

# Diplomarbeit

Höhere Technische Bundeslehr- und Versuchsanstalt Salzburg  
Abteilung für Elektrotechnik

## Entwicklung eines emissionsfreien Sportmotorrades

### Entwicklung der Zentralsteuerung / Projektleitung

Martin Kronberger 5AHET Betreuer: Dipl.-Ing. (FH) Johannes Ferner

### Entwicklung des Antriebssystems

Jakob Lackner 5AHET Betreuer: Prof. Dipl.-Ing. MBA Adolf Reinhart

### Entwicklung des Akkusystems

Simon Kern 5AHET Betreuer: Prof. Dipl.-Ing. Reinhold Benedikter

### Entwicklung der mechanischen Komponenten

Tobias Schmeisser 5AHET Betreuer: Prof. Dipl.-Ing. Peter Lindmoser



## Eidesstaatliche Erklärung

Wir erklären an Eides statt, dass wir die vorliegende Arbeit selbstständig und ohne fremde Hilfe verfasst, andere als die angegebenen Quellen nicht benutzt und die den benutzten Quellen wörtlich und inhaltlich entnommenen Stellen als solche kenntlich gemacht haben. Wir versichern, dass wir dieses Diplomarbeitsthema bisher weder im In- noch im Ausland (einer Beurteilerin oder einem Beurteiler) in irgendeiner Form als Prüfungsarbeit vorgelegt haben.

## Gendererklärung

Aus Gründen der besseren Lesbarkeit wird in dieser Diplomarbeit die Sprachform des generischen Maskulinums angewendet. Es wird an dieser Stelle darauf hingewiesen, dass die ausschließliche Verwendung der männlichen Form geschlechtsunabhängig verstanden werden soll.

---

Martin Kronberger

Ort, Datum

---

Jakob Lackner

Ort, Datum

---

Simon Kern

Ort, Datum

---

Tobias Schmeisser

Ort, Datum



# Vorwort

In immer mehr Großstädten werden Fahrzeuge mit Verbrennungsmotoren verboten. Viele Motorräder und Autos können nicht mehr produziert werden, da sie die immer strenger werdenden Abgasnormen nicht mehr einhalten können und das Thema der Klimaerwärmung wird immer präsenter und immer mehr Menschen versuchen ihren „carbon footprint“ zu verkleinern.

Doch leider gibt es für Motorradfahrer zumeist keine wirklichen Möglichkeiten, um für ihr Hobby auf eine emissionsfreie Alternativen umzusteigen. Denn zumeist ist das Preis-Leistungsverhältnis, oder auch das Produkt selbst, nicht sehr einladend. Daher ist unser Ziel die Entwicklung in diesem Bereich voranzutreiben und dadurch den Markt zu vergrößern, wodurch immer mehr und bessere Produkte angeboten werden können.



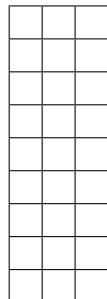
# Danksagung

TEXT DANKSAGUNG



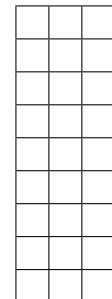
# DIPLOMARBEIT

## DOKUMENTATION



# DIPLOMA THESIS

## DOCUMENTATION





# Erklärung

Die unterfertigten Kandidaten haben gemäß §34 (3) SchUG im Verbindung mit §22 (1) Zi. 3 lit. b der Verordnung über die abschließenden Prüfungen in den berufsbildenden mittleren und höheren Schulen, BGBl. II Nr. 70 vom 24.02.2000 (Prüfungsordnung BMHS), die Ausarbeitung einer Diplomarbeit mit der umseitig angeführten Aufgabenstellung gewählt. Die Kandidaten nehmen zur Kenntnis, dass die Diplomarbeit in eigenständiger Weise und außerhalb des Unterrichtes zu bearbeiten und anzufertigen ist, wobei Ergebnisse des Unterrichtes mit einbezogen werden können. Die Abgabe der vollständigen Diplomarbeit hat bis spätestens

03.04.2020

beim zuständigen Betreuer zu erfolgen. Die Kandidaten nehmen weiters zur Kenntnis, dass gemäß §9 (6) der Prüfungsordnung BMHS nur der Schulleiter bis spätestens Ende des vorletzten Semesters den Abbruch einer Diplomarbeit anordnen kann, wenn diese aus nicht beim Prüfungskandidaten / bei den Prüfungskandidaten gelegenen Gründen nicht fertiggestellt werden kann.

Kandidaten / Kandidatinnen	Unterschrift
Martin Kronberger	
Jakob Lackner	
Simon Kern	
Tobias Schmeisser	

---

Prof. Dipl.-Ing. Reinhold Benedikter  
Prüfer

---

Prof. Dipl.-Ing. (FH) Johannes Ferner  
Prüfer

---

Prof. Dipl.-Ing. Adolf Reinhart, MBA  
Prüfer

---

Prof. Dipl.-Ing. Peter Lindmoser  
Prüfer

---

Prof. Dipl.-Ing. (FH) Roland Holzer  
Abteilungsvorstand

---

Dipl.-Ing. Dr.tech. Franz Landertshamer  
Direktor

# Inhaltsverzeichnis

<b>I Einführung</b>	<b>2</b>
1 Projektteam . . . . .	2
2 Projektbetreuer . . . . .	3
3 Aufgabeneinteilung . . . . .	3
<b>II Einleitung</b>	<b>5</b>
1 Motivation . . . . .	5
2 Zielsetzung . . . . .	5
3 Topologie des Gesamtsystems . . . . .	5
4 Leitfaden . . . . .	5
<b>III Stand der Technik</b>	<b>6</b>
1 Synchronmaschine mit Dauermagneterregung . . . . .	6
1.1 Auswertung der Antriebswelle . . . . .	6
2 Curtis Controller . . . . .	6
2.1 Allgemeines . . . . .	6
2.2 VCL . . . . .	6
2.3 Feldorientierte Regelung . . . . .	6
3 Leonard-Umformer . . . . .	6
3.1 Allgemeines . . . . .	6
3.2 . . . . .	6
4 KPI-Regler . . . . .	6
4.1 Allgemeines . . . . .	6
4.2 . . . . .	6
5 Steuereinheiten . . . . .	6
6 Bussysteme . . . . .	6
<b>IV Mechanische Umsetzung</b>	<b>7</b>
<b>V Human-Computer Interaction System</b>	<b>8</b>
1 Übersicht . . . . .	8
1.1 Grundfunktionen des Systems . . . . .	8
1.2 Steuereinheit . . . . .	9
1.3 Grundaufbau des Systems . . . . .	9
2 Spannungsversorgung . . . . .	10
2.1 Aufbau des Versorgungssystems . . . . .	10
2.1.1 12V Versorgungssystem . . . . .	10
2.1.2 5V Versorgungssystem . . . . .	10
2.1.3 Abschalten der Spannungswandler . . . . .	10
3 Steuerung der Peripherie . . . . .	11
3.1 Hardware . . . . .	11
3.1.1 Input . . . . .	11
3.1.2 Output . . . . .	12
3.2 Software . . . . .	12
3.2.1 gpiozero . . . . .	12
3.2.2 threading . . . . .	12

4	Benutzeroberfläche . . . . .	13
4.1	Hardware . . . . .	13
4.1.1	Befestigung . . . . .	14
4.2	Software . . . . .	14
4.2.1	Aufbau . . . . .	14
4.2.2	Nutzer / Berechtigungen . . . . .	15
4.3	Komponenten . . . . .	15
4.3.1	Navigationsmenü . . . . .	15
4.3.2	Balken Anzeige . . . . .	16
4.3.3	Modus Anzeige . . . . .	16
4.3.4	Graph . . . . .	17
4.3.5	Weitere Komponenten . . . . .	17
4.4	Programm Fenster . . . . .	18
4.4.1	Login . . . . .	18
4.4.2	Fahrdaten . . . . .	18
4.4.3	Akku- und Ladedaten . . . . .	19
4.4.4	Fahrdaten Diagnose . . . . .	19
4.4.5	Fehler . . . . .	20
4.4.6	Nutzer und Berechtigungen . . . . .	20
4.5	Realisierung der Benutzeroberfächer . . . . .	21
4.5.1	QML . . . . .	21
4.5.2	Qt-Quick . . . . .	21
4.5.3	Slots und Signals . . . . .	21
4.5.4	Bridge . . . . .	22
5	Kommunikation . . . . .	23
5.1	Hardware . . . . .	23
5.1.1	CAN-Modul . . . . .	23
5.1.2	Netzwerkstruktur . . . . .	23
5.2	Listener . . . . .	24
5.2.1	Konfigurieren der Schnittstelle . . . . .	24
5.2.2	Empfangen der Daten . . . . .	24
6	Fahrdatenspeicher . . . . .	25
6.1	Datenbankstruktur . . . . .	25
6.1.1	Benutzer System . . . . .	25
6.1.2	Motor Daten . . . . .	25
6.1.3	Fehler Tabelle . . . . .	27
6.1.4	Akku Daten . . . . .	27
6.2	Handler . . . . .	27
6.2.1	Konfigurieren der Schnittstelle . . . . .	27
6.2.2	SELECT Befehl . . . . .	28
6.2.3	INSERT Befehl . . . . .	28
<b>VI Antriebsstrang</b>		<b>29</b>
1	Übersicht . . . . .	29
1.1	Grundfunktionen des Systems . . . . .	29
2	Hardwareaufbau des Antriebssystems . . . . .	30
2.1	Mechanische Umsetzung . . . . .	31
2.2	Der Laststromkreis . . . . .	32
2.2.1	Elektrische Energieübertragung . . . . .	33
2.2.2	Leitungsschutzorgane . . . . .	35
2.2.3	Motorbeschreibung . . . . .	36
2.3	Der Steuerstromkreis . . . . .	37
2.3.1	Übersicht Ein- und Ausgänge . . . . .	37
2.3.2	Digitale Eingänge (Digital Inputs) . . . . .	39
2.3.3	Analoge Eingänge (Analog Inputs) . . . . .	39
2.3.4	Gas- und Bremseingänge (Throttle and Brake Inputs) . . . . .	40
2.3.5	Positionsrückmeldung vom Encoder (Position-Feedback Input) . . . . .	40
2.3.6	Prozessorversorgung und Spulenrücklauf (KSI and Coil Return) . . . . .	41

2.3.7	Analoge Ausgänge (Analog Outputs) . . . . .	41
2.3.8	Digitale und Pulsweitenmodulierbare Ausgänge (Digital and PWM Outputs) . . . . .	42
2.3.9	Spannungsversorgungs-Ausgänge (Power Supply Outputs) . . . . .	42
2.3.10	Kommunikations-Ports . . . . .	43
3	Softwareaufbau des Antriebssystems . . . . .	44
3.1	Parameterbasierte Programmierung (Programmer) . . . . .	45
3.1.1	Allgemeines . . . . .	45
3.1.2	Funktionen . . . . .	45
3.2	Drehmomentsteuerung (Torquecontrol) . . . . .	47
3.2.1	Parameter . . . . .	47
3.2.2	ECO- und Sportmodus (Speed-Mode-Select) . . . . .	49
3.3	Vehicle-Control-Language (VCL) Programmierung . . . . .	50
3.3.1	Grundfunktion . . . . .	50
3.3.2	Kommunikation (CAN-Bus) . . . . .	51
3.3.3	Speed-Mode-Select . . . . .	52
4	Inbetriebnahme . . . . .	53
4.1	Leonard-Versuchsaufbau . . . . .	53
4.2	Bleiakku-Versuchsaufbau . . . . .	54
<b>VIAkku und Ladekonzept</b>		<b>56</b>
<b>VIEndergebnis</b>		<b>57</b>
<b>A Arbeitsnachweis</b>		<b>58</b>
1 Zeitplan . . . . .		58
2 Kosten . . . . .		58
<b>B Programmcode</b>		<b>59</b>
<b>C CAD-Zeichnungen</b>		<b>60</b>
<b>D Schaltpläne</b>		<b>61</b>
<b>E Datenblätter</b>		<b>62</b>
0.1 Mean Well RSD-30H-5 . . . . .		63
0.2 Mean Well SD-350C-12 . . . . .		73
0.3 Raspberry Pi 4 Moddel B . . . . .		76
<b>Literaturverzeichnis</b>		<b>89</b>
<b>Abbildungsverzeichnis</b>		<b>89</b>
<b>Tabellenverzeichnis</b>		<b>89</b>
<b>Codeverzeichnis</b>		<b>90</b>

# Kapitel I

## Einführung

### 1 Projektteam



Martin Kronberger



Jakob Lackner



Simon Kern



Schmeisser Tobias

## 2 Projektbetreuer

### **Prof. Dipl.-Ing. Reinhold Benedikter**

unterstützte Jakob Lackner bei der Entwicklung des Akku- und Ladesystems

### **Prof. Dipl.-Ing. (FH) Johannes Ferner**

unterstützte Martin Kronberger bei der Entwicklung des Human-Computer Interaction Systems

### **Prof. Dipl.-Ing. Adolf Reinhart, MBA**

unterstützte Jakob Lackner bei der Entwicklung des Antriebssystems

### **Prof. Dipl.-Ing. Peter Lindmoser**

unterstützte Tobias Schmeisser bei der Entwicklung der mechanischen Komponenten

## 3 Aufgabeneinteilung

### **Martin Kronberger**

- Projektleitung
- Projektfindung und Projektplanung
- Projektaufteilung
- Erstellen der Einreichdokumente
- Entwickeln der Hardware des Human-Computer Interaction Systems
- Entwickeln der Software des Human-Computer Interaction Systems
- Planung und Umsetzung der elektrischen Installation
- Verfassen der Dokumentation

### **Jakob Lackner**

- Projektfindung und Projektplanung
- Projektaufteilung
- Entwicklung des Antriebssystems
- Entwicklung der Software des Motorsteuergerätes
- Erstellen der Einreichdokumente
- Verfassen der Dokumentation

**Simon Kern**

- Projektfindung und Projektplanung
- Projektaufteilung
- Entwicklung des Akkusystems
- Erstellen der Einreichdokumente
- Verfassen der Dokumentation

**Tobias Schmeisser**

- Projektfindung und Projektplanung
- Projektaufteilung
- Entwicklung der mechanischen Komponenten
- Entwicklung der Getriebemechanik
- Erstellen der Einreichdokumente
- Verfassen der Dokumentation

## Kapitel II

# Einleitung

- 1 Motivation**
- 2 Zielsetzung**
- 3 Topologie des Gesamtsystems**
- 4 Leitfaden**

# Kapitel III

# Stand der Technik

---

## 1 Synchronmaschine mit Dauermagneterregung

### 1.1 Auswertung der Antriebswelle

## 2 Curtis Controller

### 2.1 Allgemeines

### 2.2 VCL

### 2.3 Feldorientierte Regelung

## 3 Leonard-Umformer

### 3.1 Allgemeines

### 3.2

## 4 KPI-Regler

### 4.1 Allgemeines

### 4.2

## 5 Steuereinheiten

## 6 Bussysteme

## Kapitel IV

# Mechanische Umsetzung

# Kapitel V

# Human-Computer Interaction System

## 1 Übersicht

Das Human-Computer Interaction System ist, wie der Name schon sagt, die Komponente, welche als Schnittstelle zwischen dem Nutzer und dem gesamten elektrischen System dient. Durch es sollte die fehlerfreie Nutzung der Funktionen des Motorrades gewährleistet sein. Ebenso sollte es wichtige Fahrdaten und andere Informationen speichern und dem User anzeigen können. Wichtig ist das System, trotz der großen Komplexität, so intuitiv und nutzerfreundlich wie möglich zu gestalten.

### 1.1 Grundfunktionen des Systems

Die geplanten Funktionen des HCIS<sup>1</sup> lassen sich grob in vier Grundfunktionen einteilen.

- **Steuerung der Peripherie**

Die Schalter und Buttons am Lenker, welche zuvor über den Kabelbaum die Leuchten, Blinker und die Hupe gesteuert haben, werden nun über die General-purpose input/output (GPIO) des Raspberry Pi Mikrocomputers gesteuert.

- **Graphische Benutzeroberfläche**

Dient der Anzeige wichtiger Fahr- und Ladedaten, welche entweder in Echtzeit oder über die Datenbankschnittstelle abgerufen und graphisch angezeigt werden können.

- **Kommunikation mit den Steuereinheiten des Motorrades**

Über CAN-Bus werden Daten von dem Batterie Management Systems (BMS) und der Curtis Motorsteuerung empfangen und an die Benutzeroberfläche zur Anzeige und an die Datenbankschnittstelle zur Langzeitsicherung der Fahrdaten weiter gegeben.

- **Speichern der relevanten Fahrdaten über die Datenbankschnittstelle**

Die über den CAN-Bus empfangenen Daten werden sofort an die Datenbankschnittstelle (Handler) weitergegeben um die Daten für Datenauswertung und Testberichte zu speichern. Ebenso bezieht das Diagnosesystem der Benutzeroberfläche die Daten über diese Schnittstelle.

<sup>1</sup>Abkürzung: Human-Computer Interaction System

## 1.2 Steuereinheit

Als Basis zur Auswahl der Steuereinheit wurden die zuvor erläuterten Grundfunktionen herangezogen genommen. Die Ausgewählte Steuereinheit sollte diese erfüllen können und ebenso Potential zur Erweiterung der Funktionen bieten. Genauso wichtig war das eine große Flexibilität und Individualität erreicht werden kann, um nicht in der Umsetzung unserer Ideen eingeschränkt zu sein. Zur Auswahl standen verschiedene Speicherprogrammierbare Steuerungen und Mikrocomputer, doch letzten Endes überzeugte der Mikrocomputer Raspberry Pi.

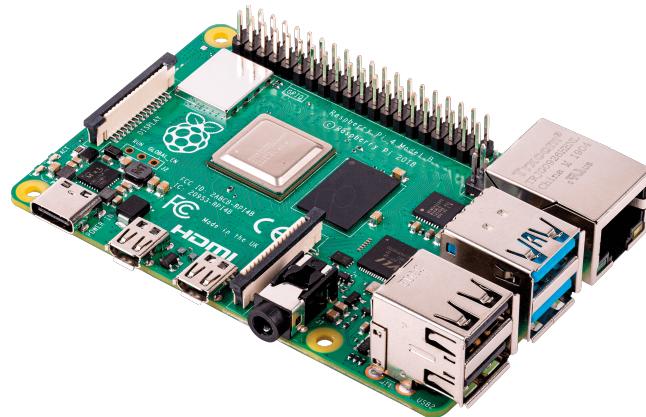


Abbildung V.1: Raspberry Pi - Steuereinheit des HCIS

## 1.3 Grundaufbau des Systems

In der Abbildung V.2 wird der Grundaufbau des Systems und die Datenverbindungen der folgenden Komponenten veranschaulicht.

- Raspberry Pi - Die Steuereinheit des Systems.  
Kommuniziert über CAN-Bus mit den anderen Steuerkomponenten des Motorrades.
- User Input - Die vorhandenen Schalter am Lenker des Motorrads werden direkt mit den Eingängen des Raspberry Pis verbunden.
- Peripherie - Die Grundkomponenten des Motorrades wie Scheinwerfer oder Hupe. Diese werden über Relais, welche an die Ausgänge des Raspberry Pis angeschlossen sind, gesteuert.
- Dashboard - Der Bildschirm zur Anzeige der verarbeiteten Informationen. Dieser wird über HDMI und USB mit dem Raspberry Pi verbunden.

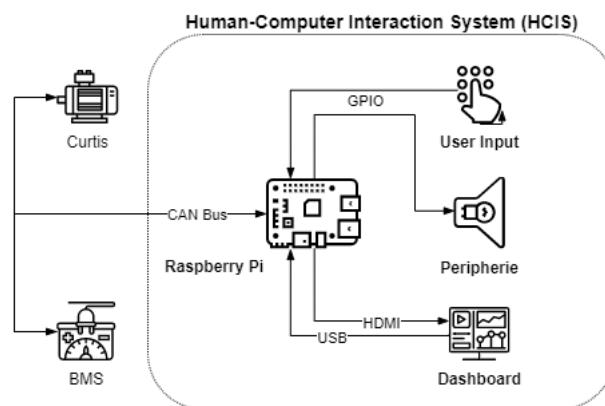


Abbildung V.2: Grundaufbau des Human-Computer Interaction Systems

Nicht in der Abbildung dargestellt ist die Versorgung der einzelnen Komponenten, welche in dem folgenden Abschnitt noch genauer erläutert wird.

## 2 Spannungsversorgung

### 2.1 Aufbau des Versorgungssystems

Das Versorgungssystem des Motorrades besteht aus zwei Spannungsebenen: Einer 12V Ebene zur Versorgung der Peripherie des Motorrades und einer 5V Ebene, welche nur den Raspberry Pi und seine Komponenten beinhaltet. Diese Ebenen werden durch DC-DC Wandler erzeugt, welche direkt an den Akku des Motorrades angeschlossen werden.

Wichtig hierbei ist, dass alle ausgewählten Spannungswandler über einen Kurzschluss- und Überstrom-Schutz verfügen. Dies macht es uns möglich diese Versorgungssysteme, solange die Drähte auch nach dem maximalen Strom der Spannungswandler dimensioniert wurden, ohne jegliche Leistungs- und Überstrom-Schutzorgane aufzubauen. Die Spannungswandler schalten bei jeglichen Fehlern ab und verbrennen die überschüssige Leistung über einen eingebauten Widerstand. Sobald der Fehler behoben wurde, schalten sich die Spannungswandler automatisch wieder ein.

#### 2.1.1 12V Versorgungssystem

Um den Spannungswandler dimensionieren zu können mussten vorher alle Bauteile, welche über die 12V versorgt werden sollten, zusammengefasst werden, um die mindestens benötigte Leistung des Spannungswandlers zu errechnen.

Bauteilbezeichnung	Spannung	Leistung
Tagfahrlicht	12V	10W
Abblendlicht	12V	10W
Aufblendlicht	12V	20W
Hupe	12V	10W
Rücklicht	12V	21W
Kennzeichenbeleuchtung	12V	5W
Blinker links	12V	2 x 10W
Blinker rechts	12V	2 x 10W
Bildschirm	12V	12W
<b>Gesamt</b>	<b>12V</b>	<b>128W</b>

Tabelle V.1: Berechnung der Leistung des 12V-Systems

Der Spannungswandler wurde nun nach der größtmöglichen Leistung, welche auftritt wenn alle Bauteile gleichzeitig auf Höchstleistung betrieben werden, ausgelegt. Diese maximale Leistung beträgt, wie in der Tabelle V.1 zu sehen, 128 Watt. Um noch Ausbaumöglichkeiten zu gewährleisten und uns nicht dem Leistungslimit des Wandlers zu nähern, haben wir uns für einen 48V-12V, 300 Watt DC-DC Wandler von Mean Well<sup>2</sup> entschieden.

#### 2.1.2 5V Versorgungssystem

Die Leistung des Raspberry Pis ist mit einem Maximum von 6.2 Watt sehr klein und daher ist die Wahl des Spannungswandlers in diesem Fall nicht wirklich davon abhängig. Auch die Komponenten, welche angeschlossen werden, haben grundsätzlich keine erwähnenswerte Wirkleistung und müssen daher nicht genau berechnet werden. Nun entschied nur mehr das Preis-Leistungs-Verhältnis sowie die Ausfallsicherheit des Spannungswandlers die Wahl. Daher haben wir uns für einen 48V-5V, 30 Watt DC-DC Wandler von Meanwell<sup>3</sup> entschieden.

#### 2.1.3 Abschalten der Spannungswandler

Das Abschalten der Spannungswandler ist nicht notwendig, da diese - wie schon in Abschnitt 2.1 erklärt - bei einem anliegenden Fehler automatisch abschalten. Ebenso wird beim Abschalten des Motorrades über die BMS jegliches andere Bauteil von der Spannungsversorgung getrennt. Was die Spannungswandler vom Entladen des Akkus abhält.

<sup>2</sup>Datenblatt: siehe Anhang 0.2

<sup>3</sup>Datenblatt: siehe Anhang 0.1

### 3 Steuerung der Peripherie

Die Grundfunktionen wie Beleuchtung, Hupe und Blinker werden hier als Peripherie bezeichnet. Diese sollten so einfach wie möglich und vom Lenker aus zu bedienen sein. Ebenso müssen sie verlässlich gesteuert werden können. Daher haben wir uns entschieden diese Funktionen ebenso über den Raspberry Pi zu steuern, da dieser bei einem Fehler der Motorsteuerung über den eingebauten Puffer gespeist werden kann und daher diese wichtigen Funktionen bis zu einem sicheren Stillstand weiter betrieben und gesteuert werden können.

Dennoch ist in der Plan in Zukunft die Motorsteuerung, welche ebenso in der Lage wäre die Ausgänge abhängig von den Eingängen zu schalten, diese Aufgabe übernehmen zu lassen, solange die Ausfallsicherheit ebenso gegeben wäre. Der Vorteil dieser Methode ist die Schaffung einer Zentralen Steuereinheit, welche alle Steueraufgaben in einem Bauteil vereinen kann.

#### 3.1 Hardware

##### 3.1.1 Input

Man kann einen GPIO Pin entweder als Eingang oder als Ausgang betreiben. Als Eingang kann er die Zustände High und Low einnehmen. Zum Beispiel von einem Schalter oder Taster. In der Regel beschaltet man die GPIOs des Raspberry Pis mit Widerständen, um Eingänge auf einen definierten Pegel zu setzen oder um den Strom zu begrenzen. Standardmäßig werden 10k Widerstände benutzt. Ob Pullup oder Pulldown ist grundsätzlich gleichgültig. Wir benutzen für das Einlesen der Eingänge 10k Pulldown Widerstände, um nicht immer eine Spannung an den Eingängen des Raspberry Pis anliegen zu haben.

