS*, vol. 25, no. 2, 02.02.2010.
- [5] Wikipedia. (11.05.2017) Clarke-transformation. [Online]. Available: <https://de.wikipedia.org/wiki/Clarke-Transformation>
- [6] P. Yedamale, “Feldorientierte steuerung ohne sensor,” *elektronik industrie*, no. 12, pp. 38–41, 2008.
- [7] Wikipedia. d/q-transformation - wikipedia. [Online]. Available: <https://de.wikipedia.org/wiki/D/q-Transformation>
- [8] H. Looser, “M4 kraftmesstechnik,” 2014, unpubliziertes Dokument.
- [9] MultiCircuitBoards. (2017) Leiterbahn / strombelastbarkeit. [Online]. Available: <https://www.multi-circuit-boards.eu/leiterplatten-design-hilfe/oberflaeche/leiterbahn-strombelastbarkeit.html>
- [10] Atmel. Atmega48a/pa/88a/pa/168a/pa/328/p. [Online]. Available: http://www.atmel.com/images/Atmel-8271-8-bit-AVR-Microcontroller-ATmega48A-48PA-88A-88PA-168A-168PA-328-328P-datasheet_Complete.pdf

A Anhang

A.1 Bedienung

Im Folgenden wird die Bedienung des Commute beschrieben. Dabei wird erst die Inbetriebnahme beschrieben, anschliessend der alltägliche Gebrauch und zuletzt wird erklärt, wie der Akku geladen werden kann.

Erste Inbetriebnahme

Bei der ersten Inbetriebnahme des Commute muss der Magic Glove kalibriert werden. Insbesondere wird dabei die Ruhestellung der Hand definiert. Der Nutzer drückt drei Sekunden auf den Taster, dann zeigen die LEDs die Ruhestellung an. Das heisst, alle LEDs leuchten, diejenigen in der Mitte am stärksten. Nun hält der Nutzer seine Hand in Ruhestellung und zwar so, wie er gerne auf dem Longboard steht. Dies wird als Nullposition definiert. Anschliessend beginnen die LEDs der Reihe nach zu blinken – in eine Richtung führt der Nutzer dazu eine Bremsbewegung aus, in die andere Richtung die Bewegung, um zu Beschleunigen. Danach ist die Kalibration abgeschlossen. Nun kann das Commute genutzt werden.

Alltägliche Handhabung

Auf dem Longboard befindet sich ein On/Off-Schalter, ebenso auf dem Magic Glove. Werden diese angeschaltet, ist das Commute betriebsbereit und der Nutzer kann losfahren. Entfernt sich der Nutzer mehr als drei bis vier Meter vom Bord, ist die Kommunikationsverbindung zwischen der Steuerung und der Antriebstechnik unterbrochen, und das Commute schaltet sich automatisch aus. Während der Fahrt wird dem Nutzer mithilfe der LEDs der Batteriestand des LiPo-Akkus angezeigt. Zudem wird die Batterie des Magic Gloves überwacht und auch über die LEDs angezeigt.

Akku aufladen

Der Akku kann praktisch über einen Stecker am Longboard aufgeladen werden. Wie gesagt muss er dafür nicht herausgelöst werden. Während des Ladevorgangs zeigen drei LEDs am Longboard den Ladestand an. Diese LEDs leuchten auch kurz auf, wenn das Longboard eingeschaltet wird und zeigen damit den Akkustand.

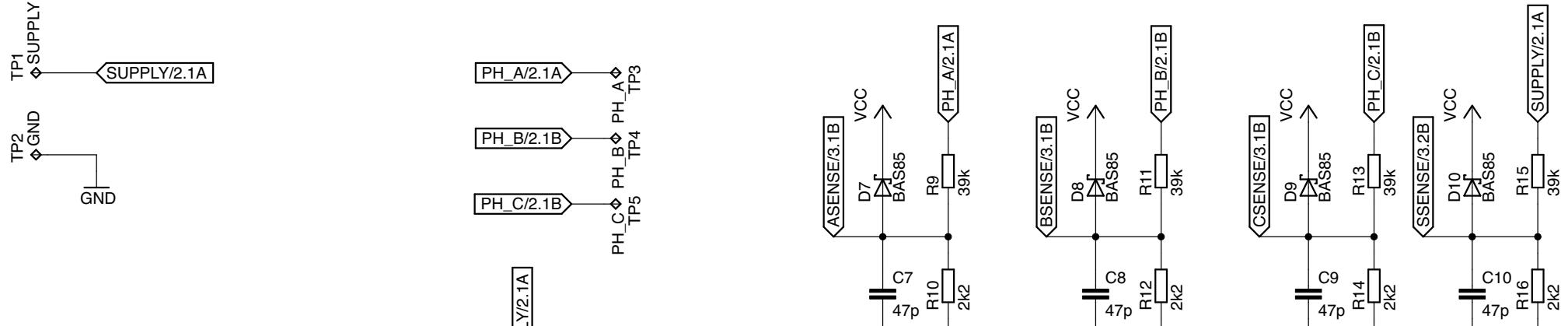
A.2 Quellcode

Der Quellcode und die Dokumentation ist unter folgenden Links aufrufbar.

