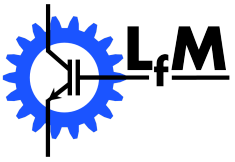

**Lehrstuhl für Mechatronik**




Prof. Dr.-Ing. Mark-M. Bakran
Universität Bayreuth

Betreuung:
Dr.-Ing. Michael Gleißner

Autoren:
Michael Rauh, Sascha Mehringer
Matr.- Nr: 1369342, 1349100

Teamprojektarbeit

# Entwicklung einer **P**V-gesteuerten, **u**niversellen **L**adestation mit **S**trom- und Phasenkonfigurationswechsel für **E**lektroautos (PULSE)

Michael Rauh, Sascha Mehringer
Matrikelnummer: 1369342, 1349100
Studiengang: Automotive und Mechatronik, Automotive und Mechatronik
Tel.: 0151/54793797, 0175/4162688
E-Mail: mitsch.rauh@gmail.com, sascha.mehringer@gmail.com

Teamprojektarbeit
Thema: Entwicklung einer **P**V-gesteuerten, **u**niversellen **L**adestation mit **S**trom- und Phasenkonfigurationswechsel für **E**lektroautos (PULSE)

Eingereicht: 16.02.2021

Betreuer: Dr.-Ing. Michael Gleißner

Prof. Dr.-Ing. Mark-M. Bakran
Lehrstuhl für Mechatronik
Universitätsstraße 30
D-95447 Bayreuth

## Erklärung:

Wir versichern hiermit, dass wir die vorliegende Arbeit selbständig verfasst und keine anderen als die im Quellenverzeichnis angegebenen Quellen benutzt habe. Stellen, die wörtlich oder sinngemäß aus veröffentlichten oder noch nicht veröffentlichten Quellen entnommen sind, sind als solche kenntlich gemacht. Die Zeichnungen oder Abbildungen in dieser Arbeit sind von uns selbst erstellt worden oder mit einem entsprechenden Quellennachweis versehen. Diese Arbeit ist in gleicher oder ähnlicher Form noch bei keiner anderen Prüfungsbehörde eingereicht worden.

| | | |
|---|---|---|
| Bayreuth, den 16.02.2021 | ..... | ..... |
| | Michael Rauh | Sascha Mehringer |

## Urheberrechtserklärung:

Hiermit versichern wir, dass wir die vorliegende Arbeit selbständig und unter Angabe aller Hilfsmittel und Referenzen angefertigt haben. Eine Ausnahme bilden die persönlichen Mitteilungen der Betreuer der Arbeit, die ständig eingeflossen sind und daher nicht im Einzelnen nachgewiesen werden. Der Universität Bayreuth, vertreten durch den Lehrstuhl für Mechatronik (Prof. Dr.-Ing. M. Bakran), haben wir das Nutzungs- und Verwertungsrecht an dieser Arbeit zu Zwecken der Lehre und Forschung, für Gutachten und Publikationen erteilt. Bei einer Veröffentlichung von Inhalten oder Auszügen der Arbeit ist in angemessener Weise auf unseren Beitrag hinzuweisen. Mit der Veröffentlichung unserer Namen und des Themas der Arbeit in der Liste der am Lehrstuhl angefertigten Arbeiten sind wir einverstanden. Wir erklären ferner, dass wir die Inhalte der Arbeit unsererseits nicht ohne Zustimmung des Lehrstuhls publizieren werden.

| | | |
|---|---|---|
| Bayreuth, den 16.02.2021 | ..... | ..... |
| | Michael Rauh | Sascha Mehringer |

## Danksagung:

Zuerst bedanken wir uns bei Prof. Dr.-Ing. Mark-M. Bakran, Inhaber des Lehrstuhls für Mechatronik der Universität Bayreuth, für die Möglichkeit, dass wir an seinem Lehrstuhl diese Teamprojektarbeit im gegebenen Umfang durchführen durften. Dank gilt ihm zusätzlich für die fachliche Betreuung und Unterstützung bei dieser Arbeit sowie für das Bereitstellen seines privaten Elektrovehikels für Versuch- und Testzwecke.

Unser Dank gilt natürlich auch unserem Betreuer Dr.-Ing. Michael Gleißner, welcher uns bei Fragen und Problemen jeglicher Art stets zur Seite stand.

Des Weiteren danken wir Dipl.-Ing. (FH) Christian Lösche für die Unterstützung bei technischen Fragen sowie seiner Hilfe bei der Anfertigung des Schaltschrankes. Zusätzlich bedanken wir uns bei allen Mitarbeitern des Lehrstuhls für Mechatronik, welche mit ihrem Fachwissen zum Gelingen der Arbeit beigetragen haben. Auch möchten wir uns bei Kira Schlesier fürs Korrekturlesen und natürlich allen Freunden und Familienmitgliedern, die uns bei dieser Arbeit unterstützt haben, bedanken.

# Inhaltsverzeichnis

**Abbildungsverzeichnis** **V**

**Tabellenverzeichnis** **VII**

**Symbole und Akronyme** **IX**

**1 Einleitung** **1**

1.1 Stand der Technik . . . . . 1

1.2 Zielsetzung . . . . . 2

**2 Grundlagen** **3**

2.1 Hardware . . . . . 3

2.1.1 Raspberry Pi 4 Model B . . . . . 3

2.1.1.1 GPIO . . . . . 4

2.1.2 8-Kanal-SSR . . . . . 4

2.1.3 Pulsares EV SimpleCharge . . . . . 5

2.1.4 Energiezähler . . . . . 8

2.1.4.1 S0-Schnittstelle . . . . . 10

2.1.5 Schütze und Relais . . . . . 10

2.1.5.1 Freigabeschütz . . . . . 10

2.1.5.2 Phasenwahlschütz . . . . . 11

2.1.5.3 Zustandsrelais . . . . . 11

2.1.6 Schaltschranknetzteil . . . . . 12

2.2 Software . . . . . 12

2.2.1 Integrierte Entwicklungsumgebung PyCharm . . . . . 12

2.2.2 Python . . . . . 12

2.2.3 Flask . . . . . 13

2.2.4 HTML und CSS . . . . . 13

2.2.5 JavaScript . . . . . 14

**3 Auslegung von Hard- und Software** **15**
3.1 Hardware . . . 15
3.1.1 Lastkreis . . . 15
3.1.2 Steuerkreis . . . 17
3.1.3 Modifizierte Ladestation . . . 18
3.1.4 Nachbau . . . 20
3.2 Software . . . 23
3.2.1 Projektstruktur und -inhalt . . . 23
3.2.2 Erstinbetriebnahme . . . 28
3.2.3 Benutzeroberfläche . . . 33
3.3 Messaufbau . . . 41

**4 Auswertung des Betriebsverhaltens** **44**
4.1 Analyse des implementierten Schaltverhaltens . . . 44
4.2 Betrachtung des Ladeverhaltens . . . 47
4.3 Testbetrieb der Solarautomatik . . . 50

**5 Zusammenfassung und Ausblick** **53**

**A Hardware** **55**
A.1 GPIO-Pinout . . . 55
A.2 GPIO-Funktionen . . . 56
A.3 Schaltplan . . . 57
A.4 Bauteile . . . 64

**B Software** **65**
B.1 server.service . . . 65
B.2 Ladespezifikation . . . 66

**C Exemplarische Ladekurven** **78**
C.1 Tesla Model 3 . . . 78
C.2 Renault Zoe . . . 83

**D Literatur** **86**

# Abbildungsverzeichnis


1.1 Ausgangszustand der experimentellen Ladestation . . . . . 2

2.1 Aufsicht des Raspberry Pi 4 Model B . . . . . 4
2.2 Schaltplan eines SSR-Kanals . . . . . 5
2.3 Funktionsschema der Kommunikation von Ladesäule und Elektrovehikel . . . . . 6
2.4 Aufsicht der EV SimpleCharge Platine von Pulsares . . . . . 7
2.5 Zusammenhang von PWM-Signal zwischen SimpleCharge und Norm . . . . . 9
2.6 finder Energiezähler . . . . . 9
2.7 Installationsschütz 40 A vierpolig . . . . . 11
2.8 Installationsschütz 25 A NC/NO . . . . . 11
2.9 Codebeispiel für einen einfachen Flask-Webserver . . . . . 13
2.10 Aufbau eines HTML-Dokuments . . . . . 14

3.1 Schematischer Aufbau der experimentellen Ladestation . . . . . 16
3.2 Gesamtübersicht der fertigen experimentellen EV-Ladestation . . . . . 19
3.3 Schema für den empfohlenen Nachbau . . . . . 22
3.4 Übersicht der Projektstruktur des Webservers . . . . . 23
3.5 Benutzeroberfläche des Raspberry Pi Imager . . . . . 29
3.6 Benutzeroberfläche des Bitvise SSH Clients . . . . . 29
3.7 Exemplarisches Ethernet-Profil für eine statische IP-Adresse . . . . . 30
3.8 Startseite des raspi-config Tools . . . . . 31
3.9 Nutzeroberfläche des Datenbankskripts . . . . . 31
3.10 SFTP-Terminal des Bitvise SSH Clients . . . . . 32
3.11 Startseite des Webinterfaces im ausgeloggten Zustand . . . . . 33
3.12 Login-Seite des Webinterfaces . . . . . 33
3.13 Startseite des Webinterfaces im eingeloggten Zustand . . . . . 34
3.14 Übersichtsseite des Webinterfaces: Statusanzeigen . . . . . 34
3.15 Übersichtsseite des Webinterfaces: Leistungsverläufe . . . . . 36
3.16 Startkonfigurationsseite des Webinterfaces . . . . . 37
3.17 Steuerungsseite des Webinterfaces: Allgemeines Interface . . . . . 38
3.18 Steuerungsseite des Webinterfaces: Interface für manuelle Steuerung . . . . . 39

3.19 Steuerungsseite des Webinterfaces: Interface für Leistungsvorgabe . . . . . 39
3.20 Steuerungsseite des Webinterfaces: Interface für Solarautomatik . . . . . 40
3.21 Schematischer Messaufbau . . . . . 41
3.22 Oszilloskop und Signalgenerator . . . . . 41
3.23 Differentialtastkopf, Strommesszange und Messwandler . . . . . 42
3.24 Zimmer Präzisions-Leistungsmessgerät LMG500 . . . . . 43

4.1 Analyse des Schaltverhaltens . . . . . 45
4.2 Schaltablauf bei Phasenumkonfiguration . . . . . 45
4.3 Anlaufstrom und zugehörige Phasenverschiebung des Tesla Model 3 . . . . . 48
4.4 Anlaufstrom und zugehörige Phasenverschiebung des Renault Zoe . . . . . 48
4.5 Vergleich der Ladekurven von Renault Zoe und Tesla Model 3 bei 16 A . . . . . 49
4.6 Variation des Ladestroms im laufenden Ladebetrieb des Tesla Model 3 . . . . . 50
4.7 Leistungsverläufe im Testbetrieb der Solarautomatik . . . . . 51

A.1 GPIO-Pinout des Raspberry Pi . . . . . 55

C.1 Ladekurven Tesla Model 3, einphasig mit 6 A . . . . . 78
C.2 Ladekurven Tesla Model 3, zweiphasig mit 6 A . . . . . 79
C.3 Ladekurven Tesla Model 3, dreiphasig mit 6 A . . . . . 79
C.4 Ladekurven Tesla Model 3, einphasig mit 12 A . . . . . 80
C.5 Ladekurven Tesla Model 3, zweiphasig mit 12 A . . . . . 80
C.6 Ladekurven Tesla Model 3, dreiphasig mit 12 A . . . . . 81
C.7 Ladekurven Tesla Model 3, einphasig mit 16 A . . . . . 81
C.8 Ladekurven Tesla Model 3, zweiphasig mit 16 A . . . . . 82
C.9 Ladekurven Tesla Model 3, dreiphasig mit 16 A . . . . . 82
C.10 Ladekurven Renault Zoe, einphasig mit 6 A . . . . . 83
C.11 Ladekurven Renault Zoe, dreiphasig mit 6 A . . . . . 83
C.12 Ladekurven Renault Zoe, einphasig mit 10 A . . . . . 84
C.13 Ladekurven Renault Zoe, dreiphasig mit 10 A . . . . . 84
C.14 Ladekurven Renault Zoe, einphasig mit 16 A . . . . . 85
C.15 Ladekurven Renault Zoe, dreiphasig mit 16 A . . . . . 85

# Tabellenverzeichnis


2.1 SimpleCharge Codierwiderstände . . . . . 6
2.2 Spannungsteiler zur Statusübermittlung . . . . . 7

3.1 Software für die Erstinbetriebnahme des Webserver . . . . . 28
3.2 Übersicht aller Steuerparameter des Webservers . . . . . 35
3.3 Standardeinstellungen der Ladestation . . . . . 37
3.4 Übersicht der Ladeleistungen verschiedener Elektroautomodelle . . . . . 38
3.5 Messgrößen für die Oszilloskopmessung . . . . . 42

4.1 Werte des DC des externen PWM-Signals . . . . . 44
4.2 Parameter für den Testbetrieb der Solarautomatik . . . . . 51

A.1 GPIO-Funktionen des Raspberry Pi 4 Model B . . . . . 56
A.2 Klemmplan der Reihenklemmen -X4, -X5 und -X6 . . . . . 63
A.3 Liste der verwendeten Bauteile für den Hardwareaufbau . . . . . 64

B.1 Parametererläuterung server.service . . . . . 65
B.2 Mögliche Stromwerte aufgrund der Stromspezifikation . . . . . 66
B.3 Ladespezifikation 1ph mit Maximalstrom 16A . . . . . 66
B.4 Ladespezifikation 1p mit Maximalstrom 32A . . . . . 67
B.5 Ladespezifikation 2ph mit Maximalstrom 16A . . . . . 68
B.6 Ladespezifikation 2ph mit Maximalstrom 32A . . . . . 69
B.7 Ladespezifikation 3ph mit Maximalstrom 16A . . . . . 70
B.8 Ladespezifikation 3ph mit Maximalstrom 32A . . . . . 71
B.9 Ladespezifikation 1ph/2ph mit Maximalstrom 16A . . . . . 72
B.10 Ladespezifikation 1ph/2ph mit Maximalstrom 32A . . . . . 73
B.11 Ladespezifikation 1ph/3ph mit Maximalstrom 16A . . . . . 74
B.12 Ladespezifikation 1ph/3ph mit Maximalstrom 32A . . . . . 75
B.13 Ladespezifikation 1ph - 3ph mit Maximalstrom 16A . . . . . 76
B.14 Ladespezifikation 1ph - 3ph mit Maximalstrom 32A . . . . . 77

# Symbole und Akronyme

## Akronyme

**GPIO** general purpose input/output

**PWM** Pulsweitenmodulation

**IDE** integrated development environment

**EV** electric vehicle

**CP** Control Pilot

**PP** Proximity Pilot

**IEC** International Electrotechnical Commission

**SAE** Society of Automotive Engineers

**MID** Measuring Instruments Directive

**SSR** Solid State Relay

**LED** light-emitting diode

**NO** Normally Open

**NC** Normally Closed

**PELV** Protective Extra Low Voltage = Schutzkleinspannung

**V2G** Vehicle to Grid

**V2H** Vehicle to Home

**RCD** Differenzstrom-Schutzschalter

**HTML** HyperText Markup Language

**CSS** Cascading Style Sheets

**SSH** Secure Shell

**SHA** secure hash algorithm

**OBD** On-Board-Diagnose

**SoC** State of Charge

**AC** alternating current

**dc** duty cycle

# 1 Einleitung

Die Elektromobilität ist derzeit allgegenwärtig, aber dennoch in vielen Bereichen noch nicht ausgereift bzw. in bestehenden Anwendungen implementiert. Dabei spielt die Rolle des electric vehicle (EV) als Energiespeicher im privaten Sektor eine große Rolle, dessen Einbindung in das Versorgernetz unter dem Überbegriff Smart Grid (zu deutsch intelligentes Stromnetz) eingeordnet wird. Dabei ist eine essentielle Komponente die Kommunikation und Steuerung aller Teilnehmer des Netzes. Darunter zählen Verbraucher, Energieerzeuger und Energiespeicher. Somit muss für eine erfolgreiche Implementierung des EV in das Smart Grid ein Informationsaustausch zwischen diesen und dem Netz erfolgen. Dafür ist die Einbindung aller neueren EV in die unter der ISO 15118 genormten Protokolle geplant, wobei jedes Vehikel eine unverkennbare Adresse erhält. Dies ermöglicht die Kommunikation mit öffentlichen oder privaten Ladenstationen und letztlich eine geplante Einbindung in das intelligente Netz. Zusätzlich muss das EV auch bidirektionales Laden (auch Vehicle to Grid (V2G) oder Vehicle to Home (V2H)) unterstützen, um die gespeicherte Energie für das Netz oder den Haushalt bereitstellen zu können. Dies ist jedoch derzeit noch wenig verbreitet. [1, 2]

## 1.1 Stand der Technik

Übliche Wallboxen, ob selbst gebaut oder fertig erworben, laden das EV mit der maximalen Anschlussleistung. Zusätzlich wird das Laden direkt bei Anschluss eines Fahrzeuges gestartet. Auch die Ladestation als Grundlage dieser Arbeit erfüllt nur diese rudimentären Funktionen. Dies erfolgt über eine einfache Steuerung zur Kommunikation der Wallbox mit dem EV und einem darauf folgenden Freigabeschütz. Die meisten Fahrzeuge unterstützen jedoch weit mehr Möglichkeiten zur Leistungsregelung, auch ohne eine Implementierung in ein neues Kommunikationssystem. Einige private Projekte der Onlinecommunity befassen sich mit verschiedenen Ansätzen zur Automatisierung ihrer Ladestation. Beispielsweise die indirekte Ansteuerung des Ladevorgangs über die von den Herstellern der Fahrzeuge und Solaranlagen mitgeführten Anwendungen. Die Programmierung für das Ladeverhalten erfolgt dabei beispielsweise auf einem Webserver. Dieser kann das Laden jedoch auch nur starten und stoppen. Die Regelung des maximalen Ladestroms findet noch in wenigen Ladestationen Anwendung, ist meist aber auch nur über herstellerspezifische Apps steuerbar. Der in Abb. 1.1 dargestellte Schaltschrank bildet die Grundlage für die in dieser Arbeit durchgeführten Versuche und Verbesserungen.

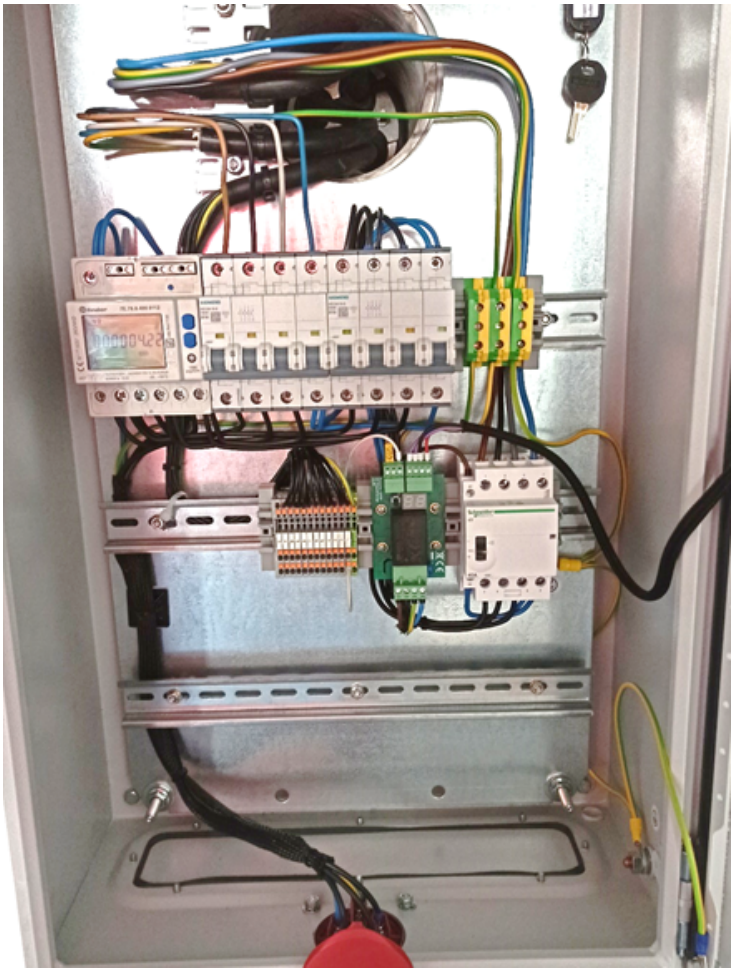


**Abb. 1.1:** Überblick des Schaltschranks der experimentellen Ladestation am Lehrstuhl für Mechatronik der Universität Bayreuth zu Beginn dieser Arbeit

## 1.2 Zielsetzung

Ziel dieser Arbeit ist es, eine Ladestation zu entwickeln, welche sich über ein Webinterface steuern lässt und eine einfache Anbindung eines in Haushalten genutzten Energieerzeugers, wie beispielsweise Photovoltaikanlagen oder Blockheizkraftwerke, ermöglicht. Die Ansteuerung soll in der Lage sein, die Ladeleistung entsprechend der überschüssigen Energie des angebundenen Erzeugers automatisch anzupassen, um diese effizienter zu nutzen. Die Ergebnisse werden anschließend für elektrisch unterwiesene Personen zum Nachbau aufbereitet und die Software als Open Source bereitgestellt.

# 2 Grundlagen

Dieses Kapitel soll die nötigen Grundlagen für den Hardwareaufbau der Ladestation vermitteln sowie die für die Erstellung des Webinterfaces zur Steuerung verwendete Programmiersprachen und die Entwicklungsumgebung kurz beschreiben.

## 2.1 Hardware

Im Folgenden wird auf die in dieser Arbeit verwendetet Hardware eingegangen. Dabei sind alle verwendeten Bauteile und Geräte für das Verständnis der Arbeit beschrieben.

### 2.1.1 Raspberry Pi 4 Model B

Der Raspberry Pi ist ein Einplatinencomputer, welcher ursprünglich für die Bereitstellung von preiswerten Computern zu Bildungszwecken entwickelt wurde. Der günstige Anschaffungspreis und die kostenlose Software sollen jedem einen Einstieg in die Anwendung eines PC‘s und das Programmieren jeglicher Art ermöglichen. Die Leistungsfähigkeit und Anwendungsvielfalt ist aber auch für fortgeschrittenen Programmierer interessant. Im Falle dieser Arbeit findet der Raspberry Pi 4 Model B mit 8 GB Arbeitsspeicher Anwendung. Als Prozessor ist ein Broadcom BCM2711 Chip mit einem 64 bit ARM v8 quad-core bei einer maximalen Taktrate von 1,5 GHz als System-on-a-Chip (SoC) auf dem Computer verbaut. Die Konnektivität ist über 2,4 GHz- und 5 GHz-WLAN, Bluetooth 5.0, Gigabit Ethernet und USB-Ports gegeben. Zusätzlich bestehen Zahlreiche Optionen zu multimedialen Anwendungen wie hochauflösende Bildschirmanschlüsse (HDMI), Kameraerweiterungen (CSI) und serielle Audio- und Videoanschlüssen. Wie in allen Vorgängerversionen ist eine 40-Pin Allzweckeingabe oder -ausgabe-Stiftleiste (general purpose input/output (GPIO)) installiert, um jegliche Erweiterungen an der Hardware zu ermöglichen. Diese ist im Folgenden genauer beschrieben.[3]

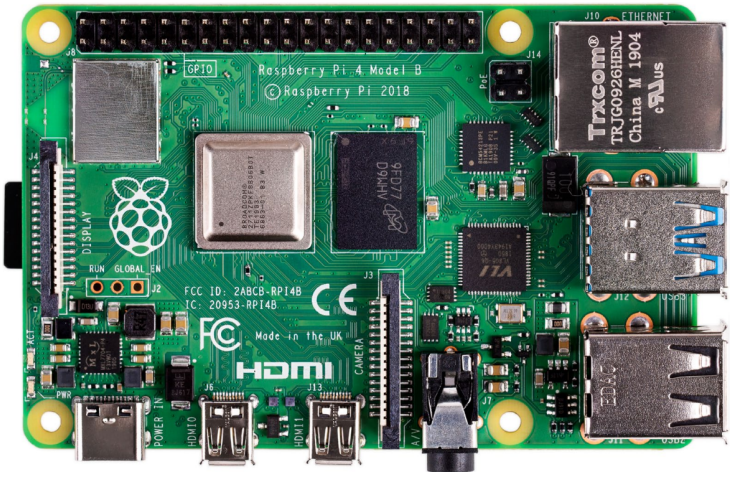



**Abb. 2.1:** Draufsicht auf die gesamte Platine des Raspberry Pi 4 Model B mit gesteckter Speicherkarte (links)

#### 2.1.1.1 GPIO

Die Ansteuerung der GPIO-Pins ist, wie die meisten Funktionen des Raspberry Pi, direkt in dem SoC integriert. Die Funktionen der Pins werden über Register angesteuert. Jeder Pin kann letztlich mehrere alternative Funktionen übernehmen, welche von der Programmierung des Anwenders abhängen. Nachteil dieser Flexibilität auf kleinem Raum ist das Fehlen einer Schutzbeschaltung für die GPIOs. Der gesamte GPIO-Block kann mit maximal 50 mA und einzelne GPIOs mit maximal 16 mA belastet werden. Auch bei der Spannungsverträglichkeit sollte nicht über 3,3 V hinaus verschaltet werden, da dies auch zu Beschädigungen führen kann (Die Werte sind nicht vom Hersteller verifiziert, sondern berufen sich nur auf Erfahrungen).[4, 5] Die alternativen Funktionen jedes einzelnen Pins und deren Anordnung auf der Platine können Tab. A.1 bzw. Abb. A.1 entnommen werden.

### 2.1.2 8-Kanal-SSR

Bei der maximalen Ausgangsspannung der GPIOs von 3,3 V ist zum Schalten von Schützen oder ähnlichem eine Schaltung zum Ansteuern höherer Spannung erforderlich. Dies ist mit einer 8-Kanal-SSR Platine realisierbar. Da auch andere Bauteile in diesem Projekt direkt 230 VAC (alternating current (AC), zu deutsch Wechselspannung) schalten, ist auch dieses Bauteil für dieses Spannungslevel ausgelegt. Die Kernfunktion der einzelnen Kanäle übernimmt ein Solid State Relay (SSR) der Firma OMRON. Genauer handelt es sich dabei um den Opto-Triac G3MB-202PL, welcher eine galvanische Trennung zwischen Last- und Steuerkreis ermöglicht. Die Ansteuerung der Opto-Triacs erfolgt mit einem 5 V Signal und muss, wie in Abb. 2.2 dargestellt, für die Ansteuerung mit 3,3 V zusätzlich beschaltet werden. Dabei handelt es sich um den Schaltplan für einen der acht Kanäle, wobei die 5 V- und `GND`-Potentiale auf der gesamten Platine für alle 8 Kanäle verbunden sind. Die parallel zum SSR geschaltete light-emitting diode (LED) mit Vorwiderstand dient lediglich der optischen Visualisierung des Schaltzustandes. Der `IN`-Pin wird über einen Spannungsteiler auf die Basis eines Transistors verschalten, welcher der LED und dem SSR das `GND`-Potential zuschalten kann. Somit wird eine anliegende Spannung von 2,5 V bis 20 V am `IN`-Pin als

HIGH-Signal und von 0 V bis 0,5 V als LOW-Signal interpretiert. Für den Bauteilschutz lastseitig ist eine 2 A-Sicherung verbaut. [6]

Um die Spannungssteilheit beim Abschalten von induktiven Lasten, wie Motoren oder Schütze, zu begrenzen, ist im OMRON G3MB-202PL ein RC-Glied, auch Snubber-Glied genannt, verbaut. Die Bauteilwerte der Beschaltung sind dem Datenblatt nicht zu entnehmen. Da der Schaltbereich bei 0,1 A bis 2 A liegt, kann es beim Schalten zu geringer Lasten und zu hoher Induktivität zum eigenständigen zünden des Opto-Triacs kommen, wodurch ein Trennen des Stromkreises verhindert wird. Beim Auftreten solcher Fehler kann ein zusätzliches RC-Glied (siehe Abb. 2.2) angefügt werden.

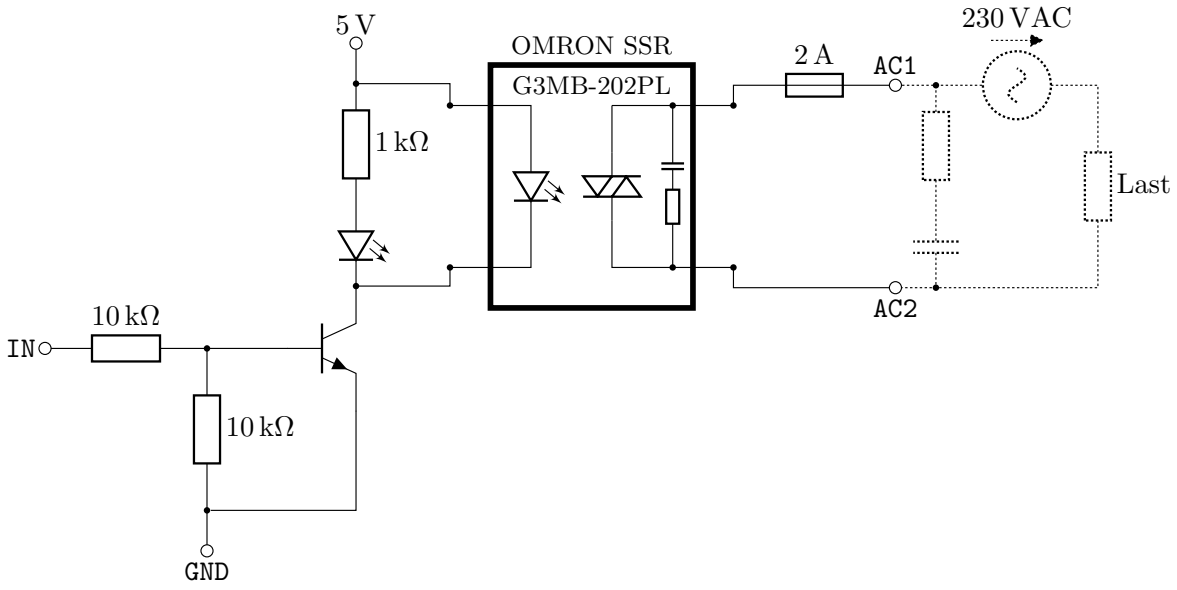



**Abb. 2.2:** Verschaltung eines Kanals auf der 8-Kanal-SSR Platine zum Schalten von 230 V Lasten mit bis zu 2 A und einer Ansteuerung über geringe Spannungen

### 2.1.3 Pulsares EV SimpleCharge

Die EV SimpleCharge Platine von Pulsares ist eine günstige Variante für die grundlegende Steuerung einer EV-Ladesäule. Sie übernimmt die Kommunikation mit dem EV und die Freigabe der AC-Ladespannung. Dies wird in der Norm als Ladebetriebsart 3 (Mode 3) gekennzeichnet. Somit ist für den Anschluss des EV nur ein Ladekabel mit IEC Typ 2 Kontaktierung (International Electrotechnical Commission) notwendig. Dabei wird ein genormtes Kommunikationsprotokoll zwischen EV und der Platine verwendet. Der IEC Typ 2-Stecker ist im Gegensatz zum CEE-Drehstromsteckverbinder (IEC 60309) mit zwei Kommunikationsanschlüssen versehen. Dabei handelt es sich um die Kontroll-/Datenleitung, auch Control Pilot (CP) genannt, und den Ladekabelerkennungskontakt, auch Proximity Pilot (PP) genannt. Es erfolgt von Seiten des EV und der Ladesäule eine Messung des Widerstands am PP zur Erde, um den vorhandene Kabelquerschnitt des Ladekabels ermitteln zu können. Die Querschnitte und zulässigen Ströme sind wie in folgender Tabelle den Widerständen des PP zugeordnet.

**Tab. 2.1:** Zuordnung der verwendeten Kabelquerschnitte mit den Codierwiderständen für die SimpleCharge Platine und die dadurch zulässigen Maximalströme

| Leitungsquerschnitt | Codierwiderstand | Zulässige Stromstärke |
|---|---|---|
| 1,5 $mm^2$ | 1500 Ω | 13 A |
| 2,5 $mm^2$ | 680 Ω | 20 A |
| 6 $mm^2$ | 220 Ω | 32 A |
| 16 $mm^2$ | 100 Ω | 63 A |

Bei der EV SimpleCharge Platine wird jedoch am PP-Eingang ein Widerstand direkt angeschlossen, um die Leitungsquerschnitte in der Ladestation zu berücksichtigen. Somit kann bei einer internen 2,5 $mm^2$ Verdrahtung trotz Anschluss eines höher belastbaren Ladekabels kein größerer Strom als 20 A freigegeben werden. Der Leitungsschutz für das Ladekabel, im Fall eines geringeren Querschnitts, wird somit vom EV übernommen.