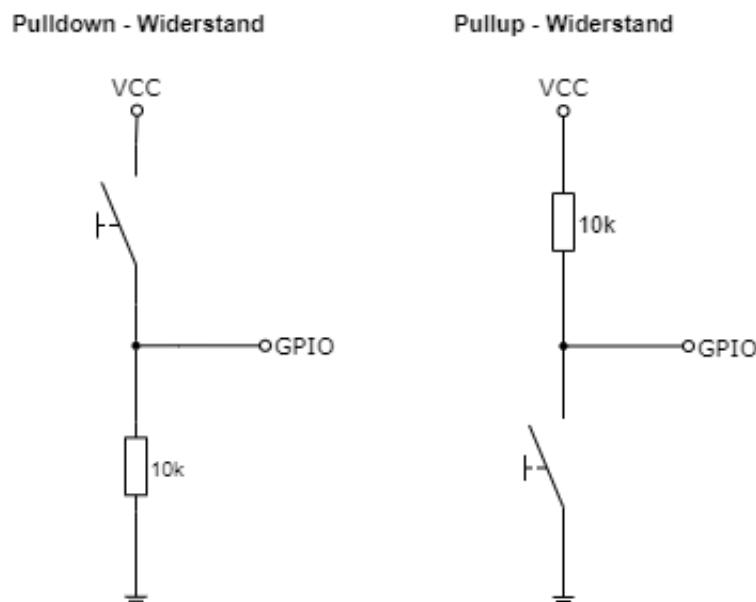


Abbildung V.3: Anschlussplan Eingänge

##### Pullup Widerstand

Bei dem Nutzen eines Pullup Widerstands wird der GPIO Pin mit einem Widerstand auf die Spannung von VCC gezogen. Der Grundzustand des Eingangs ist dann High. Mit einem Schalter oder Taster wird der Eingang dann gegen Ground gezogen. Das heißt er hat solange der Schalter geschlossen ist, liegt das Massepotential am Eingang an.

##### Pulldown Widerstand

Bei dem Nutzen eines Pulldown Widerstands wird der GPIO Pin mit einem Widerstand auf die Spannung von Ground gezogen. Der Grundzustand des Eingangs ist dann Low. Mit einem Schalter oder Taster wird der Eingang dann gegen VCC gezogen. Das heißt er hat solange der Schalter geschlossen ist, liegt das Versorgungspotential am Eingang an.

### 3.1.2 Output

Hierbei werden die GPIOs als Ausgang verwendet. Sie sind verbunden mit den Eingängen eines 4 Channel Relais Moduls, welches über die 5V direkt von dem Raspberry Pi gespeist wird. Hiermit ist es nun möglich die 12V der Peripherie zu schalten und

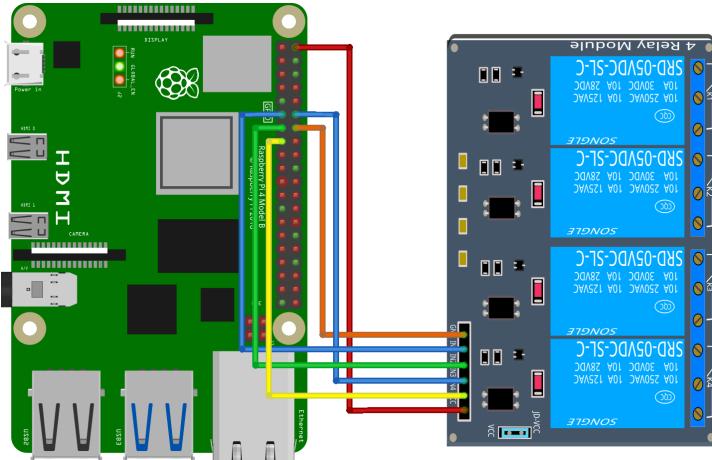


Abbildung V.4: Anschlussplan Relais

## 3.2 Software

Wichtig bei der Programmierung war in diesem Fall die dauerhafte Verfügbarkeit der Grundfunktionen sowie die einfache Integrierbarkeit von alten und neuen Bauteilen. Dies wurde erreicht durch die Anwendung verschiedener Bibliotheken.

Die Wichtigsten in dieser Anwendung waren:

### 3.2.1 gpiozero

Eine Bibliothek, welche das einfache und schnelle Integrieren neuer Ein- oder Ausgänge ermöglicht. Dadurch können schnell und einfach Änderungen an dem Steuerverhalten der Peripherie gemacht werden. Hierzu kann die Klasse Button und DigitalOutputDevice verwendet werden. Dieser können Parameter wie die gewünschte Debounce Zeit und verwendete Pull-Widerstände übergeben werden. Es werden nur drei Zeilen Code benötigt um einen Aus- oder Eingang zu definieren und anzusteuern<sup>4</sup>

### 3.2.2 threading

Eine Bibliothek, welche es ermöglicht, einen eigenen Thread<sup>5</sup> für das Programm zu öffnen, wodurch die Anwendung ohne Einflüsse oder Unterbrechungen anderer Programme weiter arbeiten kann. Der folgende Programmcode zeigt einen Ausschnitt der Klasse zum Steuern der GPIOs.

```
def start(self):
    self.thread=Thread(target=self.runner)
    self.thread.daemon=True
    self.thread.start()
```

Code Listing V.1: Code zum Starten eines Threads

In diesem Beispiel wird ein Demon Thread erzeugt und gestartet. Das bedeutet dieser Thread wird beim schließen des Programms automatisch mit geschlossen und muss dadurch nicht mehr überwacht werden. Das Target ist die Funktion, welche unabhängig ausgeführt werden soll.

<sup>4</sup>siehe: Anhang

<sup>5</sup>gleichzeitig laufende Aufgabe

## 4 Benutzeroberfläche

Die Benutzeroberfläche stellt die Verbindung zwischen dem Nutzer und dem Motorrad dar. Sie sollte während der Fahrt die Instrumententafel des Motorrades ersetzen und dem Nutzer die wichtigsten Fahrinformationen anzeigen. Sobald das Motorrad zum Stillstand gekommen ist, wird es möglich Einstellungen zu ändern und die aufgezeichneten Fahrdaten anzeigen zu lassen. Ebenso können der Akkuladestatus und Informationen über Fehler im System entnommen werden.

### 4.1 Hardware

Zur Anzeige und Bedienung wird ein 11.6 Zoll kapazitives Touch LCD Display verwendet. Es besitzt eine Full HD Auflösung (1920x1080), was für eine professionelle Darstellung essentiell ist. Ebenso hat es ein schützendes ABS Gehäuse, welches trotz fehlender IP Zertifizierung das Abdichten ermöglicht. Die Versorgungsspannung beträgt 12V, was ident zu den anderen Komponenten am Motorrad ist und daher die Versorgung sehr vereinfacht, es kann also über den gleichen Spannungswandler versorgt werden.



Abbildung V.5: Paneel Maße

Die Auflösung und die Größe des Paneels wirkt sich stark auf das Design der Benutzeroberfläche aus. Es muss die Größe der Icons und der anderen Designelemente so angepasst werden, dass sie einerseits gut ersichtlich und andererseits einfach über Berührung zu bedienen sind.

#### 4.1.1 Befestigung

Besser: In das Gehäuse des Paneels sind M4 Verschraubungen in einem Raster von 75mm x 75mm integriert und kann daher einfach an Wänden oder Platten verschraubt werden. Um den Bildschirm nun in einer ähnlichen Position wie die Instrumententafel zu befestigen wurde eine 100mm x 210mm x 1.5mm Aluminium Platte - wie in der Abbildung zu sehen - gebogen und mit Löchern versehen. Um diese Halterung nun an dem Motorrad zu befestigen werden die Verschraubungen der alten Instrumententafel verwendet.



Abbildung V.6: Befestigung des Displays

#### 4.2 Software

Bevor die Software für die Benutzeroberfläche verfasst wurde, mussten das Design, die Funktionen sowie die angezeigten Informationen geplant werden, um einen reibungslosen Arbeitsablauf beim Entwickeln des Frontends zu gewährleisten. Design Elemente wurden zuvor in Adobe Illustrator vorgefertigt. In den folgenden Seiten wird das Ergebnis dieses Prozesses erläutert.

##### 4.2.1 Aufbau

Die nachfolgende Abbildung zeigt den grundsätzlichen Programmaufbau der Benutzeroberfläche. Die einzelnen Fenster werden als Tabelle mit ihren angezeigten Informationen dargestellt. Dies ist wichtig da jede dieser Informationen vom Backend an das Frontend gesendet werden müssen. Ebenso sind in den letzten Zeilen der Tabellen die QML-Elemente zur Navigation zwischen den einzelnen Fenstern niedergeschrieben. Diese müssen auch schon in der frühen Phase der Entwicklung der Benutzeroberfläche definiert werden.

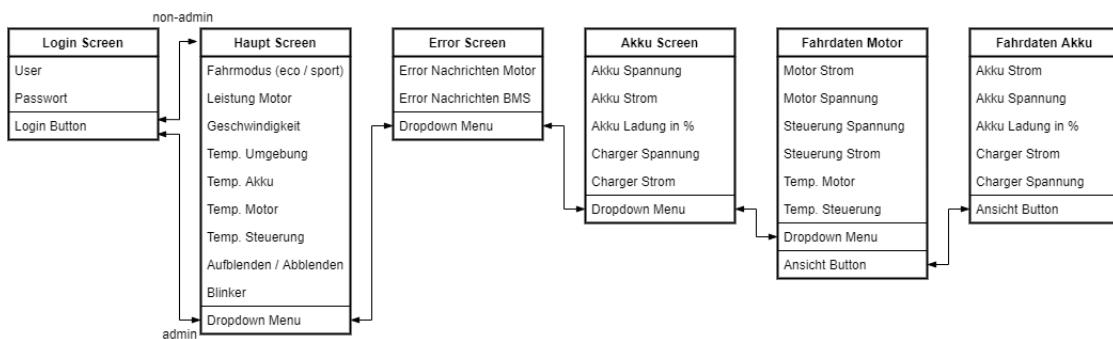


Abbildung V.7: Aufbau der Graphischen Benutzeroberfläche

#### 4.2.2 Nutzer / Berechtigungen

### 4.3 Komponenten

Komponenten sind wiederverwendbare, gekapselte QML-Elemente mit genau definierten Schnittstellen. Komponenten werden häufig durch Komponenten-Dateien definiert, das heißt durch QML-Dateien. Wichtig ist dabei die Definition von Schnittstellen sowie Properties und Signals.

#### Properties

Einer Property eines Objektes kann ein statischer Wert zugewiesen werden, der konstant bleibt, bis ihm explizit ein neuer Wert zugewiesen wird. Um QML und seine integrierte Unterstützung für dynamisches Objektverhalten optimal zu nutzen, verwenden die meisten QML-Objekte jedoch Propertybindings.

Dies sind eine Kernfunktion von QML, mit der Beziehungen zwischen verschiedenen Objekteigenschaften festgelegt werden können. Wenn sich die Abhängigkeiten einer Property im Wert ändern, wird die Eigenschaft automatisch gemäß der angegebenen Beziehung aktualisiert. Hinter den Kulissen überwacht die QML-Engine die Abhängigkeiten der Eigenschaft. Wenn eine Änderung erkannt wird, wertet die QML-Engine den Bindungsausdruck erneut aus und wendet das neue Ergebnis auf die Eigenschaft an.

#### Java-Script-Funktionen

Programmlogik kann auch in Java-Script-Funktionen definiert werden. Diese können in QML-Dokumenten definiert und von Signalhandlern, Eigenschaftsbindungen oder Funktionen in anderen QML-Objekten aufgerufen werden. Solche Methoden werden häufig als Inline-Java-Script-Funktionen bezeichnet, da ihre Implementierung im QML-Dokument statt in einer externen Java-Script-Datei enthalten ist.

#### 4.3.1 Navigationsmenü

Das Navigations-Menü ist ein Dropdown-Menü, welches zur Navigation zwischen den verschiedenen Fenstern benutzt wird. Sobald man sich eingeloggt hat wird das Menü angezeigt und die einzelnen Untermenüs können aufgerufen werden. Das Menü wird abhängig von den Berechtigungen des Benutzers angepasst.



Abbildung V.8: GUI Komponente - Navigation Menü

#### Buttons

Die Navigation wird über das QML-Element *Mousearea*, welche direkt über den Icons der einzelnen Navigationselemente platziert wurde, gesteuert. Nun kann mit dem Befehl *onClicked* eine Funktion aufgerufen werden, welche das gewünschte Fenster sichtbar macht, sowie das Navigationsmenü wieder nach oben fahren lässt.

In dieser Funktion wird ebenso die Berechtigung des Nutzers über eine Globale Variable, welche beim Anmelden durch ein Signal gesetzt wird, abgefragt. Falls die Berechtigung die ausgewählte Funktion nicht zulässt wird ein Informationstext ausgegeben und das Menü wiederum geschlossen.

#### Abmelden

Wird der Abmeldebutton gedrückt, werden die Anmeldeinformationen zurückgesetzt und dem Nutzer wird wieder das Anmeldefenster angezeigt, wo er sich nun mit anderen Anmelde-Informationen einloggen kann.

### 4.3.2 Balken Anzeige

Die Komponente Balken Anzeige wird in der Benutzeroberfläche zur Visualisierung verschiedener Daten verwendet. Mit ihr können diese übersichtlicher dargestellt werden. Diese Komponente wird in mehreren QML-Dateien verwendet, daher sind Properties zur Anpassung notwendig. Die wichtigsten davon sind:

- Wert - Der aktuelle Wert, welcher am Balken angezeigt werden sollte.
- Anfangswinkel - Der Winkel an dem der Balken entspringt.
- maximaler Wert - Der maximal zu erreichende Wert. Dieser bestimmt die Länge des Hintergrundbalkens
- Hintergrundfarbe - Farbe des Hintergrundbalkens (in der Abbildung grau)
- Balkenfarbe - Farbe des Anzeigebalkens (in der Abbildung blau)

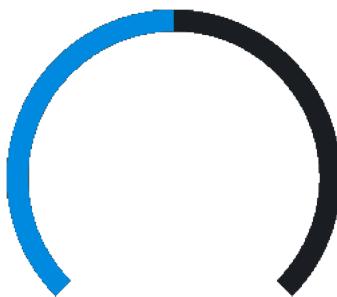


Abbildung V.9: GUI Komponente - Balken Anzeige

Über ein *Signal* kann nun über das Backend der Wert des Balkens verändert werden und in Echtzeit angezeigt werden.

### 4.3.3 Modus Anzeige

Diese Komponente befindet sich auf der Instrumententafel. Der Modus kann über einen Button am Lenker, welcher mit der Curtis Steuerung verbunden ist, geändert werden. Sie dient zum Anzeigen der derzeitig gewählten Fahrmodi und wird über ein *Signal* aus dem Backend gesteuert, welches mit der Listener Klasse verbunden ist.



Abbildung V.10: GUI Komponente - Modus Anzeige

Die Modus Anzeige ist eine einfache Komponente. Sie ist ein Item QML-Typ, dadurch wird das Nutzen von *States* möglich. Durch diese können verschiedene Eigenschaften gespeichert und über einen kurzen Befehl wiederhergestellt werden.

In diesem Fall wird der Punkt ausgefüllt und die Farbe des Textes geändert.

#### 4.3.4 Graph

Die Graph Komponente befindet sich auf dem Diagnose Fenster und dient zum Anzeigen der in der Datenbank gespeicherten Fahrdaten. Sie verfügt über einen Ladebalken, welcher nach dem Auswählen der anzuseigenden Daten den Fortschritt des Auslesens der Datenbank anzeigt. Dieses Auslesen läuft über eine Funktion der Fetcher Klasse.

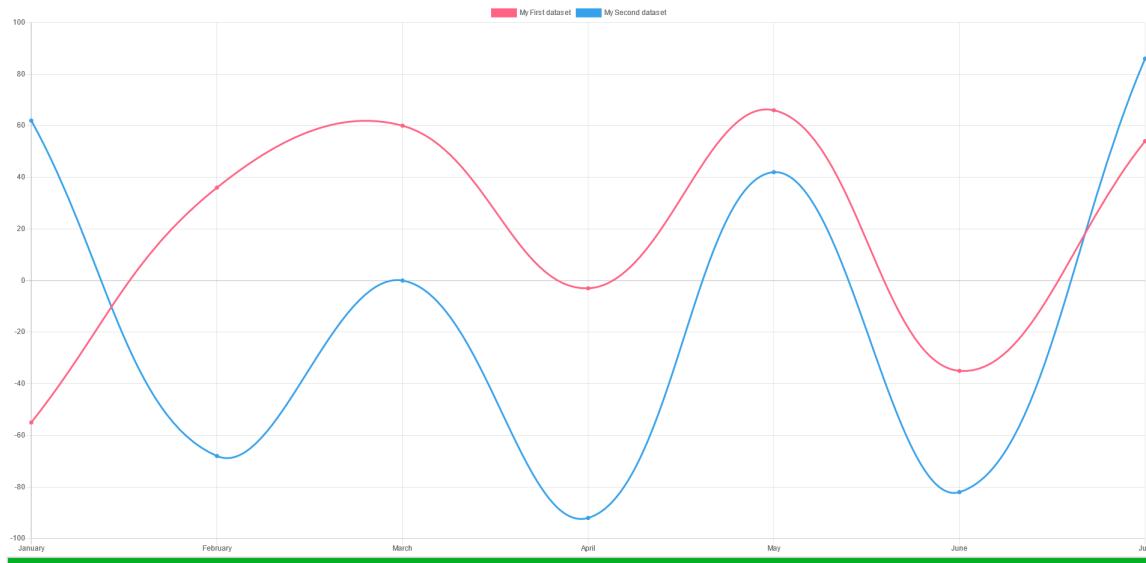


Abbildung V.11: GUI Komponente - Graph

Realisiert wurde diese Komponente über öffentliche Bibliothek *Charts.js* (Referenz), welche durch die Java Script Unterstützung von *QT*, mit der Installation eines Adapters mit dem QML Programm kompatibel ist. Dadurch wird nun das Zeichnen von verschiedenen Graphen ermöglicht.

#### 4.3.5 Weitere Komponenten

Wie schon bei der Fahr Modi Komponente, wurden weitere Objekte, welche zwischen zwei Zuständen hin und her gestalten werden müssen oder eine bestimmte Handlung ausführen sollten, über eine eigene QML Komponente verwirklicht. Diese ermöglicht nun wiederum das wiederholte verwenden dieser Komponente, sowie das Ansteuern über *States* und *Signale*. Diese Komponenten sind:

##### Umschaltkomponenten

Dies sind Komponenten, welche über ein *Signal* angesteuert werden und dann mithilfe von *States* ihr Aussehen verändern.

- Blinker Rechts
- Blinker Links
- Tagfahrlicht
- Fernlicht

##### Touch Komponenten

Dies sind Komponenten, welche eine *Mousearea* mit einer Graphik verbindet. Wird diese Komponente nun mit einem Mausklick ausgewählt, kann eine Funktion ausgeführt oder ein *Signal* ausgesendet werden.

- Abmeldebutton
- Navigationsmenübuttons
- Menü-Öffnen-Button

## 4.4 Programm Fenster

### 4.4.1 Login

Das Login Fenster dient zur Autorisierung des Benutzers. Über die User Kombobox kann der gewünschte Nutzer mit der dazugehörigen Berechtigung ausgewählt werden (siehe Abschnitt ). Das Fenster dient ebenso zur Darstellung der Logos unserer Sponsoren



Abbildung V.12: GUI Fenster - Login Menü

Um sich einzuloggen muss nur mehr das Passwort im *Textfeld* darunter eingegeben werden. Nach dem Drücken des Login Pfeiles (Neben dem Passwort Textfeld) werden die Login Daten an das Backend versendet. Diese vergleicht die Daten über die Datenbankschnittstelle<sup>6</sup> und loggt, insofern das richtige Passwort gegeben ist, den Benutzer ein. Wird jedoch das Passwort falsch eingegeben, wird eine rote Fehlermeldung angezeigt.

### 4.4.2 Fahrdaten

Dieses Fenster dient als Ersatz für die Instrumententafel des Motorrades. Es ist daher das einzige während der Fahrt ersichtliche Fenster. Es Zeigt alle wichtigen Fahrdaten wie Leistung, Geschwindigkeit und Akkuladestand, sowie den Aktuellen Fahrmodus an. Diese Anzeige kann jedem Nutzer, unabhängig der Berechtigung, angezeigt werden.

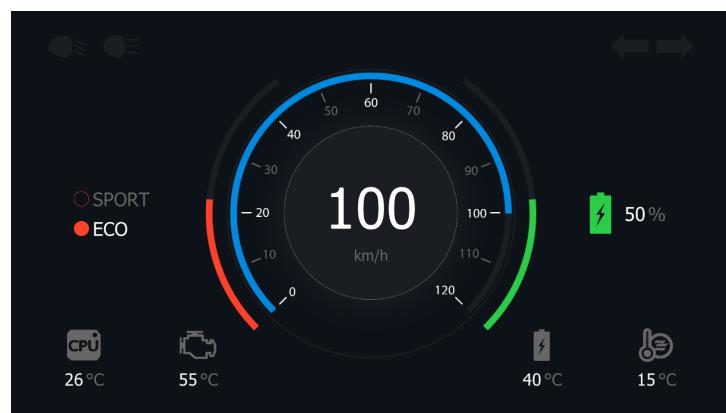


Abbildung V.13: GUI Fenster - Fahrdaten

Der Status der Blinker und der Beleuchtung wird ebenso wie die Temperaturen der verschiedenen Komponenten des Motorrades angezeigt. Derzeit ist diese Ansicht starr und kann noch nicht geändert werden. Doch es ist geplant in Zukunft weitere Designs, welche in den Einstellungen geändert werden können, zu implementieren.

<sup>6</sup>siehe Abschnitt

#### 4.4.3 Akku- und Ladedaten

Dieses Fenster dient Ähnlich wie das der Fahrdaten zur Darstellung wichtiger Daten. Es können alle wichtigen Daten bezüglich Akku und BMS auf einem Blick abgelesen werden.

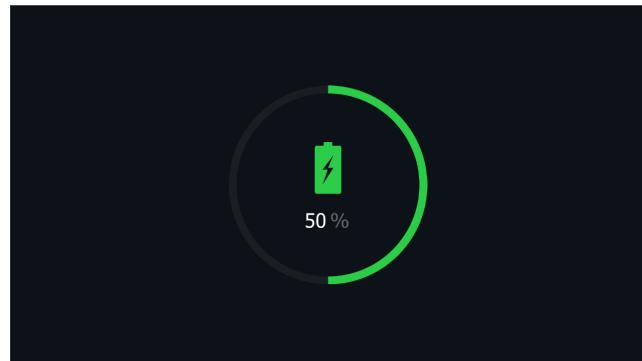


Abbildung V.14: GUI Fenster - Akkudaten

Dieses Fenster wird Standardmäßig während dem Laden des Fahrzeuges angezeigt. Es benötigt also keine bestimmte Berechtigung und kann von jedem Nutzer über das Navigations-Menü angezeigt werden.

#### 4.4.4 Fahrdaten Diagnose

Hier können die Fahrdaten, welche während der Fahrt dauerhaft von der Motorsteuerung versendet und vom Raspberry Pi verarbeitet und in einer Datenbank gespeichert werden, in Graphen angezeigt werden. Hierzu wird die *Graph Komponente*<sup>7</sup> verwendet.

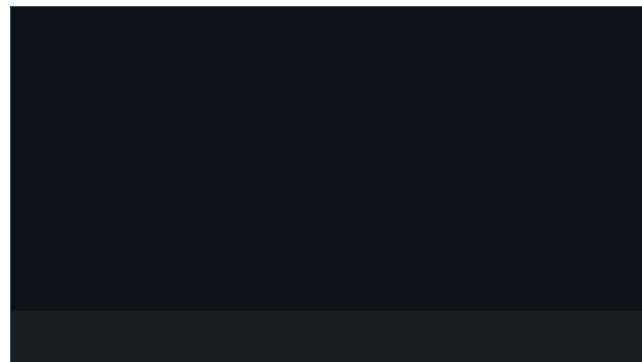


Abbildung V.15: GUI Fenster - Fahrdaten Diagnose

Diese wird, um während der Fahrt die volle Prozessorleistung zu gewährleisten, erst auf Knopfdruck geladen. In der Menüleiste am unteren Ende des Fensters können verschiedene Vorlagen ausgewählt werden, welche über fixe Datensätze verfügen. Das bedeutet es werden erst beim Auswählen des anzuseigenden Datensatzes die Daten der letzten Stunde ausgelesen und in den Graphen geladen.

---

<sup>7</sup>siehe Abschnitt 4.3.4

#### 4.4.5 Fehler

Sobald ein Fehlerdatenpaket über den CAN-Bus versendet wird, gibt der Listener Klasse die Fehlercodes an die Bridge Klasse weiter, um sie in diesem Fenster anzuzeigen. In einer *Listview* werden nun die Aktiven Fehler angezeigt. Solange Fehler anliegen werden diese angezeigt und erst sobald die zuvor angezeigten Fehler nicht mehr anliegen werden diese aus der Liste gelöscht.