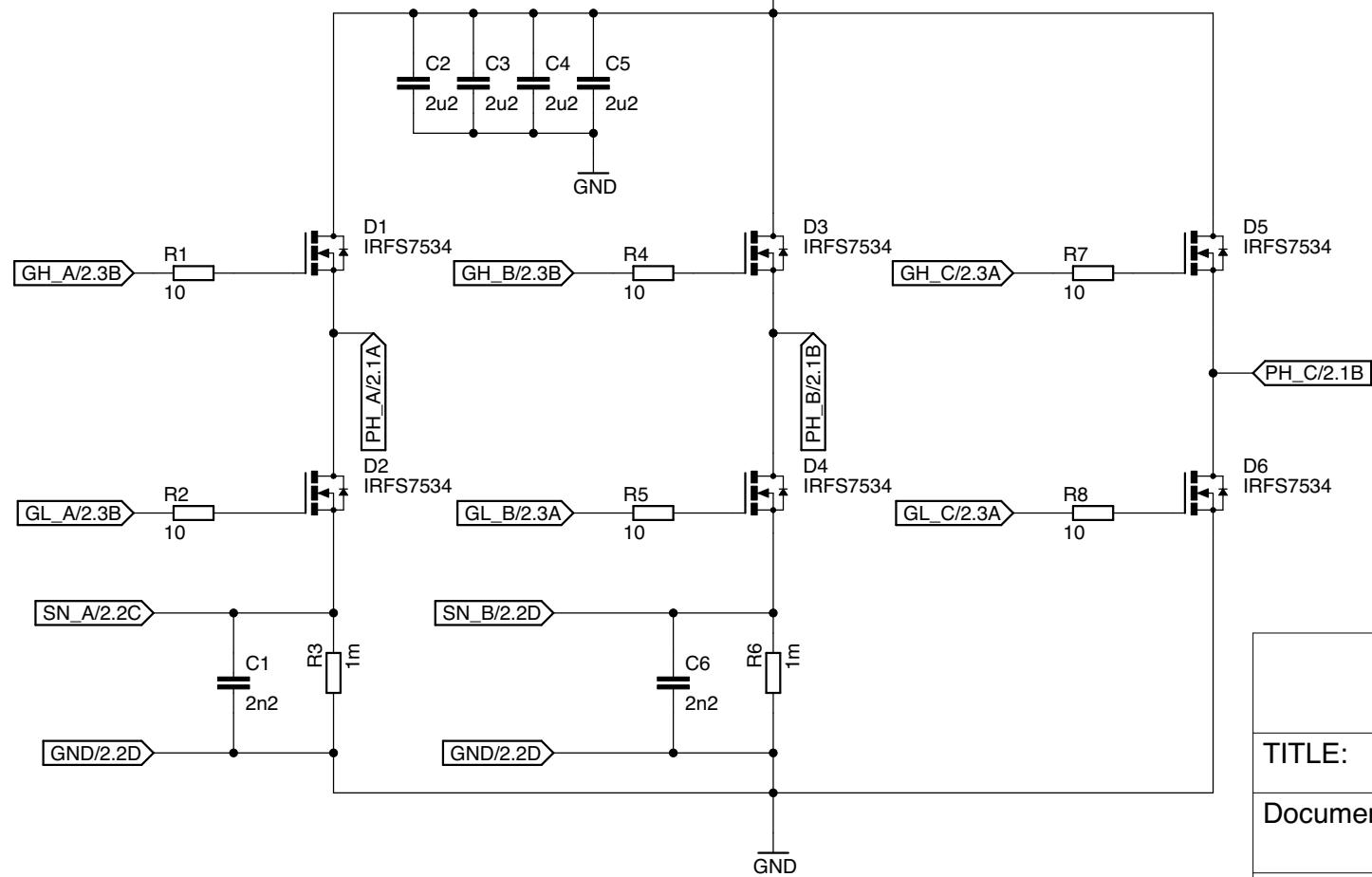
Hardware	https://github.com/noah95/skatemate-hw
Software	https://github.com/noah95/skatemate-sw
Technischer Fachbericht	https://noah95.github.io/skatemate-fachbericht/
Motorcontrol Software Dokumentation	https://noah95.github.io/skatemate-sw/

Zusätzlich liegen die Dokumente auf dem USB-Stick dieser Dokumentation bei.