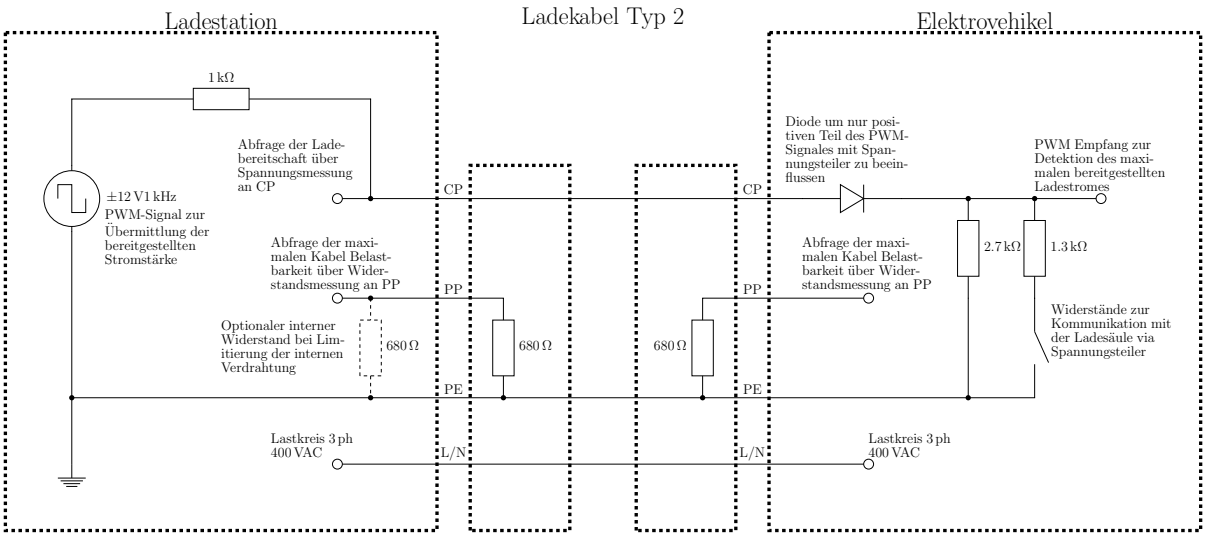



**Abb. 2.3:** Funktionsschema zur Veranschaulichung der Kommunikation einer Ladesäule mit einem Elektrovehikel nach IEC 62196-1 und IEC 61851-1 [7, 8]

Die von der Ladesäule für das EV freigegebene Stromstärke wird mit Hilfe eines Rechtecksignales (±12 V und 1 kHz) von der SimpleCharge Platine übermittelt. Dabei dient der Tastgrad (duty cycle (dc)) des Signals als Informationsträger für die bereitgestellte, maximale Stromstärke der Ladesäule und die Spannung als Statusinformation der Ladebereitschaft von EV und Ladecontroller.

Die pulsierende Rechteckspannung wird beim Verbinden des Ladekabels mittels einer Diode, einem festen Widerstand und weiteren zuschaltbaren Widerständen im Vehikel beeinflusst. Der Grundzustand ist in Abb. 2.3 dargestellt. Somit wird beim Einstecken des Kabels der Spannungspegel im positiven Spannungsbereich des Rechtecksignals beeinflusst und kann vom Vehikel durch parallel zuschaltbare

Widerstände verändert werden. Die möglichen genormten Spannungsteilerkonfigurationen und deren Informationen an die Ladesäule sind in der folgenden Tabelle aufgeführt.

**Tab. 2.2:** Kommunikationsprotokoll zwischen EV und Ladestation zur Ladestatusbestimmung und Fehlererkennung mit der Zuordnung der Spannungsteiler und Status[7]

| $U_{CP-PE}$ in V | $R_{EV}$ in Ω | $R_{CP-PE}$ in Ω | Status | Beschreibung |
|---|---|---|---|---|
| 12 | - | ∞ | A | Standby |
| 9 | 2740 | 2700 | B | Vehikel erkannt |
| 6 | 2740 ∥ 1300 | 880 | C | Bereit (Laden) |
| 3 | 2740 ∥ 270 | 240 | D | Mit Kühlung |
| 0 | 2740 ∥ 0 | 0 | E | Keine Spannung |
| -12 | - | - | F | Fehler |

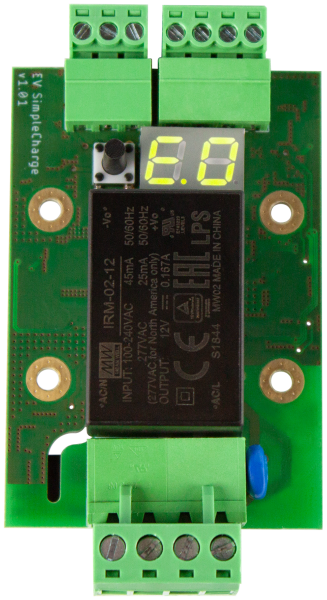



**Abb. 2.4:** EV SimpleCharge Platine von Pulsares mit Anschlussstecker mit Schraubkontakten

Wie das EV die empfangene Pulsweite interpretiert, ist in der IEC 61851-1 (International Electrotechnical Commission) festgelegt. Dabei wird der für das Vehikel maximal freigegebene Ladestrom $I_{laden}$ in Abhängigkeit des Tastgrads $D_{EV}$ durch die abschnittsweise definierte, lineare Funktion Gl.(2.1.1) beschrieben. Die Unterteilung in Bereiche unterschiedlicher Steigung existiert aufgrund der Erweiterung der Norm für Ströme bis 80 A. Dabei konnte die ursprüngliche Codierung der SAE J1772 (Society of Automotive Engineers) für Ströme bis 48 A ($D_{EV} = 0,8$) nicht geändert werden. Die von der Platine an das Elektrovehikel übertragenen maximale Stromstärke $I_{laden}$ ist durch den in der Hardware verdrahteten Widerstand limitiert (PP), kann jedoch unterhalb dieser Grenze von der Software gesteuert werden. Dies erfolgt über die Einstellung verschiedener Ladeprofile direkt an der Platine mit Hilfe eines Tasters oder, für

die Automatisierung interessanter, mittels eines an diesen Tasteranschluss gesendeten 4 Hz-PWM Signals. In diesem Fall wird ebenfalls mit dem Tastgrad des Signals die gewünschte Stromstärke übertragen. Dabei kann die Platine die empfangene maximale Stromstärke in 1 A Schritten nach der Gl.(2.1.2) auflösen und weitergeben.

$$I_{\text{laden}} = \begin{cases} 6\,\text{A} & \text{für} \quad 0,08 \geq D_{\text{EV}} \geq 0,10 \quad \text{I} \\ 0{,}6\,\text{A} \cdot D_{\text{EV}} \cdot 100 & \text{für} \quad 0,10 \geq D_{\text{EV}} \geq 0,85 \quad \text{II} \\ 2{,}5\,\text{A} \cdot (D_{\text{EV}} - 0.64) \cdot 100 & \text{für} \quad 0,85 \geq D_{\text{EV}} \geq 0,96 \quad \text{III} \\ 80\,\text{A} & \text{für} \quad 0,96 \geq D_{\text{EV}} \geq 0,97 \quad \text{IV} \end{cases} \tag{2.1.1}$$

$$D_{\text{SC\%}} = 1,579 \cdot \frac{I_{\text{SC}}}{\text{A}} - 4,474 \tag{2.1.2}$$

In Abb. 2.5 sind die PWM-Zusammenhänge der beiden Signale über ihren Tastgrad aufgetragen. Die Polaris EV SimpleCharge Platine übersetzt das empfangene PWM-Signal in das für die EV genutzte $\pm 12\,\text{V}\,1\,\text{kHz}$ Signal. Der Platine ist es möglich einen Strom von 4 A bis 6 A zu interpretieren, jedoch ist dieser Bereich nicht zur Übermittlung an das EV genormt. Soll beispielsweise ein Ladestrom von $I_{\text{SC}} = 10\,\text{A}$ vorgegeben werden, so wird für das externe PWM-Signal ein Tastgrad (in Prozent) von $D_{\text{SC\%}} = 11{,}3\,\%$ gemäß Gl.(2.1.2) gewählt und an den Tastereingang übermittelt. Die SimpleCharge-Platine übersetzt diese Signal anschließend gemäß Gl.(2.1.1) in das genormte Kommunikationsprotokoll für die Übertragung an das EV. Somit beträgt für den übermittelten Ladestrom $I_{\text{laden}}$ von 10 A der Tastgrad des PWM-Signals $D_{\text{EV}}$ der Kommunikationsleitung nach Gl.(2.1.1) (II) $0,167$.

### 2.1.4 Energiezähler

Der finder Energiezähler 7E.78.8.400.0112 ist ein Gerät zur Zählung transportierter elektrischer Energie im Dreiphasennetz. Mit Hilfe der gemessenen Spannungen und Ströme der drei Außenleiter ist es dem Zähler möglich, Momentanwerte wie Spannungen, Ströme, Leistungsfaktoren, Leistungen, Phasenfolgen und Leistungsrichtung auf dem integrierten LCD-Display anzuzeigen. Zusätzlich werden die dazugehörigen Energien gespeichert und aufsummiert. Mit der Intervallmessungen ist es möglich einen Teilzähler, unabhängig vom Gesamtzähler, zu starten. Bei vielen Energiezählern ist zur Einbindung in verschiedenste Systeme eine digitale Schnittstelle zur Übertragung der ermittelten Daten integriert. Dabei sind Kommunikationsprotokolle wie Modbus, Mbus, LAN oder Eib möglich. In diesem Fall wird auf eine simple S0-Schnittstelle zurückgegriffen. Diese wird in ihrer Funktion im folgenden genauer Erläutert. [9]

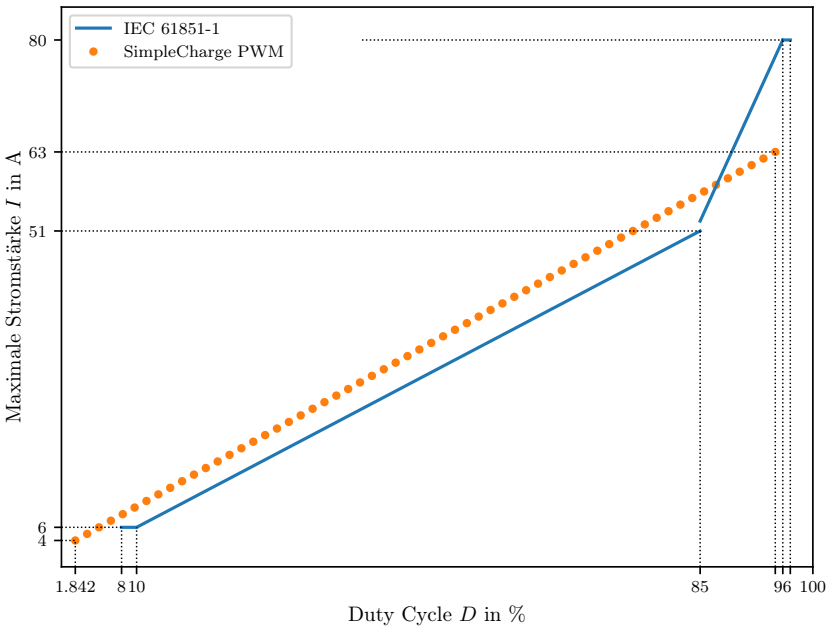



**Abb. 2.5:** Visualisierung der mittels PWM übertragenen maximalen Ladeströme an das EV von der Polaris EV SimpleCharge Platine (IEC 61851-1) und das von einer eigenen Steuerung an EV von der Polaris EV SimpleCharge übertragene PWM Signal


**Abb. 2.6:** finder Energiezähler 7E.78.8.400.0112 ohne Darstellung der S0-Schnittstelle auf der Oberseite [9]

#### 2.1.4.1 S0-Schnittstelle

Die S0-Schnittstelle ist zur Übermittlung von Daten wie Wasser-, Gas- oder Stromverbrauch innerhalb einer Gebäudeautomation. Dabei wird die gezählte Energie in Impulsen weitergeben, welche vom Hersteller in ihrer Wertigkeit festgelegt werden. Im Beispiel des vorangegangen finder Energiezählers liegt das Signal bei 100 imp/kWh. Für die Übertragungssicherheit ist die Schnittstelle als Lifezero-Signal ausgelegt und ist zusätzlich vom Messgerät selbst mit einem Optokoppler galvanisch getrennt. Ein Lifezero-Signal hat einen versetzten Nullpunkt („lebender Nullpunkt“), z.B. 1 V als definiert `LOW` und 5 V als definiert `HIGH`, wodurch dem Nullwert kein Zustand zugewiesen wird. Die Anschlusswerte belaufen sich dabei auf die Klassen A, für große Übertragungsstrecken (27 V DC), und B, für kurze Übertragungsstrecken (15 V DC). Die Impulse sind durch die Norm wie folgt definiert.

$$30\,\text{ms} \leq t_{\text{HIGH}} \leq 120\,\text{ms} \qquad 30\,\text{ms} \leq t_{\text{LOW}} \qquad t_{\text{EDGE}} \leq 5\,\text{ms} \tag{2.1.3}$$

Dabei beschreibt $t_{\text{HIGH}}$ die Dauer eines Impulses, $t_{\text{low}}$ die Mindestzeit bis zum nächsten Impuls und $t_{\text{EDGE}}$ die maximale Zeit für den Pegelwechsel zwischen `HIGH` und `LOW`.[10]
Durch Erfassen der Impulse lassen sich die momentane Leistung P in W und die gezählte Energie E in kW h gemäß Gl.(2.1.4) ermitteln. Dabei entspricht $\Delta t$ dem Zeitabstand zwischen zwei Impulsen in s, $N_{\text{S0}}$ der Anzahl an Impulsen pro kW h in $\frac{1}{\text{kWh}}$, z.B. wie hier 100 imp/kWh, und n der Anzahl an gezählten Impulsen.

$$P = \frac{1}{\Delta t} \times \frac{1000 \times 3600}{N_{\text{S0}}} \qquad E = \frac{n}{N_{\text{S0}}} \tag{2.1.4}$$

### 2.1.5 Schütze und Relais

Zum einfachen Schalten und ansteuern von elektrischen Lasten und Signalen sind Schütze und Relais noch immer zeitgemäße Varianten im Gegensatz zur Halbleitertechnik. Im Folgenden werden die in dieser Arbeit verwendeten Magnetschalter vorgestellt.

#### 2.1.5.1 Freigabeschütz

Um die Funktion der in Abschnitt 2.1.3 vorgestellten Platine zu gewährleisten, benötigt diese ein mit 230 V ansteuerbares Leistungsschütz mit vier Schließern für die drei Außenleiter und den Neutralleiter. Somit ist der Typ2-Stecker ohne eine Freigabe von der Steuerplatine komplett vom Netz getrennt. Das in diesem Fall verwendete Installationsschütz Acti 9 iCT von Schneider Electric (iCT 40A 4S 220/240V 50Hz) ist bei der Gebrauchskategorie AC-7a für 40 A und bei AC-7b für 15 A Nennstrom konzipiert. Um für ungesteuerte Ladebetriebsarten die Steuerplatine zu umgehen, bzw. das elektrische Ansteuern zu unterbinden, ist das Schütz zusätzlich mit einer mechanischen Betätigung versehen, um zwischen den Schaltzuständen `I`, `auto` und `0` zu wählen. [11]


**Abb. 2.7:** Installationsschütz A9C21844 Acti 9 iCT von Schneider Electric iCT 40A 4S 220/240V 50Hz [11]

#### 2.1.5.2 Phasenwahlschütz

Für die Schaltung von einzelnen Phasen werden finder Installationsschütze mit 25 A Nennstrom (Betriebskategorie AC-7a) herangezogen. Die hier verwendete Ausführung mit einem Schließer und einem Öffner ist somit variabel in der Verschaltung einsetzbar. Die Ansteuerung der Spule erfolgt mit 230 VAC. [12]


**Abb. 2.8:** Installationsschütz finder 22.32.0.230.4520 Schütz 1 Schließer, 1 Öffner 230 V/DC, 230 V/AC 25 A [12]

#### 2.1.5.3 Zustandsrelais

Um einen Schaltzustand mit 230 VAC über einen GPIO des Raspberry Pi einzulesen, muss das Signal vorher auf 3,3 VDC herabgestuft werden. Bei der geringen Spannung der Kontaktseite ist auch ein dafür vorgesehenes Relais mit geringem Kontaktwiderstand nötig. Dies wird mit einem Einzelrelais von Phoenix Contact (Phoenix Contact 2961480 REL-MR-230AC/21-21AU) realisiert, dessen hartvergoldete Schaltkontakte definierte Schaltsignale ab Spannungen von 100 mV (bei 10 mA) darstellen können. Das Relais wird auf einem Sockel mit Schraubkontakten angebracht. [13]

### 2.1.6 Schaltschranknetzteil

Zur Spannungsversorgung des Raspberry Pi und der SSR-Platine ist ein 5 VDC-Netzteil notwendig. Im Falle dieser Arbeit wird das Mean Well MDR-20-5 Hutschienennetzteils verwendet. Bei einem Eingangsspannungsbereich von 85 VAC bis 264 VAC und 120 VDC bis 370 VDC kann das Netzteil eine Ausgangsspannung von 5 VDC bis zu einem maximale Ausgangsstrom von 3 A aufrechterhalten. Mit Hilfe eines Widerstandspotentiometers ist die Spannung auch in einen Bereich von 4,75 V bis 5,5 V verstellbar. Für die Betriebssicherheit ist das Netzteil mit einem Kurzschlussschutz und einem Überspannungsschutz versehen. [14]

## 2.2 Software

In diesem Abschnitt werden alle für die Entwicklung der Webapplikation zur Steuerung der Ladestation nötigen Elemente sowie die dafür verwendete Entwicklungsumgebung kurz beschrieben.

### 2.2.1 Integrierte Entwicklungsumgebung PyCharm

Die integrierte Entwicklungsumgebung PyCharm (integrated development environment (IDE)) ist eine Entwicklungsumgebung des Softwareunternehmens JetBrains, welche speziell für die Programmiersprache Python entwickelt wurde. Darüber hinaus unterstützt PyCharm auch die Entwicklung von Webapplikationen mit dem Python-Paket Flask und den dazugehörigen Komponenten JavaScript, HTML und CSS. Zudem ermöglicht die Vollversion das Nutzen der Secure Shell (SSH)-Funktion, mit welcher beispielsweise extern auf einen Raspberry Pi zugegriffen werden kann. Über die SSH-Verbindung kann als Projektinterpreter die auf dem Raspberry Pi installierte Version von Python verwendet und auch konfiguriert werden. Die Software bietet außerdem die Möglichkeit einer umfangreichen Versionsverwaltung der Projektdateien mit Backupfunktion sowie das automatische Synchronisieren des Projektordners zwischen PC und Raspberry Pi. [15, 16]

### 2.2.2 Python

Python ist eine Open Source Programmiersprache, deren Entwicklung 1989 von Guido van Rossum begonnen wurde und durch die gemeinnützige Organisation Python Software Foundation koordiniert wird. Das Ziel war es, eine Programmiersprache zu schaffen, die leicht zu erlernen ist sowie eine gute Lesbarkeit des Programmtextes mit sich bringt. Python ist eine objektorientierte Programmiersprache, unterstützt aber auch andere Programmierparadigmen wie z.B. die aspektorientierte oder funktionale Programmierung. Ein Python-Programm wird gängigerweise als Skript bezeichnet. Zudem ist Python auf allen gängigen Betriebssystemen (Unix, Windows, Mac OS) verfügbar und eignet sich ideal für die Programmierung eines Raspberry Pi. [17]

### 2.2.3 Flask

Flask ist ein in Python geschriebenes Webframework, dessen Fokus auf der Erweiterbarkeit liegt. Im Gegensatz zu anderen Frameworks wie beispielsweise Django oder Web2py erlaubt Flask ein einfaches Einbinden von bereits bestehenden Bibliotheken, wodurch seine Kernfunktionalität klein und einfach gehalten werden kann. Für die meisten gängigen Funktionen wie beispielsweise Authentifizierung, Cookies, Caching oder das Verbinden mit einer Datenbank existieren Erweiterungen. Zudem ist für Test- und Entwicklungszwecke ein Webserver in Flask integriert. Wie einfach es ist, einen Webserver in Flask auf dem lokalen System zu erstellen, zeigt Abb. 2.9. Ebenso ist es möglich, den Webserver für alle sich im gleichen Netzwerk befindlichen Teilnehmer erreichbar zu machen. Für eine detaillierte Dokumentation von Flask wird auf [18] verwiesen. Anzumerken ist, dass der integrierte Webserver von Flask sich nur für die Entwicklung sowie zum Testen gedacht ist und über das Heimnetzwerk hinaus nicht verwendet werden sollte. [18, 19]


**Abb. 2.9:** Code-Beispiel für einen einfachen Flask-Webserver: (a) Code und (b) Anzeige im Webbrowser unter Lokalhost

### 2.2.4 HTML und CSS

HyperText Markup Language (HTML) ist keine Programmiersprache im klassischen Sinn, sondern eher ein Set von Anweisungen, welches vorgibt, wie Inhalte in einem Webbrowser dargestellt werden sollen. HTML-Dokumente bestehen aus zwei wesentlichen Teilen, den Informationsinhalten und einem Set von Befehlen zum Visualisieren dieser Informationsinhalte. Jedes HTML-Dokument sollte mindestens aus den folgenden vier Elementen bestehen, welche wie in Abb. 2.10 angeordnet sind.

- Dokument: <html> ... </html>
- Kopf: <head> ... </head>
- Titel: <title> ... </title>
- Körper: <body> ... </body>

```
<html lang="en">
<head>
    <title> ... </title>
</head>
<body>
    ...
</body>
</html>
```

**Abb. 2.10:** Aufbau eins HTML-Dokuments

Zur einheitlichen Formatierung von Webinhalten über mehrere Seiten hinweg können Formatvorlagen definiert werden. In diesen Vorlagen lassen sich für Elemente des HTML-Dokuments, wie beispielsweise Auswahlfelder, bestimmte Formatierungen (Schriftgröße, Ausrichtung, usw.) oder Konturen festlegen. Eine Möglichkeit ist die Verwendung von Cascading Style Sheets (CSS), welche eine weitverbreitete Art von Standard-Formatvorlagen sind. [20]

### 2.2.5 JavaScript

JavaScript ist eine Skriptsprache, welche ursprünglich im Jahr 1995 von dem US-amerikanischen Softwareunternehmen Netscape entwickelt wurde. Der Grundgedanke hinter JavaScript war das Einpflegen von dynamischen HTML-Elementen in Webbrowsern. So können mit Hilfe von JavaScript beispielsweise einzelne Elemente einer Seite nachgeladen oder aktualisiert werden, ohne die ganze Seite neu laden zu müssen. Zudem eignet sich JavaScript ideal für die Interaktion zwischen Nutzer und Seite. [21]

# 3 Auslegung von Hard- und Software

In diesem Kapitel wird zunächst der Aufbau der Hardware beschrieben. Anschließend erfolgt eine detaillierte Beschreibung der Software sowie des Webinterfaces im Bezug auf Komponenten, Inbetriebnahme und Benutzeroberfläche. Zum Schluss wird der zur Validierung des Hardwareaufbaus und der Software gewählte Messaufbau kurz erläutert.

## 3.1 Hardware

Im Folgenden ist der Hardwareaufbau der experimentellen EV-Ladestation beschrieben. Das Wissen über den Zustand vor den Ausführungen dieser Arbeit ist nicht zum Verständnis der Funktionen vorausgesetzt. Das Funktionsverständnis der Bauteile ist Abschnitt 2.1 zu entnehmen oder als gegeben erachtet. Der Übersichtlichkeit halber ist Abschnitt 3.1 an Abb. 3.1 orientiert. Für die Zuordnung der Bauteile sind diese mit Betriebsmittelkennzeichnungen versehen [22]. Die genaue Verdrahtung und der Schaltschrankaufbau sind Abschnitt A.3 zu entnehmen.

### 3.1.1 Lastkreis

Der Energietransport zum Laden des Vehikels erfolgt über den sogenannten Lastkreis. Dafür ist die Ladestation zunächst über einen üblichen CEE-16 A Industriestecker (-X1) mit Spannung versorgt. Im Vorfeld ist der dafür genutzte Anschluss mittels eines allstromsensitiven Fehlstromschutzschalters, mit einem Bemessungsdifferenzstrom von 30 mA, und einem Leitungsschutzschalter mit Auslösecharakteristik C16 abgesichert. Der spezielle Schutzschalter ist nötig, da ein reiner Differenzstromschutzschalter bei Gleichstromfehlern, nach der Gleichrichtung im Vehikel, nicht auslöst und zusätzlich bei einem solchen Fehler das Auslösen bei Wechselstromfehlern verhindert werden kann. [23]

Um die genutzte Energiemenge zu messen, ist der Einspeisung direkt ein Energiezähler (-U1, Abschnitt 2.1.4) nachgeschaltet. Mit Hilfe der S0-Schnittstellen gibt der Zähler Impulse an die Steuerung, welche auf Basis der gewählten Ausgabewerte verschiedene Messwerte interpretiert. Auf der Klemmleiste -X5 ist der Kondensator -C1 mit 100 nF zwischen die S0+ und S0- Leitung verschaltet, welcher mit dem Widerstand innerhalb der Schnittstelle ein Tiefpassfilter bildet. Folglich werden von angrenzenden Schaltungen eingekoppelte hochfrequente Signale herausgefiltert.[9]

Für die weitere Absicherung und der Möglichkeit, die nachfolgenden Bauteile manuell spannungsfrei zu schalten, ist der Lastkreis mit dem Leitungsschutzschalter -F2 versehen. Mit der Sicherung -F1 sind ein-

und dreiphasige CEE-16 A Buchsen und eine Schutzkontaktsteckdose zuschaltbar, falls diese für etwaige Anwendungen nötig sind.

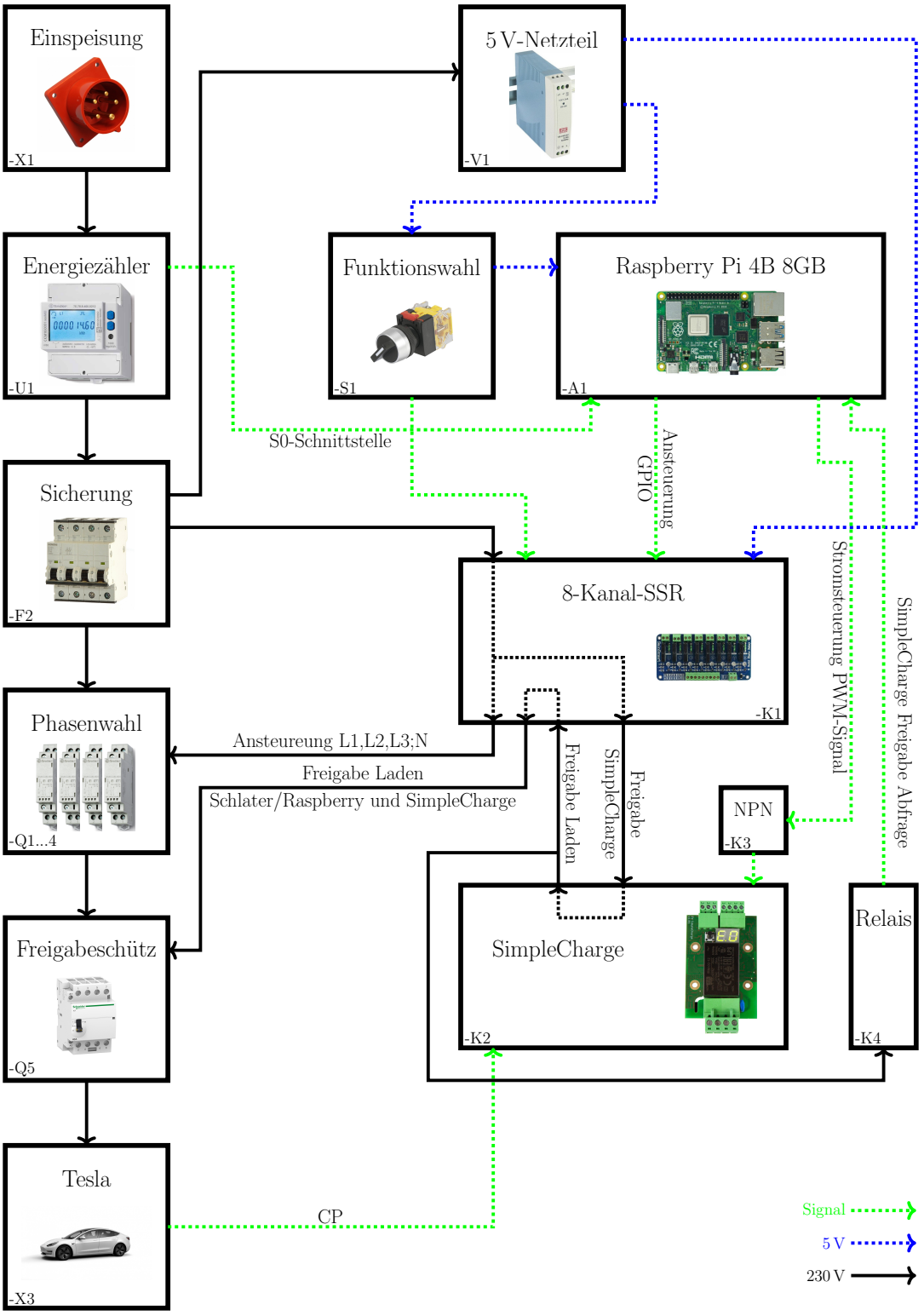



**Abb. 3.1:** Schematische Darstellung des Hardwareaufbaus der experimentellen Ladestation zur Veranschaulichung der funktionellen Zusammenhänge

Die Ansteuerung, bzw. Zuschaltung, der einzelnen Phasen und des Nullleiters für das spätere Laden ist mit vier Installationsschützen (-Q1...4, Abschnitt 2.1.5.2) realisiert. Demnach ist es möglich, dem EV jegliche Schaltkombination der drei Außenleiter und des Nullleiters anzulegen. Die Ansteuerung des Nullleiters dient dabei lediglich Testzwecken und ist für den eigentlichen Betrieb nicht notwendig. Beim Trennen des Nullleiters ist zu beachten, dass es in Schaltungen mit Sternpunkten zu Schäden führen kann, da bei unsynchroner Last Überspannungen entstehen. Die verwendeten Schütze haben einen Bemessungsstrom von 25 A bei der Betriebskategorie AC-7a. Die Auswahl eines Schützes dieser Kategorie ist aufgrund der Strommessungen mit einem Tesla Model 3 erfolgt. Da dieser beim Laden eine sehr geringe Phasenverschiebung aufweist ist das Schalten von schwach induktiver Last als Betriebskategorie ausreichend. Zusätzlich werden die Ströme nur langsam vom Gleichrichter im Auto erhöht, womit keine Einschaltströme oberhalb der Bemessung entstehen können.

Das eigentliche Zuschalten und Abschalten der Last, auf Seiten der Ladesäule, erfolgt durch das vierpolige Freigabeschütz (-Q5, Abschnitt 2.1.5.1). In diesem Fall ist das Schütz zum Schalten des Nennstromes bei geringer Induktivität der Last ausgelegt.

Im Laufe der Testphase ist die Abschaltung der Schütze aufgrund der zu hohen Induktivitäten ihrer Magnetspule teilweise nicht erfolgt. Dies führte zu einem Selbstzünden der Opto-Triacs. Zur Erweiterung der Snubber-Glieder werden zusätzliche, externe RC-Glieder (vier Stück für die Schütze -Q1...4) parallel zu den Spulen verschaltet. Die Bauteilwerte der Glieder belaufen sich auf 2200 Ω Widerstände und 220 nF Kondensatoren.

Wenn letztlich die Sicherung -F2 durchgeschaltet, mit den Phasenwahlschützen die Außenleiterkonfiguration gewählt und die Spannung mit dem Freigabeschütz der Ladebuchsen zugeschaltet ist, kann das EV mit dem Laden beginnen. Die Ansteuerung dafür erfolgt über den Steuerkreis.

### 3.1.2 Steuerkreis

Der Steuerkreis benötigt zunächst eine 5 VDC Spannungsversorgung, um den Raspberry Pi (-A1, Abschnitt 2.1.1) und die 8-Kanal-SSR-Platine (-K1, Abschnitt 2.1.2) betreiben zu können. Direkt nach der Sicherung -F2 wird 230 VAC Spannung für das 5 V-Netzteil (-V1, Abschnitt 2.1.6) abgegriffen. Um den Schutz vor elektrischen Schlag zu garantieren, ist das 0 V Potential auf der Klemmleiste -X5 mit dem PE-Leiter verbunden (Protective Extra Low Voltage = Schutzkleinspannung (PELV)). [24] Die SSR-Platine ist ohne weitere Schaltung umgehend mit der Gleichspannung versorgt. Die Spannungsversorgung des Raspberry Pi ist über den Wahlschalter -S1 trennbar. Ist dies der Fall, sind über den Schalter zwei Kanäle der SSR-Platine (`IN 1-2`) mit 5 V angesteuert und durchgeschaltet, um die SimpleCharge-Platine mit 230 VAC zu versorgen und im Falle einer Ladefreigabe diese direkt an das Freigabeschütz -Q5 weiterzugeben. Folglich kann das Ladegerät mit drei Phasen und den an der SimpleCharge-Platine maximal eingestellten Ladestrom betrieben werden. Die Schalterstellung für diesen Betriebsfall ist am Schaltschrank mit „RaspberryPi Steuerung AUS“ beschriftet.