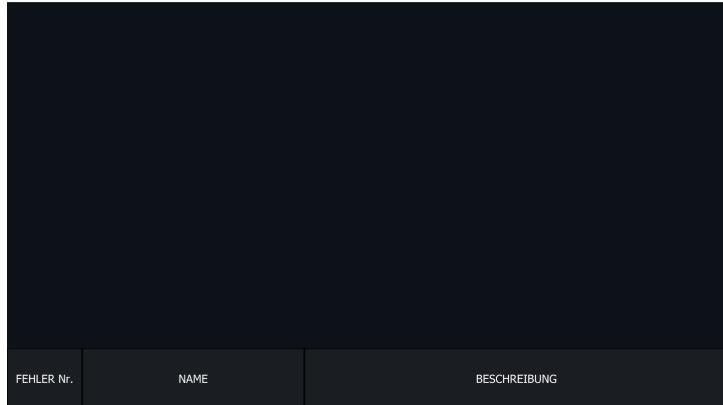


Abbildung V.16: GUI Fenster - Fehler Liste

Der Name und die Beschreibung der Fehler, wird über eine Fehlertabelle<sup>8</sup> in der Datenbank ausgelesen. Diese werden während dem Testen des Motorrades gesammelt und häufig auftretende Fehler werden in diese Tabelle eingetragen.

#### 4.4.6 Nutzer und Berechtigungen

Um sicherzustellen das das Nutzen der Benutzeroberfläche nicht für jeden uneingeschränkt möglich ist wurde ein Anmeldesystem mit drei verschiedenen Benutzern, welche jeweils verschiedene Berechtigungen besitzen, umgesetzt.

**ADMIN** - Passwort: 53AC2

Dieser Benutzer verfügt über uneingeschränkte Rechte und kann auf jede verfügbare Funktion sowie Funktionen, welche noch in Entwicklung sind zugreifen.

**USER** - Passwort: 5AHET

Dieser Benutzer verfügt über wenig eingeschränkte Rechte. Er kann auf jede fertig entwickelte Funktion zugreifen kann jedoch keine Änderungen an der Benutzeroberfläche vornehmen und hat ebenso keinen Zugriff auf experimentelle Funktionen.

**GUEST** - Passwort: 00000

Dieser Benutzer verfügt nur über sehr eingeschränkte Rechte. Er wird zum Präsentieren der Maschine benutzt und lässt den unerfahrenen Nutzer nur auf die Fahrdaten und Akku- und Ladedaten zugreifen. Ebenso sollte in Zukunft die Leistung des Motorrades beim einloggen dieses Benutzers stark eingeschränkt werden, um Unfälle zu verhindern.

<sup>8</sup>siehe Abschnitt

## 4.5 Realisierung der Benutzeroberfächer

### 4.5.1 QML

QML<sup>9</sup> ist eine deklarative Sprache, mit der Benutzeroberflächen anhand ihrer visuellen Komponenten und ihrer Interaktion und Beziehung zueinander beschrieben werden können. Es ist eine gut lesbare Sprache, die entwickelt wurde, um die dynamische Verbindung von Komponenten zu ermöglichen und die einfache Wiederverwendung und Anpassung von Komponenten innerhalb einer Benutzeroberfläche erlaubt. Es bietet Syntax mit Unterstützung für Java-Script-Ausdrücke in Kombination mit dynamischen Eigenschaftsverbindungen.

### 4.5.2 Qt-Quick

Das Qt-Quick-Modul ist die Standardbibliothek zum schreiben von QML-Anwendungen. Während das QML-Modul die Engine und die Sprachinfrastruktur bereitstellt, bietet das Qt Quick-Modul alle grundlegenden Typen, die zum Erstellen von Benutzeroberflächen mit QML erforderlich sind. Es bietet eine visuelle Zeichenfläche und Typen zum Erstellen und Animieren visueller Komponenten, zum Empfangen von Benutzereingaben, zum Erstellen von Datenmodellen und Ansichten sowie zum verzögerten Objektinstanziieren. Es können problemlos flüssige, animierte Benutzeroberflächen in QML erstellt werden. Diese Benutzeroberflächen können mit beliebigen Backend Bibliotheken verbunden werden.

### 4.5.3 Slots und Signals

Slots und Signals werden in QML zur ereignisgesteuerten Kommunikation zwischen Frontend und Backend verwendet. In der folgenden Illustration wird diese anhand eines einfachen Beispiels erklärt.

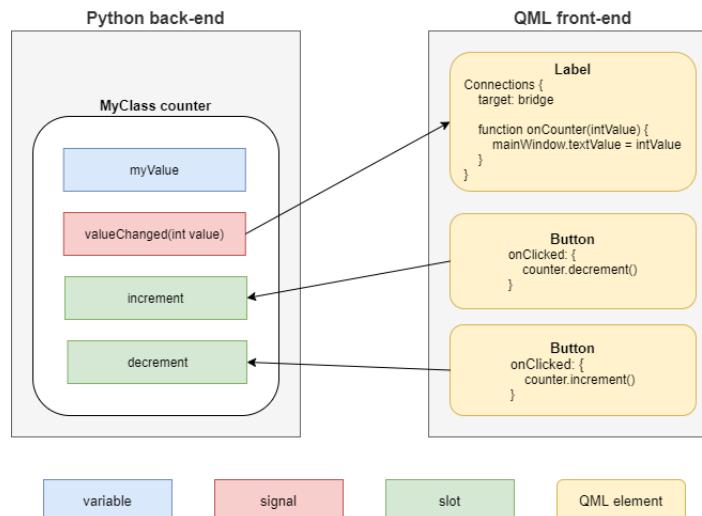


Abbildung V.17: Slots und Signals Konzept

### Signale

Diese können als Mitteilungen angesehen werden, welche über das Aufrufen der `signal.emit()` Funktion vom Backend an das Frontend gesendet wird. Im Frontend wird wiederum eine eigens definierte Funktion benötigt um dem Wert einem Property eines QML Elements zuzuweisen.

### Slots

Slots sind Call-Back Funktionen, welche im Backend definiert werden und sind über die Bridge Klasse mit dem Frontend verknüpft. Dadurch können diese Funktionen im Frontend aufgerufen und mit Signalen verbunden werden. Sie stellen daher die wichtigste Verbindung zwischen dem Programm und der Benutzeroberfläche dar.

<sup>9</sup>Qt Modeling Language

#### 4.5.4 Bridge

Die Bridge Klasse wird für die Kommunikation zwischen den einzelnen Modulen des Backends mit denen der graphischen Benutzeroberfläche.

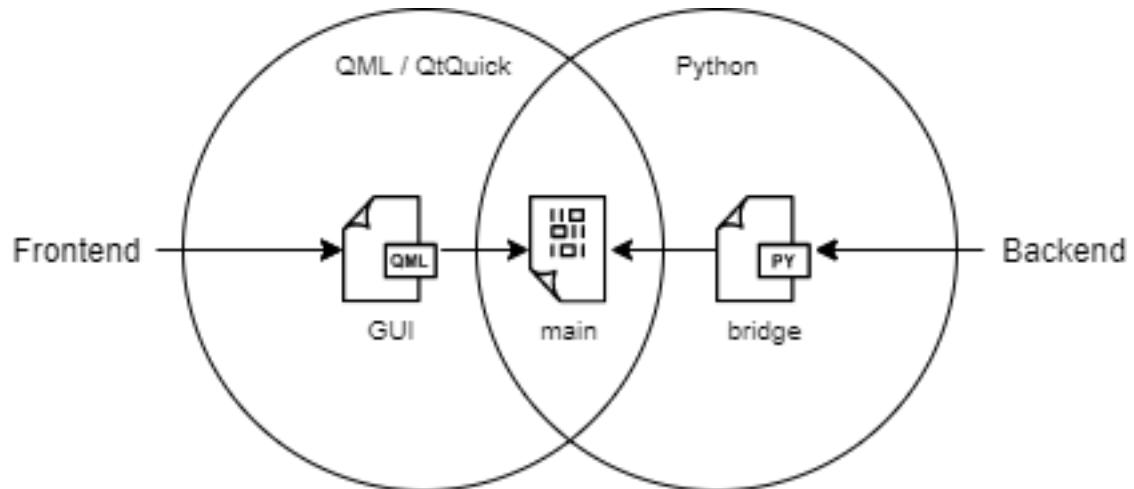


Abbildung V.18: Verbindung Frontend zu Backend

## 5 Kommunikation

Um Daten zwischen den mehreren Steuereinheiten des Motorrades zu versenden, muss eine Echtzeit-Kommunikation über ein Bussystem gewährleistet werden. Die Entscheidung ist auf das Controller Area Network Bussystem (CAN-Bus) gefallen. Ausschlaggebend für diese Entscheidung war der Curtis Motorcontroller, dieser verfügt über eine serielle Schnittstelle (RS-232) sowie ein CAN-System. Für unserer Anwendung bietet das CAN-System eine größere Ausbaufähigkeit sowie größere Übertragungsraten, weshalb wir uns letztendlich auch dafür entschieden haben.

### 5.1 Hardware

#### 5.1.1 CAN-Modul

Da der Raspberry Pi selbst nicht über ein CAN-System verfügt, erfolgt der Anschluss an die Busleitung über ein externes CAN-Modul, welches über das Serial Peripheral Interface (SPI) mit dem Raspberry Pi kommuniziert, welches wie folgt angeschlossen werden muss:

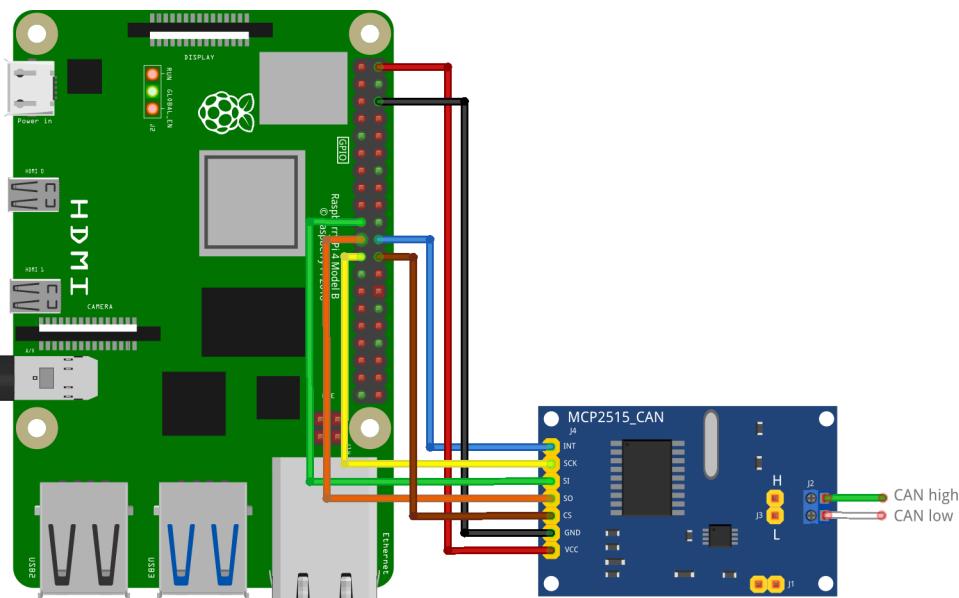


Abbildung V.19: Anschlussplan CAN-Modul

Die Kommunikation wird über zwei Komponenten ermöglicht. Einen MCP2562 Transceiver, welcher für die Verarbeitung der Nachrichten zuständig ist und ein MCP2515 CAN Interface, welches die Daten zwischenspeichert und sich um das Versenden der Nachrichten kümmert. Gemeinsam mit dem Mikrocomputer ergibt dies nun einen CAN Node, welcher fähig ist Nachrichten zu versenden und zu empfangen.

#### 5.1.2 Netzwerkstruktur

Ein CAN-Netzwerk wird standardmäßig als Bus- oder Sternkopplung aufgebaut. Wir haben uns bewusst für die Bustopologie entschieden, da Sternkopplungen nur in bestimmten Anwendungen Gebrauch finden und noch dazu markante Nachteile besitzt.

Es müsste zum Beispiel eine zentrale Steuereinheit den Nachrichtenverkehr steuern, ebenso gibt uns die niedrige Anzahl an Teilnehmern im Netzwerk nicht einmal die Möglichkeit ein anderes System zu verwenden. Erst wenn wir in Zukunft das CAN-Netzwerk um Sensoren und Aktoren erweitern würden, müsste weitere Zeit in die Planung des Netzwerks investiert werden.

## 5.2 Listener

Die Listener Klasse ist dafür zuständig den Datenverkehr am Bus zu überwachen und geordnet an die Datenbankschnittstelle (Fetcher) sowie die Schnittstelle zum Frontend (Bridge) weiterzugeben.

### 5.2.1 Konfigurieren der Schnittstelle

```
def start(self):
    self.thread=Thread(target=self.runner)
    self.thread.daemon=True
    self.thread.start()
```

Code Listing V.2: Konfigurieren des CAN Adapters

### 5.2.2 Empfangen der Daten

## 6 Fahrdatenspeicher

Das Motorrad sollte seine Fahrdaten abspeichern und darstellen können, um eine Diagnose des gesamten Aufbaus zu gewährleisten. Dafür wird eine Datenbank verwendet, nämlich die weit verbreitete Software mySQL verwendet.

### 6.1 Datenbankstruktur

#### 6.1.1 Benutzer System

users		
Attribut	Datentyp	Beschreibung
id	int	Identifikationsnummer
username	varchar(50)	Name des Benutzers
password	varchar(255)	Eingegebenes Passwort

Tabelle V.2: Datenbankstruktur der Benutzer Tabelle

#### 6.1.2 Motor Daten

##### Datenpacket 1

data1		
Attribut	Datentyp	Beschreibung
id	int	Identifikationsnummer
date_time	datetime	Datum der Erstellung
vehicle_speed	int	Geschwindigkeit des Motorrades
current_rms	int	Ausgangstrom der Motorsteuerung
controller_temp	int	Temperatur des Kontrollers
motor_temp	int	Temperatur des Motors
motor_torque	int	Drehmoment am Motor
modulation_depth	int	Modulationsgrad des Kontrollers

Tabelle V.3: Datenbankstruktur der Datenpaket 1 Tabelle

id und enabled sind nicht signiert, wobei id der Primärschlüssel mit Auto-Inkrement ist. step\_id ist ein Zeigerattribut, das für die Reihenfolge der Schritte verwendet wird, wie sie in der grafischen Oberfläche angezeigt werden. Der kaskadierte Fremdschlüssel tutorial\_id wird für die Zuordnung zu dem jeweiligen Tutorial verwendet. Für die

**Datenpacket 2**

data2		
Attribut	Datentyp	Beschreibung
id	int	Identifikationsnummer
id_data_1	int	Fremdschlüssel von Datenpacket 1
date_time	datetime	Datum der Erstellung
vehicle_speed	int	Geschwindigkeit des Motorrades
capacitor_volts	int	Eingangsspannung der Motorsteuerung
bdi_percentage	int	
interlock	int	
throttle_command	int	Stellung des Gasgriffes
brake_command	int	Stellung der Motorbremse (software)
emr_state	int	

Tabelle V.4: Datenbankstruktur der Datenpaket 2 Tabelle

**Datenpacket 3**

data3		
Attribut	Datentyp	Beschreibung
id	int	Identifikationsnummer
id_data_2	int	Fremdschlüssel von Datenpacket 2
date_time	datetime	Datum der Erstellung
vehicle_speed	int	Geschwindigkeit des Motorrades
vehicle_accelration	int	Eingangsspannung der Motorsteuerung
vehicle_odometer	int	Vom Motorrad zurückgelegte Strecke
time_to_capture_speed	int	Zeit bis zur eingestellten Geschwindigkeit
time_to_capture_distance	int	Zeit bis zur eingestellten Distanz
braking_distance_captured	int	Zeit von Bremsbeginn bis zu Stillstand
distance_since_stop	int	Zurückgelegte Distanz seit dem Stillstand

Tabelle V.5: Datenbankstruktur der Datenpaket 3 Tabelle

## Datenpacket 4 - Fehler

data4		
Attribut	Datentyp	Beschreibung
id	int	Identifikationsnummer
id_data_3	int	Fremdschlüssel von Datenpacket 3
date_time	datetime	Datum der Erstellung
vehicle_speed	int	Geschwindigkeit des Motorrades
vehicle_acceleration	int	Eingangsspannung der Motorsteuerung
vehicle_odometer	int	Vom Motorrad zurückgelegte Strecke
time_to_capture_speed	int	Zeit bis zur eingestellten Geschwindigkeit
time_to_capture_distance	int	Zeit bis zur eingestellten Distanz
braking_distance_captured	int	Zeit von Bremsbeginn bis zu Stillstand
distance_since_stop	int	Zurückgelegte Distanz seit dem Stillstand

Tabelle V.6: Datenbankstruktur der Fehler-Datenpaket Tabelle

### 6.1.3 Fehler Tabelle

Diese Tabelle ist ebenso in der Datenbank gespeichert. Sie beinhaltet die Fehlercodes der Motorsteuerung mit weiteren Informationen wie Name und Beschreibung. Sie muss per Hand angelegt werden und sollte während dem Testen der Maschine mit dokumentiert, um sie dann in das Fehler Fenster der Benutzeroberfläche zu implementieren.

### 6.1.4 Akku Daten

Da die Kommunikation der BMS noch nicht umgesetzt wurde, werden von Ihr keine Daten auf den CAN-Bus gelegt und daher muss auch keine Tabelle dafür erstellt werden. Somit werden alle Daten über Akku und Ladestand über die Motorsteuerung empfangen.

## 6.2 Handler

Die Handler Klasse hat die Aufgabe eine Verbindung zwischen der Datenbank und dem Programm herzustellen. Dazu verwendet sie die MySQL connector Bibliothek. Mit dieser können SQL Skripte direkt in die Python Klasse integriert werden ohne dass externe SQL Skript Dateien geöffnet werden müssen.

### 6.2.1 Konfigurieren der Schnittstelle

```
# Database credentials
dbhost = "localhost"
dbuser = "root"
dbpass = "8Yf!97mmnZbvYJe"
dbname = "login_data"

# Connection to the database
con = mysql.connector.connect(host=dbhost, user=dbuser, password=dbpass,
                               database=dbname)

# Creating cursor for database queries
cursor = con.cursor()
```

Code Listing V.3: Konfiguration der Datenbankschnittstelle

### 6.2.2 SELECT Befehl

Der SELECT Befehl wird zum Abrufen von Daten einer Tabelle verwendet und besitzt mehrere Parameter.

```
INSERT INTO `users` ('id', 'username', 'password')
VALUES (1, 'USER', '5AHET');
```

```
INSERT INTO `users` ('id', 'username', 'password')
VALUES (2, 'GUEST', '00000');
```

```
INSERT INTO `users` ('id', 'username', 'password')
VALUES (3, 'ADMIN', '53AC2');
```

Code Listing V.4: Konfiguration der Datenbankschnittstelle

Der Befehl kann in drei Bereiche unterteilt werden. In der ersten Zeile werden die Attribute, die von der Tabelle Tutorials ausgegeben werden sollen, definiert. Da sich diese jedoch nicht in Tutorials befinden, werden im mittleren Abschnitt alle Tabellen durch den Befehl JOIN anhand der Fremdschlüssel verbunden. Die anzugebenden Ergebnisse werden durch den *WHERE* Befehl gefiltert werden. Dieser Befehl zeigt die Attribute id, step id sowie detection aller Tutorials vom Programm mit dem Namen Windows Settings an.

### 6.2.3 INSERT Befehl

Der INSERT Befehl wird zum Schreiben von Daten in eine Tabelle verwendet.

```
INSERT INTO `users` ('id', 'username', 'password')
VALUES (1, 'USER', '5AHET');
```

```
INSERT INTO `users` ('id', 'username', 'password')
VALUES (2, 'GUEST', '00000');
```

```
INSERT INTO `users` ('id', 'username', 'password')
VALUES (3, 'ADMIN', '53AC2');
```

Code Listing V.5: Einfügen der Benutzer und Passwörter über den

Dieser Befehl kann in zwei Teile unterteilt werden. In der ersten Zeile wird die Zieltabelle mit den Attributen angegeben. In der zweiten Zeile folgen die Werte, die in der Datenbank erfasst werden sollen. Hierbei ist die Reihenfolge der Attribute zu beachten. Das Attribut id wird standardmäßig von dem Datenbanksystem vergeben, es ist daher möglich das Attribut im Befehl zu vernachlässigen. Ebenso unterstützt MySQL als Wert die Funktion *NOW()*, wodurch intern der aktuelle Zeitstempel abgespeichert wird. Dadurch kümmert sich das Datenbanksystem um die korrekten Werte für das *date\_time* Attribut.

# Kapitel VI

## Antriebsstrang

««< HEAD

### 1 Übersicht

Die Hauptaufgabe des Antriebssystems ist die Umwandlung der, vom Akkumulator zur Verfügung gestellten, elektrischen Energie in die kinetische Antriebsenergie. Diese tritt zuerst rotatorisch am Motor auf und wird zunächst über das Direkt-Getriebe umgeformt bzw. auf die passende Drehzahl gebracht, anschließend wird die Rotationsenergie mithilfe des Hinterrades auf die Straße übertragen und das ganze Motorrad beschleunigt. Neben dem Antrieb des Motorrades hat die Motorsteuerung noch weitere Bedeutung als Steuereinheit, sie fungiert als Bindeglied zwischen dem Human-Computer Interacting System und den elektrischen Anforderungen an das Gesamtsystem.

#### 1.1 Grundfunktionen des Systems

Die geplanten Aufgaben des Antriebssystems lassen sich grob in zwei Grundfunktionen einteilen:

- **Der Antrieb**

Translation ist eine Grundfunktion eines jeden Verkehrsmittels.

Durch die Umwandlung der elektrischen Energie in kinetische Energie erfährt das gesamte System eine Beschleunigung in Fahrtrichtung.

- **Die Steuereinheit**

Steuerung und Kommunikation mit anderen Betriebsmitteln,  
realisiert durch In- und Outputs, Datenübergabe mithilfe des CAN-Buses.

Um auf die einzelnen Details des Antriebssystems besser eingehen zu können, unterscheiden wir zwischen dem Hardwareaufbau und dem Softwareaufbau des Antriebssystems.

## 2 Hardwareaufbau des Antriebssystems

Der grundsätzliche Hardwareaufbau des Antriebssystems lässt sich in zwei galvanisch getrennte Stromkreise und die mechanische Umsetzung unterscheiden:

- **Mechanische Umsetzung (Kraftübertragung und Montage)**  
Umfasst das Getriebe und die Befestigung aller Komponenten am Rahmen.
- **Der Laststromkreis**  
Beinhaltet die Verbindung des Motorcontrollers mit dem Motor und dem Akkumulator.
- **Der Steuerstromkreis**  
Beinhaltet alle elektrischen Verbindungen, welche mithilfe des 35-poligen Niederleistungs-Steckers mit dem Motorcontroller verbunden sind.

## 2.1 Mechanische Umsetzung

Die Fertigung des Getriebes und die Montage der einzelnen Betriebsmittel wurde vollständig von Tobias Schmeisser übernommen.

## 2.2 Der Laststromkreis

Der Laststromkreis beinhaltet alle leistungsführenden Betriebsmittel des Antriebssystems. Hierbei unterscheiden wir zwischen den zwei wichtigsten Grundfunktionen:

- **Elektrische Energieübertragung**

Umfasst die elektrische Verbindung von Motor, Motorcontroller und Akkumulator. Realisiert durch einfache Leitungen, um Leistungen übertragen zu können.

- **Schutz der Komponenten vor Beschädigungen (Leitungsschutzorgane)**

Beinhaltet eine Schmelzsicherung zum Schutz vor Überströmen und ein Hochleistungs-Relais, um im Fehlerfall den Laststromkreis öffnen zu können und damit eine galvanische Trennung des Antriebs und der Energieversorgung gewährleisten zu können.

Im folgenden Bild ist der Grundaufbau des Laststromkreises ausführlich beschrieben:

R1 ... Vorladewiderstand

F1 ... Schmelzsicherung

S4.1 ... Hauptschütz (Hochleistungs-Relais)

S4.2 ... Vorladeschütz

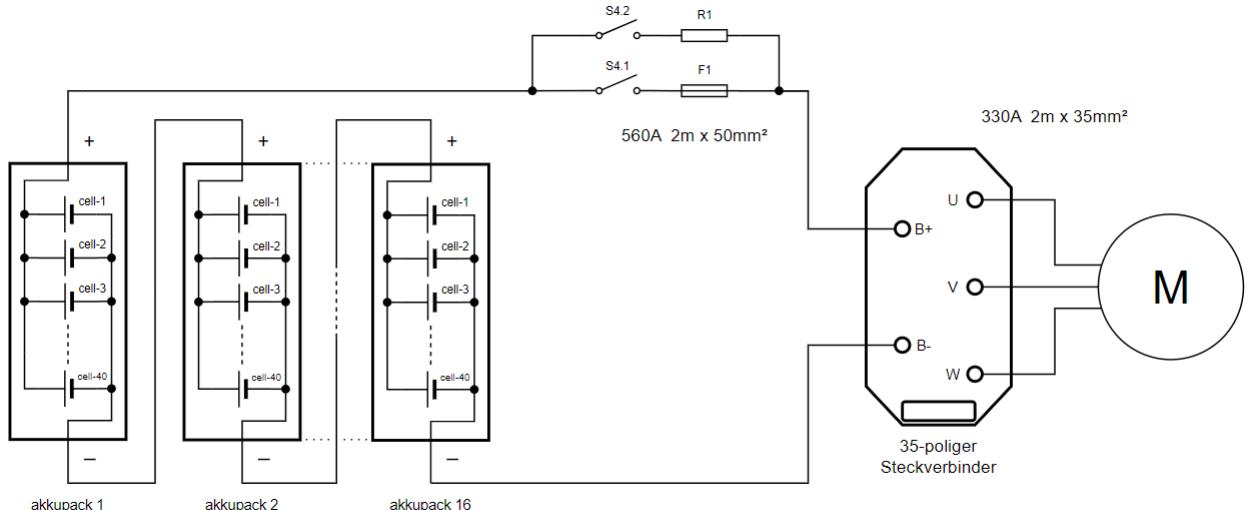


Abbildung VI.1: Grundaufbau des Laststromkreises

### 2.2.1 Elektrische Energieübertragung

Um die benötigte elektrische Energie übertragen zu können, müssen die Leitungen an den Leistungsverbrauch des Verbrauchers (Motor) angepasst werden. Bei einer zu hohen Stromaufnahme (Überlast) des Motors kann es zu einer übermäßigen Erwärmung der Leitungen bis hin zu dauerhaften Beschädigungen, wie Durchschmoren der Isolierung, oder sogar einen Leistungsbrand führen. Um dies verhindern zu können, müssen die Leitungen an die Stromaufnahme des Motors angepasst werden. Das heißt, der zulässige Dauerstrom der Leitungen muss den maximalen Dauerstrom des Motors bzw. den maximalen Dauerstrom, welcher durch den Akkumulator zur Verfügung gestellten werden kann, übersteigen.