A.3 Schemati



From Reference design DRV8301



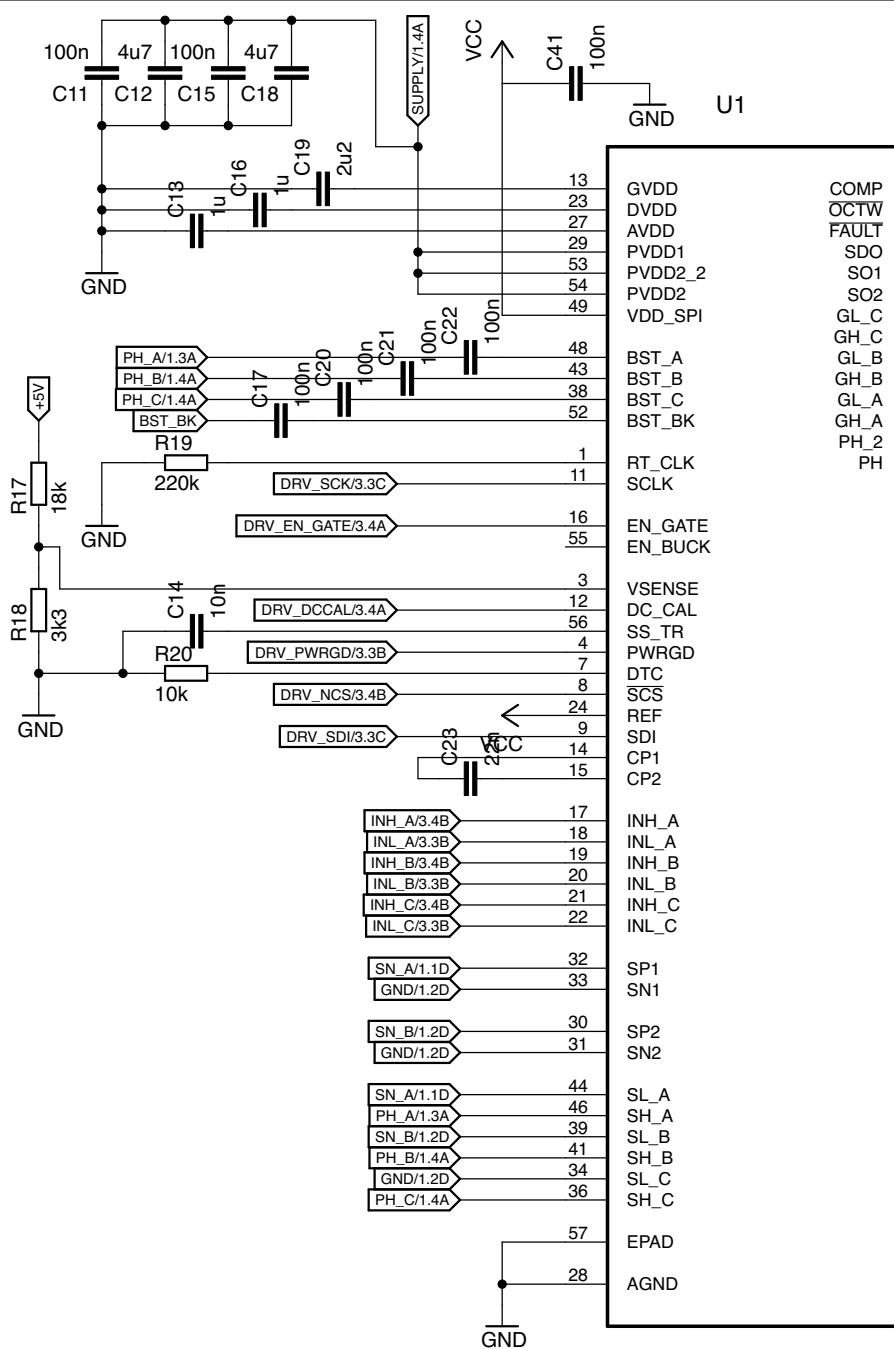
TITLE: skatemate_proto

Document Number:

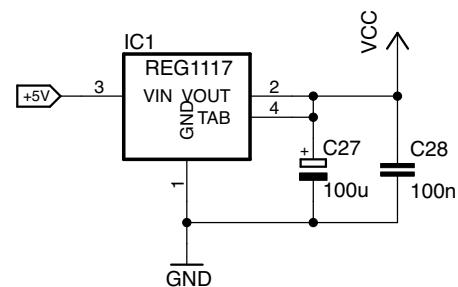
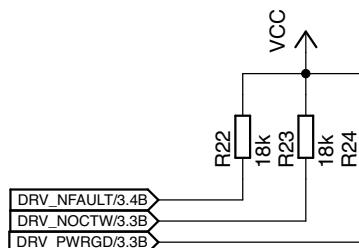
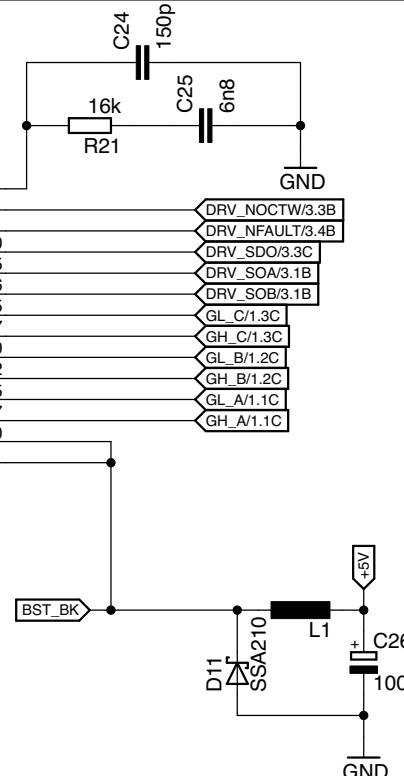
REV:

Date: 11/06/17 21:27

Sheet: 1/3



DRV8301DCAR



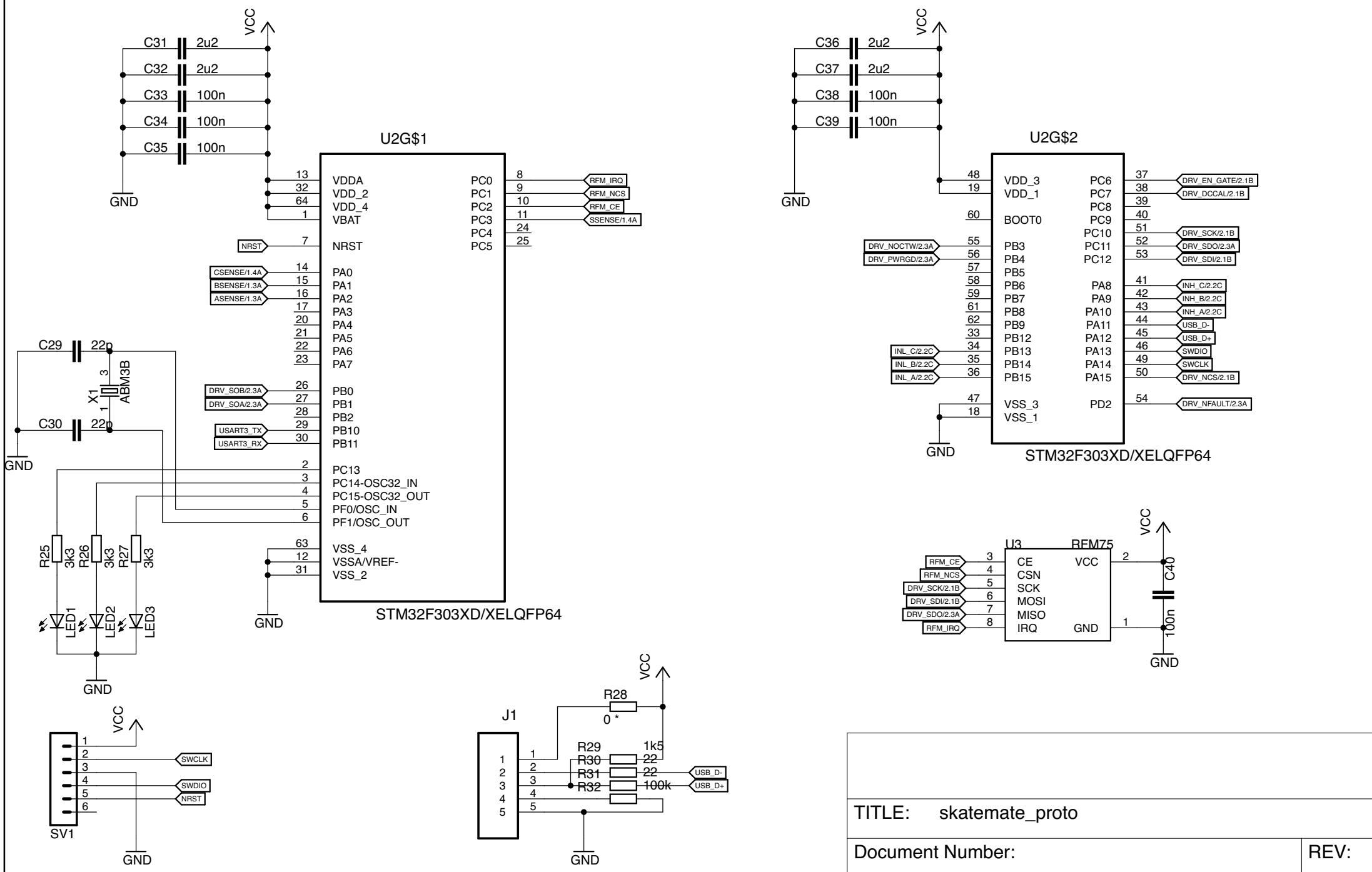
TITLE: skatemate_proto

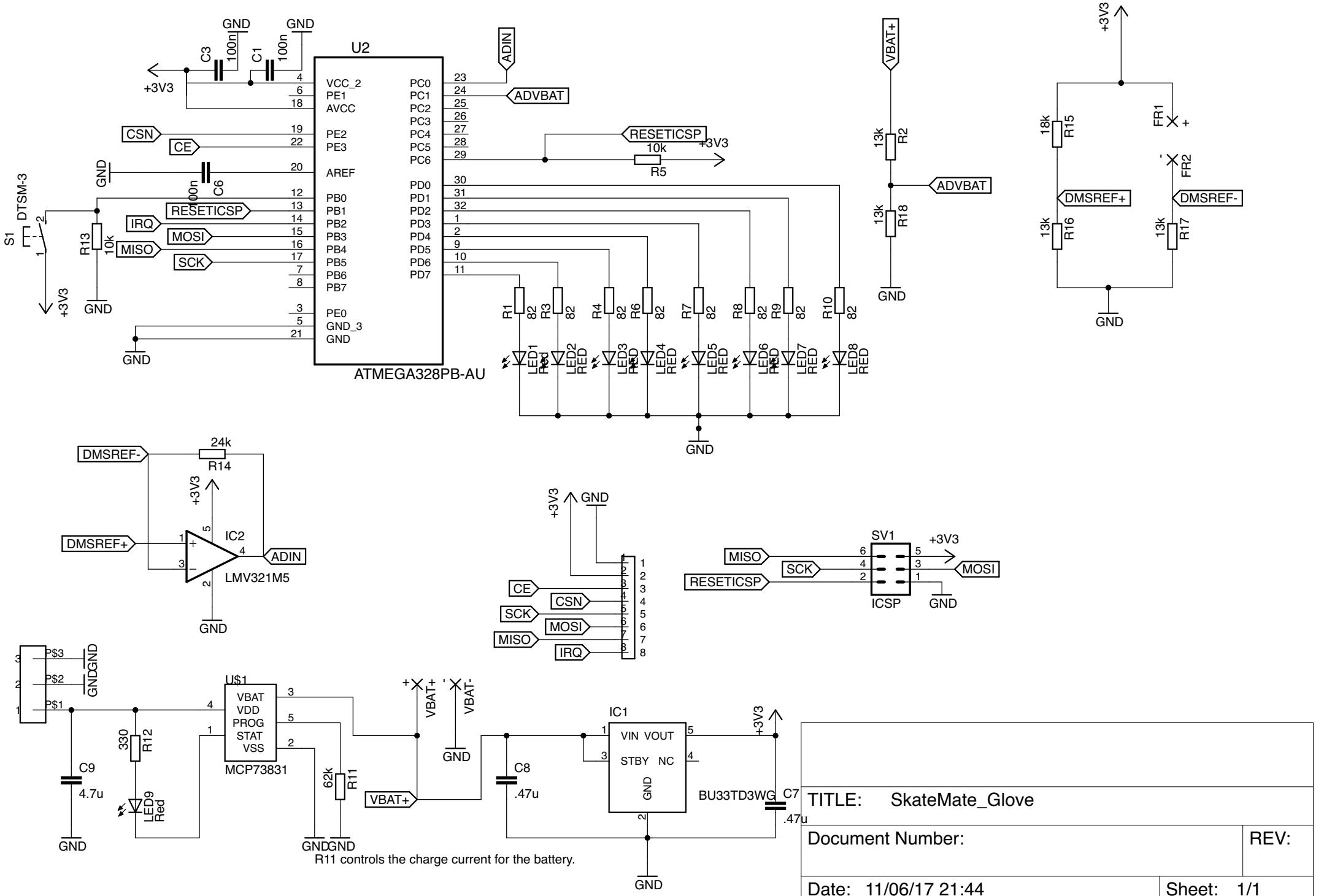
Document Number:

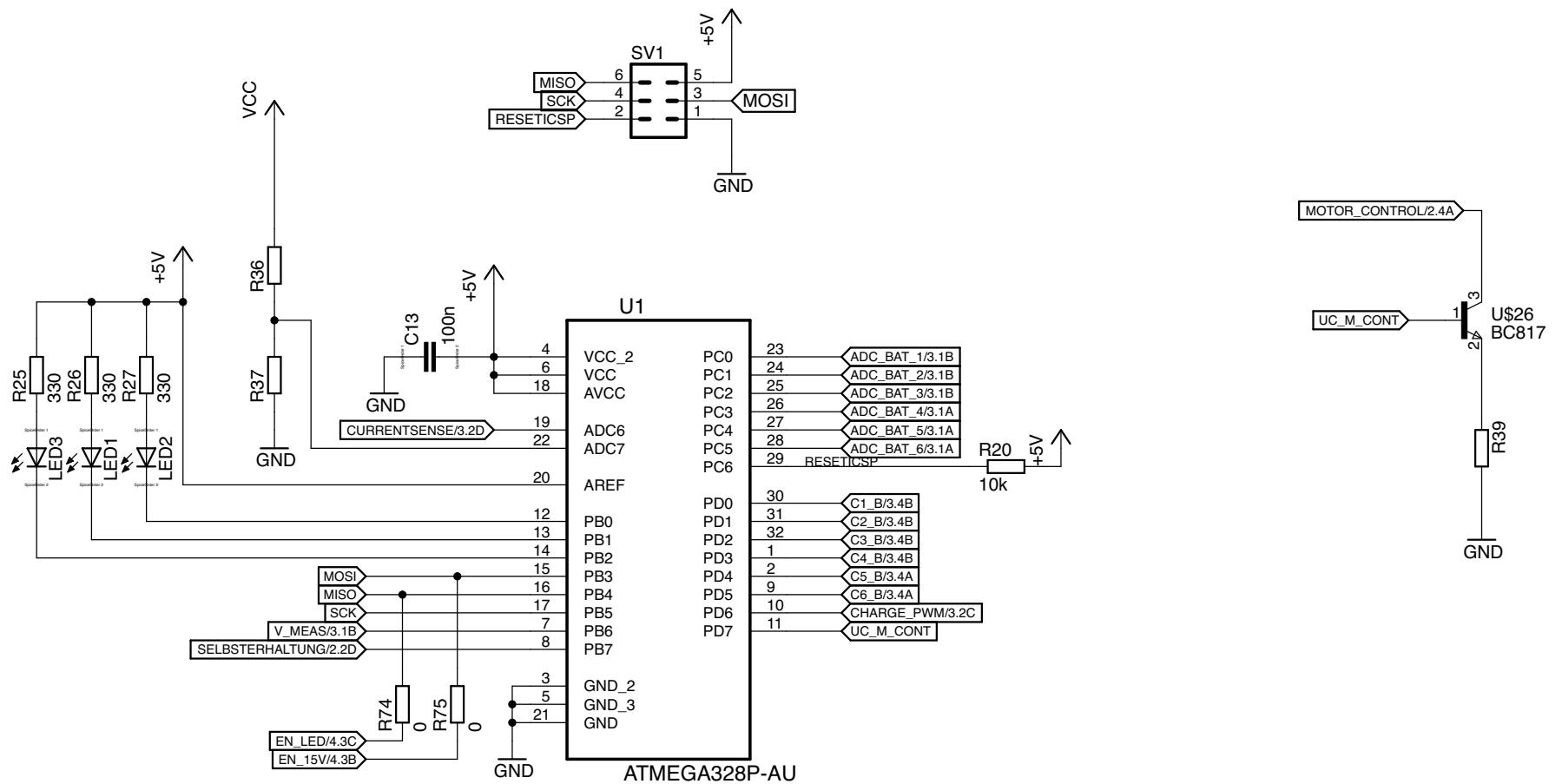
REV:

Date: 11/06/17 21:27

Sheet: 2/3







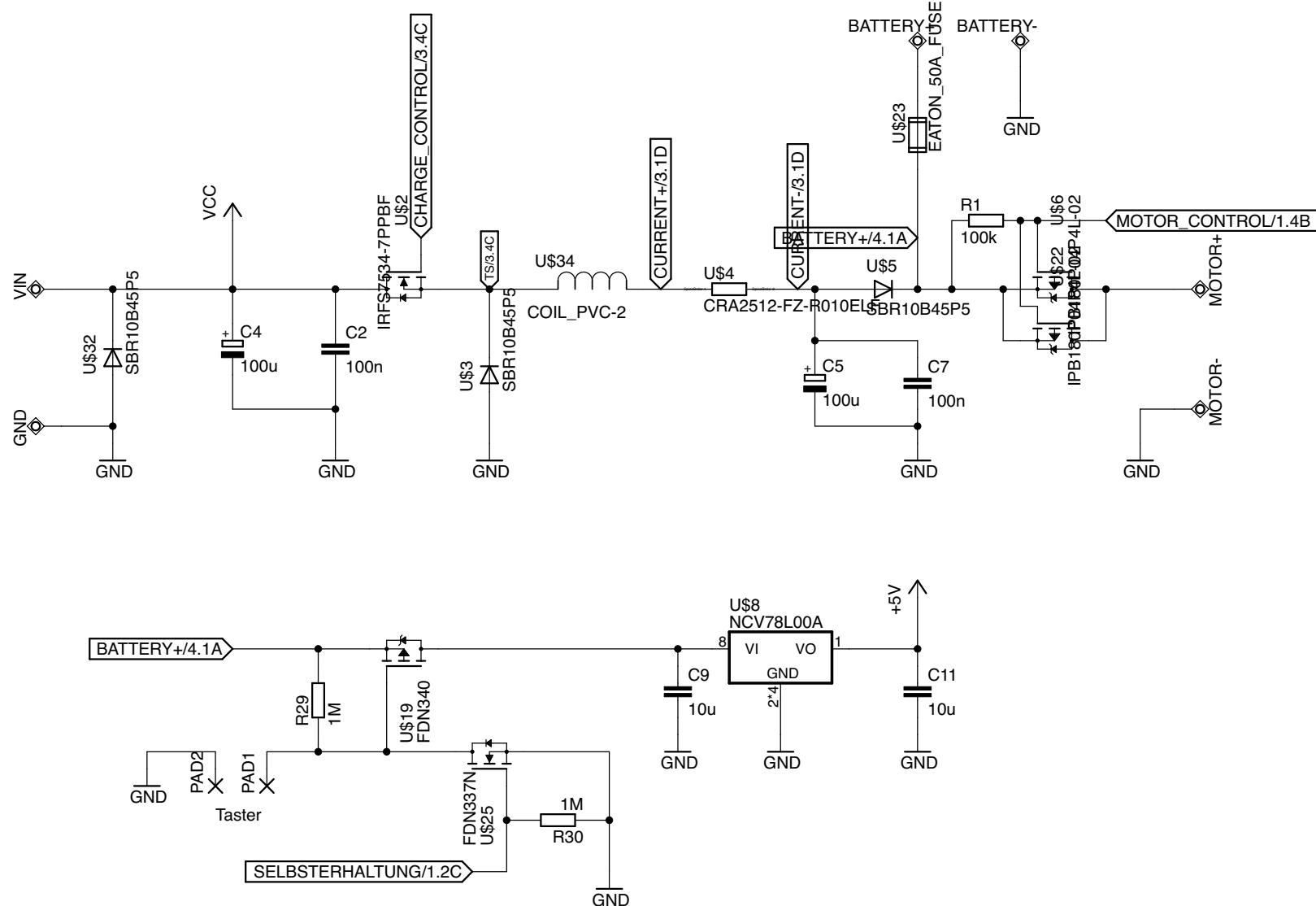
TITLE: BM_v2

Document Number:

REV:

Date: 11/06/17 21:54

Sheet: 1/4



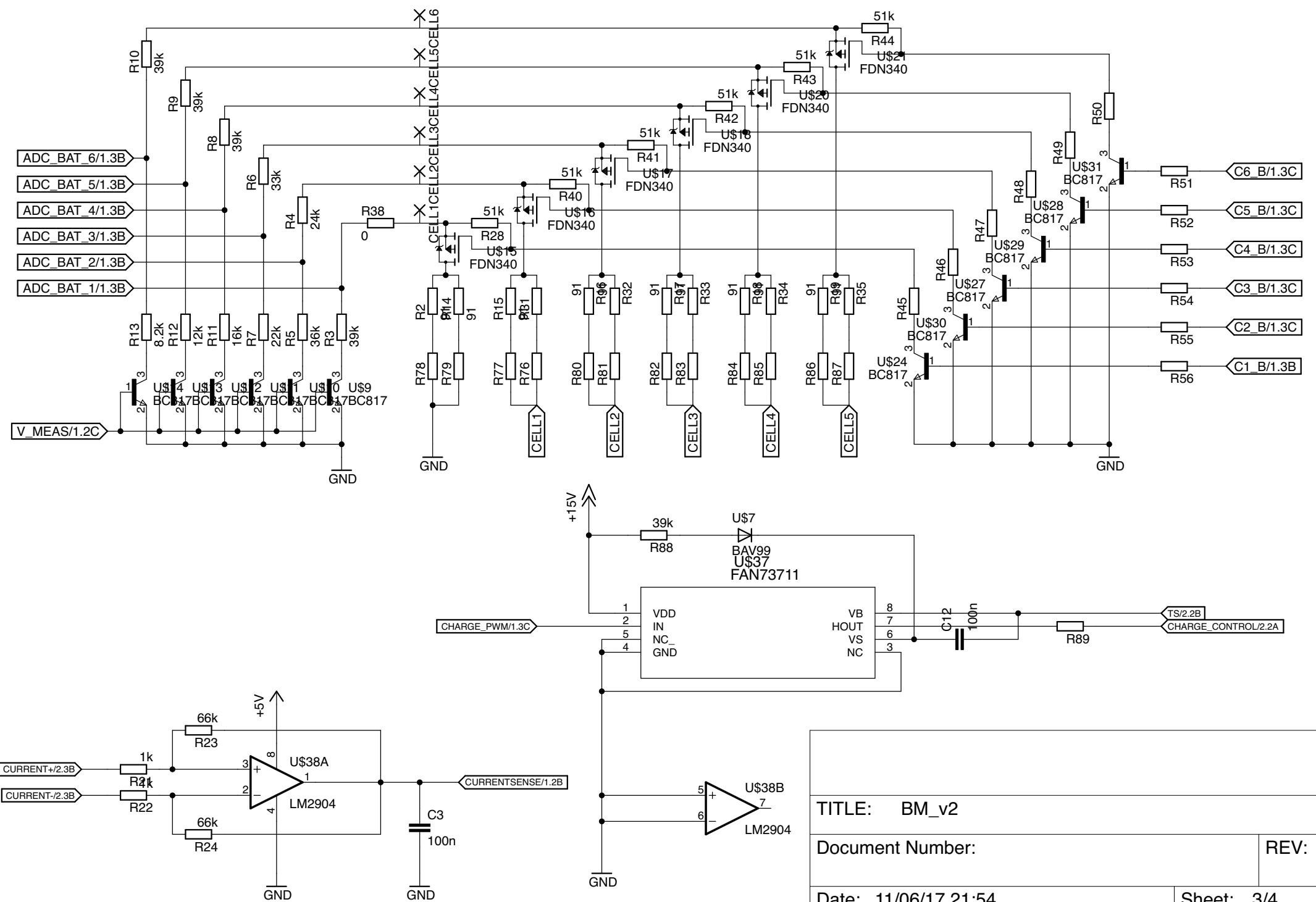
TITLE: BM_v2

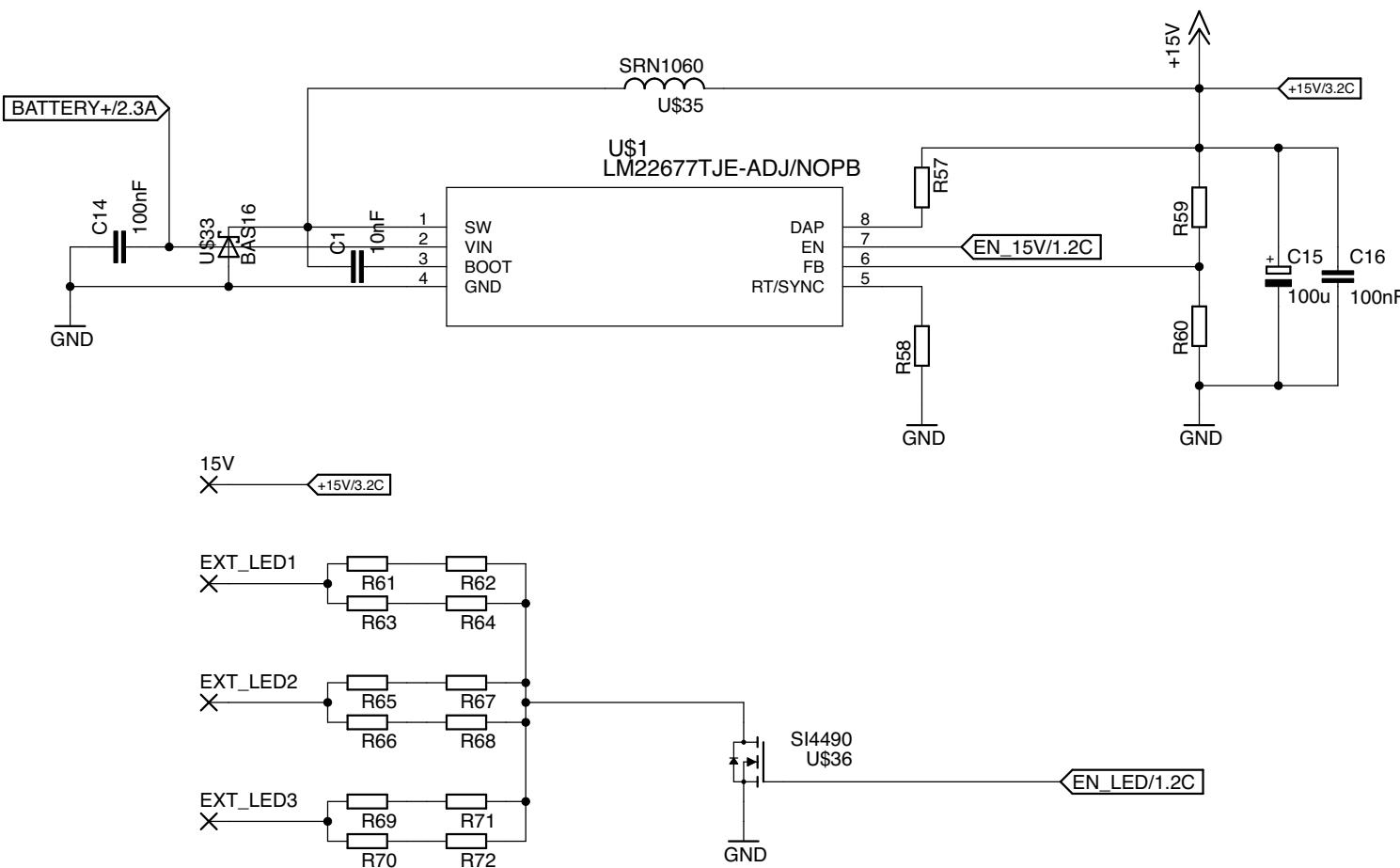
Document Number:

REV:

Date: 11/06/17 21:54

Sheet: 2/4





TITLE: BM_v2

Document Number:

REV:

Date: 11/06/17 21:54

Sheet: 4/4