Ist die Schalterstellung des Wahlschalters -S1 auf „RaspberryPi Steuerung AN“ gestellt, werden die

Funktionen mittels Software geregelt. Über die GPIOs kann der Raspberry Pi mit Hilfe der 8-Kanal-SSR-Platine die 230 V Phasenwahlschütze -Q1...4 schalten, welche als Öffner (Normally Closed (NC)) im Lastkreis integriert sind. Demnach entspricht eine Ansteuerung dem Trennen des Außenleiters. Im Detail erfolgt zunächst über vier Kanäle die Ansteuerung der Phasenwahlschütze -Q1...4 (`IN 4-7`). Die Spannungsversorgung für die SimpleCharge-Platine, welche die gesamte Kommunikation mit dem EV übernimmt, wird von der Software initialisiert (`IN 8`). Ist dies der Fall, wird nach einer erfolgreichen Kommunikation zwischen der SimpleCharge-Platine und einem EV das Laden freigegeben. Dies erfolgt über einen 230 V Ausgang der Platine und steuert damit das Freigabeschütz -Q5 an. Für flexiblere Steuermöglichkeiten ist dieses Signal über ein Relais der SRR-Platine zusätzlich geschaltet, wodurch die endgültige Ladefreigabe sowohl von der RaspberryPi-Steuerung als auch von der SimpleCharge-Platine erfolgt. Die Freigabe erfolgt, nachdem die SimpleCharge-Platine und das EV eine erfolgreiche Kommunikation, wie in Abschnitt 2.1.3 beschrieben, aufgebaut haben. Ein Eingreifen in den genormten Ablauf der Kommunikation und der Spannungsfreigabe kann beim EV einen Fehler hervorrufen. Wird beispielsweise die Ladefreigabe an das Schütz -Q5 zu lange unterbrochen, interpretiert dies das EV als Fehler innerhalb der Ladesäule. Um der Steuerung zu signalisieren, dass die SimpleCharge-Platine eine Ladefreigabe erteilt hat, wird das Signal mit einem Relais (-K4, Abschnitt 2.1.5.3) zu einem GPIO weitergeleitet. Somit kann der Freigabezustand überwacht werden.

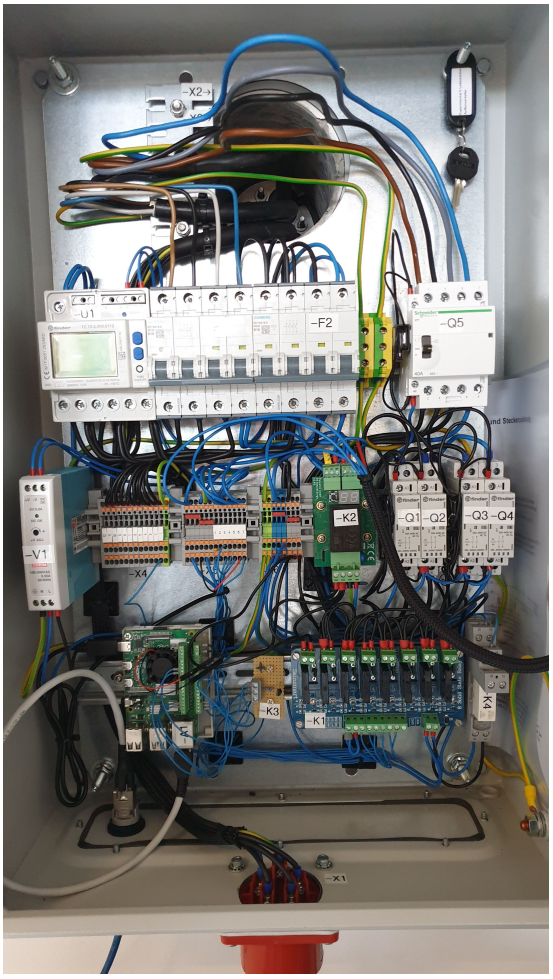
Für die Variation der maximal freigegebenen Ladestromstärke, und somit der späteren Leistungssteuerung, übermittelt die RaspberryPi-Steuerung der SimpleCharge-Platine ein PWM-Signal. Um den Eingang der SimpleCharge-Platine auf `GND`-Potential zu schalten, wird ein NPN-Transistor (-K3) mit dem PWM-Signal des RaspberryPi angesteuert. Wird dieser Transistor nicht angesteuert, wie es im erstgenannten Betriebsmodus der Fall ist, interpretiert die Platine das Signal als Freigabe des maximalen Stroms.

Alle verwendeten Bauteile sind in Abschnitt A.4 aufgeführt.

### 3.1.3 Modifizierte Ladestation

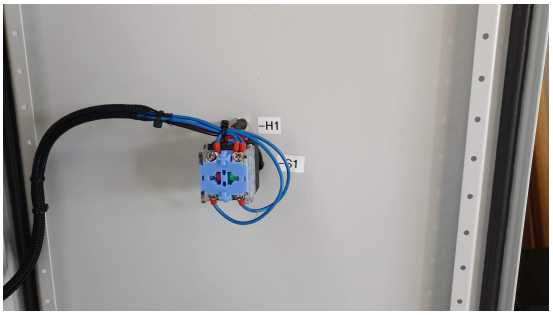
Die im Vorfeld beschriebene Hardwarekonfigurationen sind in die bereits vorhandene, experimentelle Ladestation integriert. In Abb. 3.2 wird der entgültige Aufbau dargestellt. Alle Bauteile sind für die Zuordnung mit Betriebsmittelkennzeichnungen versehen. Somit ist die Verdrahtung mit Hilfe des Schaltplans (Abschnitt A.3) leicht nachvollziehbar. Sämtliche Bauteile sind mittels Hutschienenmontage im Schaltschrank befestigt. Für den Netzwerkzugang ist eine Kabeldruchführung im Anschlussboden des Schaltschranks implementiert. Die Verbingung für den Ladeanschluss ist mit einen Durchbruch in der Wand in den Außenbereich geführt. Somit ist das Arbeiten an der Ladestation im Innenraum möglich.

Der Zugänglichkeit halber sind keine Kabelkanäle innerhalb des Schaltschrankes verbaut. Somit sind kurze Veränderungen während der Testphase möglich. Ebenfalls sind die Kabel des Lastkreises für Strommessung mit Hilfe von Messzangen zugänglich verbaut (Abb. 3.2a oben), wodurch jedoch kein zweiter Berührungsschutz vorhanden ist.




(a) Innenansicht


(b) geschlossenen Frontansicht




(c) Prgrammwahlschalter -S1

(d) externer Anschlusskasten mit Typ 2 Buchse

**Abb. 3.2:** Übersicht der gesamten experimentellen EV-Ladestation nach den beschriebenen Umbauten

### 3.1.4 Nachbau

Für den Nachbau der Ladestation in Eigenregie ist das Wissen von Elektrofachpersonal erforderlich. Es ist lebensgefährlich elektrische Anlagen solcher Art ohne das nötige Wissen anzufertigen. Zusätzlich wird im Folgenden der essentielle Aufbau beschrieben, welcher sich aus den Erfahrungen dieser Arbeit und den eigenen Gegebenheiten zusammensetzt, um die Ladestation mit den beschriebenen Funktionen nachzubilden. Die detaillierte Planung des Schaltschrankes ist der ausführenden Person überlassen.

Zunächst müssen die Anschlusswerte ermittelt werden. Im Falle dieser Arbeit ist ein 11 kW-Anschluss installiert. Für die meisten Modelle ist eine solche Ladeleistung ausreichend. Die Auswahl der Anschlussleistung beruht darauf, welche maximalen Werte der Hausanschluss zur Verfügung stellt, welche Leistung die Photovoltaikanlage erzeugen kann und welche Ladeleistung das EV umsetzen kann. Ist ein 22 kW Anschluss möglich, kann dieser mit den richtigen Leitungen und Absicherungen angebunden werden und somit ist die Ladestation auch für höhere Ladeleistungen gerüstet. Dabei werden jedoch erhebliche Mehrkosten für die Leitungen und Bauteile mit den richtigen Bemessungswerten generiert. Ist die PV-Ladung beispielsweise die höhere Priorität, kann der Anschlusswert danach dimensioniert werden und geringer ausfallen. Die gewählte Anschlussleistung muss eigenständig bei der Auslegung der nachfolgenden Bauteile berücksichtigt werden.

Der Aufbau der Ladestation ist den örtlichen Gegebenheiten anzupassen. Im Falle dieser Arbeit ist der Schaltschrank der Zugänglichkeit halber vom eigentlichen Typ-2 Anschluss getrennt. Dies ist beim Nachbau nicht notwendig. Der funktionelle Aufbau ist dem Schema in Abb. 3.3 zu entnehmen. Die genauen Anschlüsse sind im Schaltplan (Abschnitt A.3) vermerkt.

Nach der Einspeisung ist aufgrund der geltenden Vorschriften ein allstromsensitiver Differenzstrom-Schutzschalter (RCD) (-F1) zu verbauen, um auch Fehler nach dem im Fahrzeug verbauten Gleichrichter detektieren zu können. Falls ein normaler RCD vorhanden ist, kann die Gleichstromfehlerüberwachung separat verschaltet werden. Für die spätere Regelung der Leistung folgt ein Energiezähler (-U1) mit S0-Schnittstelle. Für genauere Auswertungen des Verbrauchs kann auch auf einen Energiezähler mit ModBus oder ähnlichem zurückgegriffen werden. Dies ist jedoch im verwendeten Code nicht unterstützt, muss also zusätzlich im Back-End der Steuerung eingepflegt werden und zieht erhebliche Mehrkosten mit sich. Die externe Messung der eingespeisten Solarleistung ist mit einem Energiezähler mit S0-Schnittstelle vorgesehen. Der Ablauf und die Konfiguration der Solarsteuerung ist im Abschnitt 4.3 beschrieben. Die nachfolgende Reihensicherung (-F2) ist den Anschlusswerten anzupassen. Dabei genügt eine dreipolige Sicherung mit B-Charakterisierung. Für die Ansteuerung der Phasen werden drei Leistungsschütze (-Q1...3) benötigt. Dabei werden Leistungsschütze mit den Gebrauchskategorien AC-1 oder AC-7a verwendet. Der Neutralleiter sollte nicht schaltbar sein, dies erfolgte in dieser Arbeit lediglich aus Testzwecken und ist zur Sicherheit aus dem Code entfernt. Das folgende Freigabeschütz (-Q5) ist jedoch vierpolig auszulegen, um das Ladekabel komplett vom Netz zu trennen. Ein fest verbundenes Ladekabel mit dem benötigten Querschnitt für die gewählte Ladeleistung ist zu empfehlen, da keine Ansteuerung der Steckerverriegelung seitens der Ladestation vorgesehen ist.

Der Steuerkreis unterscheidet sich beim Nachbau ebenfalls nur gering im Vergleich zur Beschreibung im vorherigen Abschnitt. Zunächst wird nach der Sicherung ein 5 V-Netzteil (-V1) für den Raspberry Pi und die Relaisplatine benötigt. Je nach verwendetem Netzteil ist auf die zusätzliche Absicherung und das Erden des Minuspols zu achten. Für die Funktionswahl zwischen gesteuertem und ungesteuertem Betrieb ist ein rastender Schalter (-S1) mit einem Schließer und einem Öffner nötig. Welche Bauform dabei verwendet wird ist unerheblich. Der Raspberry Pi 4 (-A1) dient als Steuereinheit für die gesamte Plattform. Für die Anschlüsse der verschiedenen Signale wird ein Breakoutboard benötigt. Für bessere Montage und den Schutz der Steuerung ist ein geeignetes Gehäuse empfehlenswert. Hier dient dieses zusätzlich der Hutschienenmontage. Die Netzwerkanbindung ist über W-LAN möglich, kann jedoch in metallischen Schaltschränken in der Verbindung gestört werden. Folglich ist eine Ethernetkabelverbindung stabiler. Zur Ansteuerung der Schütze und Freigabe der Steuerung werden Relais benötigt. Dabei ist aus der Erfahrung dieser Arbeit die Verwendung von elektromagnetischen Relais den SSR vorzuziehen, da diese bei der Ansteuerung keine Probleme mit zu geringen Lasten mit sich führen. Die EV-SimpleCharge Platine (-K2) dient der Kommunikation zwischen Ladestation und EV, wobei auf die korrekte Version der Firmware zu achten ist, da ältere Modelle die externe PWM-Steuerung nicht unterstützen. Um die maximale Ladeleistung des Anschlusses nutzen zu können wird der passenden Codierwiderstand gemäß Tab. 2.1 verbaut. Das PWM-Signal vom Raspberry Pi zur SimpleCharge Platine wird über einen NPN-Transitor (-K3) verschaltet. Für die Abfrage des Freigabezustandes wird ebenfalls ein Relais (-K4) verwendet. Dabei ist auf die geringe geschaltete Spannung von 3,3 V zu achten. Die genaue Verdrahtung ist wiederum dem Schaltplan zu entnehmen.

Nachdem die Hardware vorbereitet sowie deren Funktion sichergestellt ist, kann die Software auf dem Raspberry Pi installiert werden. Dies wird im Abschnitt 3.2.2 genau beschrieben. Der maximale Ladestrom, welcher von der SimpleCharge Platine freigegeben wird, muss nach der Erstinbetriebnahme an der Platine manuell von Hand eingestellt werden. Die Bedienung der Platine sollte dem zugehörigen Datenblatt entnommen werden. Die Funktion des ungesteuerten Ladebetriebs ist auch ohne die Software möglich.

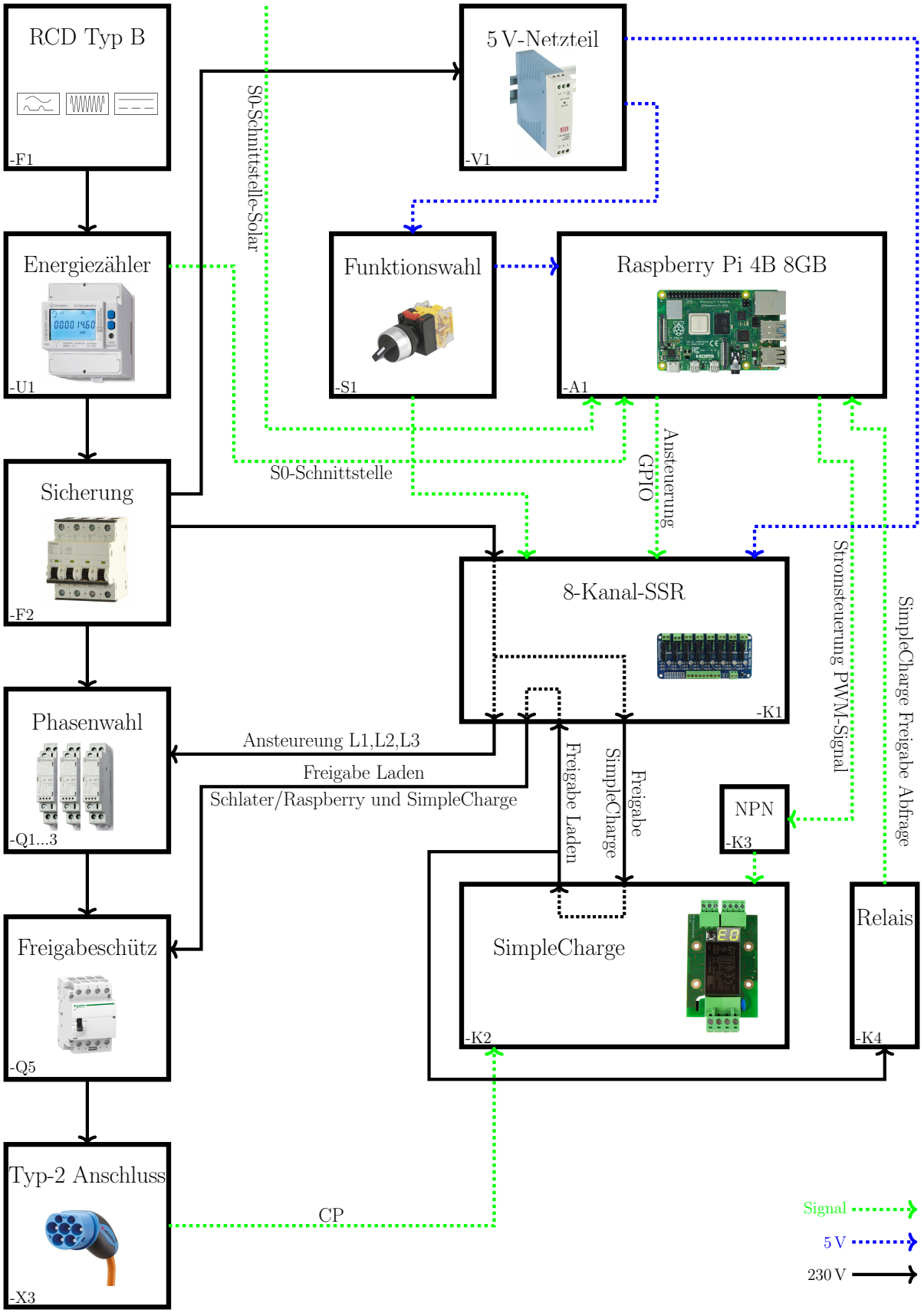



**Abb. 3.3:** Schematische Darstellung des Hardwareaufbaus für den Nachbau der Ladestation

## 3.2 Software

In diesem Abschnitt wird die Software, welche der Steuerung der in Abschnitt 3.1 beschriebenen Hardware der Ladestation dient, beschrieben. Hierzu wird zunächst auf die Projektstruktur im Allgemeinen eingegangen und anschließend die einzelnen Komponenten erklärt. Die Software wurde in der IDE PyCharm, beschrieben in Abschnitt 2.2.1, erstellt. Die für das Projekt verwendeten Programmiersprachen sind in Abschnitt 2.2 beschrieben.

### 3.2.1 Projektstruktur und -inhalt

Die Software des Projekts besteht aus den in Abb. 3.4 a) gezeigten zwei Teilen, einem Skript zur Erzeugung der Nutzerdatenbank sowie dem Webserver, welcher auf dem in Abschnitt 2.1.1 beschriebenen Raspberry Pi aufgespielt ist und die Steuerung der Ladestation übernimmt. Die Struktur der Unterordner des Webservers ist Abb. 3.4 b) zu entnehmen. In diesem Abschnitt wird auf die Funktion der einzelnen Komponenten eingegangen, jedoch auf eine detaillierte Erklärung des Inhalts der einzelnen Dateien verzichtet, da der in diesen Dateien enthaltene Code bereits ausführlich kommentiert ist.

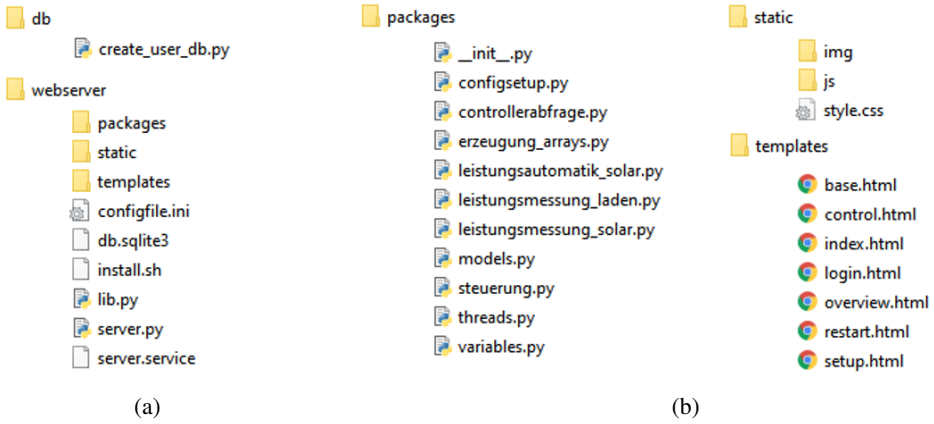



**Abb. 3.4:** Übersicht der Projektstruktur des gesamten Projektes (a) sowie der Unterstruktur des Webservers (b)

#### Datenbank

Das Skript `create_user_db.py` dient der Erzeugung, falls noch keine Datei mit dem Namen `db.sqlite3` existiert, bzw. Erweiterung der Nutzerdatenbank, falls die Datei `db.sqlite3` bereits existiert. Das Passwort wird mittels kryptologischer Hashfunktion, secure hash algorithm (SHA), SHA-256-verschlüsselt.

#### Webserver

Der Ordner `webserver`, dessen Struktur in Abb. 3.4 (a) abgebildet ist, enthält alle nötigen Komponenten der Webapplikation, welche auf dem Raspberry Pi läuft und die Steuerung der Ladestation übernimmt. Der Inhalt der drei Unterordner, dargestellt in Abb. 3.4 (b), und der weiteren Dateien im Ordner `webserver` wird im Folgenden näher erläutert.

**Packages**

Der Unterordner `packages` enthält zahlreiche für den Webserver selbst geschriebene Skripten, deren Funktion im Folgenden beschrieben werden. Der Inhalt der Skripten ist kommentiert, weshalb im Folgenden nur die wichtigsten Komponenten näher erläutert werden.

Das Skript `__init__.py` ist leer und könnte somit weggelassen werden. Würde die Datei genutzt werden, könnte mit deren Hilfe festgelegt werden, was bei einem Import des Ordners `packages` in ein anderes Python-Skript aus diesem importiert werden soll. Ist die Datei leer, so wird beim Import des Ordners alles importiert, was dieser enthält.

Im Skript `configsetup.py` sind die Funktionen `defaultconfiguration`, `updateconfiguration` und `load_configuration` definiert, welche der Interaktion des Webservers mit der Konfigurationsdatei `configfile.ini` dienen. Das Ausführen von `defaultconfiguration` über die Startkonfigurationsseite (Abb. 3.16) beschreibt die Konfigurationsdatei mit der in Tab. 3.3 aufgeführten Standardkonfiguration. Die Funktion `updateconfiguration`, ebenfalls über die Startkonfigurationsseite ausführbar, beschreibt die Konfigurationsdatei mit neuen Werten. Die Funktion `load_configuration` wird beim Start des Webservers ausgeführt. Dabei wird die Konfigurationsdatei eingelesen und alle relevanten Parameter auf die in der Konfigurationsdatei festgelegten Werte gesetzt.

Das Skript `controllerabfrage.py` enthält die Funktion `control_status`, über welche der Status des Ladecontrollers (-K2) mit Hilfe eines Relais (-K4) abgefragt und auf der Übersichtsseite (Abb. 3.14) als `Controllerladefreigabe` bzw. auf der Steuerungsseite (Abb. 3.17) als `Freigabe` angezeigt wird.

Im Skript `erzeugung_arrays.py` ist die Funktion `erzeugung_arrays` definiert. Mit deren Hilfe werden die in Anhang B.2 aufgeführten Stromwerte und möglichen Leistungskonfigurationen basierend auf der Ladespezifikation in der Konfigurationsdatei bzw. der ausgewählten Ladespezifikation auf der Steuerungsseite (Abb. 3.17) erzeugt. Das Skript ist so programmiert, dass bei doppelten Leistungskombinationen, wie z.B. einphasiges Laden mit 12 A oder zweiphasiges Laden mit 6 A, welche beide einer Ladeleistung von 2760 W entsprechen, die Kombination mit niedrigerem Strom beibehalten und die weiteren Duplikate aus der Konfigurationsmatrix entfernt werden.

Das Skript `leistungsautomatik_solar.py` enthält die Funktion `solarsteuerung`, welche in einem der beiden später erwähnten Threads periodisch ausgeführt wird. Der Zeitabstand zwischen zwei Wiederholungen entspricht dem Wert des Parameters `Stromumschaltzeit` auf der Startkonfigurationsseite (Abb. 3.16) und wird auf der Steuerungsseite im Steuermodus `Solarautomatik` (Abb. 3.20) als `Intervall Strom` angezeigt. Der Standardwert nach Tab. 3.3 beträgt 30 s. Die Zeit zwischen zwei Umkonfigurationen der Phasen wird über den Parameter `Phasenumschaltzeit` auf der Startkonfigurationsseite festgelegt, in der Solarsteuerung als `Intervall Phase` angezeigt und beträgt standardmäßig 300 s. Für den Fall, dass die Steuerung der Ladestation nicht über die Solarautomatik erfolgt, wird der Inhalt des Skripts übersprungen. Dabei ist zu erwähnen, dass bei einer Erstausführung der Solarautomatik, also bei Start des Webservers, Freigabe des Ladens durch den Benutzer oder einem Wechsel in den

Steuermodus `Solarautomatik` der Timer für eine Umkonfiguration auf Null gesetzt wird und somit direkt umkonfiguriert werden darf. Eine genaue Analyse sowie Validierung der Solarautomatik erfolgt in Abschnitt 4.3.

Die Skripten `leistungsmessung_laden.py` und `leistungsmessung_solar.py` enthalten jeweils die Funktion `pulse`, über welche die aktuelle Ladeleistung $P_{lade,ist}$ bzw. Solarleistung $P_{solar,ist}$ sowie die seit Serverstart geladene Energiemenge $E_{lade}$ bzw. durch die Solaranlage produzierte Energiemenge $E_{solar}$ ermittelt wird. Die Ermittlung dieser Größen erfolgt über die S0-Schnittstellen der Energiezähler für Ladestation bzw. Solaranlage nach Gl.(2.1.4). Anzumerken ist, dass bei der Ermittlung der Leistungsgrößen fünf Werte gemittelt werden. Hintergrund ist, dass die Energiemessung des Energiezählers vorrangig gegenüber der Erzeugung der S0-Impulse ist und somit die Zeitabstände zwischen den Impulsen selbst bei gleichbleibender Leistung schwanken, was durch das Mitteln geringer ausfällt.

Im Skript `models.py` ist festgelegt, wie ein Nutzerprofil innerhalb der Nutzerdatenbank `db.sqlite3` aufgebaut ist und wird für den Authentifizierungsprozess sowie das Erzeugen bzw. Erweitern der Nutzerdatenbank benötigt.

Über das Skript `steuerung.py` ist die Funktion `steuerung` definiert, welche den gesamten Ablauf der Steuerung der Ladestation über die Steuerungsseite ermöglicht (Abb. 3.17 bis Abb. 3.20). Eine genaue Analyse sowie Validierung der Steuerung erfolgt in Abschnitt 4.1.

Im Skript `threads.py` sind die beiden Klassen `PerpetualTimer1` und `PerpetualTimer2` sowie die Funktion `plotdata` definiert. Letztere speichert bei Ausführung die aktuellen Werte für Solarleistung sowie vorgegebener und aktueller Ladeleistung ab und bereitet diese für die grafische Darstellung auf der Übersichtsseite (Abb. 3.15) auf. Die beiden Timer dienen der periodischen Wiederholung der Funktion `solarsteuerung` mit der vorher erwähnten Wiederholzeit sowie der eben erklärten Funktion `plotdata` mit einer festen Wiederholzeit von 30 s.

Das Skript `variables.py` enthält einen Großteil der globalen Variablen für das gesamte Projekt, wie beispielsweise Pinzuweisungen. Darüber hinaus enthält es die Funktion `gpio_setup`, welche beim Start des Webservers ausgeführt wird und über welche die digitalen Ausgänge des Raspberry Pi als solche deklariert werden.

**Static**

Der Order `static` enthält für das Frontend bereitgestellte Inhalte. Dazu gehören zum einen angezeigte Logos und Bilder, welche im Unterordner `img` abgelegt sind. Zum anderen befinden sich im Ordner `js` die für dieses Projekt verwendeten Javascript-Bibliotheken. Es wäre auch ein Online-Verweis im Kopf des HTML-Templates zu den Javascript-Bibliotheken möglich, sollte sich dieser Link aber ändern, könnte über den alten Link nicht mehr online auf die Bibliothek zugegriffen werden und die damit verbundenen Inhalte des Webinterfaces würden nicht mehr funktionieren. Des Weiteren enthält der Ordner `static` ein in Abschnitt 2.2.4 beschriebenes Stylesheet `style.css` basierend auf dem Open-Source-Framework Bulma [25].

### Templates

Im Unterordner `templates` befinden sich die HTML-Dokumente für die einzelnen Seiten des Webinterfaces. Der generelle Aufbau eines HTML-Dokuments ist Abschnitt 2.2.4 zu entnehmen. Das Dokument `base.html` enthält das Grundgerüst der Seite, wie beispielsweise die `head`-Sektion, in welcher unter anderem die Javascript-Bibliotheken geladen werden oder die Formatierung der Navigationsleiste. Die restlichen HTML-Dokumente enthalten, aufbauend auf `base.html`, die Formatierung der einzelnen Seiten, wobei `control.html` die Steuerungsseite, `index.html` die Startseite, `login.html` die Login-Seite, `overview.html` die Übersichtsseite, `restart.html` die Neustartseite und `setup.html` die Startkonfigurationsseite enthält.

### Config, liby, db, install

Die Dateien `configfile.ini`, `lib.py`, `db.sqlite3` und `install.sh` sind Hilfsdateien. Die Datei `configfile.ini` enthält die Startkonfiguration, welche beim Start des Webservers aus dieser geladen wird und über die Startkonfigurationsssseite bearbeitet werden kann. Die Datei `lib.py` enthält alle für dieses Projekt verwendeten Python-Bibliotheken und wird am Anfang jedes anderen Python-Skripts importiert. Die Datei `db.sqlite3` ist die über das Skript zum Anlegen eines Nutzers erzeugte Nutzerdatenbank und dient der Verifizierung von Nutzern bei der Authentifizierung. Die Datei `install.sh` ist das Installationsskript des Webservers, welches bei Ausführen alle nötigen Python-Bibliotheken installiert und den Autostart des Webservers einrichtet.

### Server

Die Datei `server.py` enthält das Grundgerüst des Webservers. Dazu gehören das Laden der Nutzerdatenbank, der Authentifizierungsprozess, die Deklaration aller Routen und der dazugehörigen Funktionen sowie die Initialisierung der Threads für Solarautomatik und Leistungsdatenaktualisierung.

### Autostart

Um den Webserver automatisch nach dem Bootvorgang zu starten, beispielsweise nach einem Stromausfall oder Fehlerfall, muss dies dem System- und Sitzungsmanager *systemd* mitgeteilt werden. Dies erfolgt grundlegend über eine Datei mit der Endung `.service`. Die in diesem Fall verwendete `server.service` Datei ist im Abschnitt B.1, mit kurzen Kommentaren zu den Einträgen aufgeführt. Um den Autostart im Gesamten einzurichten sind die folgenden Schritte notwendig (die aufgeführten Eingaben mit dem vorangestellten $ -Zeichen müssen in der Shell des Raspberry Pi eingegeben werde):

1. Zunächst muss die Datei server.service im richtigen Verzeichnis erstellt werden. Da bei manchen Linuxinstallationen das Verzeichnis `/.local/share/systemd/user` nicht existiert, wird es mit dem folgenden Befehl erstellt, falls es nicht vorhanden ist.

   ```
   $ mkdir -p ~/.local/share/systemd/user
   ```

2. Nach dem Erstellen muss in das Verzeichnis gewechselt werden.

   ```
   $ cd ~/.local/share/systemd/user
   ```

3. Nun kann die Datei `server.service` erstellt werden.

   ```
   $ sudo touch server.service
   ```

4. Um diese zu editieren, wird sie mit einem beliebigen Texteditor geöffnet. (In diesem Fall `nano`)

   ```
   sudo nano server.service
   ```

5. Der für die `server.service` hervorgesehene Inhalt, Abschnitt B.1, ist hier einzufügen. Mit der Tastenkombination Str + X $\rightarrow$ Y $\rightarrow$ Return werden die Änderungen gespeichert und der Editor geschlossen.

6. Um den Manager nun den neuen Service mitzuteilen und zu aktivieren sind die folgenden Eingaben vorgesehene.

   ```
   $ systemctl --user daemon-reload (Erneuert die Liste der Services)

   $ systemctl --user enable server.service (aktiviert den neuen Service)
   ```

7. Da es sich um einen User Service handelt (Rootrechte sind für einen Server nicht ratsam), muss dieser noch automatisch nach dem Hochfahren angemeldet werden. In diesem Fall ist es der Benutzer `pi`.

   ```
   $ sudo loginctl enable-linger pi
   ```

8. Der Service kann nun mit Eingaben direkt angesteuert werden oder er wird nach einem Neustart ausgeführt.

   ```
   $ systemctl --user start server.service

   $ systemctl --user stop server.service (Service manuell starten oder stoppen)

   $ sudo reboot (Rasoberry Pi neustarten)
   ```

9. Die Konsolenausgaben bzw. Fehlermeldungen des Servers werden in einen Journal gespeichert. Der Status des Service und die letzten Konsolenausgaben oder das gesamte Journal des Service können abgerufen werden.

   ```
   $ systemctl --user status server.service

   $ journalctl --user-unit server.service
   ```

Möglicherweise treten bei der Anwendung des Autostarts Probleme mit Netzwerkrechten auf. Somit muss entweder ein Portforwarding im Router konfiguriert werden oder, wie im Falle dieser Arbeit, die

Einstellung des Flask-Servers beim Start auf die IP-Adresse `0.0.0.0` und den Port `5000` gesetzt werden. Diese Einstellung gibt dem Server vor, jedem Port zu lauschen und zu senden. Sollten dadurch Ports unsicher werden, sind diese über eine Firewall zu blockieren.

### 3.2.2 Erstinbetriebnahme

Um die in Abschnitt 3.1 detailliert beschriebene Hardware erstmals in Betrieb zu nehmen, sind einige Schritte nötig, welche im folgenden Abschnitt beschrieben werden. Die für die Erstinbetriebnahme benötigte Software sind ein Tool zum Formatieren der Speicherkarte im FAT32-Dateisystem, ein Tool zum Aufspielen des Betriebssystems, Python 3 sowie eine Software, welche das Herstellen einer SSH-Verbindung zum Raspberry Pi und das Transferieren von Daten zwischen PC und Raspberry Pi ermöglicht. Die hier verwendete Software ist nachfolgender Tabelle zu entnehmen.