#### Auswahlkriterien:

Bei der Auswahl von Leitungen muss man grundsätzlich den Querschnitt und die Länge der Leitung an die benötigte Stromaufnahme des Motors anpassen. Bestimmte weitere Anforderungen, wie zum Beispiel ein maximal zulässiger Spannungsabfall, müssen ebenfalls berechnet und berücksichtigt werden. Ein gutes Zusammenspielen der Leitungen mit den anderen Leitungsschutzeinrichtungen ist ebenfalls sehr wichtig, jedoch können diese Bauteile auch sehr gut an die bereits festgelegten Eigenschaften der Leitungen angepasst werden.

#### Berechnung:

In diesem Abschnitt befassen wir uns zunächst mit der Berechnung aller Ströme, die wir zur Auswahl der richtigen Leitungen benötigen. Da wir den maximalen Strom, welcher durch eine Zelle zur Verfügung gestellt werden kann, bereits aus dem Datenblatt kennen, können wir von der Seite des Akkumulators mit der Berechnung beginnen. Zuerst berechnet man also den maximalen Strom, welcher sich aus der Parallelschaltung der Zellen ergibt. Der Motor erhält nahezu die selbe Leistung, weshalb der Motorstrom durch die Annordnung der Phasen ebenfalls bestimmt werden kann. Da die Größenordnungen der beiden Ströme im Akkumulator-Stromkreis und im Motor-Stromkreis sehr unterschiedlich sind, können ebenfalls verschiedene Leitungsquerschnitte verwendet werden. Die Berechnung des Leitungswiderstands lässt nun auch auf den Spannungsabfall schließen, diese Berechnung muss jedoch bei beiden Stromkreisen erfolgen.

- **Berechnung der Ströme:**

Maximaler Strom, der von einer Zelle zur Verfügung gestellt werden kann: 14 A  
Anzahl der Zellen, die in einem Akkupack parallel verschaltet werden: 40

- **Akkumulator-Stromkreis:**

Querschnitt  $A_A = 70 \text{ mm}^2$   
Leitungslänge  $l_A = 1\text{m}$   
Spezifischer Widerstand von Aluminium: 0,0278

- **Motor-Stromkreis:**

Querschnitt  $A_M = 35 \text{ mm}^2$   
Leitungslänge  $l_M = 1\text{m}$   
Spezifischer Widerstand von Kupfer: 0,01786

Berechnung der Ströme:

$$I_{Zmax} = 14 \text{ A}$$

$$I_{Amax} = 40 \cdot I_{Zmax} = 40 \cdot 14 \text{ A} = 560 \text{ A}$$

$$I_{Mmax} = \frac{I_{Amax}}{\sqrt{3}} = \frac{560 \text{ A}}{\sqrt{3}} = 323 \text{ A}$$

Spannungsabfall Akkumulator-Stromkreis:

$$R_A = \frac{\delta_A \cdot l_A}{A_A} = \frac{0,0278 \frac{\Omega \cdot \text{mm}^2}{\text{m}} \cdot 1 \text{ m}}{70 \text{ mm}^2} = 0,397 \text{ m}\Omega$$

$$\Delta U = R_A \cdot I_{Amax} = 0,397 \text{ m}\Omega \cdot 560 \text{ A} = 222 \text{ mV}$$

$$U_{VB} = U_Q - \Delta U = 50,4 \text{ V} - 222 \text{ mV} = 50,178 \text{ V}$$

Spannungsabfall Motor-Stromkreis:

$$R_M = \frac{\delta_K \cdot l_M}{A_M} = \frac{0,01786 \frac{\Omega \cdot \text{mm}^2}{\text{m}} \cdot 1 \text{ m}}{35 \text{ mm}^2} = 0,51 \text{ m}\Omega$$

$$\Delta U = R_M \cdot I_{Mmax} = 0,51 \text{ m}\Omega \cdot 323 \text{ A} = 165 \text{ mV}$$

$$U_{VB} = U_Q - \Delta U = 50,4 \text{ V} - 165 \text{ mV} = 50,235 \text{ V}$$

### Fazit:

Bei der Anwendung als E-Motorrad spielen der Widerstand und der damit verbundene Spannungsabfall eigentlich keine Rolle, da die Leitungslängen bezogen auf den Querschnitt klein sind. Hinzu kommt noch, dass die Leitungslängen in dieser Berechnung sicherheitshalber größer angenommen wurden, als sie in Wirklichkeit umgesetzt werden, da die Leitungslängen vorab schwer abgeschätzt werden können und nach der Montage ebenfalls variieren werden. Der Curtis Controller ist ebenfalls sehr unempfindlich gegenüber Spannungsschwankungen, welche so oder so durch das Laden und Entladen des Akkumulators entstehen.

Der Aluminiumleiter verfügt über eine maximale Dauerbelastung von 160A, was deutlich unter dem Maximalstrom liegt. Bei unserer Anwendung als Motorrad ist die Dauerbelastung aber eher unwichtig, da nur kurzzeitig hohe Leistungen benötigt werden (Beschleunigungsvorgang). Weiters wird der vom Motor angeforderte Strom dauerhaft vom Batterie-Management-System und vom Curtis Controller überprüft und reguliert bzw. im Notfall abgeschaltet. Kommt es zu starken Erwärmungen im Motor, der Motorsteuerung oder dem Akkumulator, wird die Maximalleistung gedrosselt.

## 2.2.2 Leitungsschutzorgane

Die Aufgabe der Leitungsschutzorgane ist es, bei unerwarteten Überströmen oder in einem Fehlerfall den Laststromkreis zu öffnen, um damit den Motor bzw. Motorcontroller vom Akkumulator galvanisch zu trennen. Diese Maßnahme wird ergriffen, um mögliche Beschädigungen an den Komponenten oder an den Leitungen verhindern zu können. Da jedoch ungewünschte Fehlauslösungen zum sofortigen Stillstand des Motorrades führen und eventuell sogar benötigte Wartungen (Wechsel der durchgebrannten Schmelzsicherung) nach sich ziehen, müssen diese Leitungsschutzorgane sehr sorgfältig ausgewählt werden. Eine Überdimensionierung ist ebenso unerwünscht, denn dadurch steigen die Anschaffungskosten der Bauteile. Das eher größere Problem entsteht jedoch bei der Überdimensionierung der Schmelzsicherung, denn diese löst nun zu spät aus und hat damit nur mehr eine nicht geeignete Schutzfunktion.

### Hochleistungs-Relais

Bei der Auswahl des Hochleistungs-Relais muss vor allem der maximale Strom, der vom Relais geschalten werden kann, höher als der Verbraucherstrom bei maximaler Auslastung sein. Ebenfalls werden die zu erwarteten Lebenszyklen mithilfe einer speziellen Kennlinie abgeschätzt, welche diese Zyklen abhängig von bestimmten Strom- und Spannungswerten angibt. Bei einer Gleichspannung von 120V und einem Strom von 600A ergibt sich eine ungefähre Lebenszeit von 5.000 Schaltvorgängen. Die oben genannten Werte sind aber entsprechend der Kennlinie deutlich höher als die realen Leistungswerte, ebenfalls ist der Schaltvorgang meist nahezu unbelastet, da die Kondensatoren vorgeladen werden und das Hochleistungs-Relais nur in einem Fehlerfall während des Betriebs geöffnet wird. Aufgrund dessen kann auf eine Lebenszeit von deutlich über 10.000 Schaltvorgängen geschlossen werden. Natürlich muss auch die Spulenspannung und der zugehörige Leistungsverbrauch für die gewünschte Anwendung passen. Die Spulenspannung wurde passend zu den anderen Bauteilen für 12V ausgewählt, der Leistungsverbrauch befindet sich im Bereich von wenigen Watt und ist damit vernachlässigbar.

### Schmelzsicherung

Bei der Auswahl der Schmelzsicherung ist es vor allem wichtig, dass der Stromkreis bei einem Kurzschlussfall in kurzer Zeit unterbrochen wird, die zulässigen Nennströme jedoch dauerhaft geleitet werden können. Das bedeutet, der Kurzschlussstrom muss um ein Vielfaches größer sein, als der Nennstrom der Schmelzsicherung. Da jedoch die Kurzschlussimpedanz im Bereich von wenigen Milliohm liegt, ist der Kurschlusstrom mindestens 10 mal so groß wie der Nennstrom der Schmelzsicherung, was bei der ausgewählten Sicherung eine Ausschaltzeit von wenigen Millisekunden bedeutet. Normalerweise ist es ebenfalls in Bezug auf die Leitungen wichtig, dass der zulässige Dauerstrom der Leitungen größer als der Nennstrom der Schmelzsicherung ausgewählt wird, um Beschädigungen an den Leitungen verhindern zu können. In unserem Fall ist dies aber eher nebensächlich, da der vom Motor angeforderte Strom, wie bereits erwähnt, dauerhaft durch das Batterie-Managemen-System überprüft und begrenzt wird.

### Fazit:

benötigt?

### 2.2.3 Motorbeschreibung

Die Nennleistung von Motor und Motorcontroller übersteigen die Nennleistung des Akkumulators, weshalb die Nennwerte dieser Bauteile unwichtig für die Dimensionierung der Leitungen und Leitungsschutzorgane ist. Da jedoch Motor und Motorsteuerung die zentralen Elemente des E-Motorrades darstellen, werden unter diesem Punkt bestimmte Eigenschaften und Nenndaten genauer beschrieben.

Für die Anwendung in einem E-Motorrad bot sich die Verwendung eines bürstenlosen permanenterregten Gleichstrommotors an. Da jedoch eine Motorsteuerung die drei Phasen des Motors mit Drehstrom versorgt, wirkt dieser Motor eigentlich wie eine Synchronmaschine mit einem Permanentmagneten im Läufer.

Der Motor wurde von der Firma Ashwoods gebaut und hat die Bezeichnung „IPM-200-50“. Er besitzt eine mechanische Spitzenleistung von 16 kW und ein maximales Drehmoment von 74 Nm. Die maximale Drehzahl beträgt 8500 U/min und der Wirkungsgrad liegt bei circa 85%.

Die Motorsteuerung wurde von der Firma Curtis gebaut, der Controller ist ein Prototyp mit der Modelnummer „AC-F4-A-Proto-002“. Er besitzt einen maximalen Motorstrom von 450A RMS. Im Betrieb hat der Motorcontroller einen Leistungsverbrauch von rund 500W.

Betrachtet man also beide Bauteile gemeinsam, ergibt das eine maximale Leistung von 19,5 kW, sprich einen maximalen Strom von 385A bei einer Nennspannung von 50,4V.

## 2.3 Der Steuerstromkreis

### 2.3.1 Übersicht Ein- und Ausgänge

Der Steuerstromkreis umfasst mit alle elektrischen Verbindungen, welche über den 35-poligen Niederleistungs-Stecker mit dem Motorcontroller verbunden sind. Hierbei unterscheiden wir grundsätzlich zwischen drei verschiedenen Ports, welche nochmals unterkategorisiert werden können:

- **Eingänge (Inputs)**
  - Digitale Eingänge (Digital Inputs)
  - Analoge Eingänge (Analog Inputs)
  - Gas- und Bremseingänge (Throttle and Brake Inputs)
  - Positionsrückmeldung vom Encoder (Position-feedback Input)
  - Prozessorversorgung und Spulenrücklauf (KSI and Coil Return)
- **Ausgänge (Outputs)**
  - Analoge Ausgänge (Analog Outputs)
  - Digitale und Pulsweitenmodulierbare Ausgänge (Digital and PWM Outputs)
  - Spannungsversorgungs-Ausgänge (Power Supply Outputs)
- **Kommunikation (Communication)**
  - CAN-Bus (CAN-Port)
  - Serielle Schnittstelle (Serial-Port)

Der Motorcontroller verfügt über viele Pins, welche über mehrere Funktionen verfügen, es muss jedoch eine dieser Funktionen ausgewählt werden. Pin 6 zum Beispiel wird eigentlich als digitaler und phasenmodulierbarer Ausgang verwendet, bei richtiger Konfiguration kann er jedoch auch als digitaler Input verwendet werden. Weiteres kann frei konfiguriert werden, ob man mit diesem Ausgang zum Beispiel das Hochleistungs-Relais, eine Pumpe oder Bremslichter ansteuern möchte, je nach gewünschter Anwendung gibt es auch unterschiedliche Funktionen. Um den passenden Pin für eine Anwendung auswählen zu können, muss man jedoch die elektrischen Eigenschaften der Pins genauer unter die Lupe nehmen. Oftmals haben auch die Pins der selben Unterkategorie verschiedene Funktionen, Eingangsimpedanzen oder Toleranzen.

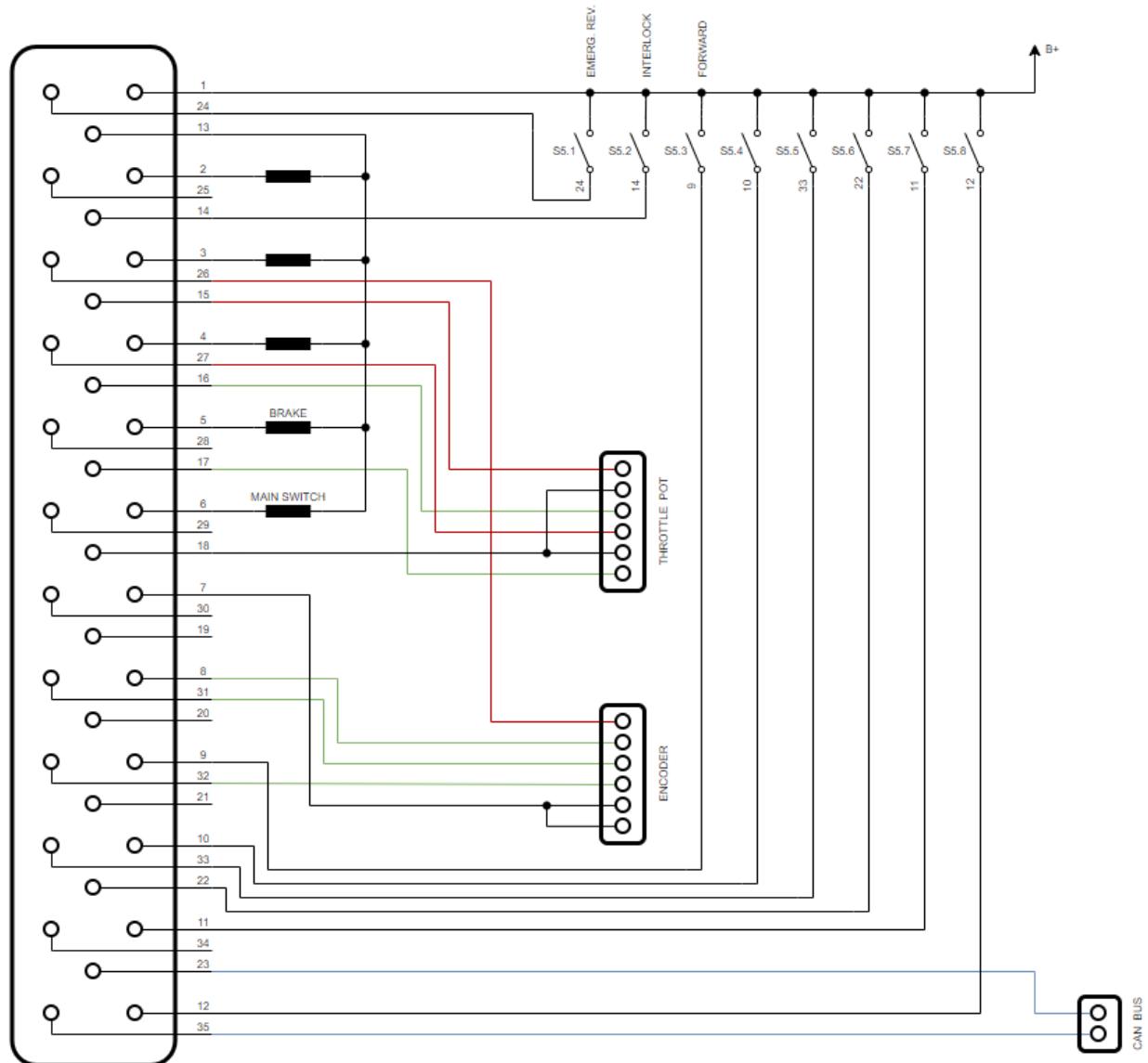


Abbildung VI.2: Grundaufbau des Steuerstromkreises

### 2.3.2 Digitale Eingänge (Digital Inputs)

Es gibt insgesamt 16 Pins, die als digitale Eingänge genutzt werden können, jedoch werden sieben Pins davon eigentlich als Ausgänge konfiguriert.

DIGITAL INPUT SPECIFICATIONS						
Name	Pin	Logic Thresholds	Input impedance*	Voltage range†	ESD Tolerance	
Switch 1	24	Rising edge= 4.4V max Falling edge= 1.5V min	24-36V models: 7.0 kΩ, 7.2 kΩ 36-48V models: 10.8 kΩ, 11.2 kΩ 48-80V models: 25.2 kΩ, 27.3 kΩ 72-96V models: n/a, 29.4 kΩ	-10V to (MaxV + 10 V)	± 8 kV (direct strike)	
Switch 2	8					
Switch 3	9					
Switch 4	10					
Switch 5	11					
Switch 6	12					
Switch 7	22					
Switch 8	33			-5V to (MaxV + 10 V)		
Switch 16	14					
Digital Out 6	19					
Digital Out 7	20					
Driver 1	6					
Driver 2	5					
Driver 3	4					
Driver 4	3					
Prop Driver	2					

Abbildung VI.3: Digital Input Specifications

### 2.3.3 Analoge Eingänge (Analog Inputs)

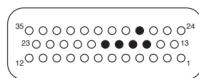
Es gibt insgesamt zwei Pins, die als analoge Eingänge verwendet werden können. Ein Pin davon wird jedoch im Normalfall für den Motortemperatur-Sensor verwendet. Die Eingänge, die für das Gas- und Bremspotentiometer verwendet werden, sind in dieser Kategorie nicht aufgelistet, obwohl diese ebenfalls als analoge Eingänge genutzt werden. Diese Pins sind jedoch speziell für die Gas- und Bremssteuerung konfiguriert und sollten im Normalfall auch dafür verwendet werden.

ANALOG INPUT SPECIFICATIONS						
Signal Name	Pin	Operating Voltage	Input impedance*	Protected Voltage	ESD Tolerance	
Analog 1	24	0 to 10V in 1024 steps	24-36 V models: 6.9 kΩ, 7.1 kΩ 36-48 V models: 10.5 kΩ, 11.0 kΩ 48-80 V models: 23.8 kΩ, 28.1 kΩ 72-96 V models: n/a, 28.1 kΩ	-10 V to (MaxV + 10 V)	± 8 kV (direct strike)	
Analog 2	8					

Abbildung VI.4: Analog Input Specifications

### 2.3.4 Gas- und Bremseingänge (Throttle and Brake Inputs)

Die zwei Gas- und Bremssteuerungs-Eingänge sind wegen der speziellen Auslegung von den analogen Eingängen abgegrenzt und können unabhängig voneinander programmiert werden. Sie sind optimiert für die Anwendung mittels Spannungssteuerung, 2-Draht Widerstandssteuerung oder 3-Draht Widerstandssteuerung. Bei der Spannungssteuerung benötigt man die Pins Pot Wiper und I/O Ground, bei der 2-Draht Widerstandssteuerung Pot Wiper und Pot Low und bei der 3-Draht Widerstandssteuerung Pot High, Pot Wiper und Pot Low. In unserem Fall benutzen wir beide Steuerungs-Eingänge für die 3-Draht Widerstandssteuerung, da der Gasdrehgriff über eine Drahtbrucherkennung verfügt. Das heißt, der Gasdrehgriff hat insgesamt zwei unabhängige 3-Draht Potentiometer-Ausgänge, welche beide für den Gaseingang benutzt werden.

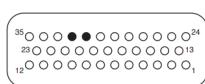


THROTTLE INPUT SPECIFICATIONS						
Signal Name	Pin	Operating Voltage	Input Impedance	S/Sink Current	Protected Voltage	ESD Tolerance
Throttle Pot High	15	0 V (shorted to Pot Low) 5 V (open circuit)	N/A	1 mA nominal (source)	-0.5 V to (MaxV + 10 V)	± 8 kV (direct strike)
Pot2 High	27					
Throttle Pot Wiper	16	0 to 6.25 V	100 kΩ min	0.76 mA nominal (source, 2-wire)	-1 V to (MaxV + 10 V)	± 8 kV (direct strike)
Pot2 Wiper	17					
Pot Low	18	0 to 0.25 V	20 Ω nom.	Faults if above 15 mA (sink)		

Abbildung VI.5: Throttle Input Specifications

### 2.3.5 Positionsrückmeldung vom Encoder (Position-Feedback Input)

Diese zwei Pins sind intern dafür konfiguriert, die aktuelle Position der Motorwelle einzulesen, um eine optimale feldorientierte Ansteuerung des Motors durchführen zu können. Dabei gibt es die Möglichkeiten über einen Quadratur-Encoder oder einen Sin/Cos-Encoder. Da im Ashwoods-Motor ein Sin/Cos-Sensor verbaut ist, wurde dies vorab beim Motorcontroller eingestellt.

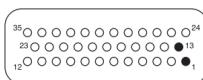


SIN/COS SENSOR INPUT SPECIFICATIONS						
Signal Name	Pin	Operating Voltage	Input Impedance	Max. Frequency	Protected Voltage	ESD Tolerance
Position Feedback A	31	0 to 5 V	150 kΩ for voltages ≤ 5 V 75 kΩ for voltages > 5 V	500 Hz	-5 V to (MaxV + 10 V)	± 8 kV (direct strike)
Position Feedback B	32					

Abbildung VI.6: Sin/Cos Sensor Input Specifications

### 2.3.6 Prozessorversorgung und Spulenrücklauf (KSI and Coil Return)

Der KSI-Eingang stellt die elektrische Versorgung aller Niederleistungs-Schaltkreise zur Verfügung. Dies beinhaltet ebenfalls die Versorgung aller Ausgänge und die Kondensator-Vorlade-Funktion, welche dazu dient, die Kondensatoren über einen Widerstand vorzuladen, um hohe Einschaltströme zu verhindern. Der Spulenrücklauf stellt die Versorgung der pulsweitenmodulierbaren Ausgänge zur Verfügung und hat die gleiche Spannung wie der KSI-Pin. Die elektrische Trennung von KSI und Coil Return muss aufrechterhalten werden, um einen Verpolungsschutz gewährleisten zu können.



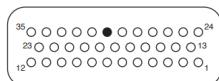
KSI and COIL RETURN INPUT SPECIFICATIONS

Signal Name	Pin	Operating Voltage	Input Current	Protected Voltage	ESD Tolerance
KSI	1	Between under- and overvoltage cutbacks	13 A max * continuous	$\pm (\text{MaxV} + 10 \text{ V})$	$\pm 8 \text{ kV}$ (direct strike)
Coil Return	13		10 A or 12 A max **	(KSI - 0.3 V) to (MaxV + 10 V)	

Abbildung VI.7: KSI and Coil Return Input Specifications

### 2.3.7 Analoge Ausgänge (Analog Outputs)

Der analoge Ausgang kann ein Spannungssignal von 0 bis 10V ausgeben. Dieser Ausgang ist für die Ausgabe über Anzeigegeräte, wie zum Beispiel eine Anzeige über den aktuellen Ladestand des Akkumulators, vorgesehen.



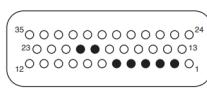
ANALOG OUTPUT SPECIFICATIONS

Signal Name	Pin	Output Voltage	Output Current	Protected Voltage	ESD Tolerance
Analog Out	30	0 to 10 V	10 mA	-1 V to (MaxV + 10 V)	$\pm 8 \text{ kV}$ (direct strike)

Abbildung VI.8: Analog Output Specifications

### 2.3.8 Digitale und Pulsweitenmodulierbare Ausgänge (Digital and PWM Outputs)

Es gibt insgesamt 7 digitale Ausgänge, wovon jedoch nur 5 für eine Pulsweitenmodulation konfiguriert werden können. Diese Ausgänge sind für induktive Lasten, wie zum Beispiel den Hauptschütz oder eine elektromagnetische Bremse, vorgesehen. Rein ohmsche Lasten können ebenfalls gesteuert werden, jedoch darf der zulässige Spitzenstrom nicht überschritten werden. Der Proportional-Driver kann bei richtiger Konfiguration auch für die Anzeige eines Tachometers hergenommen werden. Generell kann jeder Pin dieser Gruppe ebenfalls als digitaler Eingang benutzt werden.

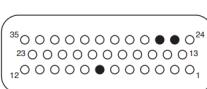


DIGITAL and PWM OUTPUT SPECIFICATIONS									
Name	Pin	PWM	PV Current	Frequency	Output Current	Protected Voltage	ESD Tolerance		
Driver 1	6	0 to 100% Duty Cycle	N/A	120 to 1000 Hz *	2A Max	-0.5 V to (MaxV + 10 V)	$\pm 8 \text{ kV}$ (direct strike)		
Driver 2	5				3A Max				
Driver 3	4				2A Max				
Driver 4	3		0 to 2A in 607 nominal steps	18 kHz					
Prop Driver	2	On / Off	N/A	N/A	1A Max				
Digital Out 6	19								
Digital Out 7	20								

Abbildung VI.9: Digital and PWM Output Specifications

### 2.3.9 Spannungsversorgungs-Ausgänge (Power Supply Outputs)

Um kleine Schaltkreise, wie zum Beispiel einen LED-Indikator oder die Positionsrückmeldung des Encoders, mit Spannung versorgen zu können, gibt es zwei dafür vorgesehene Spannungsversorgungs-Ausgänge mit je einem Pin für 5V und 12V. Für diese Anwendungen gibt es ebenfalls noch einen Rücklauf, der als I/O Ground definiert wurde.

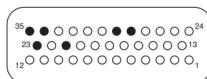


POWER SUPPLY OUTPUT SPECIFICATIONS					
Signal Name	Pin	Output Voltage	Output Current	Protected Voltage	ESD Tolerance
+12 V Out	25	11.5 to 14.5 V	100 mA max for +12 Out 100 mA max for +5 Out 200 mA max (combined total)	-1 V to (MaxV + 10 V)	$\pm 8 \text{ kV}$ (direct strike)
+5 V Out	26	5 V $\pm 10\%$			
I/O Ground	7	n/a	500 mA max	not protected	

Abbildung VI.10: Power Supply Output Specifications

### 2.3.10 Kommunikations-Ports

Für die Kommunikation mit anderen Betriebsmitteln stellt uns der Motorcontroller zwei Möglichkeiten zur Verfügung, den CAN-Bus und die serielle Schnittstelle. Da sich unser Projektteam auf die Nutzung des CAN-Busses geeinigt hat, wird die serielle Schnittstelle nicht verwendet. Die zwei Pins CAN Term High und CAN Term Low werden ebenfalls nicht benötigt, denn diese dienen nur dazu, den CAN-Bus vorübergehend funktionsunfähig zu schalten. Programmtechnisch gibt es drei Möglichkeiten zur Konfiguration des CAN-Busses, dies wird jedoch im Punkt Software genauer erklärt.



COMMUNICATIONS PORT SPECIFICATIONS

Signal Name	Pin	Supported Protocol / Devices	Data Range	Protected Voltage	ESD Tolerance
CAN H	23	CANopen, other 11-bit or 29-bit identifier protocols	up to 1 Mbit/s	-0.5 V to (MaxV + 10 V)	$\pm 8 \text{ kV}$ (direct strike)
CAN L	35			(no connection to external wiring)	
CAN Term H	21				
CAN Term L	34				
Serial TX	28	Curtis 840 Display, 1313 Handhelp Programmer, 1314 PC Programming Station	as required, 9.6 kbit/s to 56 kbit/s	-0.3 V to 12 V	
Serial RX	29				

Abbildung VI.11: Communications Port Specifications

### 3 Softwareaufbau des Antriebssystems

Der Softwareaufbau des Antriebssystems kann grob in 3 Grundfunktionen unterteilt werden:

- **Parameterbasierte Programmierung (Programmer)**

Umfasst alle Parameter und die vornehmbaren Konfigurationsmöglichkeiten.

- **Drehmomentsteuerung (Torquecontrol)**

Beinhaltet alle Parametereinstellungen, welche für die Drehmomentsteuerung ausschlaggebend sind.

- **Vehicle-Control-Language (VCL) Programmierung**

Umfasst das gesamte Programm, welches mit der Vehicle-Control-Language realisiert wurde. Ein Großteil dieses Programms beschäftigt sich hierbei mit der Kommunikation zwischen der Motorsteuerung und dem Raspberry PI.

### 3.1 Parameterbasierte Programmierung (Programmer)

#### 3.1.1 Allgemeines

In dem Curtis Integrated Toolkit befindet sich unter anderem die Funktion „Programmer“, welche sehr viele Applikationseinstellungen für die Controllerprogrammierung zur Verfügung stellt. Es gibt mehrere verschiedene Menüs, welche die jeweils zugehörigen Parametergruppen auflisten. Viele Variablen können konfiguriert werden, andere werden nur für die Ausgabe aktueller Werte wie zum Beispiel die aktuelle Geschwindigkeit oder Temperatur verwendet. Die Parameter können je nach Anwendung und Funktion verschiedene Einheiten und einstellbare Wertebereiche besitzen. Sehr vorteilhaft sind hierbei die meist vorhandenen Kurzbeschreibungen im Curtis Integrated Toolkit oder die Parameterdeklaration in der Anleitung.

#### 3.1.2 Funktionen

##### Throttle and Brake

Die Hauptfunktion dieser Parametergruppen ist die genaue Einstellung der Potentiometer-Eingänge für den Bremsvorgang, Vorwärts- und Rückwärtsbetrieb. Es können der Start- und Endwert des Potentiometer festgelegt werden, ebenso bestimmte Formänderungen der Eingangskurve, welche die Intensität des Gas- bzw. Bremsbefehls abhängig von der Potentiometerstellung beschreibt. Ebenso gibt es mehrere mögliche Sicherheitskonfigurationen, Filtereinstellungen und auch die Möglichkeit der Ansteuerung mittels VCL.

##### CAN Interface

Das CAN-Interface wird in unerer Anwendung nicht benötigt, denn die Kommunikation wurde mittels VCL-CAN umgesetzt. Die vorgefertigte Version CANopen bietet jedoch auch mehrere Einstellungsmöglichkeiten, wie ID und Taktfrequenzanpassungen. Ebenfalls gibt es jeweils 4 Datenpakete, welche für den Senden- und Empfangenvorgang konfiguriert werden können.

##### Battery Setup

In diesem Parametermenü kann die Nennspannung festgelegt werden, ebenfalls gibt es mehrere Einstellungsmöglichkeiten, welche in einem Über- oder Unterspannungsfall von Bedeutung sind. Um den Ladestand des Akkumulators besser abschätzen zu können, sollten hier auch bestimmte Eigenschaften festgelegt werden.

##### Main Contactor

Dieses Menü dient zur Konfiguration Hauptschütz (Hochleistungs-Relais). Die Ansteuerungsspannung wird hier genau beschrieben, nach dem Schließen des Schützes kann eine kleiner Spannung zum halten der Kontakte verwendet werden. Ebenfalls kann eine Zeitverzögerung eingestellt werden, welche bestimmt wie lange der KSI-Pin geöffnet sein darf, bevor der Hauptschütz geöffnet wird. Diese Funktion kann sehr wichtig sein, denn ein Wechsel des Antriebsmodus kann somit ebenfalls während des Betriebs erfolgen (Gaseingang muss aber 0% sein).

##### Pump Contactor

##### EM Brake Control

##### Emergency Reverse (EMR)

##### Interlock Braking

##### Dual Drive

##### Vehicle und Supervision

## Current Limits

## 3.2 Drehmomentsteuerung (Torquecontrol)

Die Drehmomentsteuerung wird mithilfe von 12 unterschiedlichen Parametern beschrieben. Generell können diese Parameter eher als Feineinstellungen angesehen werden, da der grundsätzliche Vorgang immer der gleiche bleibt. Mit diesen Parametern werden jedoch die maximale Geschwindigkeit, die Reaktionszeit und die Aggressivität des Motors genau definiert, um das gewünschte Beschleunigungsmuster erhalten zu können.

### 3.2.1 Parameter

Grundsätzlich lassen sich diese Parameter in zwei Hauptgruppen unterscheiden:

- **Geschwindigkeitsbegrenzer-Parameter (Speed-Limiter)**

Bestimmen die maximale Geschwindigkeit und die Begrenzungsparameter des KPI-Reglers.

- **Reaktions-Parameter (Response)**

Bestimmen die verschiedensten Ansprechzeiten und Drehzahlen für bestimmte Ereignisse.

Diese Parameter werden in den folgenden Tabellen noch genau erklärt. Für ein besseres Verständnis können ebenfalls die englischen Tabellen „SPEED LIMITER MENU“ und „RESPONSE MENU“, aber auch die zugehörigen Abbildungen „Figure 9-11“ aus der Anleitung herangezogen werden.

Parameter	Einstellungsbereich	Beschreibung
Max Speed Max_Speed_TrqM	500 - 8000 U/min 500 - 8000	Definiert die maximale Geschwindigkeit in U/min, welche mittels der Drehmomentsteuerung angesteuert werden kann. (Unabhängig von der Gasdrehgriff-Stellung)
Kp Kp_TrqM	0 – 100 % 0 – 8192	Legt fest, wie aggressiv der Drehzahlregler versucht, die Motordrehzahl auf die maximale Drehzahl zu begrenzen. Größere Werte sorgen für eine genauere Kontrolle, können jedoch zu Schwankungen führen. Bei einem zu niedrigen Kp kann die maximal Geschwindigkeit den Parameter Max Speed überschreiten.
Ki Ki_TrqM	5 – 100 % 50 – 1000	Mit diesem Parameter kann die Integralregelung genauer beschrieben werden, welche die Drehzahlbegrenzung unterschiedlich stark beeinflusst, abhängig von der aktuellen Regelabweichung. Größere Werte ermöglichen eine schnellere Regelung, können jedoch zu Schwankungen führen. Bei zu niedrigen Werten kann es lange dauern, bis sich der Motor bei einer Überdrehzahl dem Max Speed nähert.
Kd Kd_TrqM	0 – 100 % 0 – 8192	Beschreibt die Dämpfung, wenn sich das Fahrzeug der Höchstgeschwindigkeit nähert, dadurch werden Überschwingungen verringert. Bei einem zu hohen Kd kann es lange dauern, bis die Höchstgeschwindigkeit erreicht wird. Wenn Kd zu niedrig eingestellt ist, kann die Höchstgeschwindigkeit überschritten werden, insbesondere beim bergab Fahren.

Tabelle VI.1: Geschwindigkeitsbegrenzer-Parameter

Parameter	Einstellungsbereich	Beschreibung
Accel Rate Accel_Rate_TrqM	0.1 - 30.0 s 100 - 30000	Legt fest, wie lange es dauert, bis das Motordrehmoment bei Vollgas auf das Maximum angestiegen ist. Größere Werte bedeuten eine langsamere Reaktion.
Accel Release Rate Accel_Release_Rate_TrqM	0.1 - 2.0 s 100 - 2000	Legt fest, wie schnell die Verzögerung des Fahrzeugs bei einem loslassen des Gasdrehgriffs eingeleitet wird. Bei einem geringen Wert wird der Übergang abrupt eingeleitet. Ist der Wert zu hoch eingestellt, fährt das Fahrzeug für kurze Zeit weiter.
Brake Rate Brake_Rate_TrqM	0.1 - 5.0 s 100 - 5000	Legt fest, wie lange es dauert, bis das Bremsmoment bei einem startenden Bremsvorgang oder einem Fahrtrichtungswechsel aufgebaut wird. Größere Werte bedeuten einen schonenderen Bremsvorgang.
Brake Release Rate Brake_Release_Rate_TrqM	0.1 - 2.0 s 100 - 2000	Beschreibt, wie schnell sich das Bremsmoment löst, wenn das Fahrzeug vom Bremsvorgang zum Fahrbetrieb wechselt. Bei zu hohen Werten wird der Bremsvorgang noch kurzzeitig fortgeführt.
Neutral Braking Neutral_Braking_TrqM	0 – 100 % 0 - 32767	Der neutrale Bremsvorgang tritt auf, wenn der Gasdrehgriff losgelassen wird oder keine Fahrtrichtung gewählt wurde. Der neutrale Bremsparameter ist von 0 bis 100% des maximalen Rekuperationsstromes einstellbar.
Neutral Taper Speed Neutral_Taper_Speed_TrqM	200 - 6000 U/min 200 - 6000	Legt die Motordrehzahl fest, an welcher der neutrale Bremsstrom rückgespeist wird, wenn der Gasdrehgriff losgelassen wird. Bei einem zu geringen Wert kann es zu Schwankungen kommen.
Forward Full Restraint Speed Forward_Full_Restraint_Speed_TrqM	100 – 32000 U/min 100 - 32000	Legt den Geschwindigkeitspunkt fest, an dem der volle Rekuperationsstrom rückgespeist wird, um das Vorwärtsrollen des Fahrzeugs zu verhindern. Kann auch als Parameter für die Rückhaltestärke angesehen werden. Bei einem zu geringen Wert kann es zu Schwankungen kommen.
Back Full Restraint Speed Back_Full_Restraint_Speed_TrqM	100 – 32000 U/min 100 – 32000	Legt den Geschwindigkeitspunkt fest, an dem der volle Rekuperationsstrom rückgespeist wird, um das Rückwärtsrollen des Fahrzeugs zu verhindern. Kann auch als Parameter für die Rückhaltestärke angesehen werden. Bei einem zu geringen Wert kann es zu Schwankungen kommen.

Tabelle VI.2: Reaktions-Parameter

### 3.2.2 ECO- und Sportmodus (Speed-Mode-Select)

Die oben genannten Parameter werden für die Erstellung der unterschiedlichen Antriebsmodi verwendet. Für den Ecomodus werden für die Parameter höhere Reaktionszeiten und weniger Aggressivität gewählt, daraus ergibt sich ein gemütlicherer und schonenderer Fahrbetrieb, welcher gleichzeitig weniger Leistung verbraucht. Für den Sportmodus hingegen werden kürzere Reaktionszeiten und höhere Aggressivität gewählt, daraus ergibt sich ein sportlicheres Fahrverhalten, die gewünschte Geschwindigkeit wird schneller erreicht. Natürlich zieht dieser Antriebsmodus einen erhöhten Leistungsverbrauch nach sich. Der Wechsel des Antriebsmodus bedeutet jedoch eine Veränderung kritischer Parameter, weshalb der KSI-Pin aus- und eingeschalten werden muss, um den normalen Fahrbetrieb wieder aufnehmen zu können. Da jedoch ein ausschalten des KSI-Pins zum sofortigen Stillstand (Einleiten des Bremsvorgangs bis die Geschwindigkeit 0 ist) des Motorrades führt, kann der Wechsel des Antriebsmodus nur im Stillstand erfolgen. Die einzelnen Modi können beliebig konfiguriert werden und auch im Nachhinein mittels dem Curtis Integrated Toolkit bzw. VCL-Studio sehr schnell und leicht verändert oder angepasst werden. Vorerst wird es nur zwei unterschiedliche Antriebsmodi geben, da in diesem Fall nur ein Pin für die Selektierung benötigt wird (1 digitaler Eingang). Wenn in Zukunft jedoch weiterhin Interesse an weiteren unterschiedlichen Antriebsmodi besteht, können diese mit relativ wenig Aufwand hinzugefügt werden.

Hier eine Tabelle der einzelnen Parameter im ECO- und im Sportmodus:

Parameter	ECO-Modus	SPORT-Modus
Max Speed	7000 U/min	8000 U/min
Kp	30 %	40 %
Ki	30 %	40 %
Kd	15 %	10 %
Accel Rate	2 s	1 s
Accel Release Rate	1 s	0.4 s
Brake Rate	2 s	1 s
Brake Release Rate	1 s	0.4 s
Neutral Braking	15 %	10 %
Neutral Taper Speed	500 U/min	800 U/min
Forward Full Restraint Speed	800 U/min	500 U/min
Back Full Restraint Speed	800 U/min	500 U/min

Tabelle VI.3: Antriebsmodi

### Konventionelle Umsetzung

In konventionellen E-Motorrädern wird bei der Auswahl zwischen den Antriebsmodi ECO-Normal-Sport eine 7-Punkte Map hinterlegt, welche die Drehzahl abhängig vom Drehmoment beschreibt. Da dies für die schulische Umsetzung aber ebenfalls zu zeitintensiv wäre, wird dies wahrscheinlich bei späteren Optimierungen hinzugefügt.

### 3.3 Vehicle-Control-Language (VCL) Programmierung

#### 3.3.1 Grundfunktion

Die Vehicle-Control-Language ist ein eigener Abschnitt der Programmierung und von der Grundidee auch sehr unterschiedlich. Die normale Parameterprogrammierung vereinfacht die generelle Programmierung des Curtis Controllers zwar sehr, beschränkt die Anzahl der Möglichkeiten jedoch auf die Anzahl der Parameter und deren Konfigurationsmöglichkeiten. Die VCL-Programmiersprache bietet hierbei eine sehr gute Erweiterungsmöglichkeit und bei Kombination der beiden Programmierarten kann die Programmierung des Curtis Controllers optimiert und enorm erweitert werden. Die Vehicle-Control-Language ist eine Programmiersprache auf der Basis von C und das VCL-Studio sieht der Benutzeroberfläche eines normalen Programms mit integriertem Editor und Compiler, wie zum Beispiel Dev-C++, sehr ähnlich. Da in diesem Fall jedoch jeder beliebige Programmcode verfasst werden kann, bietet VCL nahezu endlose Möglichkeiten. Die wichtigsten Grundlagen, auf die man bei der Programmierung von VCL aufpassen muss, sind in den beigelegten Skripten „WIN VCL User’s Guide“, „VCL Programmer’s Guide“ und „VCL Common Functions“ im Anhang genauer erklärt. Das Wichtigste ist jedoch die Kenntnis darüber, dass mit dem VCL-Studio nicht nur die vorhandenen oder selbsterstellten VCL-Variablen verändert werden können. Die meisten Variablen, die in den Parametermenüs vorhanden sind, können ebenfalls eingelesen und beschrieben werden. Hier ist natürlich Vorsicht geboten, denn viele Parameter dürfen in keinem Fall während des Betriebs geändert werden. Bei wichtigen Variablen muss daher auch ein Aus- und Einschalten des KSI-Pins durchgeführt werden, um die Veränderung wirksam zu machen und Komplikationen zu vermeiden. Generell ist die Hauptaufgabe einer ausgefeilten VCL-Programmierung jedoch die Automatisierung des gesamten E-Motorrades. Zum Beispiel kann beim Auftritt eines Fehlers automatisch ein zugehöriger Fehlerbehebungsvorgang eingeleitet werden. Da wir in unserem Projekt jedoch nur ein sehr kleines Zeitfenster zur Verfügung haben, konnte das VCL-Programm nicht derartig ausgereift werden. In unserem Fall beschränkt sich die VCL-Programmierung fast ausschließlich auf die Kommunikation mit dem Raspberry PI, denn die Lösung über VCL-CAN schien uns einfacher und vor allem viel flexibler wie die vorgefertigte Programmierung „CANopen“ zu verwenden.

### 3.3.2 Kommunikation (CAN-Bus)

Die Kommunikation erfolgt grundsätzlich nur in eine Richtung, da dies nicht nur einfacher zu realisieren war, sondern für unsere Anwendungen auch völlig ausreicht. Der Curtis Controller sendet die ausgewählten Parameter immer in der selben Reihenfolge an den Raspberry PI, dafür wurden 3 unterschiedliche Datenpakete mit jeweils einem einzigartigen Identifier und 7 folgenden Parametern erstellt. Ein Parameter enthält immer 4 Hexadezimalzahlen, also insgesamt 2 Byte. Wenn ein Fehler am Motorcontroller erkannt wurde, wird nach dem dritten Datenpaket noch ein zusätzliches Fehler-Datenpaket eingeschoben. Wenn zum Beispiel zwei Fehler am Controller vorliegen, werden diese 2 Fehler nach dem Fehler-Identifier (FFFF) gesendet, danach werden die 5 weiteren freien Plätze des Fehler-Datenpakets mit dem Fehler-Identifier aufgefüllt. Nachdem das Fehler-Datenpaket gesendet wurde, werden die ersten 7 Fehler vom Datenspeicher gelöscht, liegt der Fehler jedoch weiterhin vor, kann dieser auch nicht gelöscht werden. Hier werden die Datenpaket mit den zugehörigen Parametern noch ausführlich dargestellt:

	Datenpaket 1	Datenpaket 2	Datenpaket 3
Identifier	FFFC	FFFD	FFFE
Parameter 1	Vehicle_Speed	Vehicle_Speed	Vehicle_Speed
Parameter 2	Current_RMS	Capacitor_Volts	Vehicle_Acceleration
Parameter 3	Controller_Temperature	BDI_Percentage	Vehicle_Odometer
Parameter 4	Motor_Temperature	Interlock	Time_to_Capture_Speed_1
Parameter 5	Motor_Power	Throttle_Command	Time_to_Capture_Distance_1
Parameter 6	Motor_Torque	Brake_Command	Braking_Distance_Captured
Parameter 7	Modulation_Depth	EMR_State	Distance_Since_Stop

Tabelle VI.4: Datenpakte Deklaration

Der gesamte VCL-Programmcode befindet sich im Anhang, auffällig ist hierbei unter anderem die enorm lange Zeitverzögerung, welche den selben Delay-Befehl 3330 mal beinhaltet. Die gewünschte CAN-Bus Übertragungsfrequenz beträgt 100ms, da jedoch der integrierte Logic-Controller eine sehr viel kürzere applikationsabhängige Taktfrequenz besitzt, erwies sich das Aussenden von Parametern in der selben Reihenfolge vorerst als sehr schwierig. Es wurden viele unterschiedliche Methoden versucht, jedoch erwies sich die Synchronisation der beiden Taktfrequenzen als die einfachste. Da ebenfalls sehr wenig hilfreiche Informationen im Internet zu finden waren, wurde die Realisierung mit einer besseren Methode vernachlässigt. Die Vereinfachung bzw. Verkürzung der langen Zeitverzögerung mit einer Schleife oder ähnlichem erwies sich ebenfalls als schwierig und fehleranfällig, wurde deshalb weggelassen.

### 3.3.3 Speed-Mode-Select

Die Speed-Mode-Select Programmierung gehört eigentlich auch zum Punkt Drehmomentsteuerung, dieses Programm wurde aber ebenfalls in der Vehicle-Control-Language realisiert. Die zugehörigen Parameter, welche ausschlaggebend für die Drehmomentsteuerung sind, wurden bereits genau erklärt. Grundsätzlich ist das VCL-Programm sehr einfach und kurz aufgebaut, die Parameter zur Drehmomentsteuerung werden abhängig von einem digitalen Eingang verändert. Der digitale Eingang wird mittels eines externen Schalters realisiert, dieser wurde auf der Lenkstange befestigt. Wenn der Schalter auf den ECO-Antriebsmodus geschalten wurde, werden die Parameter für den ECO-Antriebsmodus an den Drehmomentergler weitergegeben. Bei der Auswahl des Sport-Antriebsmodus werden die Parameter für den Sport-Antriebsmodus an den Regler weitergegeben.

Wie im folgenden Programmschnippel zu erkennen ist, ist der Aufbau dieses Programms äußerst einfach. Aufpassen muss man bei der Überschreibung der Parameter, da nicht die Werte mit den Einheiten, sondern die absoluten Werte verwendet werden müssen. Vor allem bei den Parametern des PID-Reglers sind unterschiedliche Absolutwerte vorgegeben, diese müssen auf den gewünschten Prozentwert skaliert werden.

```

;Speed-Mode-Select (Sport/ECO)
if(Switch_5 = 0) {                                     ;ECO-Antriebsmodus
    Max_Speed_TrqM = 7000                            ;7000rpm
    Kp_TrqM = 2458                                    ;30%
    Ki_TrqM = 300                                     ;30%
    Kd_TrqM = 1229                                    ;15%
    Accel_Rate_TrqM = 2000                            ;2s
    Accel_Release_Rate_TrqM = 1000                   ;1s
    Brake_Rate_TrqM = 2000                            ;2s
    Brake_Release_Rate_TrqM = 1000                   ;1s
    Neutral_Braking_TrqM = 4915                      ;15%
    Neutral_Taper_Speed_TrqM = 500                  ;500rpm
    Forward_Full_Restraint_Speed_TrqM = 800          ;800rpm
    Back_Full_Restraint_Speed_TrqM = 800             ;800rpm
}
else if(Switch_5 = 1) {                               ;Sport-Antriebsmodus
    Max_Speed_TrqM = 8000                            ;8000rpm
    Kp_TrqM = 3277                                    ;40%
    Ki_TrqM = 400                                     ;40%
    Kd_TrqM = 819                                     ;10%
    Accel_Rate_TrqM = 1000                            ;1s
    Accel_Release_Rate_TrqM = 400                    ;0.4s
    Brake_Rate_TrqM = 1000                            ;1s
    Brake_Release_Rate_TrqM = 400                    ;0.4s
    Neutral_Braking_TrqM = 3276                      ;10%
    Neutral_Taper_Speed_TrqM = 800                  ;800rpm
    Forward_Full_Restraint_Speed_TrqM = 500          ;500rpm
    Back_Full_Restraint_Speed_TrqM = 500             ;500rpm
}

```

Abbildung VI.12: ECO/Sport-Select Programmschnippel

## 4 Inbetriebnahme

### 4.1 Leonard-Versuchsaufbau

#### Grundidee:

Der Bau des Li-Ionen-Akkumulators hat sich leider sehr verzögert. Um effektiv an der Inbetriebnahme des Antriebssystems weiterarbeiten zu können, musste vorübergehend eine alternative Spannungsversorgung gefunden werden, welche für die Anforderungen der Motorsteuerung geeignet ist. Da der Motor und die Motorsteuerung jedoch eine bipolare Spannungsquelle benötigen, um ordnungsgemäß in Betrieb genommen werden zu können, erwies sich dies vorerst schwieriger als gedacht. Die Beschaffung eines bipolaren Netzteils erwies sich als zu kosten- und zeitintensiv, weshalb der Betreuungslehrer vorschlug, die bipolare Spannungsquelle mittels eines Leonard-Umformers zu realisieren.