**Tab. 3.1:** Verwendete Software zur Erstinbetriebnahme des Webservers

| Verwendung | Software | Version | Hersteller |
|---|---|---|---|
| Formatieren FAT32 | Raspberry Pi Imager | v1.5 | Raspberry Pi Foundation |
| Aufspielen Image | Raspberry Pi Imager | v1.5 | Raspberry Pi Foundation |
| SSH-Verbindung | Bitvise SSH Client | v8.43 | Bitvise Limited |
| Datentransfer | Bitvise SSH Client | v8.43 | Bitvise Limited |
| Python | Python 3 | v3.9.1 | Python Software Foundation |

Zunächst muss die SD-Karte im FAT32-Dateisystem formatiert werden. Hierfür wird der Raspberry Pi Imager verwendet, dessen Benutzeroberfläche in Abb. 3.5 dargestellt ist. Unter dem Menüpunkt `Betriebssystem` wird die Option `Löschen` ausgewählt. Im Zweiten Punkt wird die zu formatierende SD-Karte gewählt und mit `SCHREIBEN` bestätigt. Ist die SD-Karte fertig formatiert wird als nächstes im ersten Menüpunkt die Option `Raspberry Pi OS(32-bit)` ausgewählt und wie im vorherigen Schritt bestätigt. Dieser Vorgang nimmt etwas Zeit in Anspruch, da das Image während des Schreibens heruntergeladen wird. Sobald das Betriebssystem fertig geschrieben ist wird als nächstes die Boot-Partition in der Laufwerksübersicht des PCs geöffnet und dort eine leere Datei mit dem Namen `ssh` platziert. Wichtig ist, dass die Datei keine Dateiendung besitzt. Das Platzieren dieser Datei führt beim Start des Raspberry Pi dazu, dass die Verbindung des PCs mit dem Raspberry Pi via SSH freigegeben wird.


**Abb. 3.5:** Benutzeroberfläche des Raspberry Pi Imager

Als nächstes wird die Speicherkarte in den Raspberry Pi eingelegt, der Raspberry Pi über ein Ethernet-Kabel mit dem Heimnetz verbunden und anschließend gestartet. Über die Software Bitvise SSH Client, deren Benutzeroberfläche mit den nötigen Daten zum Login in Abb. 3.6 dargestellt ist, wird eine SSH-Verbindung zwischen PC und Raspberry Pi aufgebaut. Die nötigen Nutzerdaten sind `pi` als Nutzername, `raspberry` als Passwort, `raspberrypi.local` als Hostname und 22 als Port für die SSH-Verbindung, welche jetzt über `Log In` hergestellt werden kann.

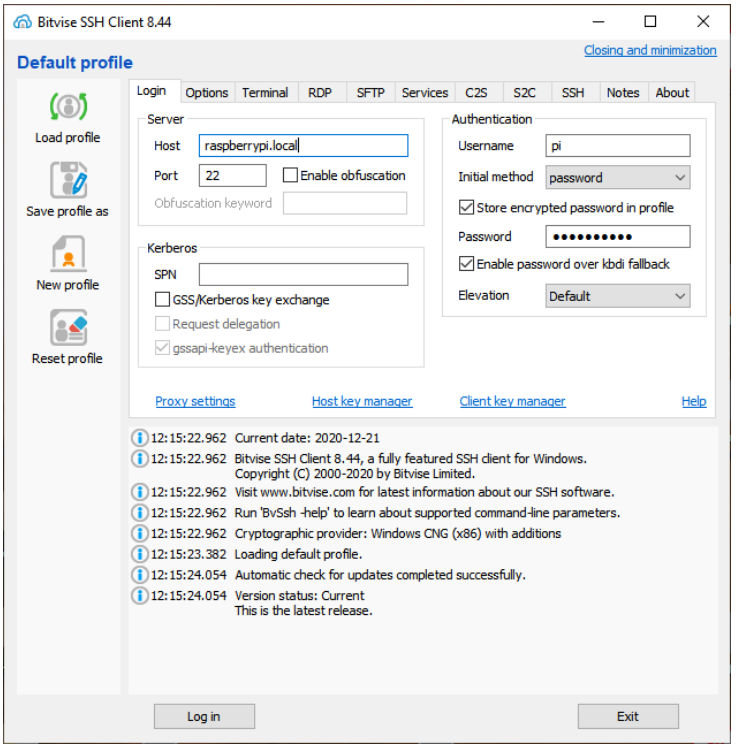



**Abb. 3.6:** Benutzeroberfläche des Bitvise SSH Clients mit nötigen Daten zur Erstkonfiguration des Raspberry Pi 4 mittels SSH-Verbindung

Jetzt kann dem Raspberry Pi eine statische IP-Adresse zugewiesen werden. Dazu muss zunächst die IP-Adresse des Routers im Heimnetz bekannt sein. Dafür empfiehlt es sich in den Router einzuloggen. Die statische IP-Adresse des Raspberry Pi sollte auf jeden Fall außerhalb der Reichweite für die automatische Zuweisung von IP-Adressen durch den Router im Heimnetz liegen. In diesem Beispiel ist die IP-Adresse des Routers `192.168.1.1` und für den Raspberry Pi wird die IP-Adresse `192.168.1.100` gewählt. Um dem Raspberry Pi jetzt die statische IP-Adresse zuzuweisen wird im Terminal des Bitvise SSH Clients nach erfolgreichem Aufbau einer SSH-Verbindung der Befehl `sudo nano /etc/dhcpcd.conf` eingegeben, wodurch die Datei `dhcpcd.conf` geöffnet wird. Mit den Pfeiltasten wird in der Datei nach unten navigiert, bis der Ausschnitt aus Abb. 3.7 erscheint. Mit den beiden IP-Adressen aus dem Beispiel wird dieser Ausschnitt gemäß Abb. 3.7 verändert. Mit der Tastenkombination `STRG + X` und anschließendem Bestätigen mit `Y` und Drücken der Enter-Taste werden die Änderungen gespeichert und die Datei geschlossen. Anschließend wird der Raspberry Pi neu gestartet und wieder eine SSH-Verbindung aufgebaut, wobei jetzt als Hostname die neue IP-Adresse angegeben werden muss.

```
# Example static IP configuration:
interface eth0
static ip_address=192.168.1.100/24
#static ip6_address=fd51:42f8:caae:d92e::ff/64
static routers=192.168.1.1
static domain_name_servers=192.168.1.1
```

**Abb. 3.7:** Exemplarisches Ethernet-Profil für die Zuweisung einer statischen IP-Adresse

Im Terminal des Bitvise SSH Clients wird als nächstes der Befehl `sudo apt-get update && sudo apt-get upgrade` eingegeben, welcher ein Update des Betriebssystems sowie aller zusätzlichen Pakete zur Folge hat. Nach kurzer Wartezeit muss mit `Y` bestätigt werden und das Update startet und kann, je nach Internetverbindung, einiges an Zeit in Anspruch nehmen. Wenn das Update beendet ist wird der Raspberry Pi nochmals neu gestartet und eine SSH-Verbindung aufgebaut. Im Terminal wird der Befehl `sudo raspi-config` eingegeben, welcher das `Raspberry PI Software Configuration Tool`, dessen Benutzeroberfläche in Abb. 3.8 zu sehen ist, öffnet. Die Navigation erfolgt über die Pfeiltasten. Zunächst wird das Tool über Menüpunkt `8` geupdated. Danach wird empfohlen, unter Menüpunkt `1` das Nutzerpasswort zu ändern. Das neue Passwort wird ab jetzt auch zum Aufbau einer SSH-Verbindung anstatt des Standardpassworts `raspberry` benötigt. Unter Menüpunkt `5` können Zeitzone, Tastaturlayout und weiteres eingestellt werden. Zum Schluss wird noch über Menüpunkt `6 - A1` das Dateisystem erweitert. Über `Finish` werden die Eingaben bestätigt und der Raspberry Pi anschließend neu gestartet.

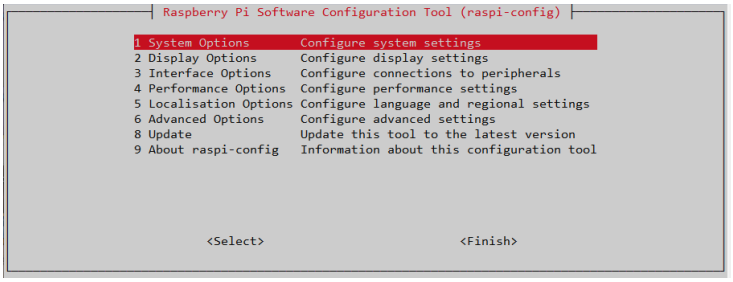



**Abb. 3.8:** Startseite des raspi-config Tools

Als Nächstes muss die Nutzerdatenbank erstellt werden. Dazu wird die Datei `create_user_db.py` im Order `db` mit Python 3 ausgeführt und es öffnet sich das in Abb. 3.9 dargestellte Fenster. Jetzt können die für den Login des Webservers gewünschten Nutzerdaten eingegeben und mit `Setup` bestätigt werden. Anschließend wird das Programm mit `Exit` beendet. Anzumerken ist, dass das Programm eine neue Datenbank erstellt, wenn keine im selben Verzeichnis existiert, oder eine vorhandene Datenbank um die eingegebenen Nutzerdaten erweitert. Das Passwort wird via SHA256 verschlüsselt. Die erzeugte Datei `db.sqlite3` wird kopiert und in den Ordner `webserver` eingefügt. Anschließend wird erneut eine SSH-Verbindung aufgebaut und das Fenster für den Datentransfer im Bitvise SSH Client, siehe Abb. 3.10, aufgerufen. Der Ordner `webserver` wird jetzt in das Verzeichnis `/home/pi` auf den Raspberry Pi kopiert. Wichtig ist, dass das richtige Verzeichnis gewählt wird und der Name des Ordners nicht verändert wird.

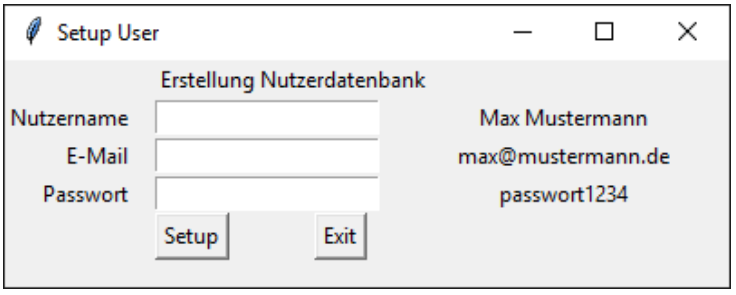



**Abb. 3.9:** Nutzeroberfläche des Datenbankskripts zur Erzeugung bzw. Erweiterung der Nutzerdatenbank

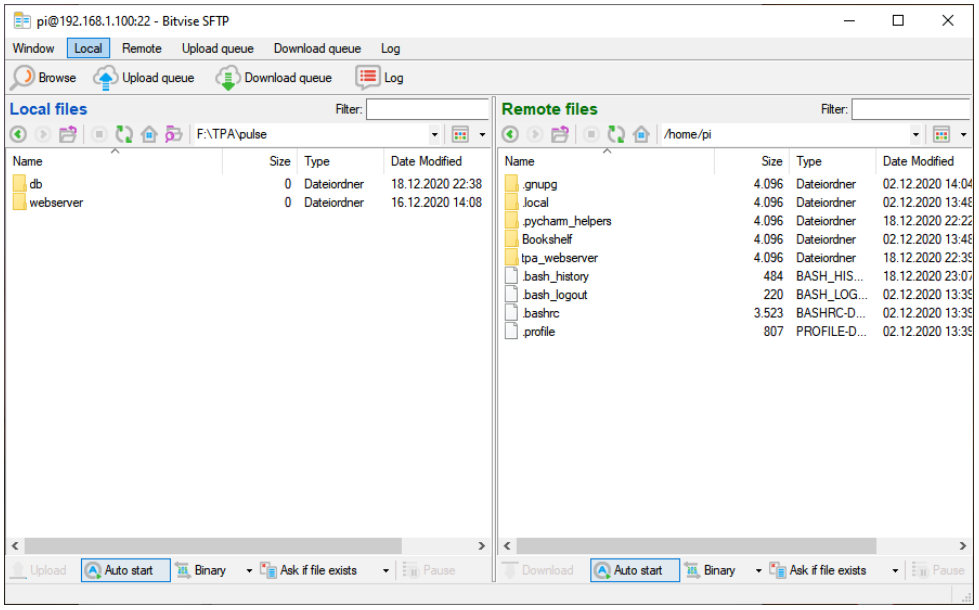



**Abb. 3.10:** SFTP-Terminal des Bitvise SSH Clients zum Aufspielen des Webservers

Im Terminal wird mit dem Befehl `cd webserver` in den Ordner `webserver` gewechselt und über die Eingabe von `bash install.sh` die Installationsdatei `install.sh` ausgeführt. Dadurch werden alle nötigen Pakete für den Webserver installiert sowie der in Abschnitt 3.2.1 beschriebene Autostart des Webservers implementiert. Wird der Raspberry Pi jetzt neu gestartet, sollte der Webservers von selbst anlaufen und unter der statischen IP-Adresse des Raspberry Pi mit dem Port 5000 von jedem Gerät im Heimnetz erreichbar sein. Für dieses Beispiel wäre der Server im Browser über die Adresse `192.168.1.100:5000` zu erreichen.

### 3.2.3 Benutzeroberfläche

Der Webserver ist jetzt für das Beispiel aus Abschnitt 3.2.2 unter der Adresse `192.168.1.100:5000` erreichbar, bei deren Eingabe in einem Webbrowser sich die Startseite aus Abb. 3.11 öffnet. Als nicht eingeloggter Nutzer stehen in der Navigationsseite im oberen, rechten Teil des Webinterfaces die Menüpunkte `Home`, `Übersicht` und `Login` zur Verfügung.

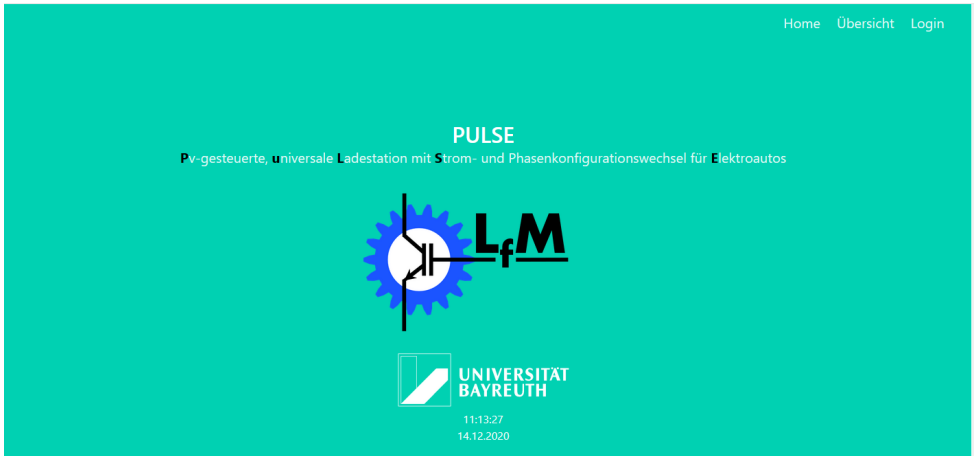



**Abb. 3.11:** Startseite im ausgeloggten Zustand

Der Login im Webinterface, Abb. 3.12 erfolgt über die Nutzerdaten `E-Mail` und `Passwort`, welche mit der in der Datenbank hinterlegten Nutzerdaten abgeglichen werden. Bei falscher Eingabe erscheint der Hinweis, die Nutzerdaten zu überprüfen, bei richtiger Eingabe wird der Nutzer eingeloggt und auf die Startseite weitergeleitet. Anzumerken ist, dass nach einer Inaktivität von zehn Minuten, beispielsweise beim Schließen des Browsers ohne Logout, der Nutzer automatisch abgemeldet wird.

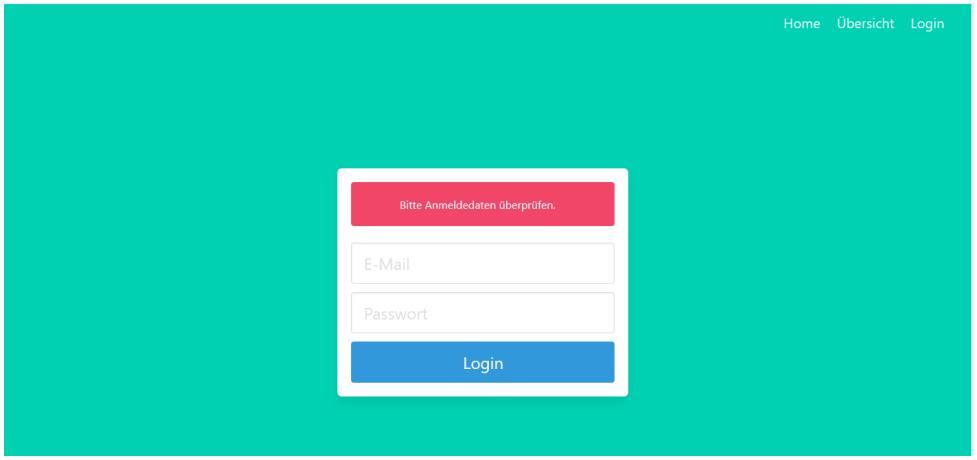



**Abb. 3.12:** Login-Seite

Im eingeloggten Zustand stehen in der Navigationsleiste die Menüpunkte `Home`, `Übersicht`, `Steuerung`, `Startkonfiguration`, `Logout` und `Neustart` zur Verfügung, wie in Abb. 3.13 zu sehen. Die Menüpunkte `Steuerung`, `Startkonfiguration`, `Logout` und `Neustart` durch die Login-Authentifizierung

geschützt sind. Über den Menüpunkt `Logout` wird der aktuelle Nutzer ausgeloggt und auf die Startseite weitergeleitet. Über den Menüpunkt `Neustart` wird der Webserver ohne Vorwarnung neu gestartet, um beispielsweise Änderungen in den Startkonfigurationen zu übernehmen. Alle aktiven Nutzer werden bei einem Neustart **nicht** ausgeloggt.

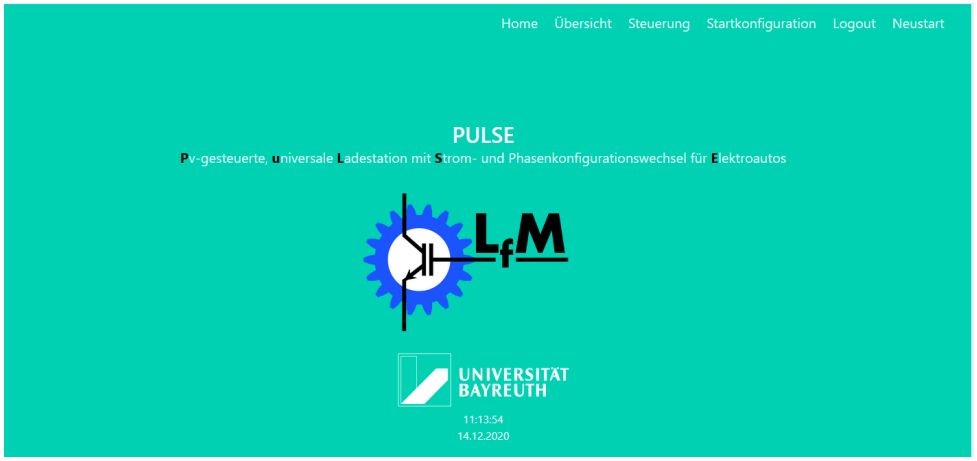



**Abb. 3.13:** Startseite im eingeloggten Zustand

Die Übersichtsseite, welche auch im ausgeloggten Zustand einsehbar ist, ist zweigeteilt und dient der Anzeige des aktuellen Betriebszustandes der Ladestation und hat keinerlei Einfluss auf die Steuerung. Der obere Teil, zu sehen in Abb. 3.14, zeigt den Zustand sämtlicher für den Betrieb relevanter Parameter. Eine Erklärung aller einzelnen Elemente ist Tab. 3.2 zu entnehmen.

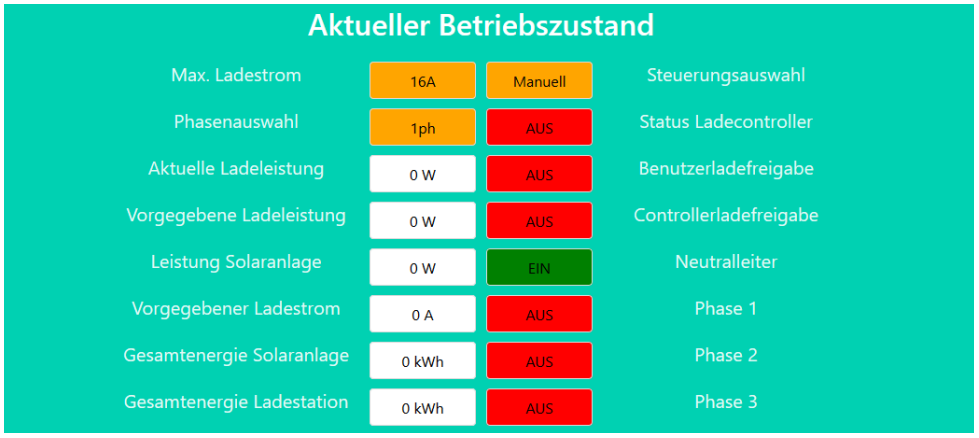



**Abb. 3.14:** Übersichtsseite: Statusanzeigen

**Tab. 3.2:** Übersicht aller Steuerparameter des Webservers

| Parameter | Varianten | Erklärung |
|---|---|---|
| Max. Ladestrom bzw. | 16 A | Maximaler Ladestrom auf 16 A begrenzt |
| Ladespezifikation | 32 A | Maximaler Ladestrom auf 32 A begrenzt |
| Phasenauswahl bzw. | 1ph | Nur einphasiges Laden |
| Ladespezifikation | 2ph | Nur zweiphasiges Laden |
| | 3ph | Nur dreiphasiges Laden |
| | 1ph/2ph | Ein- oder zweiphasiges Laden |
| | 1ph/3ph | Ein- oder dreiphasiges Laden |
| | 1ph - 3ph | Ein-, zwei- oder dreiphasiges Laden |
| Aktuelle Ladeleistung $P_{\text{lade,ist}}$ | - | Anzeige aktuelle Ladeleistung in W |
| Vorgegebene Ladeleistung $P_{\text{lade,soll}}$ | - | Anzeige vorgegebene Ladeleistung in W |
| Leistung Solaranlage $P_{\text{solar}}$ | - | Anzeige aktuelle Solarleistung in W |
| Vorgegebener Ladestrom | - | Anzeige vorgegebener Ladestrom in A |
| Gesamtenergie Soloaranlage | - | Anzeige Solarenergie seit Serverstart in kW h |
| Gesamtenergie Ladestation | - | Anzeige Ladeenergie seit Serverstart in kW h |
| Steuerungsauswahl | Manuell | Manuelle Steuerung (Testbetrieb) |
| | Leistungsvorgabe | Betrieb mit Leistungsvorgabe |
| | Solarautomatik | Betrieb mit Solarautomatik |
| Status Ladecontroller | An | Ladecontroller eingeschaltet |
| | Aus | Ladecontroller ausgeschaltet |
| Benutzerladefreigabe | Ein | Laden durch Benutzer freigegeben |
| | Aus | Laden durch Benutzer gesperrt |
| Controllerladefreigabe bzw. | Ein | Laden durch Controller freigegeben |
| Freigabe | Aus | Laden durch Controller gesperrt |
| Neutralleiter | Ein | Steuerschütz geschlossen |
| Phase 1 | Ein | Steuerschütz geschlossen |
| | Aus | Steuerschütz geöffnet |
| Phase 2 | Ein | Steuerschütz geschlossen |
| | Aus | Steuerschütz geöffnet |
| Phase 3 | Ein | Steuerschütz geschlossen |
| | Aus | Steuerschütz geöffnet |
| Eigenverbrauch | - | Berücksichtigter Eigenverbrauch in Solarbetrieb |
| Intervall Strom | - | Pausenzeiten Stromänderung in Solarbetrieb |
| Intervall Phase | - | Pausenzeiten Phasenumkonfiguration in Solarbetrieb |
| S0-Schnittstelle Laden | - | Impulse pro kW h Ladeleistungszähler |
| S0-Schnittstelle Solar | - | Impulse pro kW h Solarleistungszähler |

Der untere Teil, dargestellt in Abb. 3.15, bildet die Verläufe von vorgegebener Ladeleistung (blau) sowie aktueller Lade- (orange) und Solarleistung (grün) grafisch ab. Anzumerken ist, dass der Graph bei jedem Neustart des Webserver zurückgesetzt wird und maximal die letzten sechs Stunden abbildet. Die Leistungswerte werden alle 30 Sekunden aufgezeichnet. Die Grafik kann über das Bedienfeld im oberen Teil, welches durch Berühren der Grafik mit dem Mauszeiger sichtbar wird, angepasst und auch abgespeichert werden. Zudem ist es möglich, die Leistungswerte, aus welchen der Graph erstellt wurde, über den Button `Leistungsverläufe herunterladen` als Textdatei zu exportieren. Dabei wird für jeden der drei Leistungsverläufe ein Textdokument erstellt, welches die Leistungswerte als Integer getrennt durch Komma enthält.

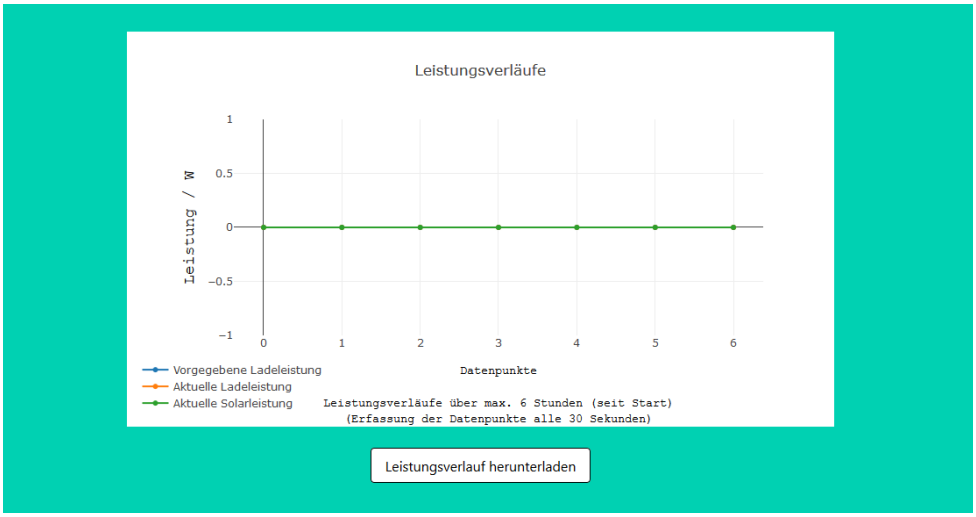



**Abb. 3.15:** Übersichtsseite: Leistungsverläufe

Die Startkonfigurationsseite, dargestellt in Abb. 3.16, ist ebenfalls nur als eingeloggter Nutzer erreichbar. Über diese Seite kann die Startkonfiguration, welche die Ladestation bei Start des Webservers übernimmt, geändert oder zurückgesetzt werden. Die beim Aufruf der Seite anzeigten Einstellungen entsprechen den Standardeinstellungen und sind in Tab. 3.3 zusammengefasst. Die Erklärung der einzelnen Optionen ist Tab. 3.2 zu entnehmen. Über die Schaltfläche `Standard` wird die Startkonfiguration auf Standardeinstellungen zurückgesetzt und über die Schaltfläche `Übernehmen` werden die aktuell eingegebenen Einstellungen als neue Startkonfiguration übernommen, sofern die Eingabe gültig ist. Bei Eingabe einer eigenen Startkonfiguration sind die Hinweise im unteren Bereich der Seite zu beachten. Ist die Eingabe fehlerhaft, so wird dies im oberen Bereich durch eine rote Fehlermeldung mitgeteilt. Ist die Eingabe korrekt erscheint die in Abb. 3.16 zu sehende, grüne Bestätigungsmeldung, dass die neuen Einstellungen übernommen wurden. Durch einen Neustart wird die geänderte Startkonfiguration übernommen. Anzumerken ist, dass auch nach Änderung der Startkonfiguration immer die Standardeinstellungen bei Aufruf der Startkonfigurationsseite angezeigt werden.

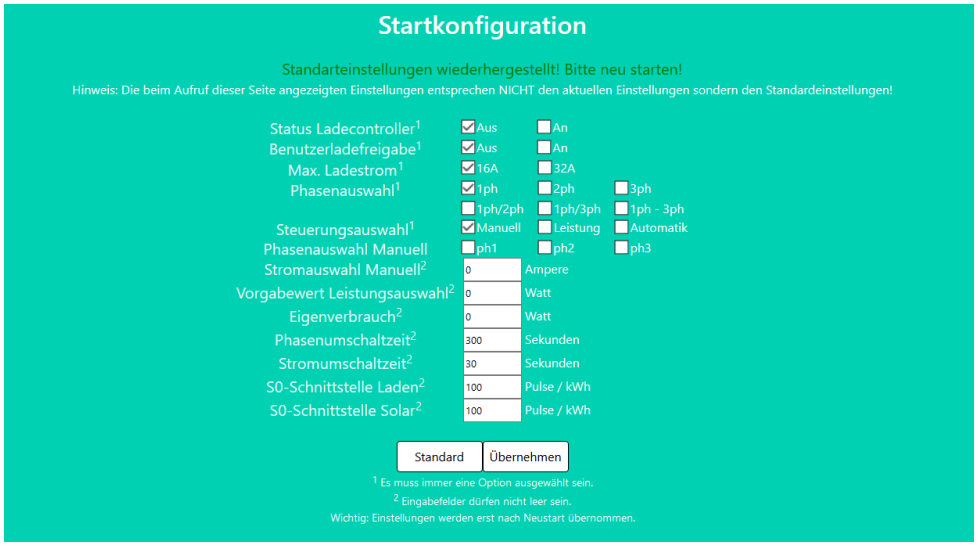



**Abb. 3.16:** Startkonfigurationsseite

**Tab. 3.3:** Standardeinstellungen der Ladestation

| Parameter | Standardwert / -einstellung |
|---|---|
| Max. Ladestrom | 16A |
| Phasenauswahl | 1ph |
| Vorgegebene Ladeleistung | 0 W |
| Vergegebener Ladestrom | 0 A |
| Steuerungsauswahl | Manuell |
| Status Ladecontroller | Aus |
| Benutzerladefreigabe | Aus |
| Controllerladefreigabe | Aus |
| Neutralleiter | Ein |
| Phase 1 | Aus |
| Phase 2 | Aus |
| Phase 3 | Aus |
| Eigenverbrauch | 0 W |
| Intervall Strom | 0 s |
| Intervall Phase | 300 s |
| S0-Ladeleistung | $100 \frac{1}{\mathrm{kWh}}$ |
| S0-Solarleistung | $100 \frac{1}{\mathrm{kWh}}$ |

Die Steuerungsseite, ebenfalls nur als eingeloggter Nutzer aufrufbar, besteht aus zwei Teilen. Der allgemeine, obere Teil (Abb. 3.17) ist für alle drei Steuervarianten gleich. Der untere Bereich variiert, basierend auf der ausgewählten Steuerung. Im allgemeinen Teil kann der Ladecontroller an- bzw. ausgeschaltet sowie die Benutzerladefreigabe, die Ladespezifikation und die Steuervariante geändert werden. Dabei bietet

das Bedienfeld `Ladespezifikation` die Möglichkeit sowohl zwischen einem maximalen Ladestrom als auch den zur Verfügung stehenden Phasen für den Ladevorgang zu wechseln, deren einzelne Bedeutung bereits in Tab. 3.2 erläutert wurde.

Außerdem werden die Controllerladefreigabe, die aktuelle sowie vorgegebene Ladeleistung und die aktuelle Solarleistung angezeigt. Eine genaue Erklärung der einzelnen Elemente ist Tab. 3.2 zu entnehmen. Die sich durch die Ladespezifikation ergebenden, möglichen Kombinationen von Phasen und Strom für die beiden Steuerungsmöglichkeiten `Leistungsvorgabe` und `Solarautomatik` sind Anhang B.2 zu entnehmen. Eine detaillierte Erklärung des Schaltverhaltens bei Konfigurations-, Ladespezifikations- und Steuerungswechsel erfolgt in Abschnitt 4.1. Anzumerken ist, dass der Ladevorgang von Seiten des Webinterfaces freigegeben ist, sobald die Benutzerladefreigabe besteht **und** eine Leistung vorgegeben ist, also der Ladestrom ungleich Null ist und mindestens eine Phase freigegeben ist. Der Ladevorgang startet, sobald das Feld `Freigabe` grün ist, also der Ladecontroller und das Auto dem Ladevorgang zustimmen. Die Auswahlmöglichkeiten basieren auf den Ladespezifikationen von Elektroautos verschiedener Hersteller und sind in nachfolgender Tabelle zusammengefasst.