#### Fazit:

Der erste Schritt bei der Inbetriebnahme des Motors ist der Testlauf (Commissionierung) zur Ausmessung und Einstellung der Motorparameter. Bei der Durchführung dieses Testlaufs stoppte die Motorsteuerung jedoch immer wieder nach kurzer Zeit. Auch viele weitere Versuche bei geänderten Testparametern oder zusätzlichen parallelgeschalteten Kondensatoren brachten keine weiteren Erkenntnisse. Aufgrund des schwankenden Spannungspegels während der gescheiterten Test-Durchläufe nahmen wir genauere Messungen mittels eines Oszilloskops vor, um mögliche Fehlerursachen herausfinden zu können. Bei der Untersuchung der Gleichspannungs-Speisung konnten wir feststellen, dass in einem Zeitbereich von circa 40ms ein unerwarteter Einschwingvorgang zu beobachten war, welcher einer negativen Sinus-Schwingung sehr ähnelte. Es trat zuerst eine negative Flanke in einem Zeitbereich von 12ms und mit einer Spannungsunterhöhung von circa 30V auf, dann folgte eine Spannungsüberhöhung mit etwa den selben Werten. Nach weiteren 16ms war der Schwingungsvorgang wieder auf die Eingangsspannung von 50V zurückgefallen. Aufgrund dessen konnten wir rückschließen, dass unsere Leonard-Spannungsquelle zu träge für die Motorsteuerung ist. Außerdem entstand aus den hohen Induktivitäten des Motors und den langen Leitungen kombiniert mit den großen Kondensatoren der Motorsteuerung eine Art Schwingkreis, welcher den Trägheitseffekt zusätzlich verstärkte.

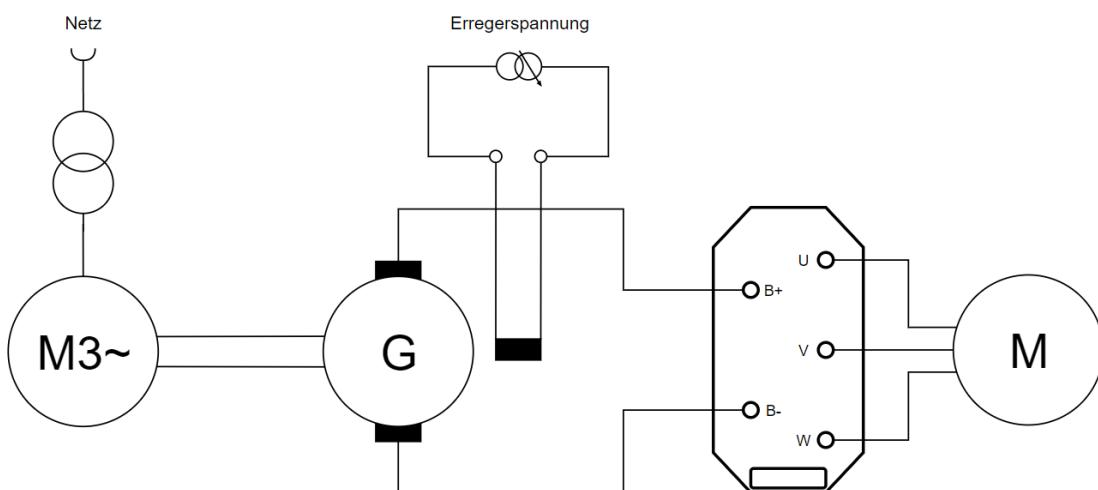


Abbildung VI.13: Leonardumformer Versuchsaufbau

## 4.2 Bleiakku-Versuchsaufbau

### Grundidee:

Der Leonard-Veruchsaufbau hat aufgrund der Trägheit der Spannungsquelle nicht funktioniert. Übrig blieb deshalb nur die Realisierung der Spannungsquelle mithilfe eines Ersatz-Akkumulators. Zufälligerweise konnten wir im Projektraum vier Bleiakkumulatoren ausfindig machen, welche wir nun seriell zu einer 48V-Spannungsquelle verschalten wollten. Die Bleiakkus waren zwar teils angeschlagen bzw. sehr tief entladen, mittels eines intelligenten Ladegeräts konnten aber 3 von 4 Akkus wieder erfolgreich aufgeladen werden. Mit einer von Zuhause mitgebrachten Autobatterie und zusätzlichen Starterkabeln konnte letztendlich aber die 48V-Spannungsquelle realisiert werden.

### Fazit:

Der Aufbau mit den Bleiakkumulatoren hat vorübergehend ganz gut funktioniert. Ein Problem stellten vorerst aber die großen Einschaltströme dar, welche bei der Schließung des Stromkreises zu Lichtbögen führten. Um Beschädigungen an den Kondensatoren zu verhindern, konnte dieses Problem jedoch durch Vorladen der Kondensatoren mithilfe eines 48V Netzgerätes behoben werden.

### Erste Inbetriebnahme:

Der Testlauf zur Einstellung der Motorparameter hat mithilfe der Bleiakkumulatoren beim ersten Versuch erfolgreich funktioniert. Der Motor konnte nach weiteren Konfigurationen letztendlich auch eine bestimmte Drehzahl abhängig von einem Spannungssignal anfahren.

Abbildung VI.14: Bleiakku

===== »»> 909ce2e364172eacda8c0ef77d3e60d4f81ec294

## Kapitel VII

# Akku und Ladekonzept

## Kapitel VIII

# Endergebnis

## **Anhang A**

# **Arbeitsnachweis**

**1 Zeitplan**

**2 Kosten**

## Anhang B

# Programmcode

## **Anhang C**

# **CAD-Zeichnungen**

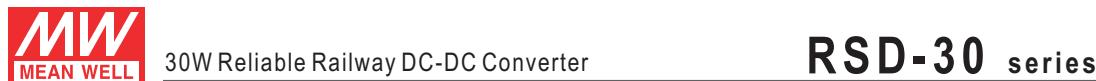
## Anhang D

# Schaltpläne

# Anhang E

# Datenblätter

## 0.1 Mean Well RSD-30H-5



### ■ Features

- Compliance to EN50155 and EN45545-2 railway standard
- Ultra compact and 1U low profile(25mm)
- 4:1 wide input range
- No minimum load required
- Protections: Short circuit / Overload / Over voltage / Input reverse polarity
- 4000VDC I/O isolation (reinforced isolation)
- Half encapsulated , cooling by free air convection
- -40~+70°C wide working temperature
- Built-in constant current limiting circuit
- LED indicator for power on
- 3 years warranty

### ■ Applications

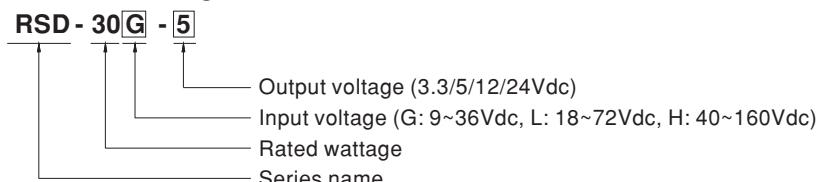
- Bus,tram,metro or railway system
- Wireless network
- Telecom or datacom system
- Highly vibrating, highly dusty, extremely low or high temperature harsh environment

### ■ Description

RSD-30 is a 30W enclosed type DC-DC reliable railway converter. This series is compliant with EN50155/IEC60571 railway standard, constituting three types of models with 4:1 wide but different input ranges 9~36V/18~72V/40~160V, suitable for railway and all kinds of transportation systems exploiting the frequently used standard input voltages such as 12V, 24V, 36V, 48V, 72V, 96V and 110V. Various output voltages, 3.3V, 5V, 12V and 24V are available for selection.

This series has the capability of working under -40~+70°C, low ripple and noise, supreme EMC characteristics, 4KVDC I/P-OP, low enclosure profile 25mm and an interior with semi-potted silicone. It does not only well fits the in-car systems or the facilities by rails for railway, trams and buses but also can be used in the harsh environment with high vibration, high dust, extremely low or high temperature, etc.

### ■ Model Encoding





30W Reliable Railway DC-DC Converter

**RSD-30 series****SPECIFICATION**

MODEL	RSD-30G-3.3	RSD-30G-5	RSD-30G-12	RSD-30G-24	RSD-30L-3.3	RSD-30L-5	RSD-30L-12	RSD-30L-24							
OUTPUT	<b>DC VOLTAGE</b>	3.3V	5V	12V	24V	3.3V	5V	12V	24V						
	<b>RATED CURRENT</b>	6A	6A	2.5A	1.25A	6A	6A	2.5A	1.25A						
	<b>CURRENT RANGE</b>	0 ~ 6A	0 ~ 6A	0 ~ 2.5A	0 ~ 1.25A	0 ~ 6A	0 ~ 6A	0 ~ 2.5A	0 ~ 1.25A						
	<b>RATED POWER</b>	19.8W	30W	30W	30W	19.8W	30W	30W	30W						
	<b>RIPPLE &amp; NOISE (max.) Note.2</b>	70mVp-p	70mVp-p	60mVp-p	50mVp-p	70mVp-p	70mVp-p	60mVp-p	50mVp-p						
	<b>VOLTAGE TOLERANCE Note.3</b>	±2.0%	±2.0%	±2.0%	±2.0%	±2.0%	±2.0%	±2.0%	±2.0%						
	<b>LINE REGULATION</b>	±0.5%	±0.5%	±0.3%	±0.2%	±0.5%	±0.5%	±0.3%	±0.2%						
	<b>LOAD REGULATION</b>	±0.5%	±0.5%	±0.3%	±0.2%	±0.5%	±0.5%	±0.3%	±0.2%						
	<b>SETUP, RISE TIME</b>	120ms, 85ms at full load													
<b>HOLD UP TIME (Typ.)</b>		G type comply with S1 level(3ms) @full load, S2 level(10ms) @80% load; L type comply with S2 level(10ms) @full load													
INPUT	<b>VOLTAGE RANGE CONTINUOUS</b>	9 ~ 36VDC				18 ~ 72VDC									
	<b>EFFICIENCY (Typ.)</b>	84%	85%	86.5%	89%	84%	86%	90%	91%						
	<b>DC CURRENT (Typ.)</b>	1.1A/24V				0.52A/48V									
	<b>INRUSH CURRENT (Typ.)</b>	20A/24VDC				20A/48VDC									
PROTECTION	<b>OVERLOAD</b>	105 ~ 135% rated output power													
		Protection type : Constant current limiting, recovers automatically after fault condition is removed													
	<b>OVER VOLTAGE</b>	3.8 ~ 4.5V	5.75 ~ 7V	13.8 ~ 16.2V	27.6 ~ 32.4V	3.8 ~ 4.5V	5.75 ~ 7V	13.8 ~ 16.2V	27.6 ~ 32.4V						
ENVIRONMENT	<b>WORKING TEMP.</b>	-40 ~ +55°C (no derating); +70°C @ 60% load by free air convection; +70°C (no derating with external base plate)													
	<b>WORKING HUMIDITY</b>	5 ~ 95% RH non-condensing													
	<b>STORAGE TEMP., HUMIDITY</b>	-40 ~ +85°C, 10 ~ 95% RH non-condensing													
	<b>TEMP. COEFFICIENT</b>	±0.03%/°C (0 ~ 50°C)													
	<b>VIBRATION</b>	10 ~ 500Hz, 5G 10min./1cycle, 60min. each along X, Y, Z axes; Mounting : compliance to IEC61373													
SAFETY & EMC (Note 4)	<b>SAFETY STANDARDS</b>	Meet IEC60950-1 (LVD)													
	<b>WITHSTAND VOLTAGE</b>	I/P-O/P:4KVDC I/P-FG:2.5KVDC O/P-FG:2.5KVDC													
	<b>ISOLATION RESISTANCE</b>	I/P-O/P, I/P-FG, O/P-FG:100M Ohms / 500VDC / 25°C / 70% RH													
	<b>EMC EMISSION</b>	<b>Parameter</b>	<b>Standard</b>	<b>Test Level / Note</b>											
		Conducted	EN55032	Class A											
		Radiated	EN55032	Class B											
		Harmonic Current	EN6100-3-2	Class A											
	<b>EMC IMMUNITY</b>	Voltage Flicker	EN6100-3-3	-----											
		<b>Parameter</b>	<b>Standard</b>	<b>Test Level / Note</b>											
		ESD	EN61000-4-2	Level 3, ±8KV air; Level 3, ±6KV contact											
		Radiated Field	EN61000-4-3	Level X											
		EFT / Burst	EN61000-4-4	Level 3, 2KV at power											
		Surge	EN61000-4-5	Level 3, 1KV Line-Line, Level 3, 2KV Line-Earth											
OTHERS	<b>Conducted</b>	EN61000-4-6	Level 3												
	<b>RAILWAY STANDARD</b>	Compliance to EN45545-2 for fire protection; Meet EN50155 / IEC60571 including IEC61373 for shock & vibration, EN50121-3-2 for EMC													
NOTE	<b>MTBF</b>	396.9K hrs min. MIL-HDBK-217F (25°C)													
	<b>DIMENSION</b>	113*60*25mm (L*W*H)													
	<b>PACKING</b>	0.25Kg; 56pcs/15Kg/0.83CUFT													
	1. All parameters NOT specially mentioned are measured at 24.48VDC input, rated load and 25°C of ambient temperature. 2. Ripple & noise are measured at 20MHz of bandwidth by using a 12" twisted pair-wire terminated with a 0.1uf & 47uf parallel capacitor. 3. Tolerance : includes set up tolerance, line regulation and load regulation. 4. The power supply is considered a component which will be installed into a final equipment. All the EMC tests are been executed by mounting the unit on a 360mm*360mm metal plate with 1mm of thickness. The final equipment must be re-confirmed that it still meets EMC directives. For guidance on how to perform these EMC tests, please refer to "EMI testing of component power supplies." (as available on <a href="http://www.meanwell.com">http://www.meanwell.com</a> ) 5. Strongly recommended that external output capacitance should not exceed 5000uF.														



30W Reliable Railway DC-DC Converter

**RSD-30 series****SPECIFICATION**

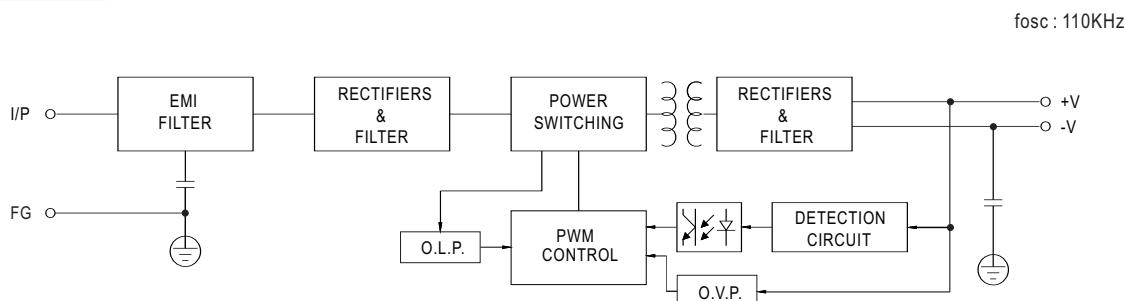
MODEL	RSD-30H-3.3	RSD-30H-5	RSD-30H-12	RSD-30H-24
OUTPUT	DC VOLTAGE	3.3V	5V	12V
	RATED CURRENT	6A	6A	2.5A
	CURRENT RANGE	0 ~ 6A	0 ~ 6A	0 ~ 2.5A
	RATED POWER	19.8W	30W	30W
	RIPPLE & NOISE (max.) Note.2	70mVp-p	70mVp-p	60mVp-p
	VOLTAGE TOLERANCE Note.3	±2.0%	±2.0%	±2.0%
	LINE REGULATION	±0.5%	±0.5%	±0.3%
	LOAD REGULATION	±0.5%	±0.5%	±0.3%
	SETUP, RISE TIME	120ms, 85ms at full load		
	HOLD UP TIME (Typ.)	H-type comply with S2 level(10ms) @ full load		
INPUT	VOLTAGE RANGE CONTINUOUS	40 ~ 160VDC		
	EFFICIENCY (Typ.)	87%	89%	89%
	DC CURRENT (Typ.)	0.23A/110V	0.35A/110V	
	INRUSH CURRENT (Typ.)	20A/110VDC		
PROTECTION	OVERLOAD	105 ~ 135% rated output power Protection type : Constant current limiting, recovers automatically after fault condition is removed		
		3.8 ~ 4.5V	5.75 ~ 7V	13.8 ~ 16.2V
	OVER VOLTAGE	27.6 ~ 32.4V Protection type : Shut down o/p voltage, re-power on to recover		
ENVIRONMENT	WORKING TEMP.	-40 ~ +55°C (no derating) ; +70°C @ 60% load by free air convection ; +70°C (no derating with external base plate)		
	WORKING HUMIDITY	5 ~ 95% RH non-condensing		
	STORAGE TEMP., HUMIDITY	-40 ~ +85°C, 10 ~ 95% RH non-condensing		
	TEMP. COEFFICIENT	±0.03%/°C (0 ~ 50°C)		
	VIBRATION	10 ~ 500Hz, 5G 10min./1cycle, 60min. each along X, Y, Z axes ; Mounting : compliance to IEC61373		
SAFETY & EMC (Note 4)	SAFETY STANDARDS	Meet IEC60950-1 (LVD)		
	WITHSTAND VOLTAGE	I/P-O/P:4KVDC I/P-FG:2.5KVDC O/P-FG:2.5KVDC		
	ISOLATION RESISTANCE	I/P-O/P, I/P-FG, O/P-FG:100M Ohms / 500VDC / 25°C / 70% RH		
	EMC EMISSION	Parameter	Standard	Test Level / Note
		Conducted	EN55032	Class A
		Radiated	EN55032	Class B
		Harmonic Current	EN6100-3-2	Class A
		Voltage Flicker	EN6100-3-3	-----
	EMC IMMUNITY	Parameter	Standard	Test Level / Note
		ESD	EN61000-4-2	Level 3, ±8KV air ; Level 3, ±6KV contact
		Radiated Field	EN61000-4-3	Level X
		EFT / Burst	EN61000-4-4	Level 3, 2KV at power Level 4, 2KV at signal
		Surge	EN61000-4-5	Level 3, 1KV Line-Line, Level 3, 2KV Line-Earth
		Conducted	EN61000-4-6	Level 3
	RAILWAY STANDARD	Compliance to EN45545-2 for fire protection ; Meet EN50155 / IEC60571 including IEC61373 for shock & vibration, EN50121-3-2 for EMC		
OTHERS	MTBF	396.9K hrs min. MIL-HDBK-217F (25°C)		
	DIMENSION	113*60*25mm (L*W*H)		
	PACKING	0.25Kg; 56pcs/15Kg/0.83CUFT		
NOTE	1. All parameters NOT specially mentioned are measured at 110VDC input, rated load and 25°C of ambient temperature. 2. Ripple & noise are measured at 20MHz of bandwidth by using a 12" twisted pair-wire terminated with a 0.1uf & 47uf parallel capacitor. 3. Tolerance : includes set up tolerance, line regulation and load regulation. 4. The power supply is considered a component which will be installed into a final equipment. All the EMC tests are been executed by mounting the unit on a 360mm*360mm metal plate with 1mm of thickness. The final equipment must be re-confirmed that it still meets EMC directives. For guidance on how to perform these EMC tests, please refer to "EMI testing of component power supplies." (as available on <a href="http://www.meanwell.com">http://www.meanwell.com</a> ) 5. Strongly recommended that external output capacitance should not exceed 5000uF.			



30W Reliable Railway DC-DC Converter

**RSD-30 series**

### ■ Block Diagram



### ■ Input Fuse

There is one fuse connected in series to the positive input line, which is used to protect against abnormal surge. Fuse specifications of each model are shown as below.

Type	Fuse Type	Reference and Rating
G	Time-Lag	CONQUE MST, 6.3A, 250V
L	Time-Lag	CONQUE MST, 3.15A, 250V
H	Time-Lag	CONQUE MST, 2A, 250V

### ■ Input Reverse Polarity Protection

There is a MOSFET connected in series to the negative input line. If the input polarity is connected reversely, the MOSFET opens and there will be no output to protect the unit.

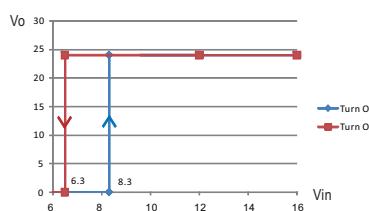
### ■ Input Range and Transient Ability

The series has a wide range input capability. With  $\pm 40\%$  of rated input voltage, it can withstand that for 1 second.

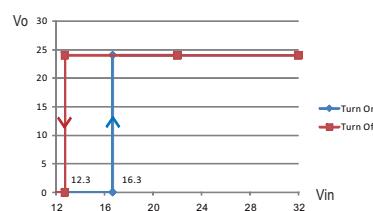
### ■ Input Under-Voltage Protection

If input voltage drops below  $V_{min}$ , the internal control IC shuts down and there is no output voltage. It recovers automatically when input voltage reaches above  $V_{min}$ , please refer to the curve below.

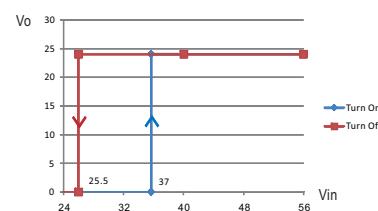
RSD-30G-24



RSD-30L-24



RSD-30H-24



### ■ Inrush Current

Inrush current is suppressed by a resistor during the initial start-up, and then the resistor is bypassed by a MOSFET to reduce power consumption after accomplishing the start-up.



30W Reliable Railway DC-DC Converter

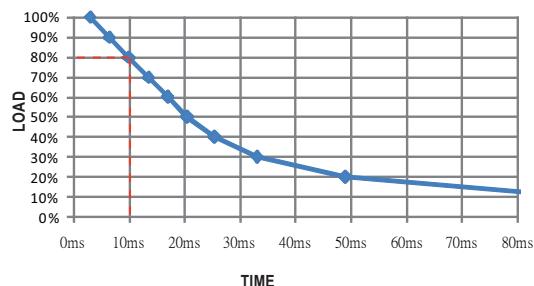
**RSD-30 series**

### ■ Hold-up Time

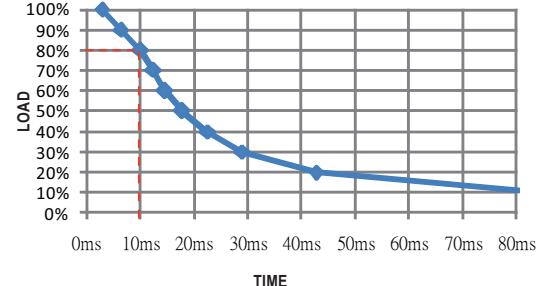
H type is in compliance with S2 level (10ms), while G and L types are in compliance with S1 level (3ms) at full load output condition.

To fulfil the requirements of S2 level (10ms), G types require de-rating their output load to 80%, please refer to the curve diagrams below.

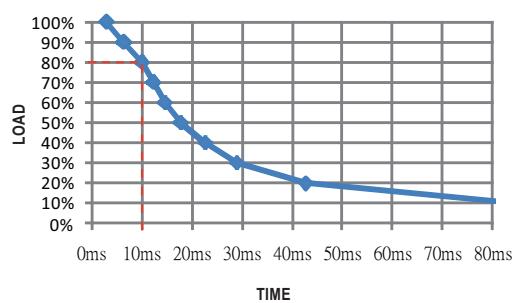
RSD-30G-3.3



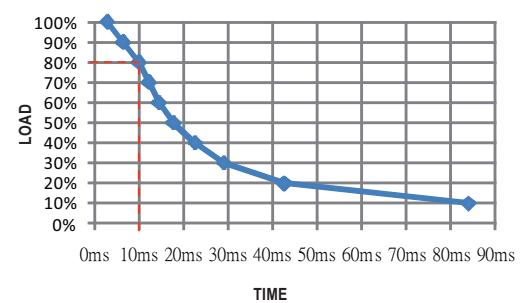
RSD-30G-5



RSD-30G-12



RSD-30G-24



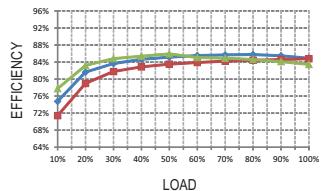
### ■ Output Voltage Adjustment

This function is optional, which the standard product does not have it. If you do need the function, please contact MW for details.

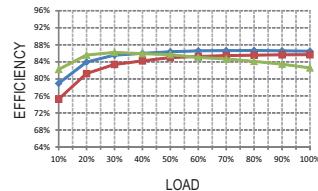
### ■ Efficiency vs Load & Vin Curve

The efficiency vs load & Vin curves of each model are shown as below.

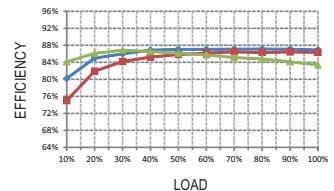
RSD-30G-3.3



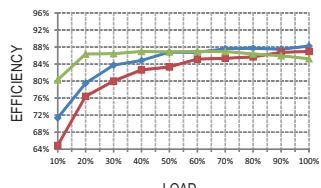
RSD-30G-5



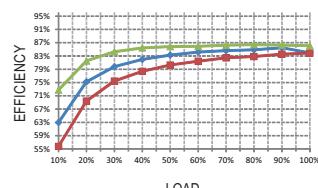
RSD-30G-12



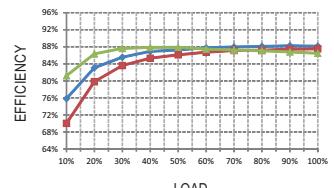
RSD-30G-24



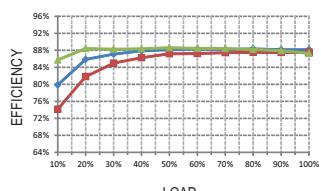
RSD-30L-3.3



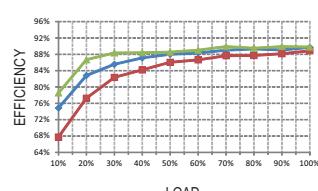
RSD-30L-5



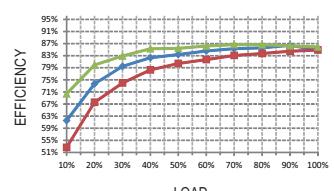
RSD-30L-12



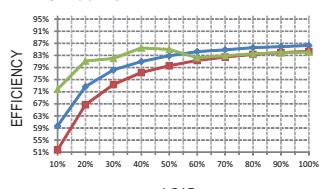
RSD-30L-24



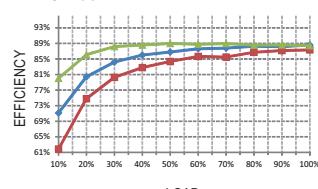
RSD-30H-3.3



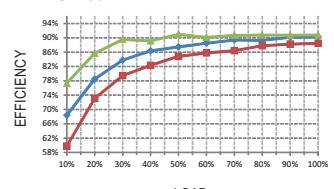
RSD-30H-5



RSD-30H-12



RSD-30H-24

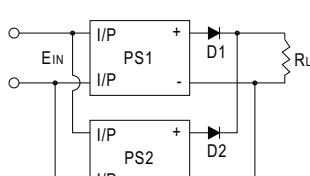


### ■ Parallel and Series Connection

#### A. Operation in Parallel

Since RSD-30 series don't have built-in parallel circuit, it can only use external circuits to achieve the redundant operation but not increase the current rating.

- Add a diode at the positive-output of each power supply (as shown as below), the current rating of the diode should be larger than the maximum output current rating and attached to a suitable heat sink. This is only for redundant use (increase the reliability of the system) and users have to check suitability of the circuit by themselves.

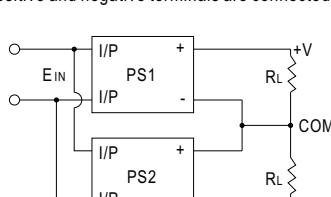


- When using S.P.S. in parallel connection, the leakage current will increase at the same time. This could pose as a shock hazard for the user. So please contact the supplier if you have this kind of application.

#### B. Operation in Series

RSD-30 can be operated in series. Here are the methods of doing it:

- Positive and negative terminals are connected as shown as below. According to the connection, you can get the positive and negative output voltages for your loads.

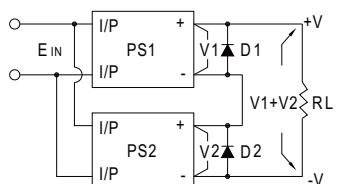




## 30W Reliable Railway DC-DC Converter

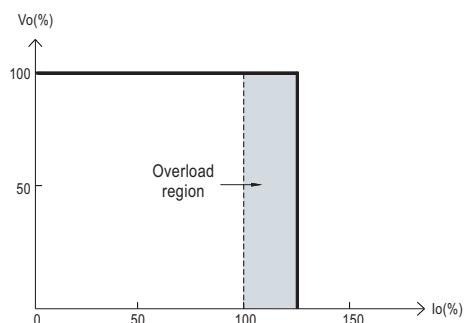
## RSD-30 series

2.Increase the output voltage (current does not change). Because RSD-30 series have no reverse blocking diode in the unit, you should add an external blocking diode to prevent the damage of every unit while starting up. The voltage rating of the external diode should be larger than  $V_1+V_2$  (as shown as below).



### ■ Overload Protection

If the output draw up to 105~135% of its output power rating, the converter will go into overload protection which is constant current mode. After the faulty condition is removed, it will recover automatically. Please refer to the diagram below for the detail operation characteristic. Please note that it's not suitable to operate within the overload region continuously, or it may cause to over temperature and reduce the life of the power supply unit or even damage it.



### ■ Over Voltage Protection

The converter shuts off to protect itself when the output voltage drawn exceeds 115~140% of its output rating. It must be repowered on to recover.

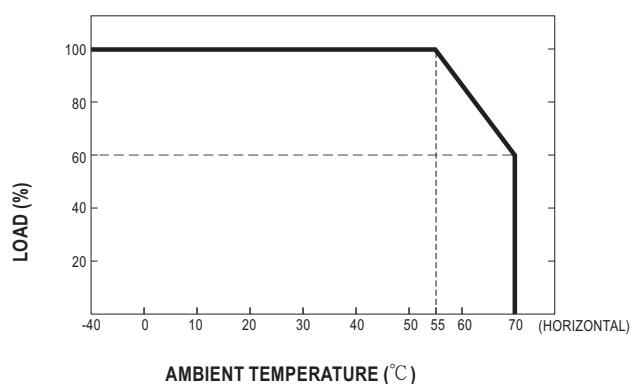
### ■ LED Indicator

Equipped with a built-in LED indicator, the converter provides an easy way for users to check its condition through the LED indicator.  
Green : normal operation; No signal: no power or failure.

### ■ Derating Curve

#### a.Single unit operation

If the unit has no iron plate mounted on its bottom, the maximum ambient temperature for the unit will be  $55^{\circ}\text{C}$  as operating under full load condition. It requires de-rating output current when ambient temperature is between  $55\text{--}70^{\circ}\text{C}$ , please refer to the de-rating curve as below.

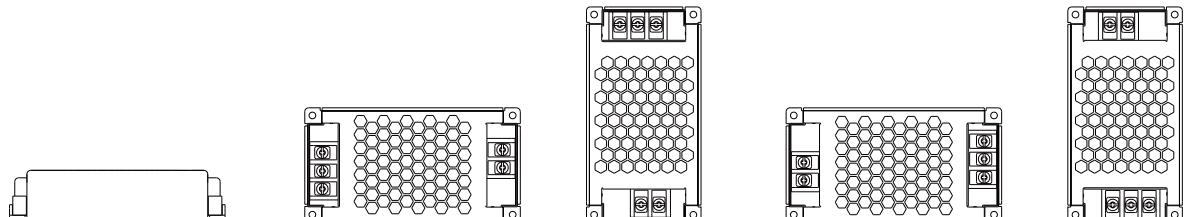




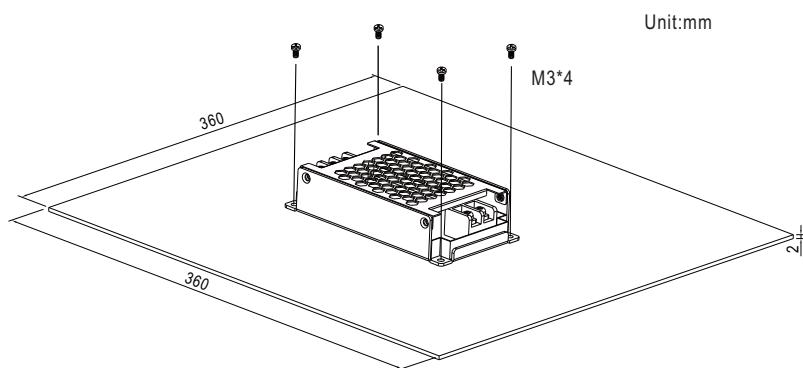
## 30W Reliable Railway DC-DC Converter

**RSD-30 series**

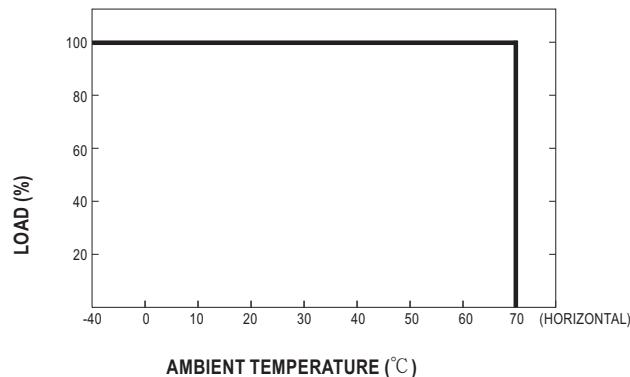
Suitable installation methods are shown as below. Since RSD-30 is a semi-potted model, its thermal performances for the following installation methods are similar and share the same derating curve.

**b. Operate with additional iron plate**

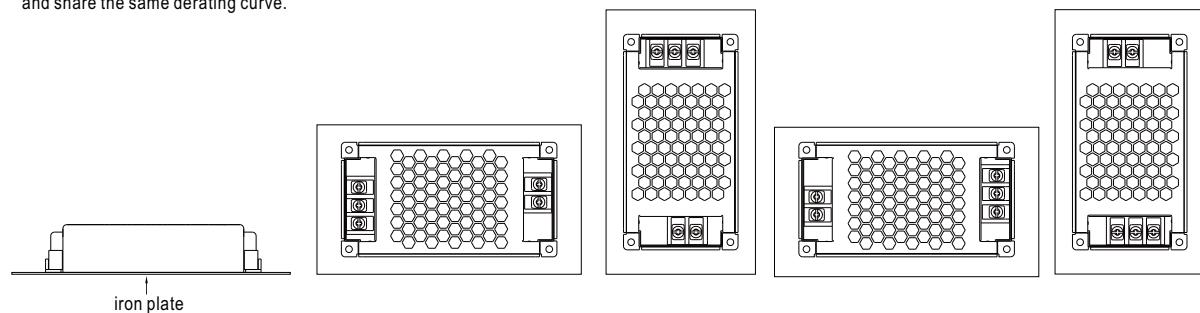
If it is necessary to fulfil the requirements of EN50155 TX level that operate the unit fully-loaded at 70°C, RSD-30 series must be installed onto an iron plate on the bottom. The size of the suggested iron plate is shown as below. In order for optimal thermal performance, the iron plate must have an even & smooth surface and RSD-30 series must be firmly mounted at the center of the iron plate.



The load vs ambient temperature curve is shown as below.



Suitable installation methods are shown as below. Since RSD-30 is a semi-potted model, its thermal performances for the following installation methods are similar and share the same derating curve.





30W Reliable Railway DC-DC Converter

**RSD-30 series**

### ■ Immunity to Environmental Conditions

Test method	Standard	Test conditions	Status
Cooling Test	EN 50155 section 12.2.3 (Column 2, Class TX) EN 60068-2-1	Temperature: -40°C Dwell Time: 2 hrs/cycle	No damage
Dry Heat Test	EN 50155 section 12.2.4 (Column 2, Class TX) EN 50155 section 12.2.4 (Column 3, Class TX & Column 4, Class TX) EN 60068-2-2	Temperature: 70°C / 85°C Duration: 6 hrs / 10min	PASS
Damp Heat Test, Cyclic	EN 50155 section 12.2.5 EN 60068-2-30	Temperature: 25°C ~ 55°C Humidity: 90% ~ 100% RH Duration: 48 hrs	PASS
Vibration Test	EN 50155 section 12.2.11 EN 61373	Temperature: 19°C Humidity: 65% Duration: 10 mins	PASS
Increased Vibration Test	EN 50155 section 12.2.11 EN 61373	Temperature: 19°C Humidity: 65% Duration: 5 hrs	PASS
Shock Test	EN 50155 section 12.2.11 EN 61373	Temperature: 21 ± 3°C Humidity: 65 ± 5% Duration: 30ms * 18	PASS
Low Temperature Storage Test	EN 50155 section 12.2.3 (Column 2, Class TX) EN 60068-2-1	Temperature: -40°C Dwell Time: 16 hrs	PASS
Salt Mist Test	EN 50155 section 12.2.10 (Class ST4)	Temperature: 35°C ± 2°C Duration: 96 hrs	PASS

### ■ EN45545-2 Fire Test Conditions

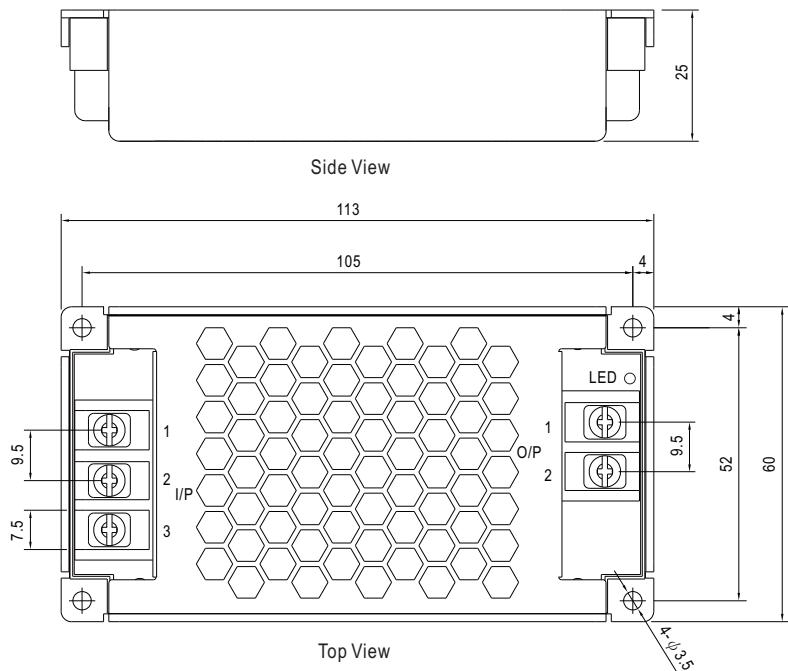
Test Items		Hazard Level			
Items		Standard	HL1	HL2	
R24	Oxygen index test	EN 45545-2:2013+A1:2015 EN ISO 4589-2:1996	PASS	PASS	PASS
R25	Glow-wire test	EN 45545-2:2013+A1:2015 EN 60695-2-11:2000	PASS	PASS	PASS
R26	Vertical flame test	EN 45545-2:2013+A1:2015 EN 60695-11:2003	PASS	PASS	PASS



30W Reliable Railway DC-DC Converter

**RSD-30 series****■ Mechanical Specification**

Case No.253A Unit:mm



Input Terminal Pin No. Assignment :

Pin No.	Assignment
1	DC INPUT V+
2	DC INPUT V-
3	FG $\pm$

Output Terminal Pin No. Assignment :

Pin No.	Assignment
1	DC OUTPUT -V
2	DC OUTPUT +V

**■ Installation Manual**Please refer to : <http://www.meanwell.com/manual.html>

## 0.2 Mean Well SD-350C-12



350W Single Output DC-DC Converter

**SD-350 series**

## ■ Features :

- 2:1 wide input range
- Protections: Short circuit / Overload / Over voltage / Over temperature
- 1500VAC I/O isolation
- Forced air cooling by built-in DC Fan
- 100% full load burn-in test
- 24V and 48V input voltage design refer to LVD
- 2 years warranty



EN62368-1

**CB** (for D type only)  
IEC62368-1

TPTC004

**SPECIFICATION**

MODEL	SD-350B				SD-350C				
OUTPUT	DC VOLTAGE	5V	12V	24V	48V	5V	12V	24V	48V
	RATED CURRENT	57A	27.5A	14.6A	7.3A	60A	27.5A	14.6A	7.3A
	CURRENT RANGE	0 ~ 57A	0 ~ 27.5A	0 ~ 14.6A	0 ~ 7.3A	0 ~ 60A	0 ~ 27.5A	0 ~ 14.6A	0 ~ 7.3A
	RATED POWER	285W	330W	350.4W	350.4W	300W	330W	350.4W	350.4W
	RISSLE & NOISE (max.) Note.2	100mVp-p	120mVp-p	150mVp-p	200mVp-p	100mVp-p	120mVp-p	150mVp-p	200mVp-p
	VOLTAGE ADJ. RANGE	4.5 ~ 5.5VDC	11 ~ 16VDC	23 ~ 30VDC	43 ~ 53VDC	4.5 ~ 5.5VDC	11 ~ 16VDC	23 ~ 30VDC	43 ~ 53VDC
	VOLTAGE TOLERANCE Note.3	±2.0%	±1.0%	±1.0%	±1.0%	±2.0%	±1.0%	±1.0%	±1.0%
	LINER REGULATION	±0.5%	±0.3%	±0.2%	±0.2%	±0.5%	±0.3%	±0.2%	±0.2%
INPUT	LOAD REGULATION	±1.0%	±1.0%	±1.0%	±1.0%	±1.0%	±1.0%	±1.0%	±1.0%
	SETUP, RISE TIME	300ms, 50ms at full load							
PROTECTION	VOLTAGE RANGE	B:19 ~ 36VDC	C:36 ~ 72VDC	D:72 ~ 144VDC					
	EFFICIENCY (Typ.)	74%	80%	80%	84%	76%	81%	81%	82%
	DC CURRENT (Typ.)	14.4A/24V	16A/24V	17.6A/24V	17.6A/24V	7.6A/48V	8.8A/48V	9.0A/48V	9.0A/48V
	INRUSH CURRENT (Typ.)	C:45A/48VDC D:45A/96VDC							
ENVIRONMENT	OVERLOAD	105 ~ 135% rated output power Protection type : Shut down o/p voltage, re-power on to recover							
		5.75 ~ 6.75V	16.8 ~ 20V	31.5 ~ 37.5V	53 ~ 65V	5.75 ~ 6.75V	16.8 ~ 20V	31.5 ~ 37.5V	53 ~ 65V
	OVER VOLTAGE	Protection type : Shut down o/p voltage, re-power on to recover							
SAFETY & EMC (Note 4)	OVER TEMPERATURE	Shut down o/p voltage, recovers automatically after temperature goes down							
	WORKING TEMP.	-20 ~ +60°C (Refer to "Derating Curve")							
	WORKING HUMIDITY	20 ~ 90% RH non-condensing							
	STORAGE TEMP., HUMIDITY	-40 ~ +85°C, 10 ~ 95% RH							
	TEMP. COEFFICIENT	±0.03%/°C (0 ~ 50°C)							
OTHERS	VIBRATION	10 ~ 500Hz, 2G 10min./1cycle, 60min. each along X, Y, Z axes							
	SAFETY STANDARDS	IEC62368-1 CB approved by TUV (for D type only), EAC TP TC 004 approved							
	WITHSTAND VOLTAGE	I/P-O/P:1.5KVAC I/P-FG:2KVAC O/P-FG:0.5KVAC							
	ISOLATION RESISTANCE	I/P-O/P, I/P-FG, O/P-FG:100M Ohms / 500VDC / 25°C/ 70% RH							
NOTE	EMC EMISSION	Compliance to EN55032 (CISPR32) Class B, EAC TP TC 020							
	EMC IMMUNITY	Compliance to EN61000-4-2,3,4,6,8, light industry level, criteria A, EAC TP TC 020							
	MTBF	209.4 hrs min. MIL-HDBK-217F (25°C)							
DIMENSION	215*115*50mm (L*W*H)								
	PACKING	1.1Kg; 12pcs/14.4Kg/0.92CUFT							
1. All parameters NOT specially mentioned are measured at 24,48,96VDC input, rated load and 25°C of ambient temperature. 2. Ripple & noise are measured at 20MHz of bandwidth by using a 12" twisted pair-wire terminated with a 0.1uf & 47uf parallel capacitor. 3. Tolerance : includes set up tolerance, line regulation and load regulation. 4. The power supply is considered a component which will be installed into a final equipment. All the EMC tests are been executed by mounting the unit on a 360mm*360mm metal plate with 1mm of thickness. The final equipment must be re-confirmed that it still meets EMC directives. For guidance on how to perform these EMC tests, please refer to "EMI testing of component power supplies." (as available on <a href="http://www.meanwell.com">http://www.meanwell.com</a> ) 5. The ambient temperature derating of 3.5°C/1000m with fanless models and of 5°C/1000m with fan models for operating altitude higher than 2000m(6500ft)									

File Name:SD-350-SPEC 2020-03-37



350W Single Output DC-DC Converter

SD-350 series



## ■ Features :

- 2:1 wide input range
- Protections: Short circuit / Overload / Over voltage / Over temperature
- 1500VAC I/O isolation
- Forced air cooling by built-in DC Fan
- 100% full load burn-in test
- 24V(B) and 48V(C) input voltage design refer to LVD
- 2 years warranty

CB (for D type only)  
IEC62368-1**SPECIFICATION**

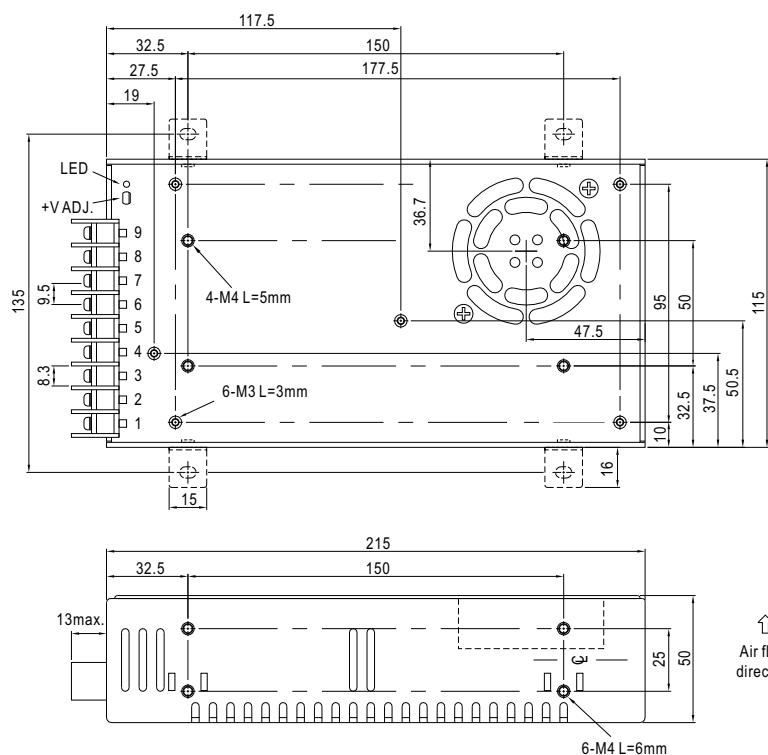
MODEL	SD-350D			
OUTPUT	DC VOLTAGE	5V	12V	24V
	RATED CURRENT	60A	29.2A	14.6A
	CURRENT RANGE	0 ~ 60A	0 ~ 29.2A	0 ~ 14.6A
	RATED POWER	300W	350.4W	350.4W
	RIPPLE & NOISE (max.) Note.2	100mVp-p	120mVp-p	150mVp-p
	VOLTAGE ADJ. RANGE	4.5 ~ 5.5VDC	11 ~ 16VDC	23 ~ 30VDC
	VOLTAGE TOLERANCE Note.3	±2.0%	±1.0%	±1.0%
	LINE REGULATION	±0.5%	±0.3%	±0.2%
	LOAD REGULATION	±1.0%	±1.0%	±1.0%
	SETUP, RISE TIME	300ms, 50ms at full load		
INPUT	VOLTAGE RANGE	B:19 ~ 36VDC	C:36 ~ 72VDC	D:72 ~ 144VDC
	EFFICIENCY (Typ.)	78%	83%	87%
	DC CURRENT (Typ.)	6A/96V	6A/96V	6A/96V
	INRUSH CURRENT (Typ.)	C:45A/48VDC	D:45A/96VDC	
PROTECTION	OVERLOAD	105 ~ 135% rated output power Protection type : Shut down o/p voltage, re-power on to recover		
	OVER VOLTAGE	5.75 ~ 6.75V	16.8 ~ 20V	31.5 ~ 37.5V
	OVER TEMPERATURE	53 ~ 65V Protection type : Shut down o/p voltage, re-power on to recover		
		Shut down o/p voltage, recovers automatically after temperature goes down		
ENVIRONMENT	WORKING TEMP.	-20 ~ +60°C (Refer to "Derating Curve")		
	WORKING HUMIDITY	20 ~ 90% RH non-condensing		
	STORAGE TEMP., HUMIDITY	-40 ~ +85°C, 10 ~ 95% RH		
	TEMP. COEFFICIENT	±0.03%/°C (0 ~ 50°C)		
	VIBRATION	10 ~ 500Hz, 2G 10min./cycle, 60min. each along X, Y, Z axes		
SAFETY & EMC (Note 4)	SAFETY STANDARDS	IEC62368-1 CB approved by TUV (for D type only), EAC TP TC 004 approved		
	WITHSTAND VOLTAGE	I/P-O/P:1.5KVAC	I/P-FG:2KVAC	O/P-FG:0.5KVAC
	ISOLATION RESISTANCE	I/P-O/P, I/P-FG, O/P-FG:100M Ohms / 500VDC / 25°C / 70% RH		
	EMC EMISSION	Compliance to EN55022 (CISPR22) Class B, EAC TP TC 020		
	EMC IMMUNITY	Compliance to EN61000-4-2,3,4,6,8, light industry level, criteria A, EAC TP TC 020		
OTHERS	MTBF	209.4K hrs min. MIL-HDBK-217F (25°C)		
	DIMENSION	215*115*50mm (L*W*H)		
	PACKING	1.1Kg; 12pcs/14.4Kg/0.92CUFT		
NOTE	1. All parameters NOT specially mentioned are measured at 24,48,96VDC input, rated load and 25°C of ambient temperature. 2. Ripple & noise are measured at 20MHz of bandwidth by using a 12" twisted pair-wire terminated with a 0.1uf & 47uf parallel capacitor. 3. Tolerance : includes set up tolerance, line regulation and load regulation. 4. The power supply is considered a component which will be installed into a final equipment. All the EMC tests are been executed by mounting the unit on a 360mm*360mm metal plate with 1mm of thickness. The final equipment must be re-confirmed that it still meets EMC directives. For guidance on how to perform these EMC tests, please refer to "EMI testing of component power supplies." (as available on <a href="http://www.meanwell.com">http://www.meanwell.com</a> ) 5. The ambient temperature derating of 3.5°C/1000m with fanless models and of 5°C/1000m with fan models for operating altitude higher than 2000m(6500ft).			