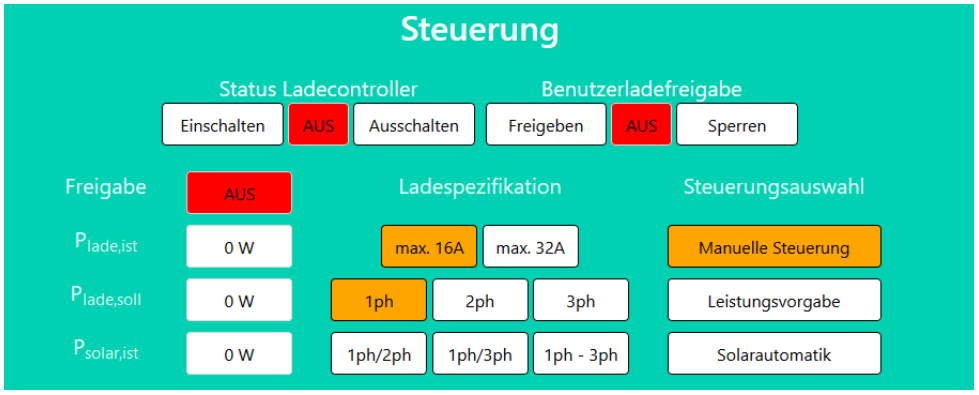



**Abb. 3.17:** Steuerungsseite: Allgemeines Interface

**Tab. 3.4:** Übersicht der Ladeleistungen verschiedener Elektroautomodelle bei Wechselspannung. Die maximalen Ladeleistungen sind [26] entnommen.
[1] Basierend auf Recherchen in diversen Online-Foren zum Thema Elektromobilität
[2] Wurde durch Testbetrieb der Ladestation aufgrund eines Wackelkontakts der dritten Phasen festgestellt

| Hersteller | Modell | Ladeleistung | Konfigurationen |
|---|---|---|---|
| VW | e-up! | max. 7,2 kW | einphasig mit bis zu 32 A [1]<br>zweiphasig mit bis zu 16 A pro Phase [26] |
| VW | e-Golf | max. 7,2 kW | einphasig mit bis zu 32 A [1]<br>zweiphasig mit bis zu 16 A pro Phase [26] |
| Tesla | Model 3 | max. 11 kW | einphasig mit bis zu 32 A [1]<br>zweiphasig mit bis zu 16 A pro Phase [2]<br>dreiphasig mit bis zu 16 A pro Phase [26] |
| Renault | Zoe | max. 22 kW | einphasig mit bis zu 32 A [27]<br>dreiphasig mit bis zu 32 A pro Phase [26] |

Das Bedienfeld für die manuelle Steuerung der Ladestation, dargestellt in Abb. 3.18, ermöglicht das manuelle Zu- und Abschalten einzelner Phasen sowie die Änderung des Ladestroms. Anzumerken ist, dass dieser Steuermodus rein für Testzwecke implementiert wurde und nach Möglichkeit nicht verwendet werden sollte, da in diesem Modus keine Schaltabläufe integriert sind. Darüber hinaus ist die Stromvorgabe zwar an den in der Ladespezifikation festgelegten, maximalen Ladestrom gebunden, die Steuerung der Phasen jedoch nicht. So ist es beispielsweise möglich, bei rein einphasiger Ladespezifikation eine weitere Phase zuzuschalten, was bei Autos, welche nur einphasig oder dreiphasig laden können, zu einer nicht zulässigen Einstellung führen würde. Eine Beschreibung, wie auch im manuellen Steuermodus ein Schaltablauf etabliert werden kann, welcher nicht zu einem Abbruch des Ladevorgangs führt, ist Abschnitt 4.1 zu entnehmen.

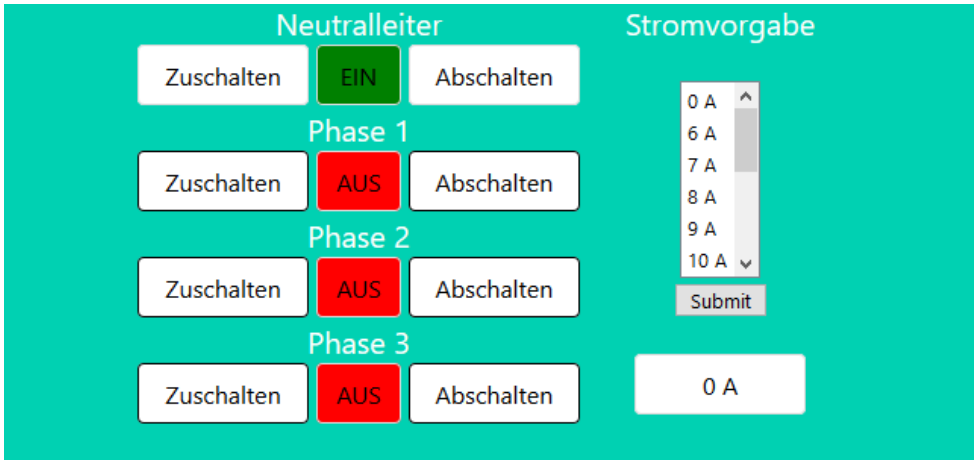



**Abb. 3.18:** Steuerungsseite: Interface für manuelle Steuerung

Das Interface des Leistungsvorgabemodus (Abb. 3.19) besteht aus einer Statusanzeige, welche den Status aller Phasen sowie den vorgegebenen Ladestrom anzeigt, und der Auswahlliste für die gewünschte Ladeleistung. Die gelisteten Leistungen ergeben sich über die ausgewählte Ladespezifikation. Einer Auflistung aller möglicher Ladespezifikationen und die dazugehörigen Leistungskombinationen ist Anhang B.2 zu entnehmen.

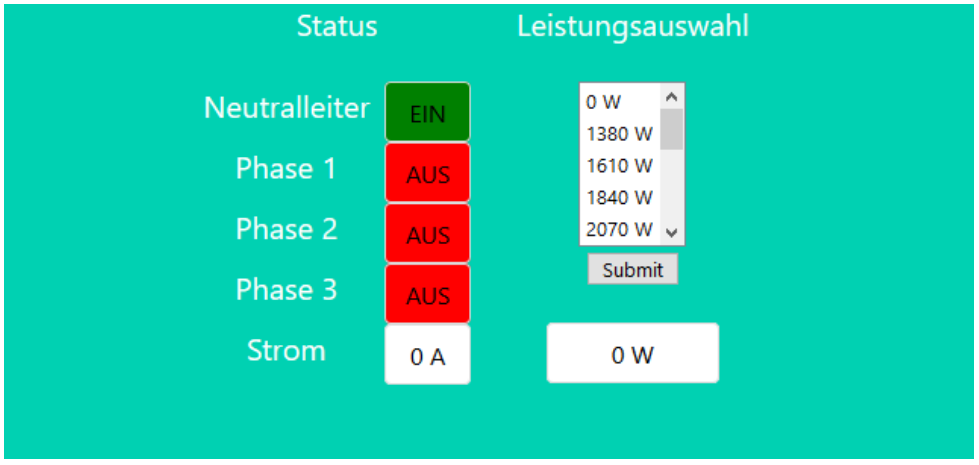



**Abb. 3.19:** Steuerungsseite: Interface für Leistungsvorgabe

Das Interface der Solarautomatik nach Abb. 3.20 ist eine reine Statusanzeige. Es enthält, wie das der Leistungsvorgabe, eine Statusanzeige über den Zustand aller Phasen und den aktuell vorgegebenen Ladestrom. Darüber hinaus werden die vorgegebenen Werte für Eigenverbrauch, Strom- und Phasenumkonfigurationszeit angezeigt, welche über die Startkonfiguration festgelegt sind. Die Solarautomatik ist, wie auch die Leistungsvorgabe, an die durch die Ladespezifikation vorgegebenen Leistungskombinationen nach Anhang B.2 gebunden, wobei immer versucht wird, die vorgegebenen Ladeleistung nach Gl.(3.2.1) einzustellen. $P_{\text{lade,soll}}$ entspricht der vorgegebenen Ladeleistung, $P_{\text{solar,ist}}$ der aktuellen Leistung der Solaranlage sowie $P_{\text{Eigenverbrauch}}$ dem Vorgabewert für den Eigenverbrauch. Dabei wird der Strom für das in Abb. 3.20 gezeigte Beispiel alle 30 s und die Phasenkonfiguration alle 300 s geändert werden.

$$P_{\text{lade,soll}} \leq P_{\text{solar,ist}} - P_{\text{Eigenverbrauch}} \tag{3.2.1}$$

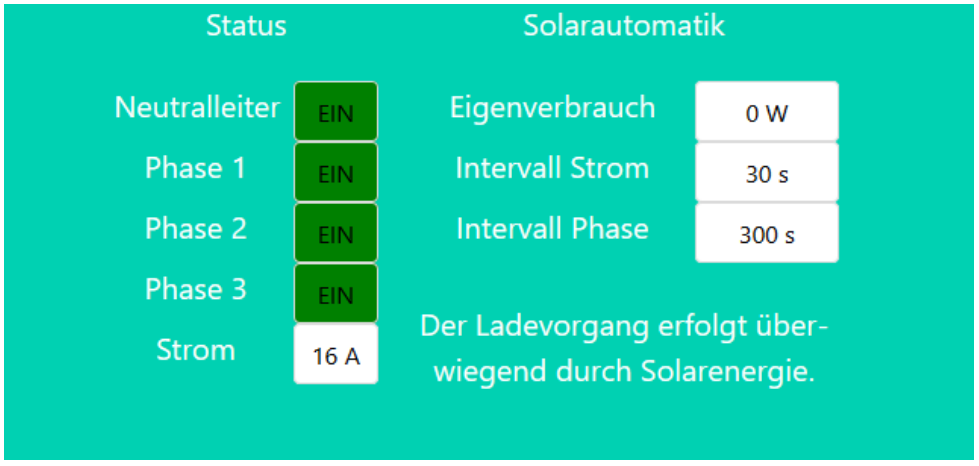



**Abb. 3.20:** Steuerungsseite: Interface für Solarautomatik

## 3.3 Messaufbau

Für die Planung, Durchführung und Validierung des Hardwareaufbaus nach Abschnitt 3.1, zum Testen der Softwareprogrammierung nach Abschnitt 3.2 sowie einem Testladebetrieb des fertigen Aufbaus ist es notwendig, verschiedene Messung an der Ladestation durchzuführen. Aus den Messungen ergeben sich letztlich verschiedene Daten wie beispielsweise Schaltleistungen, Steuerzeiten und Stranggrößen. Abb. 3.21 stellt schematisch den Messaufbau im Labor des Lehrstuhls für Mechatronik der Universität Bayreuth dar. Dabei werden die in Tab. 3.5 gelisteten Größen mit dem Achtkanaloszilloskop `WaveRunner 8108 HD` der Firma `Teledyne LeCroy` [28] (Abb. 3.22 (a)) aufgenommen. Für die Messung des externen PWM-Signals `PWM0`, der Steuersignale der drei Phasenschütze `SSR_L1`, `SSR_L2` und `SSR_L3` sowie der Kommunikationsleitung `CP` werden die aktiven Differential-Tastköpfe `TT-SI 9101` und `TT-SI 9110` der Firma `TESTEC` [29] (Abb. 3.23 (a)) verwendet. Die Erfassung der Strangströme $I_{L1}$, $I_{L2}$ und $I_{L3}$ erfolgt mit Hilfe von Strommesszangen vom Typ `TCP312A` (Abb. 3.23 (b)) und den zugehörigen Messwandlern `TCPA300` (Abb. 3.23 (c)) der Firma `Tektronix` [30]. Darüber hinaus werden für Testzwecke die S0-Pulse für Lade- und Solarleistung mittels des Signalgenerators `WaveStation 2022` der Firma `Teledyne LeCroy` (Abb. 3.22 (b)) [31] simuliert.

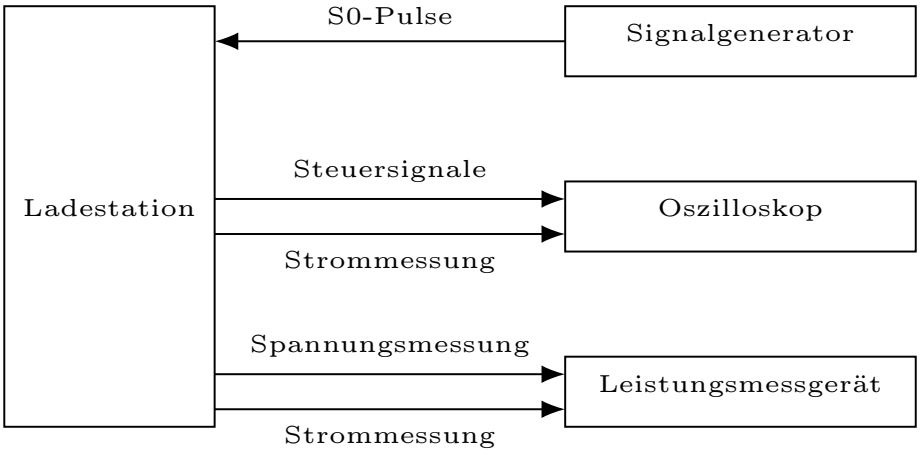



**Abb. 3.21:** Schematischer Messaufbau

(a) (b)

**Abb. 3.22:** WaveRunner 8108 HD 12bit Oszilloskop (a) [28] und Signalgenerator WaveStation 2022 25 MHz (b) [31]

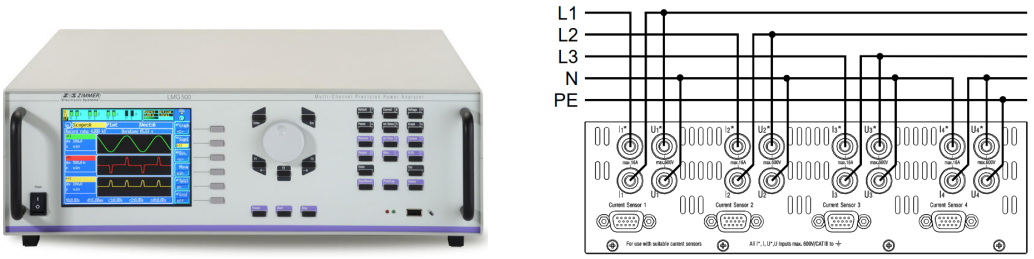
**Tab. 3.5:** Messgrößen für Oszilloskopaufnahmen; Beschreibung der Signale und Klemmen sowie Seitenreferenz nach Anhang A.3

| Signal | Abgriffspunkt | Bezug | Schaltplan | Beschreibung |
|---|---|---|---|---|
| PWM0 | 12 | 6 | 4/6 | PWM-Signal zur Steuerung des Ladecontrollers |
| SSR_L1 | 22 | 6 | 4/6 | Steuersignal Phasenschütz L1 |
| SSR_L2 | 35 | 6 | 4/6 | Steuersignal Phasenschütz L2 |
| SSR_L3 | 37 | 6 | 4/6 | Steuersignal Phasenschütz L3 |
| CP | -X5-7 | GND | 6/6 | Kommunikationsleitung Auto - Ladecontroller |
| $I_{L1}$ | -X3-U | - | 1/6 | Strangstrom L1 |
| $I_{L2}$ | -X3-V | - | 1/6 | Strangstrom L2 |
| $I_{L3}$ | -X3-W | - | 1/6 | Strangstrom L3 |

(a) (b) (c)

**Abb. 3.23:** Aktiver 100 MHz Differtialtastkopf (a) [29], Tektronix Strommesszange TCP312A (b) und TCPA300 (c) Messwandler [30]

Zusätzlich erfolgt eine detaillierte Leistungsanalyse der Ladestation mit Hilfe des Präzisions-Leistungsmessgeräts `LMG500` der Firma `ZES Zimmer` (Abb. 3.24 (a)). Das Leistungsmessgerät ist dem Anschluss der Ladestation mit Hilfe von Messadaptern vorgeschaltet. Dabei fließen bei den Messungen fälschlicherweise die Auswirkungen des Steuerkreises mit ein, was aber im Gegensatz zur eigentlichen Ladeleistung von 1,38 kW bis 11 kW vernachlässigbar ist. Die verwendete Anschlussart des Leistungsmessgerätes ist in Abb. 3.24 (b) dargestellt. Somit ist es dem Messgerät möglich die Spannungen und Ströme der drei Phasen zu messen und gleichzeitig sämtliche Leistungs-, Energie- und Verschiebungswerte zu berechnen.




(a) (b)

**Abb. 3.24:** Zimmer Präzisions-Leistungsmessgerät LMG500 mit Darstellung der Bedienseite (a) und den vier Anschlusskanälen (b) [32]

# 4 Auswertung des Betriebsverhaltens

Im folgenden Abschnitt soll der Betrieb der Ladestation analysiert werden. Dafür wird zunächst auf das implementierte Schaltverhalten eingegangen und anschließend der Ladebetrieb an sich betrachtet. Abschließend wird der Betrieb im Steuermodus `Solarautomatik` gesondert analysiert. Der Testladebetrieb erfolgt an einem Renault Zoe und einem Tesla Model 3.

## 4.1 Analyse des implementierten Schaltverhaltens

Für die Analyse des implementierten Schaltverhaltens ist es zunächst sinnvoll, noch einmal die externe Steuerung des Ladecontrollers (Abschnitt 2.1.3) über ein PWM-Signal mit einer Frequenz von 4 Hz aufzugreifen, welches die in Tab. 4.1 aufgeführten drei Möglichkeiten mit sich bringt.

**Tab. 4.1:** Werte des DC des externen PWM-Signals zur Steuerung des Ladecontrollers nach [33]

| Modus | Dutycycle | Erklärung |
|---|---|---|
| 1 | 100 % | Ladecontroller wird gesperrt |
| 2 | gemäß Gl.(2.1.2) | Ladecontroller freigegeben mit externer Stromvorgabe |
| 3 | 0 % | Ladecontroller freigegeben mit einprogrammiertem Ladestrom |

Abb. 4.1 zeigt die mittels Oszilloskop aufgenommenen Steuersignale für die Phasenschütze Q1 (blau), Q2 (braun) und Q3 (orange) sowie das PWM-Signal zur externen Steuerung des Ladecontrollers (violett) und die Kommunikationsleitung zwischen Auto und Ladestation CP (grün). Der Übersichtlichkeit halber ist die Amplitude von CP halbiert. Wie gut in Abb. 4.1 oben links zu erkennen ist beträgt die Zeitdifferenz zwischen der Freigabe des Ladecontrollers über das PWM-Signal (Übergang von Modus 1 auf Modus 2 nach Tab. 4.1) und der Freigabe des Ladevorgangs durch den Ladecontrollers ($-12$ V auf $+12$ V) ungefähr 1 s. Bei einem Sperren des Ladecontrollers, Abb. 4.1 unten links, beträgt die Reaktionsdauer ca. 2 s. Im Datenblatt ([33]) wird eine maximale Reaktionszeit beim An- / Ausschalten von 2,6 s genannt. Basierend auf diesen Beobachtungen wurde für das Zu- / Abschalten von Phasen, dargestellt in Abb. 4.1 oben bzw. unten rechts, das in Abb. 4.2 visualisierte Schaltverhalten implementiert. Dabei wird zunächst der Ladecontroller über das PWM-Signal gesperrt und anschließend eine Wartezeit von 3 s eingehalten, damit der Controller auf jeden Fall den Ladevorgang beendet und somit das Ladeschütz Q5 geöffnet hat. Dadurch

wird zum einen vermieden, dass durch ein nicht korrektes Beenden des Ladevorgangs das Auto in einen Fehlerzustand geht, welcher sich bei Tests mit Zoe und Tesla nur dadurch aufheben lies, in dem das Auto von der Ladesäule ab- und anschließend wieder angesteckt wurde. Zum anderen wird dadurch sicher gestellt, dass die Steuerschütze für die einzelnen Phasen (Q1 bis Q4) keine Last schalten müssen. Nach Ablauf der Wartezeit werden die Phasenschütze neu gestellt und anschließend der Ladecontroller wieder über das PWM-Signal freigegeben. Soll der Ladestrom angepasst werden, so kann dies einfach über eine Änderung des Dutycycles des PWM-Signals vorgenommen werden, wobei keine Schaltabläufe nötig sind.

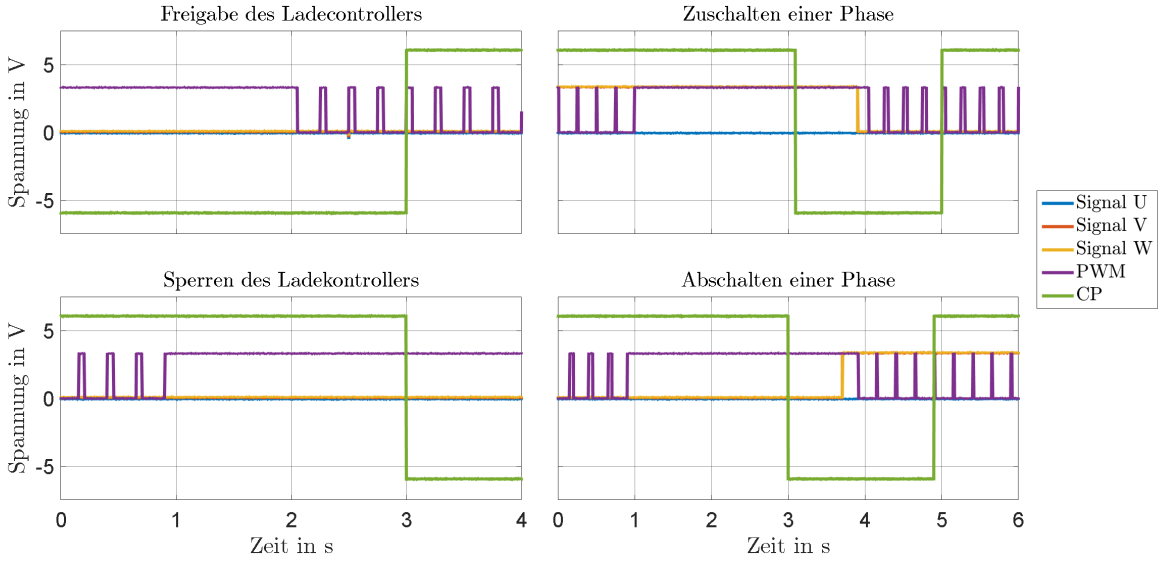



**Abb. 4.1:** Analyse des Schaltverhaltens der Steuerung im Bezug auf Freigabe und Sperren des Ladecontrollers sowie einer Änderung der Phasenkonfiguration. Der Übersichtlichkeit halber ist die Amplitude des Signals `CP` (grün) halbiert.

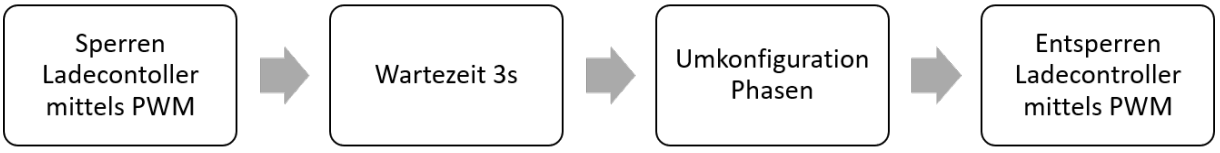



**Abb. 4.2:** Schaltablauf Phasenumkonfiguration

Basierend auf den Erkenntnissen aus den vorherigen Analysen wurden spezielle Schaltabläufe für die einzelnen Komponenten der Steuerungsseiten (Abb. 3.17 bis Abb. 3.20), welche im Skript `steuerung.py` (Abschnitt 3.2.1) definiert sind, implementiert, welche im Folgenden näher erläutert werden sollen.

Für das An- / Abschalten des Ladecontrollers über das Element `Status Ladecontroller` wurde kein spezielles Schaltverhalten implementiert, da ein Abschalten des Ladecontrollers gleich einem Beenden des Ladevorgangs durch den Ladecontroller ist. Somit wird über dieses Bedienfeld lediglich die Spannungsversorgung für den Ladecontroller zu- bzw. abgeschaltet.

Durch das Entziehen der Nutzerladefreigabe über das Bedienfeld `Benutzerladefreigabe` wird der Ladecontroller, falls dies noch nicht der Fall ist, über das externe PWM-Signal gesperrt. Anschließend werden aus oben bereits erläuterten Gründen 3 s gewartet und anschließend das Ladeschütz Q5 gesperrt. Bei einer Freigabe durch den Nutzer wird der Ladecontroller für eine Stromvorgabe $I_{\text{lade}} > 0\,\text{A}$ über das externe PWM-Signal freigegeben.

Wird eine Änderung der `Ladespezifikation` vorgenommen, so wird der Ladecontroller, falls dies noch nicht der Fall ist, über das externe PWM-Signal gesperrt und anschließend die Wartezeit von 3 s eingehalten. Anschließend werden alle Phasen sowie das Ladeschütz abgeschaltet, Ladestrom auf 0 A und Ladeleistung auf 0 W gesetzt sowie die Nutzerladefreigabe entzogen. Zum Schluss werden die Leistungskombinationen basierend auf der geänderten Ladespezifikation neu ermittelt. Dieses Vorgehen hat den Hintergrund, dass bei einer Änderung der Ladespezifikation die aktuellen Einstellungen nicht mehr zulässig sein könnten und somit zu einem Fehler oder im schlimmsten Fall zu Schäden an Ladestation und/oder Auto führen könnten. Dieses Schaltverhalten wird in allen drei Steuermodi bei einer Änderung der Ladespezifikation ausgeführt.

Bei einem Wechsel der Steuerung über das Bedienfeld `Steuerungsauswahl` spielt die Ausgangssituation eine wesentliche Rolle. Wird vom Steuermodus `Leistungsvorgabe` oder `Solarautomatik` in einen der anderen beiden gewechselt, so ist weiter nichts zu beachten, da diese beiden Steuermodi an die Leistungskonfigurationen nach Anhang B.2 gebunden sind. Die aktuelle Ladeleistung wird einfach beibehalten und ein gegebenenfalls aktiver Ladevorgang nicht unterbrochen. Wird jedoch vom Steuermodus `Manuelle Steuerung` in einen der anderen beiden gewechselt, so wird das gleiche Schaltverhalten wie bei einer Änderung der Ladespezifikation angewendet, da die manuelle Steuerung nicht an die Leistungskonfigurationen aus Anhang B.2 gebunden ist und somit unzulässige Kombinationen der einzelnen Phasen vorliegen könnten.

Wie bereits in Abschnitt 3.2.3 erwähnt stellt die `Manuelle Steuerung` einen Testmodus dar, welche nach Möglichkeit nicht verwendet werden sollte, da in diesem keine Schaltabläufe integriert sind und dieser, wie auch vorher schon erwähnt, zwar an den in der Ladespezifikation festgelegten, maximalen Ladestrom, aber nicht an die ausgewählte Phasenkonfiguration gebunden ist. Dadurch können während eines aktiven Ladevorgangs einzelne Phasen zu- oder abgeschaltet werden, was zu Störungen und im schlimmsten Fall zu Schäden an Auto und/oder Ladestation führen kann. Zudem ist es aufgrund des in der vorher erwähnten `Benutzerladefreigabe` integrierten Schaltverhaltens möglich, den Ladevorgang durch eine Stromvorgabe zu starten, obwohl keine der drei Phasen aktiv ist. Dies führt bei Tesla Model und Renault Zoe zu einem Fehlerfall, welcher manuell durch ein Ab- und anschließend erneutes Anstecken des Autos an die Ladesäule quittiert werden kann. Nichts desto trotz kann hier händisch ein Schaltverhalten zum Umgehen dieser Störung bei einer Änderung der Phasenkonfiguration durchgeführt werden. Dazu wird zunächst die Nutzerladefreigabe entzogen und einige Sekunden gewartet, bis der Ladecontroller den Ladevorgang beendet hat. Anschließend können die Phasen umkonfiguriert werden, wobei darauf zu achten ist, dass keine unzulässige Einstellung gewählt wird. Der Ladevorgang kann jetzt wieder über die

Benutzerladefreigabe gestartet werden.

Das Schaltverhalten im Steuermodus `Leistungsvorgabe` bei einer Änderung der Phasenkonfiguration aufgrund einer Leistungsänderung entspricht dem aus Abb. 4.2. Wie bereits in Abschnitt 3.2.3 erwähnt entsprechen die möglichen Leistungen in diesem Modus den Kombinationen aus Anhang B.2 basierend auf der ausgewählten Ladespezifikation. Muss bei einer Änderung der Leistung lediglich der Strom angepasst werden, so geschieht dies über eine Anpassung des Dutycycles des externen PWM-Signals ohne eine Unterbrechung des Ladevorgangs.

Bei einem Betrieb der Ladestation mittels `Solarautommatik` liegt das gleiche Schaltverhalten für Strom- und Phasenänderung wie im Steuermodus `Leistungsvorgabe` zugrunde. Auch hier entsprechen die möglichen Leistungen in den Kombinationen aus Anhang B.2 basierend auf der ausgewählten Ladespezifikation. Eine genaue Analyse des Ladevorgongs im Modus `Solarautomatik` erfolgt in Abschnitt 4.3.

## 4.2 Betrachtung des Ladeverhaltens

Für die Untersuchung der Ströme und Spannungen beim Laden stehen, wie bereits erwähnt, ein Tesla Model 3 und ein Renault Zoe zur Verfügung. Eine Auswahl exemplarischer Ladekurven von Renault Zoe und Tesla Model 3 bei unterschiedlichen Ladeströmen und aktiven Phasen ist Anhang C zu entnehmen. Besonders interessant ist dabei der dreiphasige Anlauf des Ladevorgangs. Die Messungen der Ströme und deren Phasenverschiebungen beim Start des Ladevorgangs für den Tesla Model 3 sowie den Renault Zoe wurden mit dem in Abschnitt 3.3 erwähnten Leistungsmessgerät durchgeführt und sind in Abb. 4.3 sowie Abb. 4.4 dargestellt. Für die Abschätzung der Bauteilbelastungen sind die Messungen nötig. Der Tesla Model 3 weist eine geringe Phasenverschiebung von $-2{,}65°$ nach dem Anlauf des Ladevorgangs auf. Dies ändert sich auch bei geringeren Ladeleistungen nicht. Der Renault Zoe hingegen hat nach Anlauf der Ladeleistung eine Verschiebung von $-16°$. Dies ist vermutlich weitestgehend auf die Verwendung der Motorspulen als Induktivität des Ladewandlers zurückzuführen. Da aber kaum weitere Hersteller eine solche Technik verwenden, ist anzunehmen, dass der Renault Zoe die größte Phasenverschiebung der auf den Markt erhältlichen Ladewandler aufweist. Somit sind die Auslegungen der Bauteile nach den vorhandenen Messwerten ausreichend.

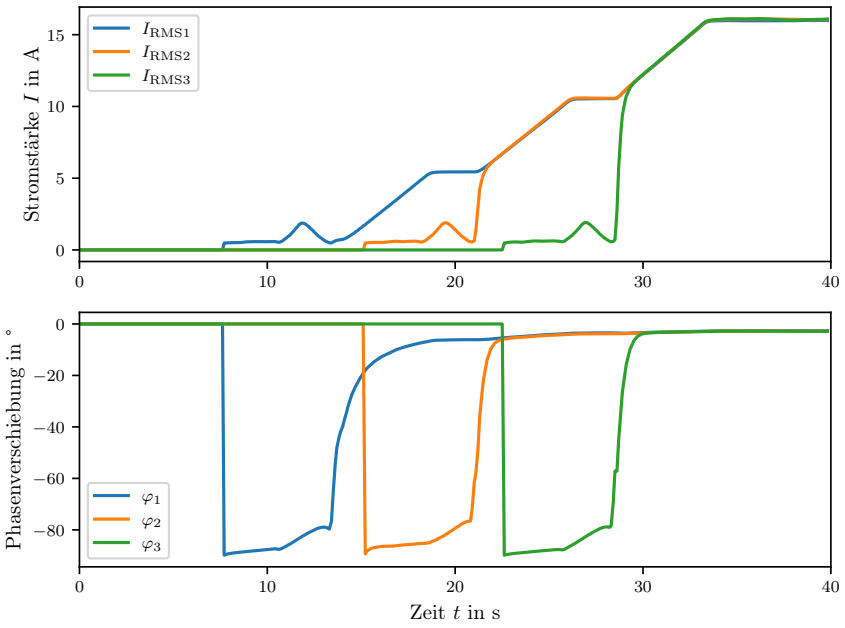



**Abb. 4.3:** Dreiphasiger Anlauf des Ladens eines Tesla Model 3 mit maximalen Strom von 16 A und zusätzlicher Messung der Phasenverschiebung

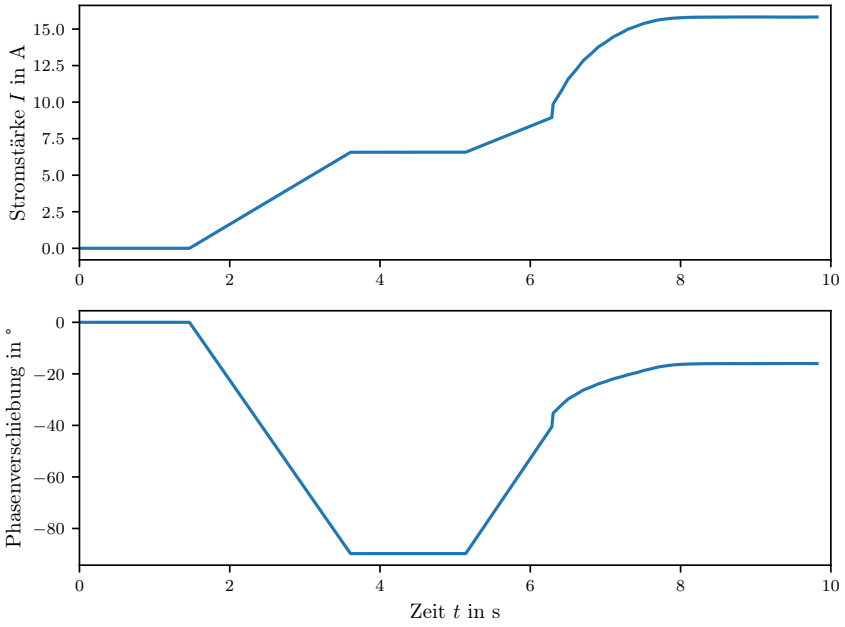



**Abb. 4.4:** Dreiphasiger Anlauf des Ladens eines Renault Zoe (Darstellung nur eines Außenleiters, da sich alle drei Außenleiter gleich verhalten) mit maximalen Strom von 16 A und zusätzlicher Messung der Phasenverschiebung

Für eine weitere Analyse des Ladeverhaltens wird anschließend der stationäre Ladebetrieb nach Anlauf betrachtet. Abb. 4.5 zeigt die Ladekurven von Renault Zoe und Tesla Model 3 im stationären Betrieb bei einem vorgegebenen Ladestrom von 16 A. Zusätzlich wird zwischen einphasigem und dreiphasigem Laden unterschieden. Bei beiden Fahrzeugen stellt sich ein sinusförmiger Strom ein, wobei die Verläufe der Ströme des Tesla Model 3 deutlich glatter als die des Renault Zoe sind. Dies wird auch durch den Mittelwert des Stroms $I_{\mathrm{RMS}}$ unterstrichen, da der des Renault Zoe leicht unter dem des Tesla Model 3 liegt. Werden die exemplarischen Ladekurven des Tesla Model 3 und Renault Zoe aus Anhang C verglichen so fällt auf, dass beim Renault Zoe der $I_{\mathrm{RMS}}$ außer bei einem Vorgabestrom von 6 A immer unter dem Vorgabewert liegt, wohingegen der des Tesla Model 3 nahe am Vorgabewert, meist etwas darüber, liegt. Da der Renault Zoe auf einen maximalen Ladestrom von 32 A ausgelegt ist (Tab. 3.4), kann es durchaus sein, dass sich die Kurvenverläufe der Strangströme zum oberen Ende des Ladeleistungsbereichs hin verbessern, da auch bereits deutliche Verbesserungen von 6 A zu 16 A zu sehen sind. Dies konnte aber aufgrund der Auslegung der Ladestation auf einen Maximalstrom von 16 A nicht überprüft werden.