350W Single Output DC-DC Converter

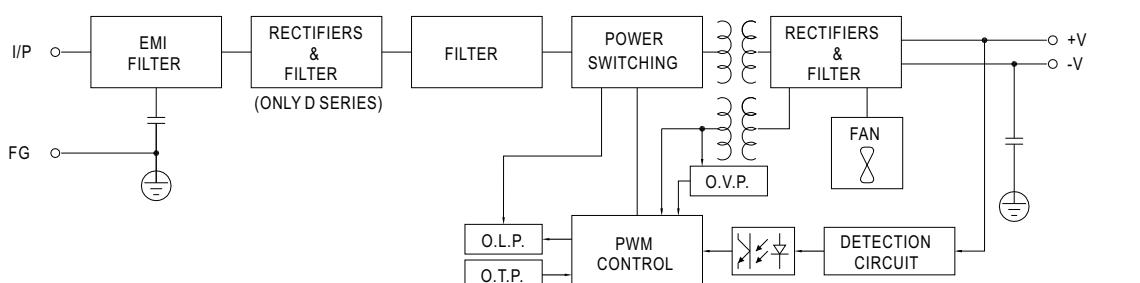
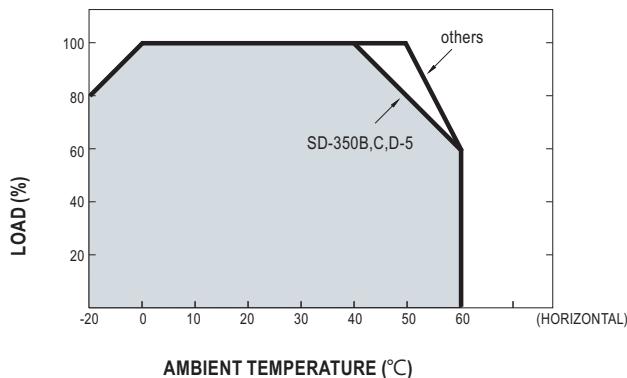
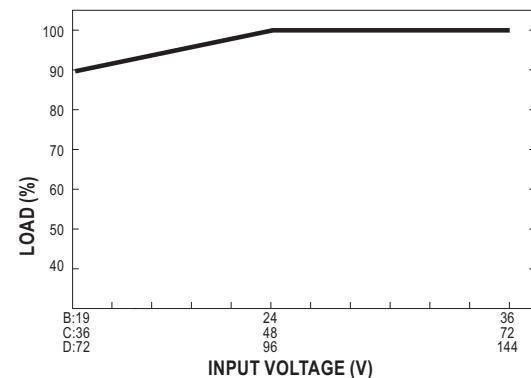
**SD-350 series****Mechanical Specification**

Case No. 912B Unit:mm



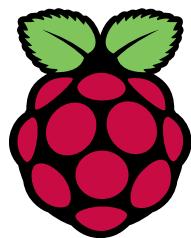
## Terminal Pin No. Assignment :

Pin No.	Assignment	Pin No.	Assignment
1	DC INPUT V+	4,5,6	DC OUTPUT V-
2	DC INPUT V-	7,8,9	DC OUTPUT V+
3	FG $\pm$		

**Block Diagram****Derating Curve****Static Characteristics**

### 0.3 Raspberry Pi 4 Moddel B

## DATASHEET



## Raspberry Pi 4 Model B

**Release 1**

**June 2019**

**Copyright 2019 Raspberry Pi (Trading) Ltd. All rights reserved.**



Raspberry Pi 4 Model B Datasheet  
Copyright Raspberry Pi (Trading) Ltd. 2019

---

Table 1: Release History

<b>Release</b>	<b>Date</b>	<b>Description</b>
1	21/06/2019	First release

The latest release of this document can be found at <https://www.raspberrypi.org>



## Contents

<b>1</b>	<b>Introduction</b>	<b>5</b>
<b>2</b>	<b>Features</b>	<b>6</b>
2.1	Hardware . . . . .	6
2.2	Interfaces . . . . .	6
2.3	Software . . . . .	7
<b>3</b>	<b>Mechanical Specification</b>	<b>7</b>
<b>4</b>	<b>Electrical Specification</b>	<b>7</b>
4.1	Power Requirements . . . . .	9
<b>5</b>	<b>Peripherals</b>	<b>9</b>
5.1	GPIO Interface . . . . .	9
5.1.1	GPIO Pin Assignments . . . . .	9
5.1.2	GPIO Alternate Functions . . . . .	10
5.1.3	Display Parallel Interface (DPI) . . . . .	11
5.1.4	SD/SDIO Interface . . . . .	11
5.2	Camera and Display Interfaces . . . . .	11
5.3	USB . . . . .	11
5.4	HDMI . . . . .	11
5.5	Audio and Composite (TV Out) . . . . .	11
5.6	Temperature Range and Thermals . . . . .	11
<b>6</b>	<b>Availability</b>	<b>12</b>
<b>7</b>	<b>Support</b>	<b>12</b>



## List of Figures

1	Mechanical Dimensions . . . . .	7
2	Digital IO Characteristics . . . . .	8
3	GPIO Connector Pinout . . . . .	9



## List of Tables

1	Release History . . . . .	1
2	Absolute Maximum Ratings . . . . .	8
3	DC Characteristics . . . . .	8
4	Digital I/O Pin AC Characteristics . . . . .	8
5	Raspberry Pi 4 GPIO Alternate Functions . . . . .	10



## 1 Introduction

The Raspberry Pi 4 Model B (Pi4B) is the first of a new generation of Raspberry Pi computers supporting more RAM and with significantly enhanced CPU, GPU and I/O performance; all within a similar form factor, power envelope and cost as the previous generation Raspberry Pi 3B+.

The Pi4B is available with either 1, 2 and 4 Gigabytes of LPDDR4 SDRAM.



## 2 Features

### 2.1 Hardware

- Quad core 64-bit ARM-Cortex A72 running at 1.5GHz
- 1, 2 and 4 Gigabyte LPDDR4 RAM options
- H.265 (HEVC) hardware decode (up to 4Kp60)
- H.264 hardware decode (up to 1080p60)
- VideoCore VI 3D Graphics
- Supports dual HDMI display output up to 4Kp60

### 2.2 Interfaces

- 802.11 b/g/n/ac Wireless LAN
- Bluetooth 5.0 with BLE
- 1x SD Card
- 2x micro-HDMI ports supporting dual displays up to 4Kp60 resolution
- 2x USB2 ports
- 2x USB3 ports
- 1x Gigabit Ethernet port (supports PoE with add-on PoE HAT)
- 1x Raspberry Pi camera port (2-lane MIPI CSI)
- 1x Raspberry Pi display port (2-lane MIPI DSI)
- 28x user GPIO supporting various interface options:
  - Up to 6x UART
  - Up to 6x I2C
  - Up to 5x SPI
  - 1x SDIO interface
  - 1x DPI (Parallel RGB Display)
  - 1x PCM
  - Up to 2x PWM channels
  - Up to 3x GPCLK outputs



### 2.3 Software

- ARMv8 Instruction Set
  - Mature Linux software stack
  - Actively developed and maintained
    - Recent Linux kernel support
    - Many drivers upstreamed
    - Stable and well supported userland
    - Availability of GPU functions using standard APIs

### 3 Mechanical Specification

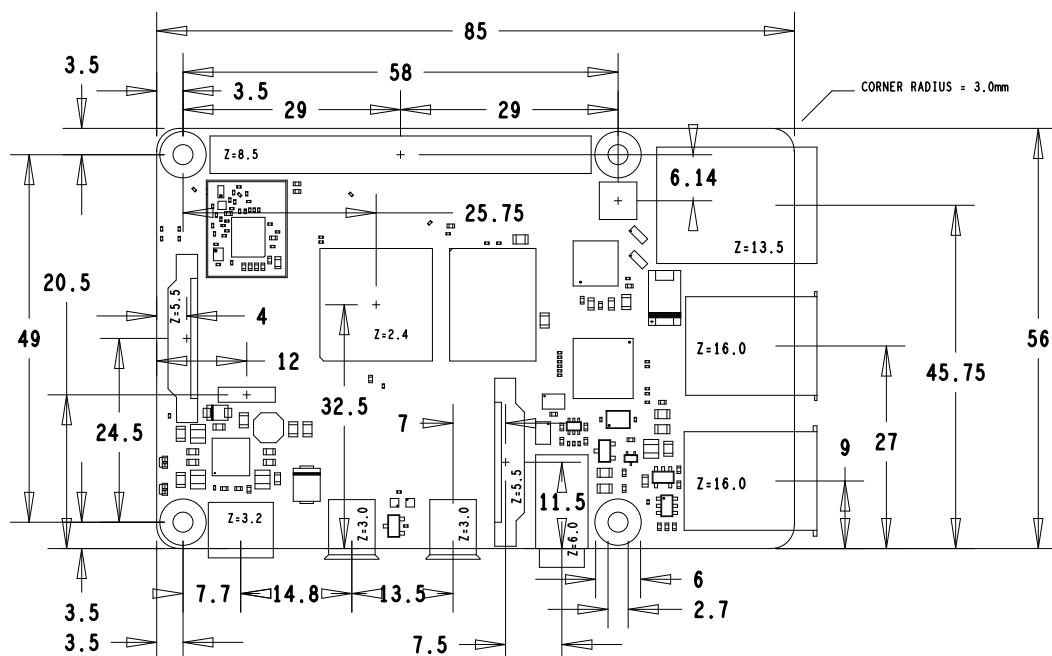


Figure 1: Mechanical Dimensions

## 4 Electrical Specification

**Caution!** Stresses above those listed in Table 2 may cause permanent damage to the device. This is a stress rating only; functional operation of the device under these or any other conditions above those listed in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.


 Raspberry Pi 4 Model B Datasheet  
 Copyright Raspberry Pi (Trading) Ltd. 2019

Symbol	Parameter	Minimum	Maximum	Unit
VIN	5V Input Voltage	-0.5	6.0	V

Table 2: Absolute Maximum Ratings

Please note that VDD\_IO is the GPIO bank voltage which is tied to the on-board 3.3V supply rail.

Symbol	Parameter	Conditions	Minimum	Typical	Maximum	Unit
$V_{IL}$	Input low voltage <sup>a</sup>	$VDD\_IO = 3.3V$	-	-	TBD	V
$V_{IH}$	Input high voltage <sup>a</sup>	$VDD\_IO = 3.3V$	TBD	-	-	V
$I_{IL}$	Input leakage current	$TA = +85^{\circ}C$	-	-	TBD	$\mu A$
$C_{IN}$	Input capacitance	-	-	TBD	-	pF
$V_{OL}$	Output low voltage <sup>b</sup>	$VDD\_IO = 3.3V, IOL = -2mA$	-	-	TBD	V
$V_{OH}$	Output high voltage <sup>b</sup>	$VDD\_IO = 3.3V, IOH = 2mA$	TBD	-	-	V
$I_{OL}$	Output low current <sup>c</sup>	$VDD\_IO = 3.3V, VO = 0.4V$	TBD	-	-	mA
$I_{OH}$	Output high current <sup>c</sup>	$VDD\_IO = 3.3V, VO = 2.3V$	TBD	-	-	mA
$R_{PU}$	Pullup resistor	-	TBD	-	TBD	k $\Omega$
$R_{PD}$	Pulldown resistor	-	TBD	-	TBD	k $\Omega$

<sup>a</sup> Hysteresis enabled

<sup>b</sup> Default drive strength (8mA)

<sup>c</sup> Maximum drive strength (16mA)

Table 3: DC Characteristics

Pin Name	Symbol	Parameter	Minimum	Typical	Maximum	Unit
Digital outputs	$t_{rise}$	10-90% rise time <sup>a</sup>	-	TBD	-	ns
Digital outputs	$t_{fall}$	90-10% fall time <sup>a</sup>	-	TBD	-	ns

<sup>a</sup> Default drive strength,  $CL = 5pF, VDD\_IO = 3.3V$

Table 4: Digital I/O Pin AC Characteristics



Figure 2: Digital IO Characteristics



## 4.1 Power Requirements

The Pi4B requires a good quality USB-C power supply capable of delivering 5V at 3A. If attached downstream USB devices consume less than 500mA, a 5V, 2.5A supply may be used.

## 5 Peripherals

### 5.1 GPIO Interface

The Pi4B makes 28 BCM2711 GPIOs available via a standard Raspberry Pi 40-pin header. This header is backwards compatible with all previous Raspberry Pi boards with a 40-way header.

#### 5.1.1 GPIO Pin Assignments

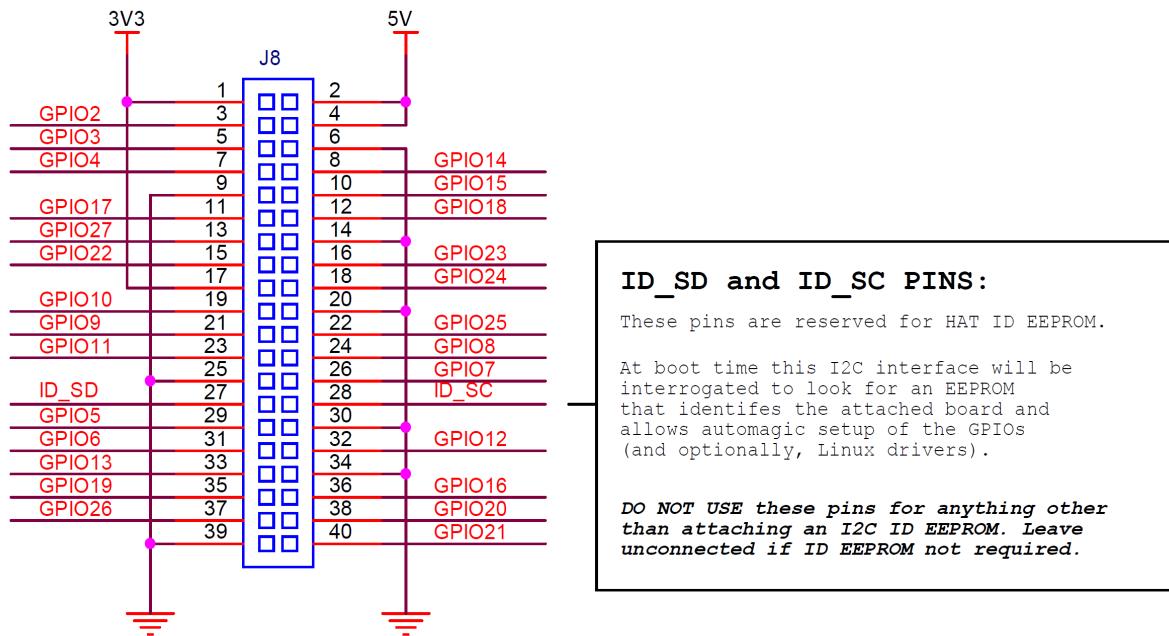


Figure 3: GPIO Connector Pinout

As well as being able to be used as straightforward software controlled input and output (with programmable pulls), GPIO pins can be switched (multiplexed) into various other modes backed by dedicated peripheral blocks such as I2C, UART and SPI.

In addition to the standard peripheral options found on legacy Pis, extra I2C, UART and SPI peripherals have been added to the BCM2711 chip and are available as further mux options on the Pi4. This gives users much more flexibility when attaching add-on hardware as compared to older models.



### 5.1.2 GPIO Alternate Functions

<b>GPIO</b>	<b>Pull</b>	<b>Default</b>					
		<b>ALT0</b>	<b>ALT1</b>	<b>ALT2</b>	<b>ALT3</b>	<b>ALT4</b>	<b>ALT5</b>
0	High	SDA0	SA5	PCLK	SPI3_CE0_N	TXD2	SDA6
1	High	SCL0	SA4	DE	SPI3_MISO	RXD2	SCL6
2	High	SDA1	SA3	LCD_VSYNC	SPI3_MOSI	CTS2	SDA3
3	High	SCL1	SA2	LCD_HSYNC	SPI3_SCLK	RTS2	SCL3
4	High	GPCLK0	SA1	DPI_D0	SPI4_CE0_N	TXD3	SDA3
5	High	GPCLK1	SA0	DPI_D1	SPI4_MISO	RXD3	SCL3
6	High	GPCLK2	SOE_N	DPI_D2	SPI4_MOSI	CTS3	SDA4
7	High	SPI0_CE1_N	SWE_N	DPI_D3	SPI4_SCLK	RTS3	SCL4
8	High	SPI0_CE0_N	SD0	DPI_D4	-	TXD4	SDA4
9	Low	SPI0_MISO	SD1	DPI_D5	-	RXD4	SCL4
10	Low	SPI0_MOSI	SD2	DPI_D6	-	CTS4	SDA5
11	Low	SPI0_SCLK	SD3	DPI_D7	-	RTS4	SCL5
12	Low	PWM0	SD4	DPI_D8	SPI5_CE0_N	TXD5	SDA5
13	Low	PWM1	SD5	DPI_D9	SPI5_MISO	RXD5	SCL5
14	Low	TXD0	SD6	DPI_D10	SPI5_MOSI	CTS5	TXD1
15	Low	RXD0	SD7	DPI_D11	SPI5_SCLK	RTS5	RXD1
16	Low	FL0	SD8	DPI_D12	CTS0	SPI1_CE2_N	CTS1
17	Low	FL1	SD9	DPI_D13	RTS0	SPI1_CE1_N	RTS1
18	Low	PCM_CLK	SD10	DPI_D14	SPI6_CE0_N	SPI1_CE0_N	PWM0
19	Low	PCM_FS	SD11	DPI_D15	SPI6_MISO	SPI1_MISO	PWM1
20	Low	PCM_DIN	SD12	DPI_D16	SPI6_MOSI	SPI1_MOSI	GPCLK0
21	Low	PCM_DOUT	SD13	DPI_D17	SPI6_SCLK	SPI1_SCLK	GPCLK1
22	Low	SD0_CLK	SD14	DPI_D18	SD1_CLK	ARM_TRST	SDA6
23	Low	SD0_CMD	SD15	DPI_D19	SD1_CMD	ARM_RTCK	SCL6
24	Low	SD0_DAT0	SD16	DPI_D20	SD1_DAT0	ARM_TDO	SPI3_CE1_N
25	Low	SD0_DAT1	SD17	DPI_D21	SD1_DAT1	ARM_TCK	SPI4_CE1_N
26	Low	SD0_DAT2	TE0	DPI_D22	SD1_DAT2	ARM_TDI	SPI5_CE1_N
27	Low	SD0_DAT3	TE1	DPI_D23	SD1_DAT3	ARM_TMS	SPI6_CE1_N

Table 5: Raspberry Pi 4 GPIO Alternate Functions

Table 5 details the default pin pull state and available alternate GPIO functions. Most of these alternate peripheral functions are described in detail in the BCM2711 Peripherals Specification document which can be downloaded from the hardware documentation section of the website.



### 5.1.3 Display Parallel Interface (DPI)

A standard parallel RGB (DPI) interface is available the GPIOs. This up-to-24-bit parallel interface can support a secondary display.

### 5.1.4 SD/SDIO Interface

The Pi4B has a dedicated SD card socket which supports 1.8V, DDR50 mode (at a peak bandwidth of 50 Megabytes / sec). In addition, a legacy SDIO interface is available on the GPIO pins.

## 5.2 Camera and Display Interfaces

The Pi4B has 1x Raspberry Pi 2-lane MIPI CSI Camera and 1x Raspberry Pi 2-lane MIPI DSI Display connector. These connectors are backwards compatible with legacy Raspberry Pi boards, and support all of the available Raspberry Pi camera and display peripherals.

## 5.3 USB

The Pi4B has 2x USB2 and 2x USB3 type-A sockets. Downstream USB current is limited to approximately 1.1A in aggregate over the four sockets.

## 5.4 HDMI

The Pi4B has 2x micro-HDMI ports, both of which support CEC and HDMI 2.0 with resolutions up to 4Kp60.

## 5.5 Audio and Composite (TV Out)

The Pi4B supports near-CD-quality analogue audio output and composite TV-output via a 4-ring TRS 'A/V' jack.

The analog audio output can drive 32 Ohm headphones directly.

## 5.6 Temperature Range and Thermals

The recommended ambient operating temperature range is 0 to 50 degrees Celcius.

To reduce thermal output when idling or under light load, the Pi4B reduces the CPU clock speed and voltage. During heavier load the speed and voltage (and hence thermal output) are increased. The internal governor will throttle back both the CPU speed and voltage to make sure the CPU temperature never exceeds 85 degrees C.

The Pi4B will operate perfectly well without any extra cooling and is designed for sprint performance - expecting a light use case on average and ramping up the CPU speed when needed (e.g. when loading a webpage). If a user wishes to load the system continually or operate it at a high temperature at full performance, further cooling may be needed.



Raspberry Pi 4 Model B Datasheet  
Copyright Raspberry Pi (Trading) Ltd. 2019

## 6 Availability

Raspberry Pi guarantee availability Pi4B until at least January 2026.

## 7 Support

For support please see the hardware documentation section of the Raspberry Pi website and post questions to the Raspberry Pi forum.

# Abbildungsverzeichnis

V.1	Raspberry Pi - Steuereinheit des HCIS . . . . .	9
V.2	Grundaufbau des Human-Computer Interaction Systems . . . . .	9
V.3	Anschlussplan Eingänge . . . . .	11
V.4	Anschlussplan Relais . . . . .	12
V.5	Panel Maße . . . . .	13
V.6	Befestigung des Displays . . . . .	14
V.7	Aufbau der Graphischen Benutzeroberfläche . . . . .	14
V.8	GUI Komponente - Navigation Menü . . . . .	15
V.9	GUI Komponente - Balken Anzeige . . . . .	16
V.10	GUI Komponente - Modus Anzeige . . . . .	16
V.11	GUI Komponente - Graph . . . . .	17
V.12	GUI Fenster - Login Menü . . . . .	18
V.13	GUI Fenster - Fahrdaten . . . . .	18
V.14	GUI Fenster - Akkudaten . . . . .	19
V.15	GUI Fenster - Fahrdaten Diagnose . . . . .	19
V.16	GUI Fenster - Fehler Liste . . . . .	20
V.17	Slots und Signals Konzept . . . . .	21
V.18	Verbindung Frontend zu Backend . . . . .	22
V.19	Anschlussplan CAN-Modul . . . . .	23
VI.1	Grundaufbau des Laststromkreises . . . . .	32
VI.2	Grundaufbau des Steuerstromkreises . . . . .	38
VI.3	Digital Input Specifications . . . . .	39
VI.4	Analog Input Specifications . . . . .	39
VI.5	Throttle Input Specifications . . . . .	40
VI.6	Sin/Cos Sensor Input Specifications . . . . .	40
VI.7	KSI and Coil Return Input Specifications . . . . .	41
VI.8	Analog Output Specifications . . . . .	41
VI.9	Digital and PWM Output Specifications . . . . .	42
VI.10	Power Supply Output Specifications . . . . .	42
VI.11	Communications Port Specifications . . . . .	43
VI.12	ECO/Sport-Select Programmschnippel . . . . .	52
VI.13	Leonardumformer Versuchsaufbau . . . . .	53
VI.14	BleiaKKU . . . . .	54

# Tabellenverzeichnis

V.1	Berechnung der Leistung des 12V-Systems . . . . .	10
V.2	Datenbankstruktur der Benutzer Tabelle . . . . .	25
V.3	Datenbankstruktur der Datenpaket 1 Tabelle . . . . .	25
V.4	Datenbankstruktur der Datenpaket 2 Tabelle . . . . .	26
V.5	Datenbankstruktur der Datenpaket 3 Tabelle . . . . .	26
V.6	Datenbankstruktur der Fehler-Datenpaket Tabelle . . . . .	27
VI.1	Geschwindigkeitsbegrenzer-Parameter . . . . .	47
VI.2	Reaktions-Parameter . . . . .	48
VI.3	Antriebsmodi . . . . .	49
VI.4	Datenpakete Deklaration . . . . .	51

# Code Listings

V.1	Code zum Starten eines Threads . . . . .	12
V.2	Konfigurieren des CAN Adapters . . . . .	24
V.3	Konfiguration der Datenbankschnittstelle . . . . .	27
V.4	Konfiguration der Datenbankschnittstelle . . . . .	28
V.5	Einfügen der Benutzer und Passwörter über den . . . . .	28