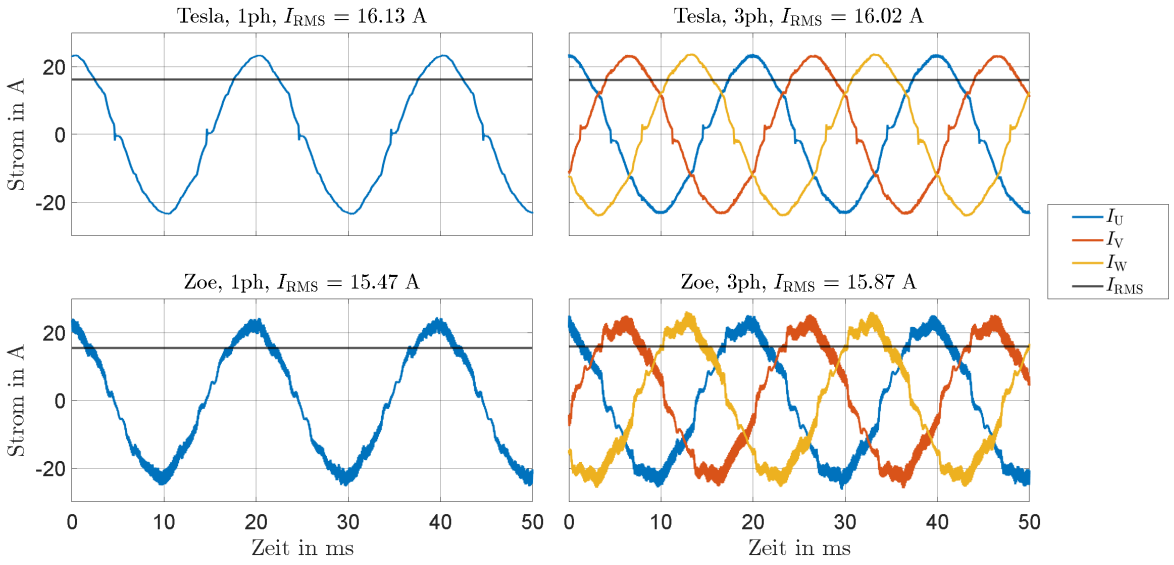



**Abb. 4.5:** Vergleich der Ladekurven von Renault Zoe und Tesla Model 3 bei 16 A

Zum Schluss wird die Änderung des vorgegebenen, effektiven Ladestroms $I_{\mathrm{RMS}}$ im laufenden Betrieb betrachtet, wofür die Effektivwerte der einzelnen Strangströme aufgezeichnet werden. Dabei wird zwischen vier verschiedenen Szenarien unterschieden. Zum einen erfolgt eine schrittweise Erhöhung bzw. Verringerung des vorgegebenen Ladestroms von 6 A auf 16 A bzw. von 16 A auf 6 A in 1 A-Schritten. Zum anderen wird eine Erhöhung bzw. Verringerung von 6 A auf 16 A bzw. von 16 A auf 6 A in einem Schritt durchgeführt. Das Ergebnis ist in Abb. 4.6 dargestellt. Die Aufnahmen wurden beim Ladebetrieb des Tesla Model 3 erstellt, wobei das Verhalten des Renault Zoe bei gleicher Stromvariation im laufenden Betrieb diesen Aufnahmen ähnelt. Wie unschwer zu erkennen ist erfolgt die Anpassung des Ladestroms nach einer Erhöhung bzw. Verringerung der Vorgabe von 1 A innerhalb kürzester Zeit. Die Anpassung

des Ladestromstrom bei einer Verringerung der Vorgabe von 16 A auf 6 A in einem Schritte erfolgt nahezu sofort. Wird die Vorgabe in einem Schritt von 6 A auf 16 A angehoben, so erhöht das Auto seine Stromaufnahme linear bis zum neuen Maximalwert über einen Zeitraum von etwas mehr als 10 s. Somit bestätigen diese Messungen die vorherigen Annahmen, dass eine Änderung des Ladestroms im laufenden Betrieb ohne jegliche Schaltabläufe vorgenommen werden kann.

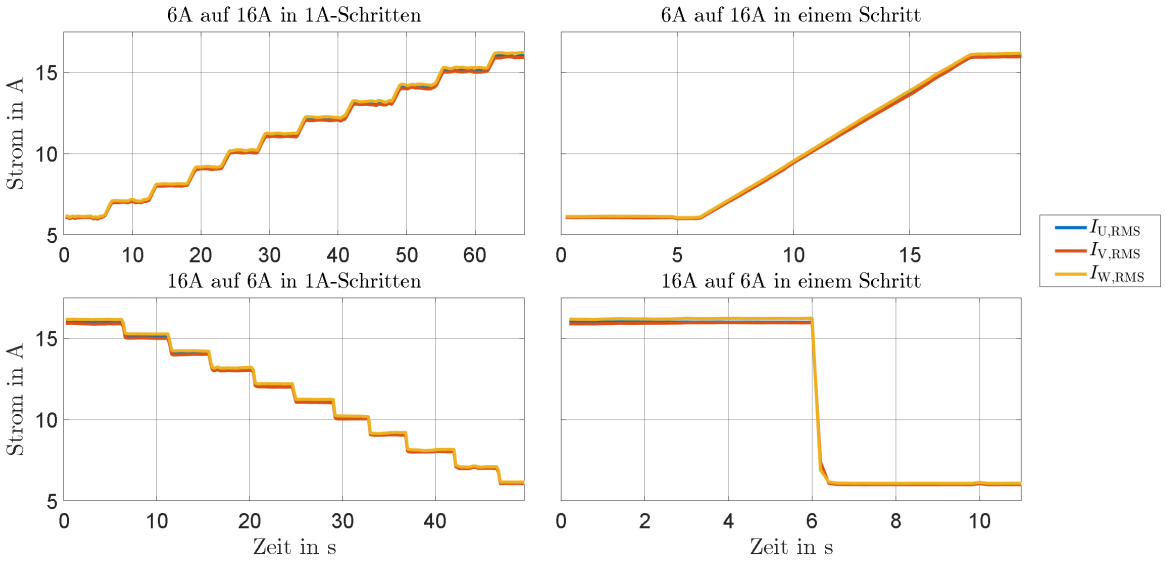



**Abb. 4.6:** Variation des vorgegebenen, effektiven Ladestroms $I_{RMS}$ beim Ladebetrieb des Tesla Model 3

## 4.3 Testbetrieb der Solarautomatik

Wie bereits in Abschnitt 3.2.3 und 4.1 sind die in der `Solarautomatik` möglichen Kombinationen von Phasen und Strom über die Ladespezifikation festgelegt und in Anhang B.2 zusammengefasst. Ebenfalls ist aus Abschnitt 3.2.1 und 4.1 bekannt, dass die `Solarautomatik` in einem periodischen Thread mit der Wiederholzeit `Intervall Strom`, welches auch dem Zeitabstand zwischen zwei Stromänderungen entspricht, ausgeführt wird. Der Abstand zwischen zwei Änderungen der Phasenkonfiguration ist durch den Parameter `Intervall Phase` vorgegeben. Nach einer Änderung der Phasenkonfiguration steht für die verbleibende Zeit `Intervall Phase` nur noch der Strom gemäß der Ladespezifikation nach Tab. B.2 als Steuerparameter zur Verfügung. Ist der minimale Strom eingestellt, lässt sich die Leistung bis zur nächsten Umkonfiguration der Phasen nicht mehr verringern. Darüber hinaus wird, in Abschnitt 3.2.1 näher erläutert, bei einer Erstausführung der Solarautomatik der Timer für die Phasenumkonfiguration zurückgesetzt und direkt umkonfiguriert. Zusätzlich lässt sich über den Parameter `Eigenverbrauch` ein fester Leistungswert vorgeben, welcher bei der Einstellung der Ladeleistung gemäß Gl.(3.2.1) berücksichtigt wird. Das Schaltverhalten bei einer Umkonfiguration der Phasen entspricht dem Ablauf nach Abb. 4.2 während eine Änderung des Stroms im laufenden Betrieb durch die Anpassung des externen PWM-Signals vorgenommen werden kann.

Abb. 4.7 zeigt den Testbetrieb der Solarautomatik während des Ladevorgangs mit dem Tesla Model 3. Die Grafik basiert auf den zum Download bereitstehenden Leistungsdaten des Leistungsgraphen auf der Übersichtsseite (Abb. 3.15). Zusätzlich wurde die Abbildung noch um die Umkonfigurationspunkte (schwarz) erweitert. Die S0-Pulse der aktuellen Solarleistung $P_{\text{solar,ist}}$ (blau) wurden dabei mittels des in Abschnitt 3.3 beschriebenen Signalgenerators simuliert.

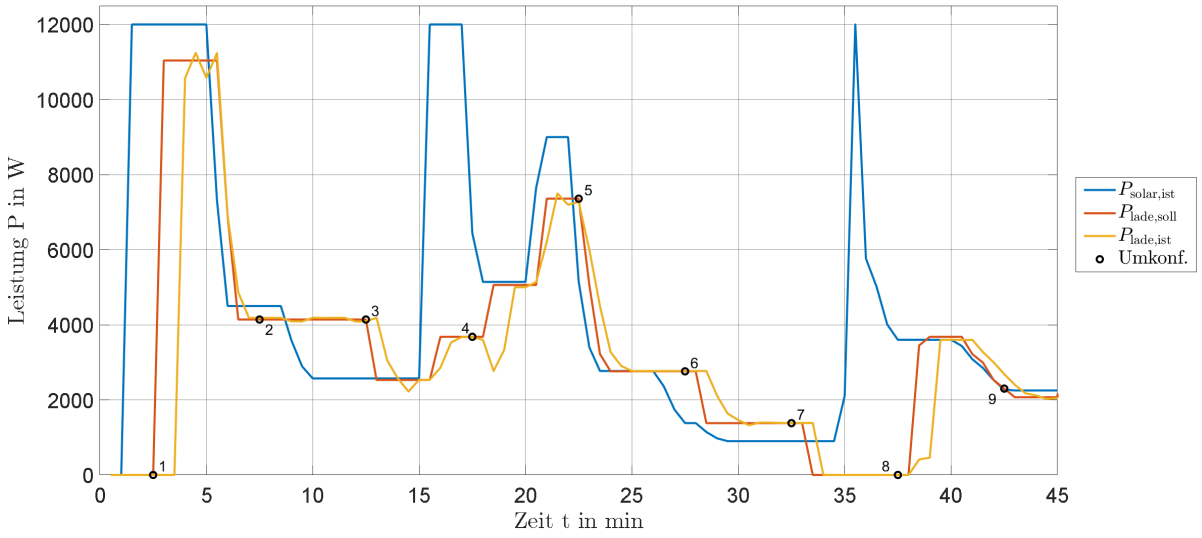



**Abb. 4.7:** Leistungsverläufe im Testbetrieb der Solarautomatik

Tab. 4.2 enthält die für den Testbetrieb gewählten Parameter für die Steuerung. Aufgrund der gewählten Ladespezifikation stehen die Kombinationen nach Tab. B.13 zur Verfügung und somit beträgt die minimal Ladeleistung $P_{\text{lade,min}}$ 1380 W und die maximal Ladeleistung $P_{\text{lade,max}}$ 11 040 W. In der Abbildung ist zu erkennen, dass die aktuelle Ladeleistung $P_{\text{lade,ist}}$ (orange) der vorgegebenen Ladeleistung $P_{\text{lade,soll}}$ (rot) mit etwas Verzögerung folgt und deren Wert letztlich annimmt. Grund dafür ist, dass die Reaktion von Ladecontroller und Auto auf eine Leistungsänderung etwas Zeit in Anspruch nimmt und zudem bei einer Umkonfiguration der Phasen der Ladevorgang kurz unterbrochen wird. Die Schwankungen der aktuellen Ladeleistung trotz konstanter Vorgabe liegen an den in Abschnitt 3.2.1 erwähnten Schwankungen der Zeitabstände der S0-Pulse bei konstanter Ladeleistung.

**Tab. 4.2:** Ausgewählte Parameter für den Testbetrieb der Solarautomatik am Tesla

| Parameter | Wert |
|---|---|
| Eigenverbrauch | 0 W |
| Intervall Strom | 30 s |
| Intervall Phase | 300 s |
| Maximaler Ladestrom | 16 A |
| Phasenspezifikation | 1ph - 3ph |

Für den Testbetrieb wird zunächst der Ladevorgang freigegeben und eine aktuelle Solarleistung von 12 000 W simuliert. Zum Zeitpunkt $t = 2{,}5$ min (Umkonfigurationspunkt 1) wird der Solarautomatikbetrieb gestartet und direkt die maximale Ladeleistung $P_{\text{lade,max}}$ von 11 040 W für die aktuelle Ladespezifikation vorgegeben, was einem dreiphasigen Laden mit 16 A pro Phase entspricht. Anschließend wird die Solarleistung auf 4500 W verringert und $P_{\text{lade,soll}}$ wird automatisch auf 4140 W angepasst, was einem dreiphasigen Laden mit 6 A entspricht und somit keine Umkonfiguration der Phasen notwendig ist. Am Umkonfigurationspunkt 2 ($t = 7{,}5$ min) wird die aktuelle Konfiguration beibehalten. Eine weitere Verringerung der Solarleistung auf 2571 W hat zur Folge, dass die vorgegebene Ladeleistung vorerst nicht geändert wird, da bereits der minimale Strom von 6 A eingestellt ist. Erst nach Erreichen von Umkonfigurationspunkt 3 ($t = 12{,}5$ min) wird $P_{\text{lade,soll}}$ auf 2530 W angepasst, was einem einphasigen Laden mit 11 A entspricht. Der erneute Anstieg von $P_{\text{solar,ist}}$ auf 12 000 W und ein direkt darauf folgenden Rückgang auf 5142 W wird durch die Solarautomatik geglättet, indem zunächst der Ladestrom im einphasigen Betrieb auf 16 A erhöht wird, was einem $P_{\text{lade,soll}}$ von 3680 W entspricht. Bei Umkonfigurationspunkt 4 ($t = 17{,}5$ min) wird auf zweiphasiges Laden mit 11 A, also 5060 W, umgestellt. Anschließend wird nach erneutem Anstieg der Solarleistung der Ladestrom auf 16 A erhöht und somit ein $P_{\text{lade,soll}}$ von 7360 W eingestellt. Ein weiteres Abfallen der Solarleistung auf 2770 W hat zur Folge, dass an Umkonfigurationspunkt 5 ($t = 22{,}5$ min) auf eine Ladeleistung von 2760 W, also zweiphasige mit 6 A, umkonfiguriert wird. Die Solarleistung sinkt weiter kontinuierlich auf 900 W. Dies hat zur Folge, dass zunächst an Umkonfigurationspunkt 6 ($t = 27{,}5$ min) auf ein einphasiges Laden mit 6 A, also der minimalen Ladeleistung von 1380 W, und anschließend aufgrund der Unterschreitung von $P_{\text{lade,min}}$ durch die $P_{\text{solar,ist}}$ der Ladevorgang an Umkonfigurationspunkt 7 ($t = 32{,}5$ min) unterbrochen wird. Bei Umkonfigurationspunkt 8 ($t = 37{,}5$ min) wird der Ladevorgang wieder im einphasigen Betrieb mit 16 A, also 3680 W, gestartet und die Ladeleistung zum Ende hin auf 2070 W (einphasig mit 9 A) der Solarleistung entsprechend angepasst und somit findet an Umkonfigurationspunkt 9 ($t = 42{,}5$ min) keine Änderung der Phasenkonfiguration statt.

# 5 Zusammenfassung und Ausblick

Diese Arbeit stellt eine Komplettanleitung für den Selbstbau einer Ladestation basierend auf der Typ-2-Verbindung inklusive Webinterface und einer einfachen Anbindungsmöglichkeit für eine Solaranlage dar. Dabei ist nochmals darauf hinzuweisen, dass für den Nachbau der Ladestation in Eigenregie das Wissen von Elektrofachpersonal erforderlich ist, da es lebensgefährlich ist, elektrische Anlagen solcher Art ohne das nötige Wissen anzufertigen.
Im Zuge dieser Arbeit wird zunächst auf die für den Hardwareaufbau nötigen Komponenten eingegangen und einige Grundlagen für die Erstellung des Webinterfaces geschaffen. Anschließend erfolgt der Aufbau, sowie dessen gleichzeitige Dokumentation inklusive Nachbauanleitung, der Ladestation, welche auf einen Maximalstrom von 16 A ausgelegt wird. Auf dem Hardwareaufbau basierend wurde ein Webinterface, ausgelegt auf einen Maximalstrom von 32 A, zur Steuerung der Ladestation über einen Raspberry Pi 4 entworfen, dessen Funktionsumfang und Benutzeroberfläche ebenfalls genau dokumentiert sind. Mittels eines eingebauten Schalters kann zwischen der Steuerung über das Webinterface und einer Steuerung rein über den Ladecontroller gewechselt werden. Das Webinterface ermöglicht eine Steuerung der Ladestation über Geräte, welche sich im gleichen Netzwerk wie der Raspberry Pi befinden. Neben einem Authentifizierungssystem, diversen Statusanzeigen, einer benutzerdefinierten Startkonfiguration und der Auswahlmöglichkeit unterschiedlicher Ladespezifikationen im Bezug auf verwendete Phasen und maximalen Ladestrom bilden die drei möglichen Steuermodi das Herzstück des Webinterfaces. Dabei spielt die Solarautomatik die wichtigste Rolle, da sich in diesem Modus die vorgegebene Ladeleistung automatisch an die aktuelle Leistung der angebundenen Solaranlage anpasst.
Zur Validierung von Hard- und Software wurde ein geeigneter Messaufbau gewählt, mit dessen Hilfe zunächst das implementierte Schaltverhalten bei einer Umkonfiguration der einzelnen Phasen sowie die Anpassung des Ladestroms im laufenden Betrieb analysiert wurde. Anschließend wurde das Ladeverhalten an einem Renault Zoe sowie einem Tesla Model 3 bei unterschiedlichen Ladeleistungen durch Variation des Ladestroms und/oder der aktiven Phasen untersucht. Zum Schluss wurde die Solarautomatik am Tesla Model 3 auf ihre Funktionalität durch die Simulation einer Solarleistung über einen Signalgenerator geprüft.

Die vorliegende Arbeit soll das Grundgerüst für den Bau einer Ladestation für Elektroautos in Eigenregie bilden, welche darüber hinaus die einfache Integrierung einer Solaranlage ermöglicht. Da bei der Konzeptionierung darauf geachtet wurde, dass die Kommunikation zwischen Auto und Ladestation ausschließlich

über die IEC Typ 2 Kontaktierung erfolgt, ist diese kompatibel zu allen gängigen Elektroautos mit diesem Anschluss. Dieser Punkt bietet jedoch noch großes Potential, da beispielsweise durch die Implementierung der OBD2-Schnittstelle (On-Board-Diagnose) Autodaten wie der aktuelle State of Charge (SoC), zu deutsch Ladezustandsbereich, ausgelesen werden können, und somit zum Beispiel eine Wirkungsgradmessung bei verschiedenen Ladeleistungen und SoC durchgeführt werden könnte. Gleiches kann auch mit bereits existierenden Python-Paketen wie dem Paket `tesla-api` erreicht werden, mit dessen Hilfe online auf die vom Hersteller zur Verfügung gestellten Autodaten wie z.B. geladenen Energiemenge, SoC oder Temperatur zugegriffen werden kann. Auch sollte darüber nachgedacht werden, die Solaranlage nicht über die S0-Schnittstelle des Energiezähler anzubinden, sondern über die Modbus-Schnittstelle, was den Vorteil bringen würde, dass genauere Daten von Seiten der Solaranlage zur Verfügung stehen würden. Im Hinblick auf die Steuerung der Ladestation über das Webinterface und speziell für die Solarautomatik ergeben sich im wesentlichen zwei Punkte für weitere Untersuchungen. Zum einen sollte versucht werden, doppelte Leistungswerte wie beispielsweise einphasiges Laden mit 12 A oder zweiphasiges Laden mit 6 A mit in die Solarautomatik zu integrieren, um eine einfachere Anpassung an die Solarleistung zu ermöglichen. Auch sollte hierbei an den Umkonfigurationspunkten der Phasen versucht werden, einen Strom in der Mitte des zur Verfügung stehenden Strombereichs einzustellen, damit anschließend während der Stromregelung ein ausreichend großer Spielraum nach oben und unten zur Verfügung steht. Zum anderen wurden keinerlei Untersuchungen über die Auswirkungen der Ladeunterbrechungen auf die Akkumulatoren der Autos im Bezug auf Alterungseffekte angestellt.

Für die Zukunft von Ladesäulen ist die Verbindung mit der Norm `ISO 15118` erwähnenswert, da diese unter dem Stichwort Smart Grid, zu deutsch intelligentes Stromnetz, zu einer zukunftssicheren Ladeinfrastruktur führen soll. Dabei soll das Auto beispielsweise aktiv mit der Ladesäule kommunizieren und mitteilen, wie viel Energie bis zu welchem Zeitpunkt benötigt wird, so dass nicht zwangsläufig mit voller Leistung geladen werden muss. Weitere Ansätze wie V2G oder V2H bieten zusätzlich die Möglichkeit, Fahrzeuge als aktive Speicher im Netz zu integrieren.

# A Hardware

## A.1 GPIO-Pinout

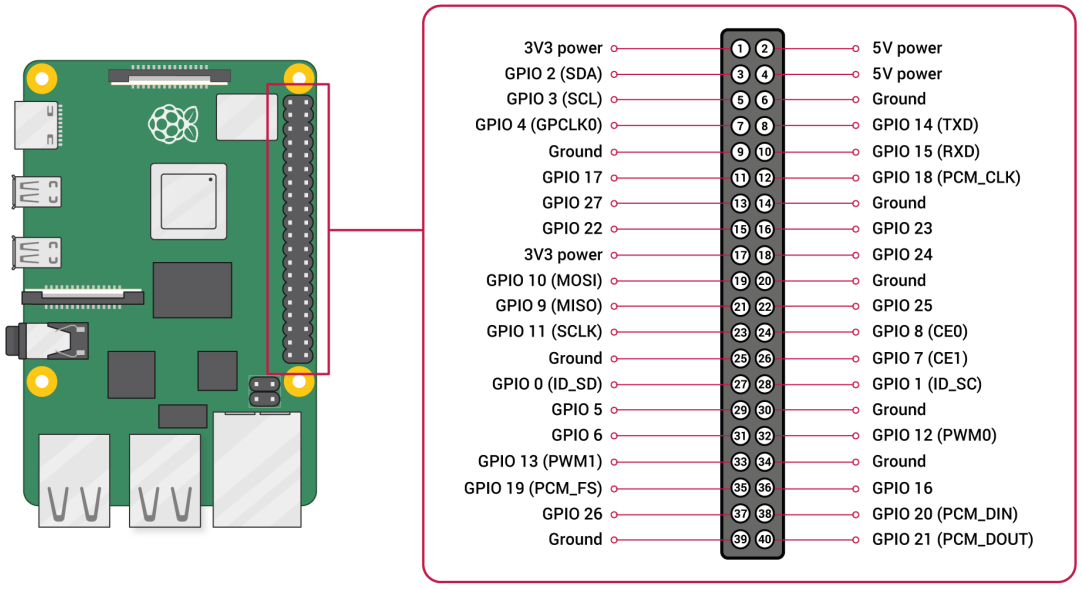


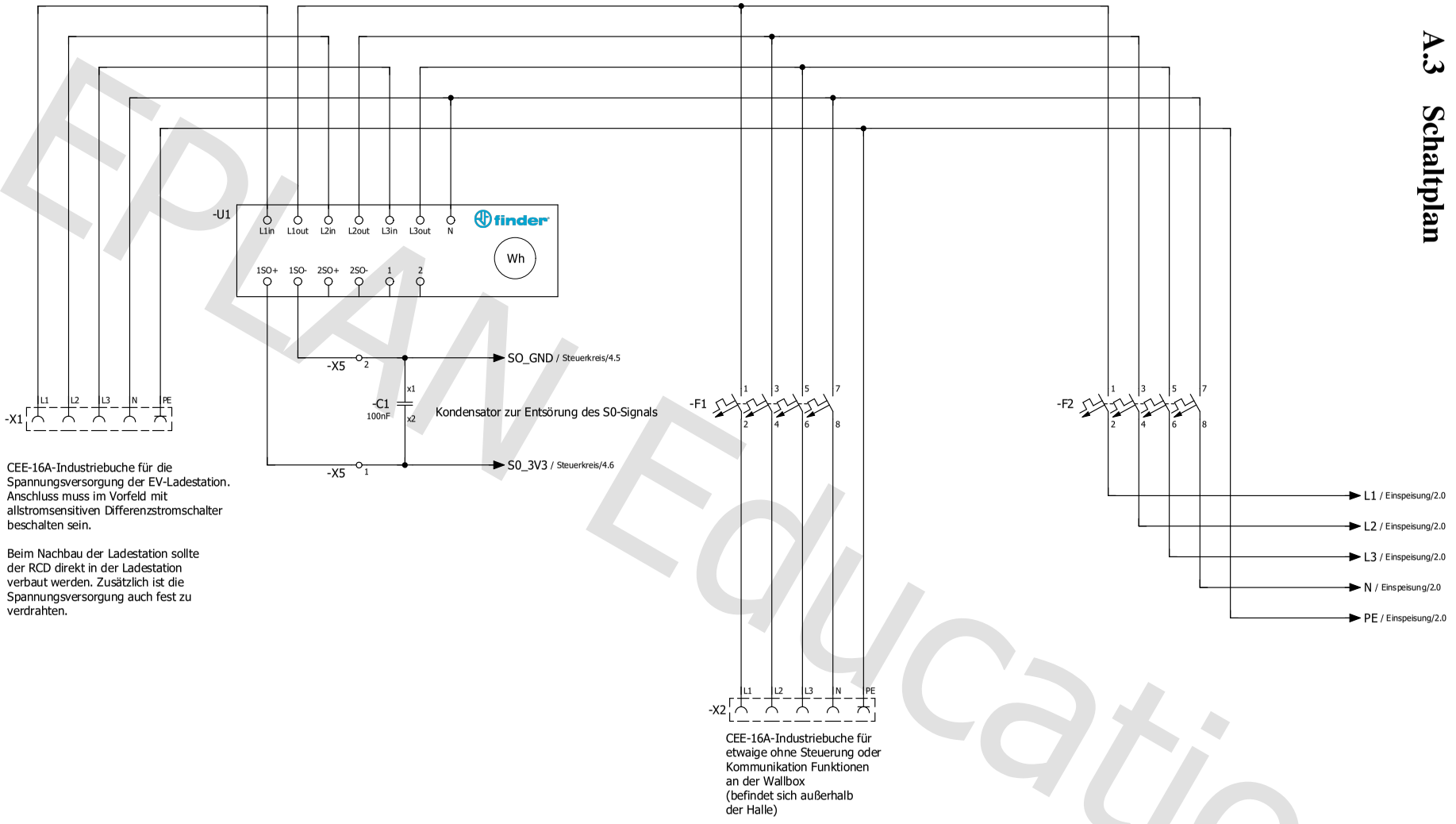

**Abb. A.1:** Darstellung der GPIO-Stiftleiste (Pinout) des Raspberry Pi 4 Model B mit Platinendraufsicht zur Orientierung und Nummerierung der Pins. Die Funktionen in den Klammern sind in der Hardware verbunden und sollten bei Bedarf der Funktion nur an diesen Pins genutzt werden.[34]

## A.2 GPIO-Funktionen

| GPIO | Default Pull | ALT0 | ALT1 | ALT2 | ALT3 | ALT4 | ALT5 |
|---|---|---|---|---|---|---|---|
| 0 | High | SDA0 | SA5 | PCLK | SPI_CE0_N | TXD2 | SD6 |
| 1 | High | SCL0 | SA4 | DE | SPI3_MISO | RXD2 | SCL6 |
| 2 | High | SDA1 | SA3 | LCD_V | SPI3_MOSI | CTS2 | SDA3 |
| 3 | High | SCL1 | SA2 | LCD_H | 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 | CTS5 | TXD1 |
| 14 | Low | TXD0 | SD6 | DPI_D10 | SPI5_MOSI | RXD5 | SCL5 |
| 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 |

**Tab. A.1:** GPIO-Funktionen des Raspberry Pi 4 Model B und deren Zuordnung [3]

# A.3 Schaltplan

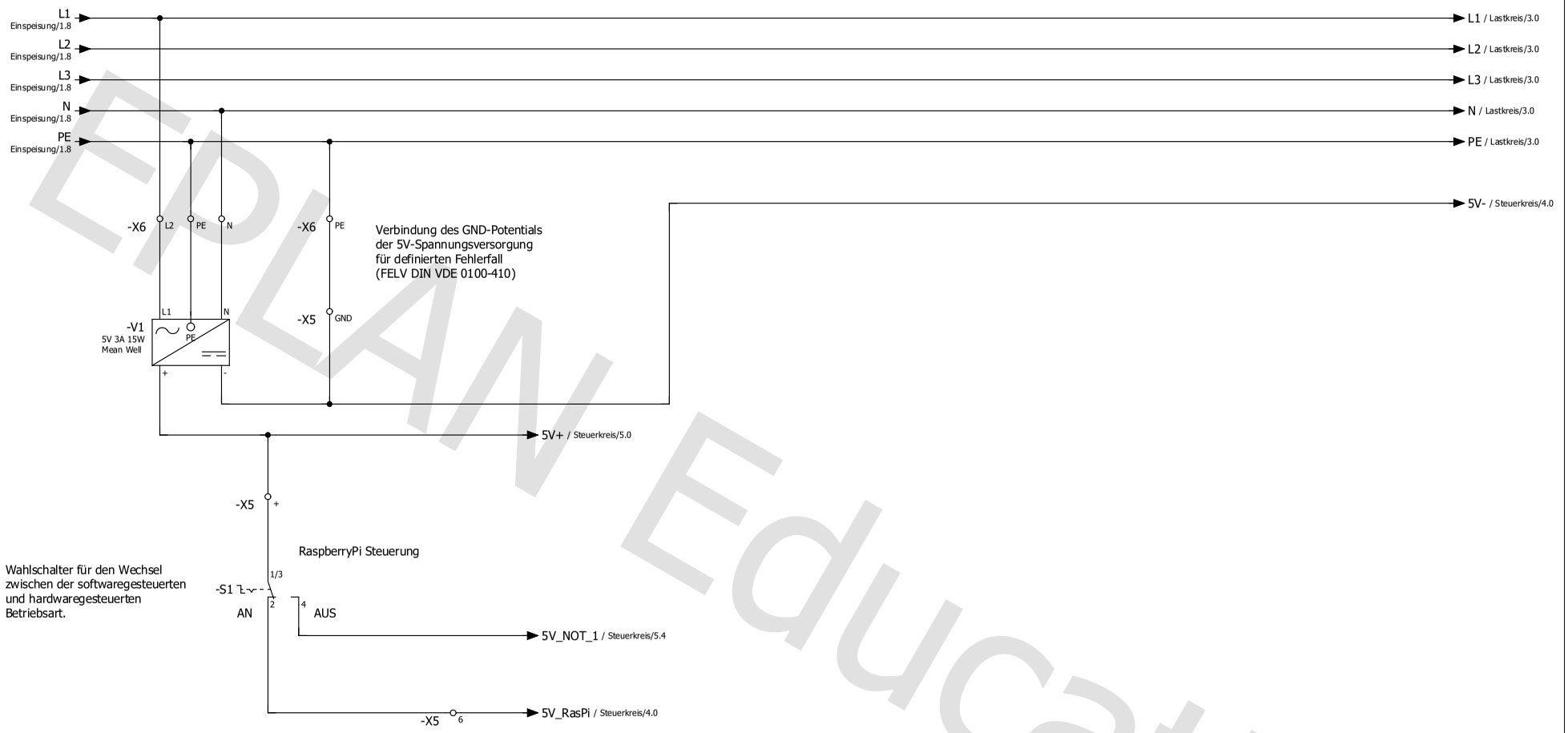







Einspeisung/2

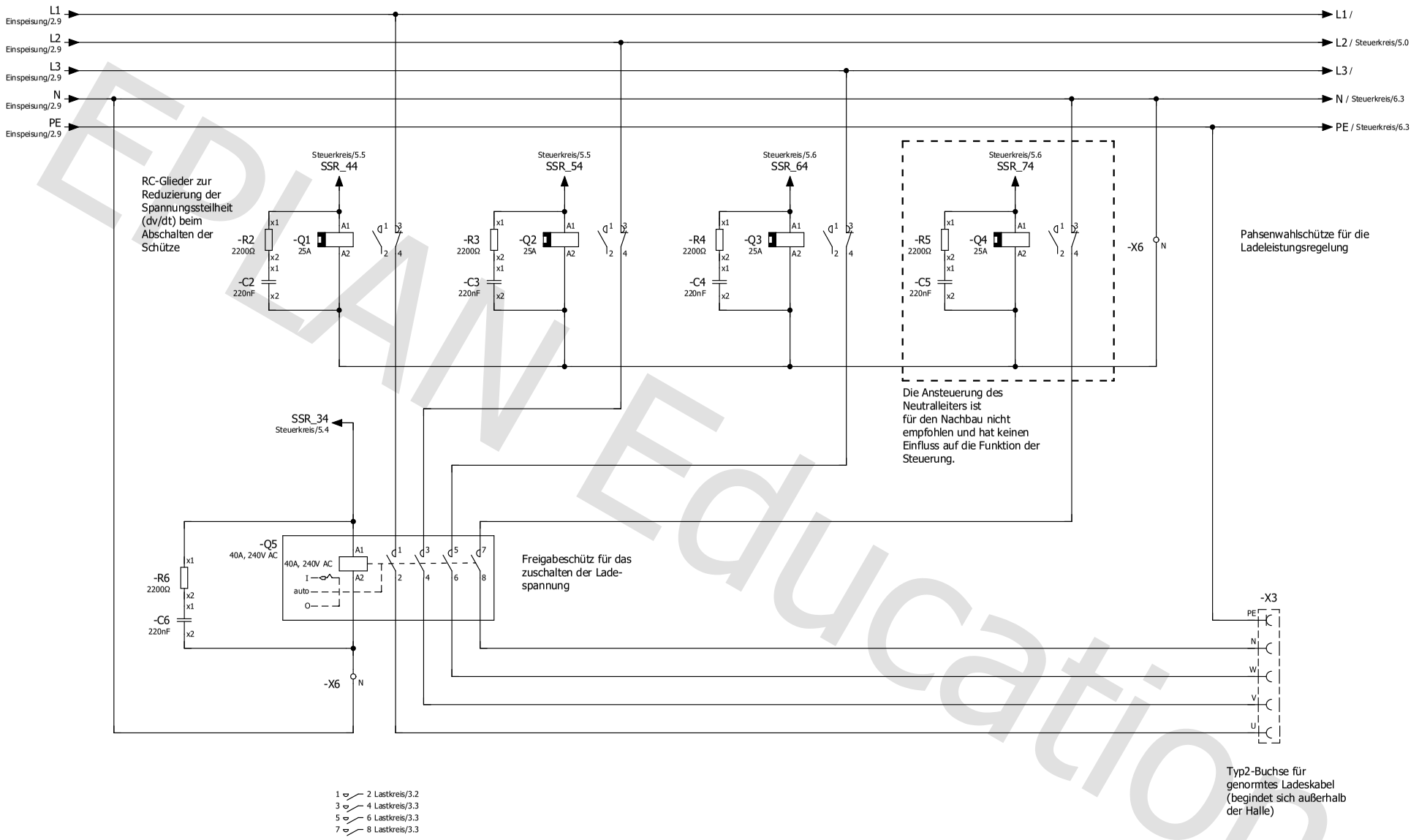
| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | | Datum | 02.12.2020 | | Lehrstuhl für Mechatronik | Einspeisung der Versorungsspannung<br>Energiezähler<br>Leitungsschutzschalter | | Einspeisung |
| | | | Bearb. | Sascha Mehringer | | | | | |
| | | | Gepr. | | SimpleCharge | | | 1 | Blatt 1 |
| Änderung | Datum | Name | Urspr. | | Ersatz von | Ersetzt durch | | | Seite 1 / 6 |






Einspeisung/1 Lastkreis/3

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | | Datum | 02.12.2020 | | Lehrstuhl für Mechatronik | Schaltschranknetzteil für 5V Steuerspannung<br>Wahlschalter für Betriebsart | | Einspeisung |
| | | | Bearb. | Sascha Mehringer | | | | | |
| | | | Gepr. | | SimpleCharge | | | 1 | Blatt 2 |
| Änderung | Datum | Name | Urspr. | | Ersatz von | Ersetzt durch | | | Seite 2 / 6 |






Einspeisung/2

Steuerkreis/4

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | | | Datum | 02.12.2020 | | Lehrstuhl für Mechatronik | Phasenwahlschütz<br>Freigabeschütz<br>RC-Glieder<br>Typ2-Buchse | Lastkreis |
| | | | Bearb. | Sascha Mehringer | | | | |
| | | | Gepr. | | SimpleCharge | | | Blatt 3 |
| Änderung | Datum | Name | Urspr. | | Ersatz von | Ersetzt durch | | Seite 3 / 6 |

5V_RasPi
Einspeisung/2.3

Raspberry Pi Steuerung

-A1
Raspberry Pi 4B 8GB

5V(USB-C)

## Raspberry Pi 4 B J8 GPIO Header

| Pin# | NAME | | NAME | Pin# |
|---|---|---|---|---|
| 01 | 3.3v DC Power | | DC Power 5v | 02 |
| 03 | GPIO02 (SDA1, I²C) | | DC Power 5v | 04 |
| 05 | GPIO03 (SCL1, I²C) | | Ground | 06 |
| 07 | GPIO04 (GPCLK0) | | (TXD0, UART) GPIO14 | 08 |
| 09 | Ground | | (RXD0, UART) GPIO15 | 10 |
| 11 | GPIO17 | | (PWM0) GPIO18 | 12 |
| 13 | GPIO27 | | Ground | 14 |
| 15 | GPIO22 | | GPIO23 | 16 |
| 17 | 3.3v DC Power | | GPIO24 | 18 |
| 19 | GPIO10 (SPI0_MOSI) | | Ground | 20 |
| 21 | GPIO09 (SPI0_MISO) | | GPIO25 | 22 |
| 23 | GPIO11 (SPI0_CLK) | | (SPI0_CE0_N) GPIO08 | 24 |
| 25 | Ground | | (SPI0_CE1_N) GPIO07 | 26 |
| 27 | GPIO00 (SDA0, I²C) | | (SCL0, I²C) GPIO01 | 28 |
| 29 | GPIO05 | | Ground | 30 |
| 31 | GPIO06 | | (PWM0) GPIO12 | 32 |
| 33 | GPIO13 (PWM1) | | Ground | 34 |
| 35 | GPIO19 | | GPIO16 | 36 |
| 37 | GPIO26 | | GPIO20 | 38 |
| 39 | Ground | | GPIO21 | 40 |

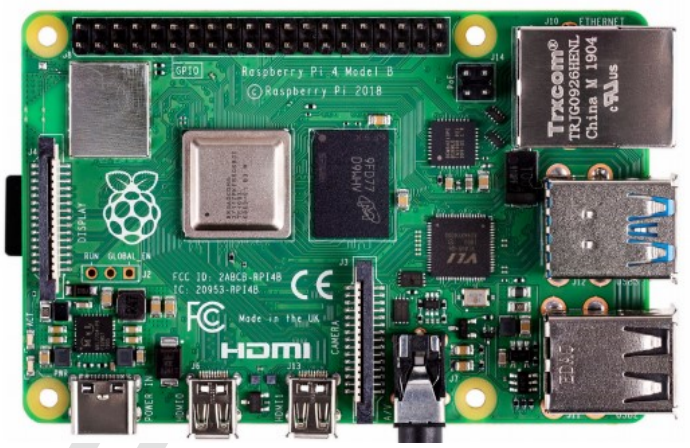
## Raspberry Pi 4 B J14 PoE Header

| | | | | |
|---|---|---|---|---|
| 01 | TR01 | | TR00 | 02 |
| 03 | TR03 | | TR02 | 04 |

### Pinout Grouping Legend

Inter-Integrated Circuit Serial Bus
Serial Peripheral Interface Bus
Ungrouped/Un-Allocated GPIO
Universal Asynchronous Receiver-Transmitter
Reserved for EEPROM

Rev. 2
19/06/2019 CGS
www.element14.com/RaspberryPi




GND(USB-C)

-X5

5V-
Einspeisung/2.9

5V- / Steuerkreis/5.0

-M1

Lüfter zur Kühlung der Steuerungseinheit

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40

-K4
Steuerkreis/6.7
13
14

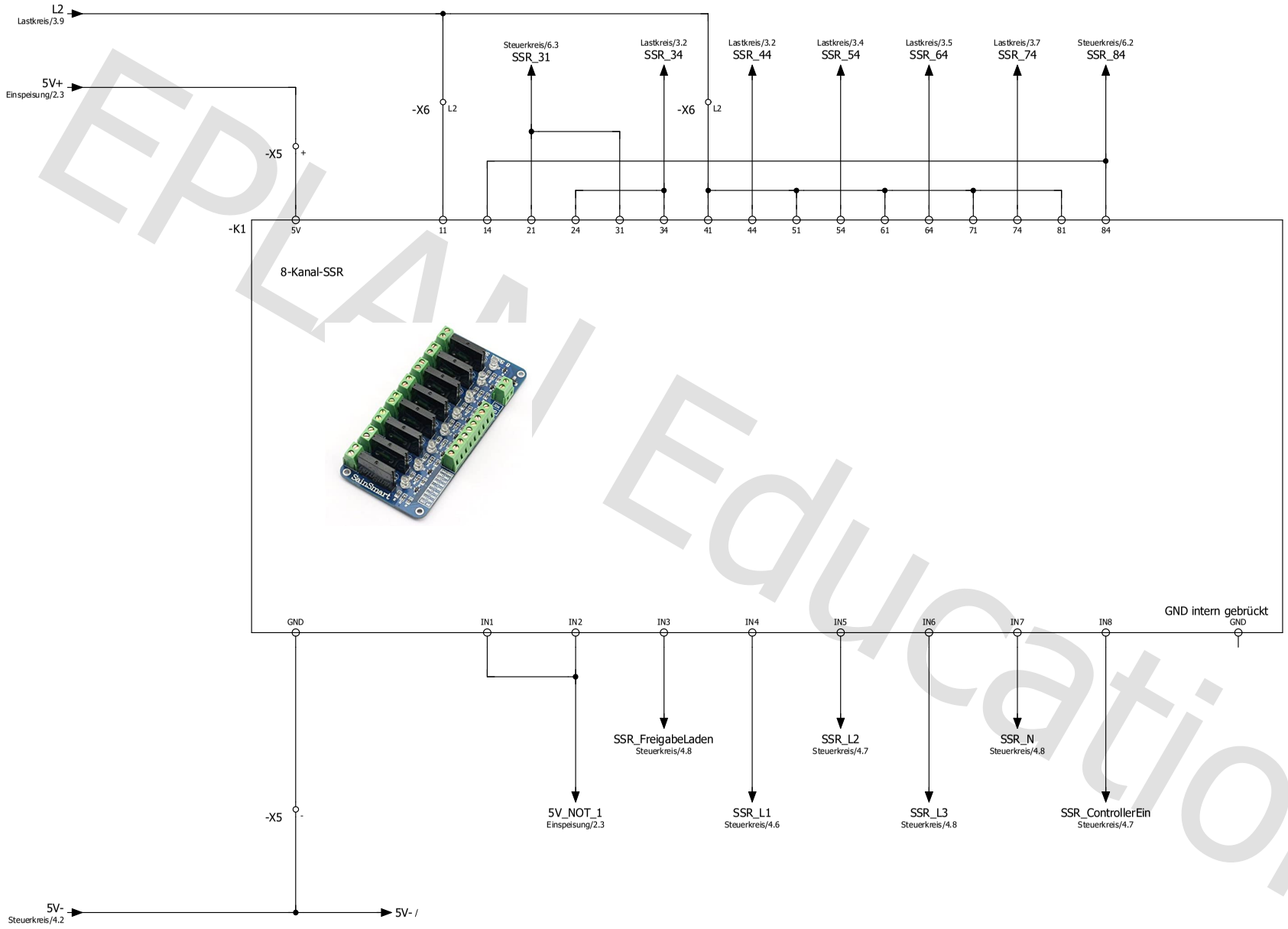
Steuerkreis/6.2 / PWM0
Einspeisung/1.3 / SO_GND
Einspeisung/1.3 / S0_3V3
/ S0_3V3_Solar
Steuerkreis/5.5 / SSR_L1
Steuerkreis/5.5 / SSR_L2
Steuerkreis/5.7 / SSR_ControllerEin
Steuerkreis/5.6 / SSR_L3
Steuerkreis/5.4 / SSR_FreigabeLaden
Steuerkreis/5.6 / SSR_N

Anschluss der S0-Schnittstelle des Energiezählers für die eingespeiste Solarleistung bei Nachbau

Lastkreis/3

Steuerkreis/5

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | | | Datum | 02.12.2020 | | Lehrstuhl für Mechatronik | Raspberry Pi Steuerung Platinenlüfter | Steuerkreis |
| | | | Bearb. | Sascha Mehringer | | | | |
| | | | Gepr. | | SimpleCharge | | | Blatt 4 |
| Änderung | Datum | Name | Urspr. | | Ersatz von | Ersetzt durch | | Seite 4 / 6 |

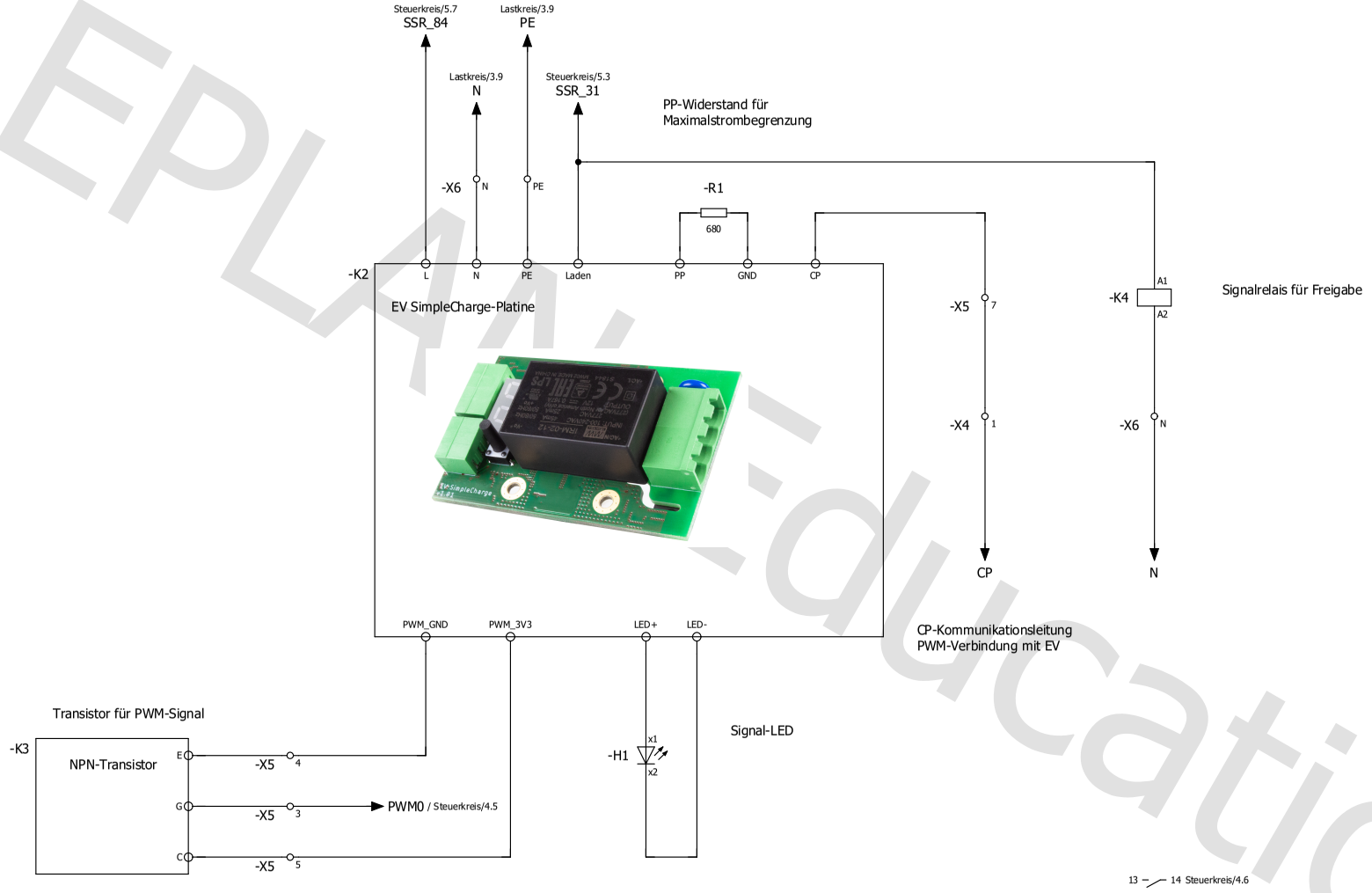





Steuerkreis/4

Steuerkreis/6

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | | | Datum | 02.12.2020 | | Lehrstuhl für Mechatronik | 8-Kanal-SSR | Steuerkreis |
| | | | Bearb. | Sascha Mehringer | | | | |
| | | | Gepr. | | SimpleCharge | | | Blatt 5 |
| Änderung | Datum | Name | Urspr. | | Ersatz von / Ersetzt durch | | 1 | Seite 5 / 6 |






Steuerkreis/5

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | | | Datum | 02.12.2020 | | Lehrstuhl für Mechatronik | EV SimpleCharge-Platine<br>Transistor für PWM-Signal<br>Signal-LED<br>Zustandsrelais | Steuerkreis |
| | | | Bearb. | Sascha Mehringer | | | | |
| | | | Gepr. | | SimpleCharge | | | Blatt 6 |
| Änderung | Datum | Name | Urspr. | | Ersatz von / Ersetzt durch | | 1 | Seite 6 / 6 |

**Tab. A.2:** Klemmplan für die Reihenklemmen -X4, -X5 und -X6 innerhalb des Schaltschrankes

| | | | | | | |
|---|---|---|---|---|---|---|
| • | • | | 12 | | • | -X4 |
| • | • | | 11 | | • | |
| • | • | | 10 | | • | |
| • | • | | 9 | | • | |
| • | • | | 8 | | • | |
| • | • | | 7 | | • K12 | |
| • | • | | 6 | | • K11 | |
| • | • | | 5 | | • Verriegelung - | |
| • | • | | 4 | | • Verriegelung + | |
| • | • | | 3 | | • | |
| • | • | | 2 | | • PP | |
| • -X5:7 | • | | 1 | | • CP | |
| • | • | | PE | | • | |
| • | • | | + | | • MeanWell + | -X5 |
| • SSR -K1:VCC | • | | + | | •-S1:1/3 | |
| • | • | | + | | • | |
| • Raspberry GND | • | | - | | • MeanWell - | |
| • SSR -K1:GND | • | | - | | • -X6:PE | |
| • | • | | - | | • | |
| • S0+ -A1:16 | • -C1:1 | | 1 | | • -U1:1S0+ | |
| • S0- -A1:14 | • -C1:2 | | 2 | | • -U1:1S0- | |
| • PWM -A1:12 | • PWM -K3:G | | 3 | | • | |
| • PWM -K3:E | • | | 4 | | • -K2:PWM+ | |
| • PWM -K3:C | • | | 5 | | • -K2:PWM- | |
| • Raspberry VCC | • | | 6 | | • -S1:2 | |
| • -X4:1 | • | | 7 | | • -K2:CP | |
| • MeanWell PE | • | | PE | | • -X5:- | -X6 |
| • | • | | PE | | • | |
| • MeanWell N | • | | N | | • -F2:4 | |
| • -K2:N | • | | N | | • -Q5:N | |
| • -Q1-4:N | • -K4:N | | N | | • | |
| • MeanWell L | • | | L2 | | • -F2:L2 | |
| • -K1:1 | • | | L2 | | • | |
| • -K1:4-8 | • | | L2 | | • | |

## A.4 Bauteile

**Tab. A.3:** Aufführung der in dieser Arbeit verwendeten Bauteile zur Realisierung des Hardwareaufbaus

| Nr. | Bezeichnung | Hersteller | Artikelnummer | Anzahl |
|---|---|---|---|---|
| 1 | RCD | EATON | FRCDM-40/4/003-G/B | 1 |
| 2 | Energiezähler | finder | 7E.78.8.400.0112 | 1 |
| 3 | Reihenleitungsschutzschalter | Siemens | 5SY6416-6 MCB B16 | 2 |
| 4 | Installationsschütz 1pol | finder | 22.32.0.230.4520 | 4 |
| 5 | Installationsschütz 4pol | Schneider Elektronik | A9C21844 Acti 9 iCT | 1 |
| 6 | Typ 2 Anschlussbuchse | Phoenix Contact | T2M3SE12-3AC32A-0 | 1 |
| 7 | Schaltschranknetzteil | Mean Well | MDR-20-5 | 1 |
| 8 | USB-C Stromkabel | Joy-it | K-1473 | 1 |
| 9 | Wahlschalter rastend | Tru Components | LAS0-A3Y-11X/21 | 1 |
| 10 | Raspberry Pi 4 8GB | RASPBERRY PI | PI4 MODEL B/8GB | 1 |
| 11 | 64 GB Speicherkarte | Samsung | MB-MC64GA/EU | 1 |
| 12 | Raspberry Hutschienengehäuse | GeeekPi | Z-0272 | 1 |
| 13 | Breakout Board | OONO | MD-D1352-1 | 1 |
| 14 | 8-Kanal-SSR | SainSmart | 8-Kanal 5V OMRON | 1 |
| 15 | Kommunikationsplatine | Pulsares | EV SimpleCharge | 1 |
| 16 | Hutschienenadapter | Wekon | WEKON-113 | 1 |
| 17 | Relais | Phoenix Contact | REL-MR-230AC/21-21AU | 1 |
| 18 | Relaissockel | Phoenix Contact | RIF-1-BSC/2X21 | 1 |
| 19 | Snubberglied | Schütz24 | RC-K3N 230 | 5 |
| 20 | Netzwerkkabeldurchführung | Harting | 09454521560 | 1 |

# B Software

## B.1 server.service

```
[Unit]
Description=ServerBoot
After=network.target

[Service]
Type=simple
ExecStart=/usr/bin/python3 -u server.py
WorkingDirectory=/home/pi/scws/
StandardOutput=inherit
StandardError=inherit
Restart=always

[Install]
WantedBy=default.target
```

Listing B.1: server.service

**Tab. B.1:** Erklärung der in server.service verwendeten Parameter

| Parameter | Erläuterung |
|---|---|
| Description | Verwendeter Name für den Service |
| After | Liste der targets auf welche vor der Ausführung gewartet wird |
| Type | Art der Programmausführung |
| ExecStart | Vom Service ausgeführter Befehl |
| WorkingDirectory | Arbeitsverzeichnis für den Befehl |
| StandardOutput | Ziel für Standardausgaben |
| StandardError | Ziel für Fehlerausgaben |
| Restart | Neustartzeitpunkte des Service |
| WantedBy | Zuordnung des Service auf Benutzer |

## B.2 Ladespezifikation

Im Folgenden werden die aufgrund der gewählten Ladespezifikation auf der Steuerungsseite (Abb. 3.17) zur Verfügung Leistungskombinationen gelistet. Dabei enthält Tab. B.2 die aufgrund des ausgewählten Maximalstroms möglichen Stromwerte. Die Tabellen B.3 bis B.14 enthalten die durch die Kombination von Maximalstrom und möglichen Phasen über das in Abschnitt 3.2.1 beschriebene Skript `erzeugung_arrays.py` ermittelten Leistungskombination, sortiert nach aufsteigender Leistung. Dabei steht bei den einzelnen Phasen eine `1` für aktiv und eine `0` für inaktiv. Der Leistungswert ergibt sich über $P = n \cdot I \cdot 230\,\mathrm{V}$, wobei n für die Anzahl aktiver Phasen steht.

**Tab. B.2:** Mögliche Stromwerte aufgrund der Stromspezifikation

| Spezifikation | Stromkonfiguration mit Schrittweite 1A |
|---|---|
| max. 16A | 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16 |
| max. 32A | 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32 |

**Tab. B.3:** Ladespezifikation 1ph mit Maximalstrom 16A

| Phase 1 | Phase 2 | Phase 3 | $I$ / A | $P$ / W |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 6 | 1380 |
| 1 | 0 | 0 | 7 | 1610 |
| 1 | 0 | 0 | 8 | 1840 |
| 1 | 0 | 0 | 9 | 2070 |
| 1 | 0 | 0 | 10 | 2300 |
| 1 | 0 | 0 | 11 | 2530 |
| 1 | 0 | 0 | 12 | 2760 |
| 1 | 0 | 0 | 13 | 2990 |
| 1 | 0 | 0 | 14 | 3220 |
| 1 | 0 | 0 | 15 | 3450 |
| 1 | 0 | 0 | 16 | 3680 |

**Tab. B.4:** Ladespezifikation 1ph mit Maximalstrom 32A

| Phase 1 | Phase 2 | Phase 3 | $I$ / A | $P$ / W |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 6 | 1380 |
| 1 | 0 | 0 | 7 | 1610 |
| 1 | 0 | 0 | 8 | 1840 |
| 1 | 0 | 0 | 9 | 2070 |
| 1 | 0 | 0 | 10 | 2300 |
| 1 | 0 | 0 | 11 | 2530 |
| 1 | 0 | 0 | 12 | 2760 |
| 1 | 0 | 0 | 13 | 2990 |
| 1 | 0 | 0 | 14 | 3220 |
| 1 | 0 | 0 | 15 | 3450 |
| 1 | 0 | 0 | 16 | 3680 |
| 1 | 0 | 0 | 17 | 3910 |
| 1 | 0 | 0 | 18 | 4140 |
| 1 | 0 | 0 | 19 | 4370 |
| 1 | 0 | 0 | 20 | 4600 |
| 1 | 0 | 0 | 21 | 4830 |
| 1 | 0 | 0 | 22 | 5060 |
| 1 | 0 | 0 | 23 | 5290 |
| 1 | 0 | 0 | 24 | 5520 |
| 1 | 0 | 0 | 25 | 5750 |
| 1 | 0 | 0 | 26 | 5980 |
| 1 | 0 | 0 | 27 | 6210 |
| 1 | 0 | 0 | 28 | 6440 |
| 1 | 0 | 0 | 29 | 6670 |
| 1 | 0 | 0 | 30 | 6900 |
| 1 | 0 | 0 | 31 | 7130 |
| 1 | 0 | 0 | 32 | 7360 |

**Tab. B.5:** Ladespezifikation 2ph mit Maximalstrom 16A

| Phase 1 | Phase 2 | Phase 3 | $I$ / A | $P$ / W |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 |
| 1 | 1 | 0 | 6 | 2760 |
| 1 | 1 | 0 | 7 | 3220 |
| 1 | 1 | 0 | 8 | 3680 |
| 1 | 1 | 0 | 9 | 4140 |
| 1 | 1 | 0 | 10 | 4600 |
| 1 | 1 | 0 | 11 | 5060 |
| 1 | 1 | 0 | 12 | 5520 |
| 1 | 1 | 0 | 13 | 5980 |
| 1 | 1 | 0 | 14 | 6440 |
| 1 | 1 | 0 | 15 | 6900 |
| 1 | 1 | 0 | 16 | 7360 |

**Tab. B.6:** Ladespezifikation 2ph mit Maximalstrom 32A

| Phase 1 | Phase 2 | Phase 3 | $I$ / A | $P$ / W |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 |
| 1 | 1 | 0 | 6 | 2760 |
| 1 | 1 | 0 | 7 | 3220 |
| 1 | 1 | 0 | 8 | 3680 |
| 1 | 1 | 0 | 9 | 4140 |
| 1 | 1 | 0 | 10 | 4600 |
| 1 | 1 | 0 | 11 | 5060 |
| 1 | 1 | 0 | 12 | 5520 |
| 1 | 1 | 0 | 13 | 5980 |
| 1 | 1 | 0 | 14 | 6440 |
| 1 | 1 | 0 | 15 | 6900 |
| 1 | 1 | 0 | 16 | 7360 |
| 1 | 1 | 0 | 17 | 7820 |
| 1 | 1 | 0 | 18 | 8280 |
| 1 | 1 | 0 | 19 | 8740 |
| 1 | 1 | 0 | 20 | 9200 |
| 1 | 1 | 0 | 21 | 9660 |
| 1 | 1 | 0 | 22 | 10120 |
| 1 | 1 | 0 | 23 | 10580 |
| 1 | 1 | 0 | 24 | 11040 |
| 1 | 1 | 0 | 25 | 11500 |
| 1 | 1 | 0 | 26 | 11960 |
| 1 | 1 | 0 | 27 | 12420 |
| 1 | 1 | 0 | 28 | 12880 |
| 1 | 1 | 0 | 29 | 13340 |
| 1 | 1 | 0 | 30 | 13800 |
| 1 | 1 | 0 | 31 | 14260 |
| 1 | 1 | 0 | 32 | 14720 |

**Tab. B.7:** Ladespezifikation 3ph mit Maximalstrom 16A

| Phase 1 | Phase 2 | Phase 3 | $I$ / A | $P$ / W |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 |
| 1 | 1 | 1 | 6 | 4140 |
| 1 | 1 | 1 | 7 | 4830 |
| 1 | 1 | 1 | 8 | 5520 |
| 1 | 1 | 1 | 9 | 6210 |
| 1 | 1 | 1 | 10 | 6900 |
| 1 | 1 | 1 | 11 | 7590 |
| 1 | 1 | 1 | 12 | 8280 |
| 1 | 1 | 1 | 13 | 8970 |
| 1 | 1 | 1 | 14 | 9660 |
| 1 | 1 | 1 | 15 | 10350 |
| 1 | 1 | 1 | 16 | 11040 |

**Tab. B.8:** Ladespezifikation 3ph mit Maximalstrom 32A

| Phase 1 | Phase 2 | Phase 3 | $I$ / A | $P$ / W |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 |
| 1 | 1 | 1 | 6 | 4140 |
| 1 | 1 | 1 | 7 | 4830 |
| 1 | 1 | 1 | 8 | 5520 |
| 1 | 1 | 1 | 9 | 6210 |
| 1 | 1 | 1 | 10 | 6900 |
| 1 | 1 | 1 | 11 | 7590 |
| 1 | 1 | 1 | 12 | 8280 |
| 1 | 1 | 1 | 13 | 8970 |
| 1 | 1 | 1 | 14 | 9660 |
| 1 | 1 | 1 | 15 | 10350 |
| 1 | 1 | 1 | 16 | 11040 |
| 1 | 1 | 1 | 17 | 11730 |
| 1 | 1 | 1 | 18 | 12420 |
| 1 | 1 | 1 | 19 | 13110 |
| 1 | 1 | 1 | 20 | 13800 |
| 1 | 1 | 1 | 21 | 14490 |
| 1 | 1 | 1 | 22 | 15180 |
| 1 | 1 | 1 | 23 | 15870 |
| 1 | 1 | 1 | 24 | 16560 |
| 1 | 1 | 1 | 25 | 17250 |
| 1 | 1 | 1 | 26 | 17940 |
| 1 | 1 | 1 | 27 | 18630 |
| 1 | 1 | 1 | 28 | 19320 |
| 1 | 1 | 1 | 29 | 20010 |
| 1 | 1 | 1 | 30 | 20700 |
| 1 | 1 | 1 | 31 | 21390 |
| 1 | 1 | 1 | 32 | 22080 |

**Tab. B.9:** Ladespezifikation 1ph/2ph mit Maximalstrom 16A

| Phase 1 | Phase 2 | Phase 3 | $I$ / A | $P$ / W |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 6 | 1380 |
| 1 | 0 | 0 | 7 | 1610 |
| 1 | 0 | 0 | 8 | 1840 |
| 1 | 0 | 0 | 9 | 2070 |
| 1 | 0 | 0 | 10 | 2300 |
| 1 | 0 | 0 | 11 | 2530 |
| 1 | 1 | 0 | 6 | 2760 |
| 1 | 0 | 0 | 13 | 2990 |
| 1 | 1 | 0 | 7 | 3220 |
| 1 | 0 | 0 | 15 | 3450 |
| 1 | 1 | 0 | 8 | 3680 |
| 1 | 1 | 0 | 9 | 4140 |
| 1 | 1 | 0 | 10 | 4600 |
| 1 | 1 | 0 | 11 | 5060 |
| 1 | 1 | 0 | 12 | 5520 |
| 1 | 1 | 0 | 13 | 5980 |
| 1 | 1 | 0 | 14 | 6440 |
| 1 | 1 | 0 | 15 | 6900 |
| 1 | 1 | 0 | 16 | 7360 |

**Tab. B.10:** Ladespezifikation 1ph/2ph mit Maximalstrom 32A

| Phase 1 | Phase 2 | Phase 3 | $I$ / A | $P$ / W | Phase 1 | Phase 2 | Phase 3 | $I$ / A | $P$ / W |
|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 27 | 6210 |
| 1 | 0 | 0 | 6 | 1380 | 1 | 1 | 0 | 14 | 6440 |
| 1 | 0 | 0 | 7 | 1610 | 1 | 0 | 0 | 29 | 6670 |
| 1 | 0 | 0 | 8 | 1840 | 1 | 1 | 0 | 15 | 6900 |
| 1 | 0 | 0 | 9 | 2070 | 1 | 0 | 0 | 31 | 7130 |
| 1 | 0 | 0 | 10 | 2300 | 1 | 1 | 0 | 16 | 7360 |
| 1 | 0 | 0 | 11 | 2530 | 1 | 1 | 0 | 17 | 7820 |
| 1 | 1 | 0 | 6 | 2760 | 1 | 1 | 0 | 18 | 8280 |
| 1 | 0 | 0 | 13 | 2990 | 1 | 1 | 0 | 19 | 8740 |
| 1 | 1 | 0 | 7 | 3220 | 1 | 1 | 0 | 20 | 9200 |
| 1 | 0 | 0 | 15 | 3450 | 1 | 1 | 0 | 21 | 9660 |
| 1 | 1 | 0 | 8 | 3680 | 1 | 1 | 0 | 22 | 10120 |
| 1 | 0 | 0 | 17 | 3910 | 1 | 1 | 0 | 23 | 10580 |
| 1 | 1 | 0 | 9 | 4140 | 1 | 1 | 0 | 24 | 11040 |
| 1 | 0 | 0 | 19 | 4370 | 1 | 1 | 0 | 25 | 11500 |
| 1 | 1 | 0 | 10 | 4600 | 1 | 1 | 0 | 26 | 11960 |
| 1 | 0 | 0 | 21 | 4830 | 1 | 1 | 0 | 27 | 12420 |
| 1 | 1 | 0 | 11 | 5060 | 1 | 1 | 0 | 28 | 12880 |
| 1 | 0 | 0 | 23 | 5290 | 1 | 1 | 0 | 29 | 13340 |
| 1 | 1 | 0 | 12 | 5520 | 1 | 1 | 0 | 30 | 13800 |
| 1 | 0 | 0 | 25 | 5750 | 1 | 1 | 0 | 31 | 14260 |
| 1 | 1 | 0 | 13 | 5980 | 1 | 1 | 0 | 32 | 14720 |

**Tab. B.11:** Ladespezifikation 1ph/3ph mit Maximalstrom 16A

| Phase 1 | Phase 2 | Phase 3 | $I$ / A | $P$ / W |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 6 | 1380 |
| 1 | 0 | 0 | 7 | 1610 |
| 1 | 0 | 0 | 8 | 1840 |
| 1 | 0 | 0 | 9 | 2070 |
| 1 | 0 | 0 | 10 | 2300 |
| 1 | 0 | 0 | 11 | 2530 |
| 1 | 0 | 0 | 12 | 2760 |
| 1 | 0 | 0 | 13 | 2990 |
| 1 | 0 | 0 | 14 | 3220 |
| 1 | 0 | 0 | 15 | 3450 |
| 1 | 0 | 0 | 16 | 3680 |
| 1 | 1 | 1 | 6 | 4140 |
| 1 | 1 | 1 | 7 | 4830 |
| 1 | 1 | 1 | 8 | 5520 |
| 1 | 1 | 1 | 9 | 6210 |
| 1 | 1 | 1 | 10 | 6900 |
| 1 | 1 | 1 | 11 | 7590 |
| 1 | 1 | 1 | 12 | 8280 |
| 1 | 1 | 1 | 13 | 8970 |
| 1 | 1 | 1 | 14 | 9660 |
| 1 | 1 | 1 | 15 | 10350 |
| 1 | 1 | 1 | 16 | 11040 |

**Tab. B.12:** Ladespezifikation 1ph/3ph mit Maximalstrom 32A

| Phase 1 | Phase 2 | Phase 3 | $I$ / A | $P$ / W | Phase 1 | Phase 2 | Phase 3 | $I$ / A | $P$ / W |
|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 10 | 6900 |
| 1 | 0 | 0 | 6 | 1380 | 1 | 0 | 0 | 31 | 7130 |
| 1 | 0 | 0 | 7 | 1610 | 1 | 0 | 0 | 32 | 7360 |
| 1 | 0 | 0 | 8 | 1840 | 1 | 1 | 1 | 11 | 7590 |
| 1 | 0 | 0 | 9 | 2070 | 1 | 1 | 1 | 12 | 8280 |
| 1 | 0 | 0 | 10 | 2300 | 1 | 1 | 1 | 13 | 8970 |
| 1 | 0 | 0 | 11 | 2530 | 1 | 1 | 1 | 14 | 9660 |
| 1 | 0 | 0 | 12 | 2760 | 1 | 1 | 1 | 15 | 10350 |
| 1 | 0 | 0 | 13 | 2990 | 1 | 1 | 1 | 16 | 11040 |
| 1 | 0 | 0 | 14 | 3220 | 1 | 1 | 1 | 17 | 11730 |
| 1 | 0 | 0 | 15 | 3450 | 1 | 1 | 1 | 18 | 12420 |
| 1 | 0 | 0 | 16 | 3680 | 1 | 1 | 1 | 19 | 13110 |
| 1 | 0 | 0 | 17 | 3910 | 1 | 1 | 1 | 20 | 13800 |
| 1 | 1 | 1 | 6 | 4140 | 1 | 1 | 1 | 21 | 14490 |
| 1 | 0 | 0 | 19 | 4370 | 1 | 1 | 1 | 22 | 15180 |
| 1 | 0 | 0 | 20 | 4600 | 1 | 1 | 1 | 23 | 15870 |
| 1 | 1 | 1 | 7 | 4830 | 1 | 1 | 1 | 24 | 16560 |
| 1 | 0 | 0 | 22 | 5060 | 1 | 1 | 1 | 25 | 17250 |
| 1 | 0 | 0 | 23 | 5290 | 1 | 1 | 1 | 26 | 17940 |
| 1 | 1 | 1 | 8 | 5520 | 1 | 1 | 1 | 27 | 18630 |
| 1 | 0 | 0 | 25 | 5750 | 1 | 1 | 1 | 28 | 19320 |
| 1 | 0 | 0 | 26 | 5980 | 1 | 1 | 1 | 29 | 20010 |
| 1 | 1 | 1 | 9 | 6210 | 1 | 1 | 1 | 30 | 20700 |
| 1 | 0 | 0 | 28 | 6440 | 1 | 1 | 1 | 31 | 21390 |
| 1 | 0 | 0 | 29 | 6670 | 1 | 1 | 1 | 32 | 22080 |

**Tab. B.13:** Ladespezifikation 1ph - 3ph mit Maximalstrom 16A

| Phase 1 | Phase 2 | Phase 3 | $I$ / A | $P$ / W |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 6 | 1380 |
| 1 | 0 | 0 | 7 | 1610 |
| 1 | 0 | 0 | 8 | 1840 |
| 1 | 0 | 0 | 9 | 2070 |
| 1 | 0 | 0 | 10 | 2300 |
| 1 | 0 | 0 | 11 | 2530 |
| 1 | 1 | 0 | 6 | 2760 |
| 1 | 0 | 0 | 13 | 2990 |
| 1 | 1 | 0 | 7 | 3220 |
| 1 | 0 | 0 | 15 | 3450 |
| 1 | 1 | 0 | 8 | 3680 |
| 1 | 1 | 1 | 6 | 4140 |
| 1 | 1 | 0 | 10 | 4600 |
| 1 | 1 | 1 | 7 | 4830 |
| 1 | 1 | 0 | 11 | 5060 |
| 1 | 1 | 1 | 8 | 5520 |
| 1 | 1 | 0 | 13 | 5980 |
| 1 | 1 | 1 | 9 | 6210 |
| 1 | 1 | 0 | 14 | 6440 |
| 1 | 1 | 1 | 10 | 6900 |
| 1 | 1 | 0 | 16 | 7360 |
| 1 | 1 | 1 | 11 | 7590 |
| 1 | 1 | 1 | 12 | 8280 |
| 1 | 1 | 1 | 13 | 8970 |
| 1 | 1 | 1 | 14 | 9660 |
| 1 | 1 | 1 | 15 | 10350 |
| 1 | 1 | 1 | 16 | 11040 |

**Tab. B.14:** Ladespezifikation 1ph - 3ph mit Maximalstrom 32A

| Phase 1 | Phase 2 | Phase 3 | $I$ / A | $P$ / W | Phase 1 | Phase 2 | Phase 3 | $I$ / A | $P$ / W |
|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 19 | 8740 |
| 1 | 0 | 0 | 6 | 1380 | 1 | 1 | 1 | 13 | 8970 |
| 1 | 0 | 0 | 7 | 1610 | 1 | 1 | 0 | 20 | 9200 |
| 1 | 0 | 0 | 8 | 1840 | 1 | 1 | 1 | 14 | 9660 |
| 1 | 0 | 0 | 9 | 2070 | 1 | 1 | 0 | 22 | 10120 |
| 1 | 0 | 0 | 10 | 2300 | 1 | 1 | 1 | 15 | 10350 |
| 1 | 0 | 0 | 11 | 2530 | 1 | 1 | 0 | 23 | 10580 |
| 1 | 1 | 0 | 6 | 2760 | 1 | 1 | 1 | 16 | 11040 |
| 1 | 0 | 0 | 13 | 2990 | 1 | 1 | 0 | 25 | 11500 |
| 1 | 1 | 0 | 7 | 3220 | 1 | 1 | 1 | 17 | 11730 |
| 1 | 0 | 0 | 15 | 3450 | 1 | 1 | 0 | 26 | 11960 |
| 1 | 1 | 0 | 8 | 3680 | 1 | 1 | 1 | 18 | 12420 |
| 1 | 0 | 0 | 17 | 3910 | 1 | 1 | 0 | 28 | 12880 |
| 1 | 1 | 1 | 6 | 4140 | 1 | 1 | 1 | 19 | 13110 |
| 1 | 0 | 0 | 19 | 4370 | 1 | 1 | 0 | 29 | 13340 |
| 1 | 1 | 0 | 10 | 4600 | 1 | 1 | 1 | 20 | 13800 |
| 1 | 1 | 1 | 7 | 4830 | 1 | 1 | 0 | 31 | 14260 |
| 1 | 1 | 0 | 11 | 5060 | 1 | 1 | 1 | 21 | 14490 |
| 1 | 0 | 0 | 23 | 5290 | 1 | 1 | 0 | 32 | 14720 |
| 1 | 1 | 1 | 8 | 5520 | 1 | 1 | 1 | 22 | 15180 |
| 1 | 0 | 0 | 25 | 5750 | 1 | 1 | 1 | 23 | 15870 |
| 1 | 1 | 0 | 13 | 5980 | 1 | 1 | 1 | 24 | 16560 |
| 1 | 1 | 1 | 9 | 6210 | 1 | 1 | 1 | 25 | 17250 |
| 1 | 1 | 0 | 14 | 6440 | 1 | 1 | 1 | 26 | 17940 |
| 1 | 0 | 0 | 29 | 6670 | 1 | 1 | 1 | 27 | 18630 |
| 1 | 1 | 1 | 10 | 6900 | 1 | 1 | 1 | 28 | 19320 |
| 1 | 0 | 0 | 31 | 7130 | 1 | 1 | 1 | 29 | 20010 |
| 1 | 1 | 0 | 16 | 7360 | 1 | 1 | 1 | 30 | 20700 |
| 1 | 1 | 1 | 11 | 7590 | 1 | 1 | 1 | 31 | 21390 |
| 1 | 1 | 0 | 17 | 7820 | 1 | 1 | 1 | 32 | 22080 |
| 1 | 1 | 1 | 12 | 8280 | | | | | |

# C Exemplarische Ladekurven

Im Folgenden sind einige exemplarische Ladekurven des Tesla Model 3 sowie des Renault Zoe bei unterschiedlichen Phasenkonfigurationen und Ladeströmen dargestellt. Es wurden die Stromverläufe aller drei Phasen (U, V und W), $I_U$, $I_V$ und $I_W$, über zwei Perioden aufgezeichnet sowie der sich für den Strom ergebende Mittelwert $I_{RMS}$ angetragen.

## C.1 Tesla Model 3

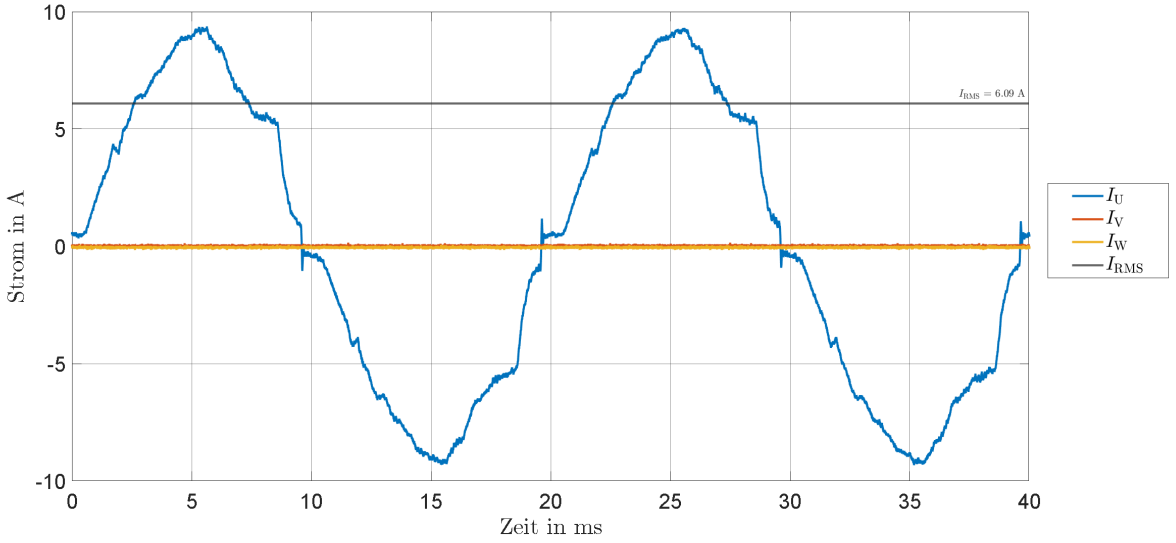



**Abb. C.1:** Ladekurven des Tesla Model 3 während des einphasigen Ladens mit einem vorgegebenen Ladestrom von 6 A und einem $I_{RMS}$ von 6,09 A

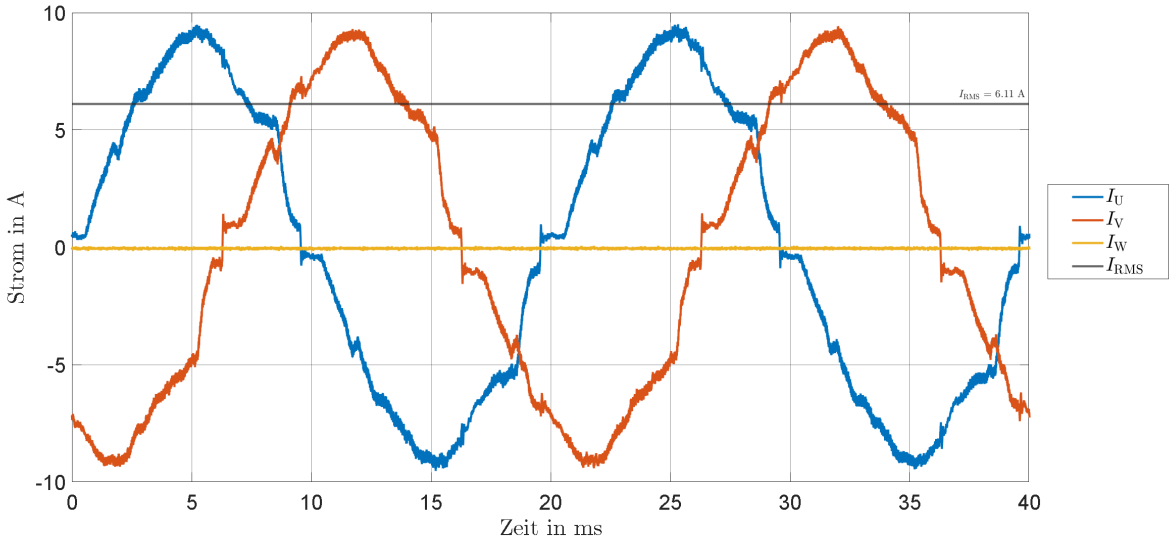



**Abb. C.2:** Ladekurven des Tesla Model 3 während des zweiphasigen Ladens mit einem vorgegebenen Ladestrom von 6 A und einem $I_{RMS}$ von 6,11 A

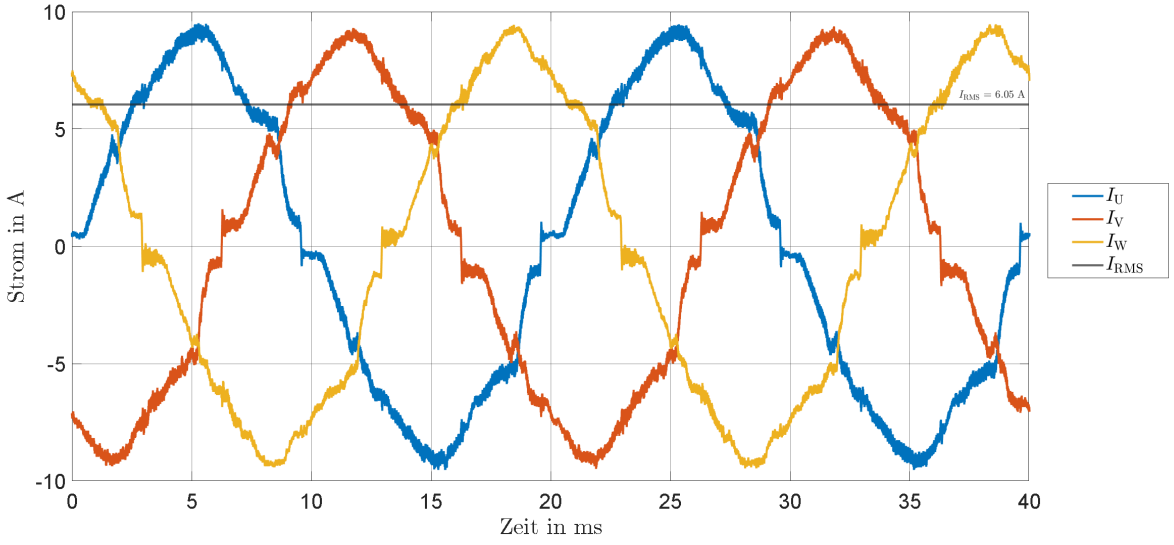



**Abb. C.3:** Ladekurven des Tesla Model 3 während des dreiphasigen Ladens mit einem vorgegebenen Ladestrom von 6 A und einem $I_{RMS}$ von 6,05 A

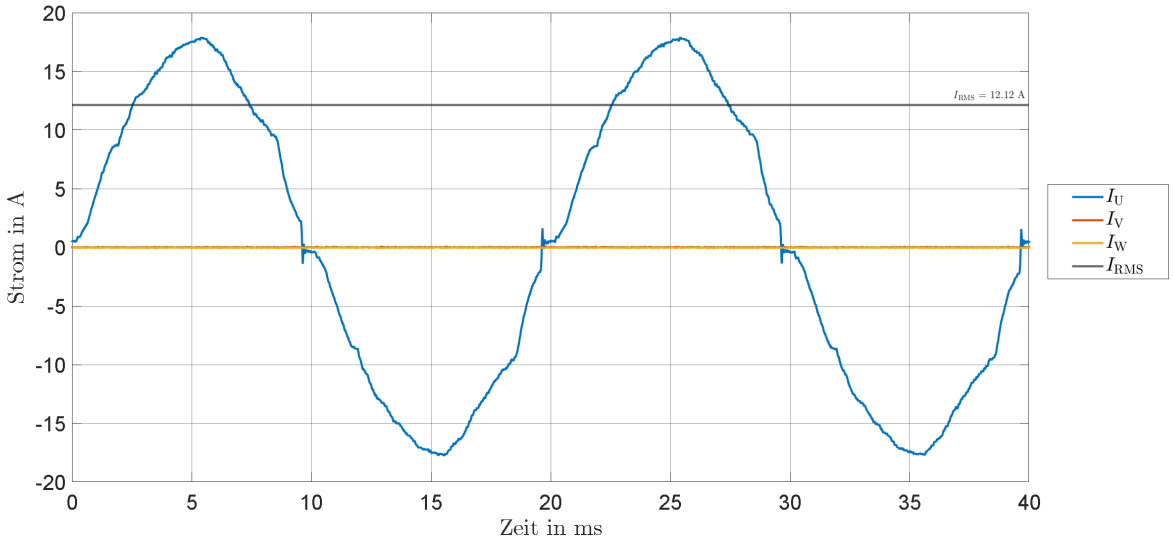



**Abb. C.4:** Ladekurven des Tesla Model 3 während des einphasigen Ladens mit einem vorgegebenen Ladestrom von 12 A und einem $I_{RMS}$ von 12,12 A

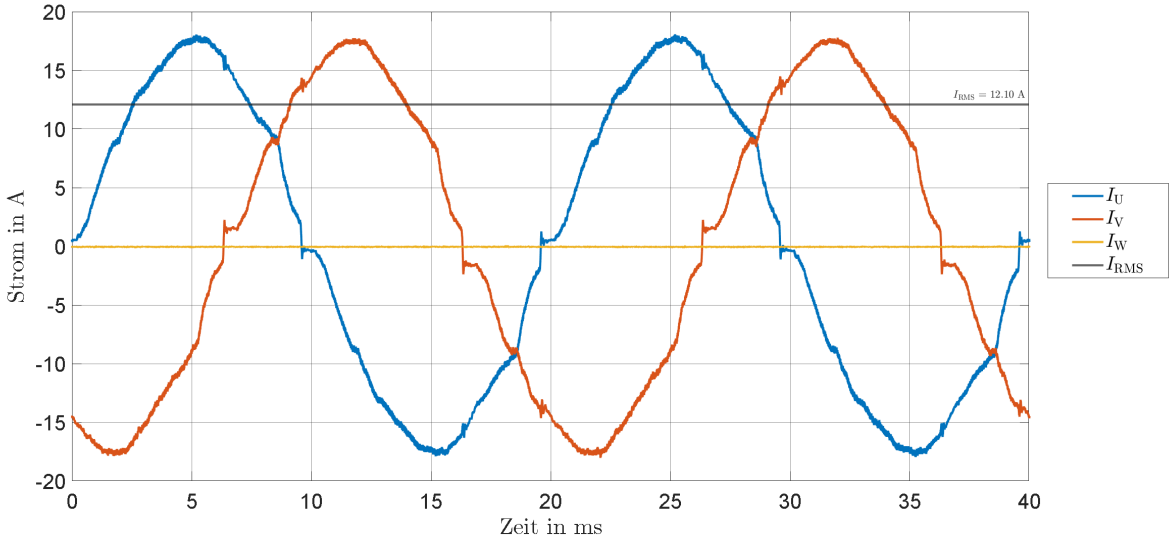



**Abb. C.5:** Ladekurven des Tesla Model 3 während des zweiphasigen Ladens mit einem vorgegebenen Ladestrom von 12 A und einem $I_{RMS}$ von 12,10 A

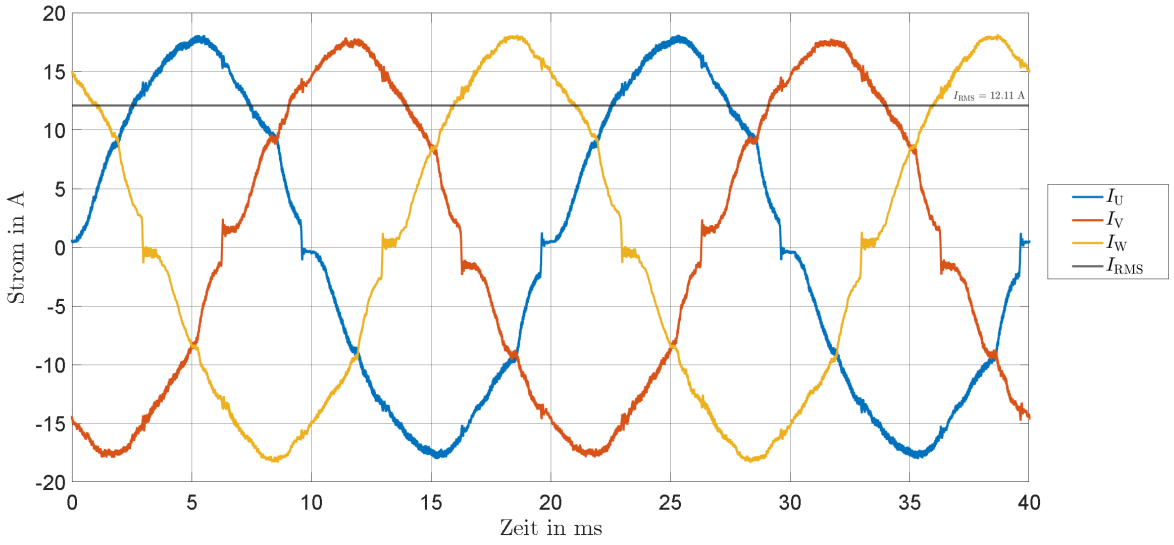



**Abb. C.6:** Ladekurven des Tesla Model 3 während des dreiphasigen Ladens mit einem vorgegebenen Ladestrom von 12 A und einem $I_{RMS}$ von 12,11 A

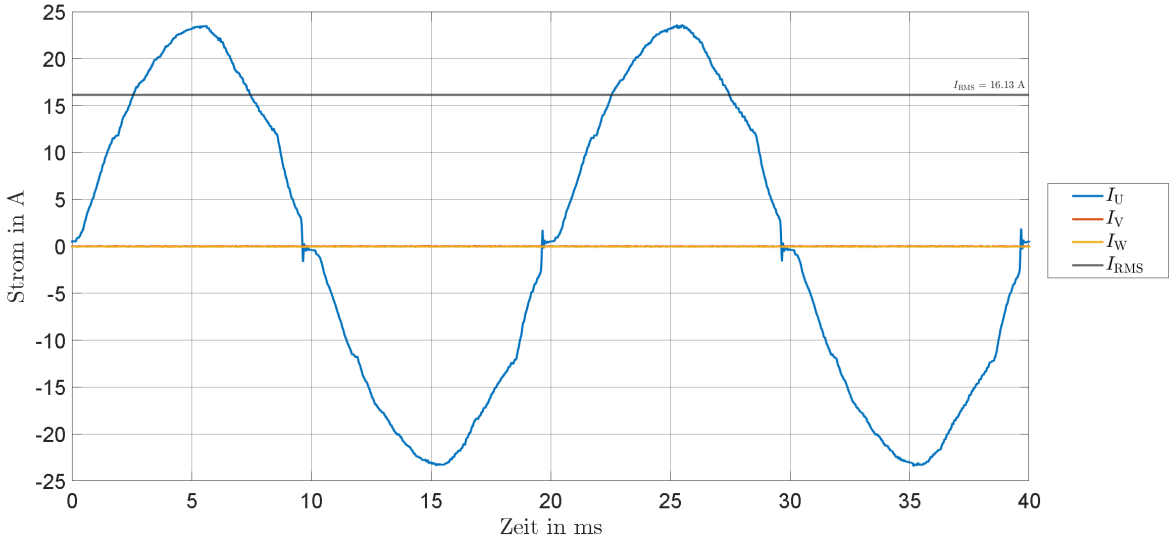



**Abb. C.7:** Ladekurven des Tesla Model 3 während des einphasigen Ladens mit einem vorgegebenen Ladestrom von 16 A und einem $I_{RMS}$ von 16,13 A

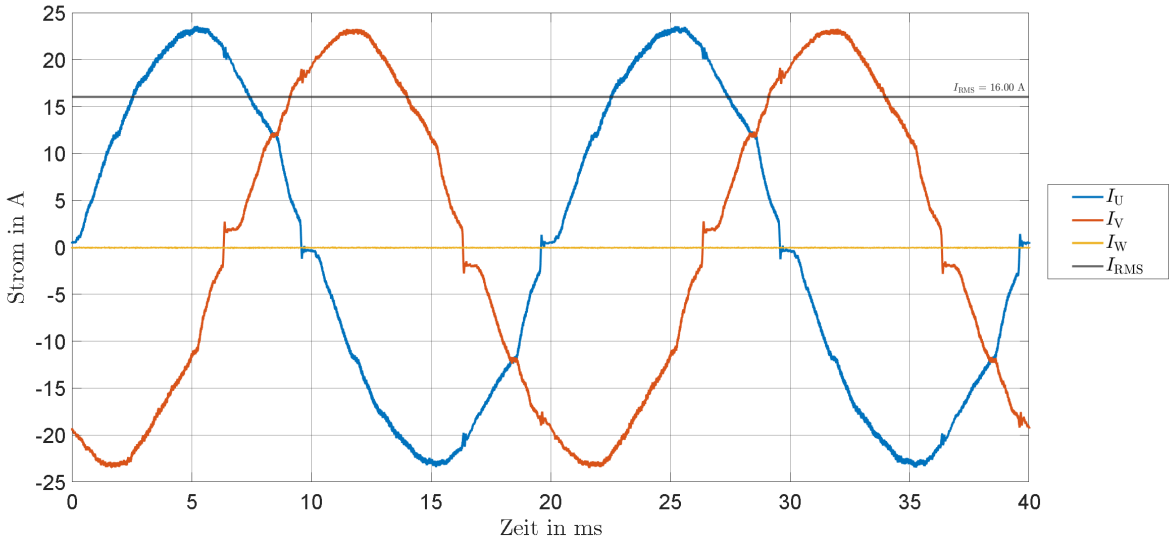



**Abb. C.8:** Ladekurven des Tesla Model 3 während des zweiphasigen Ladens mit einem vorgegebenen Ladestrom von 16 A und einem $I_{RMS}$ von 16,00 A

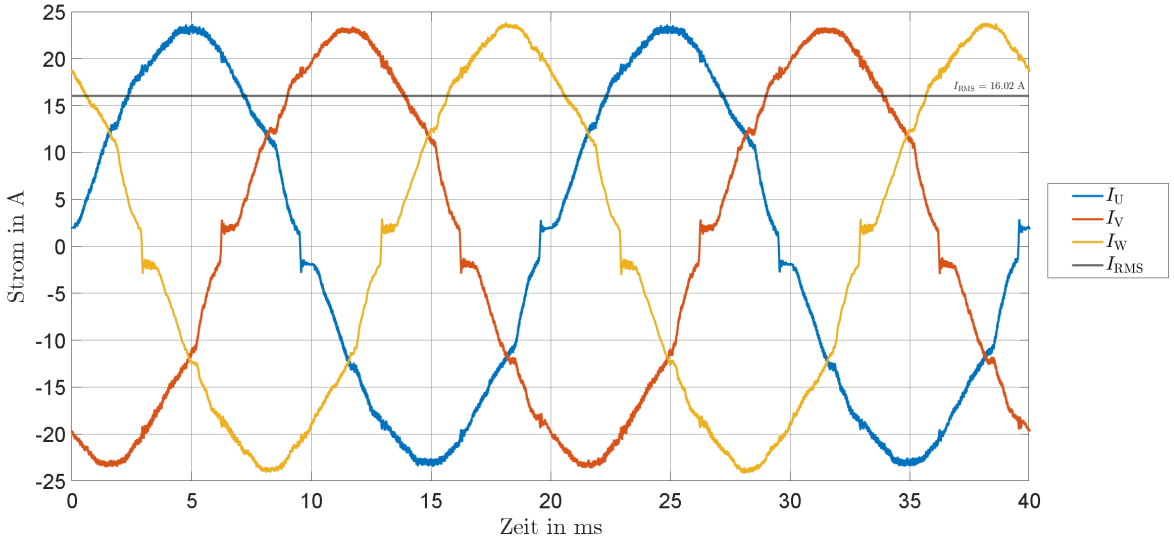



**Abb. C.9:** Ladekurven des Tesla Model 3 während des dreiphasigen Ladens mit einem vorgegebenen Ladestrom von 16 A und einem $I_{RMS}$ von 16,02 A

## C.2 Renault Zoe

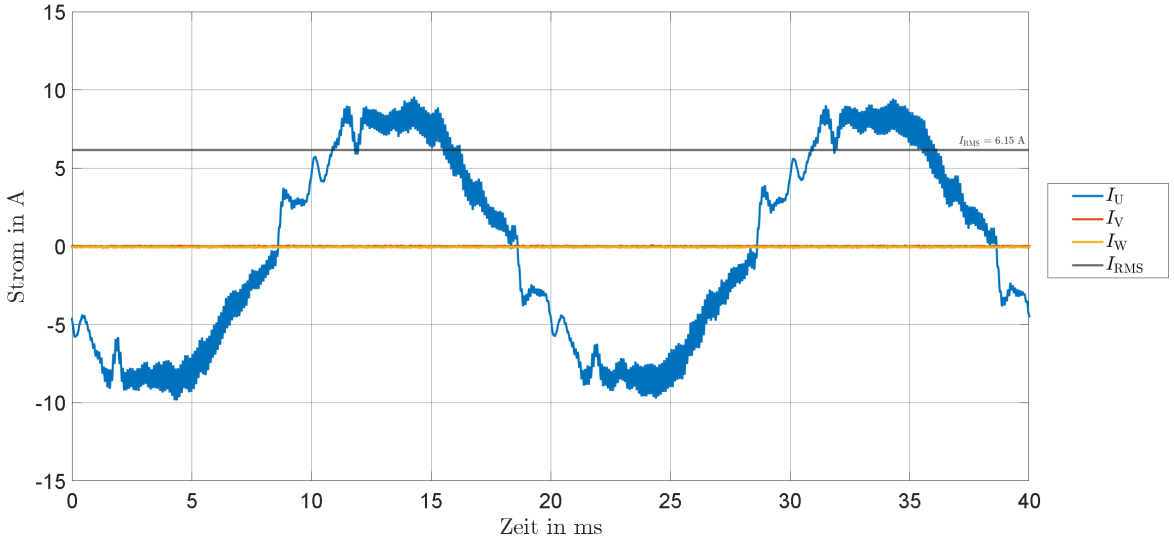



**Abb. C.10:** Ladekurven des Renault Zoe während des einphasigen Ladens mit einem vorgegebenen Ladestrom von 6 A und einem $I_{RMS}$ von 6,15 A

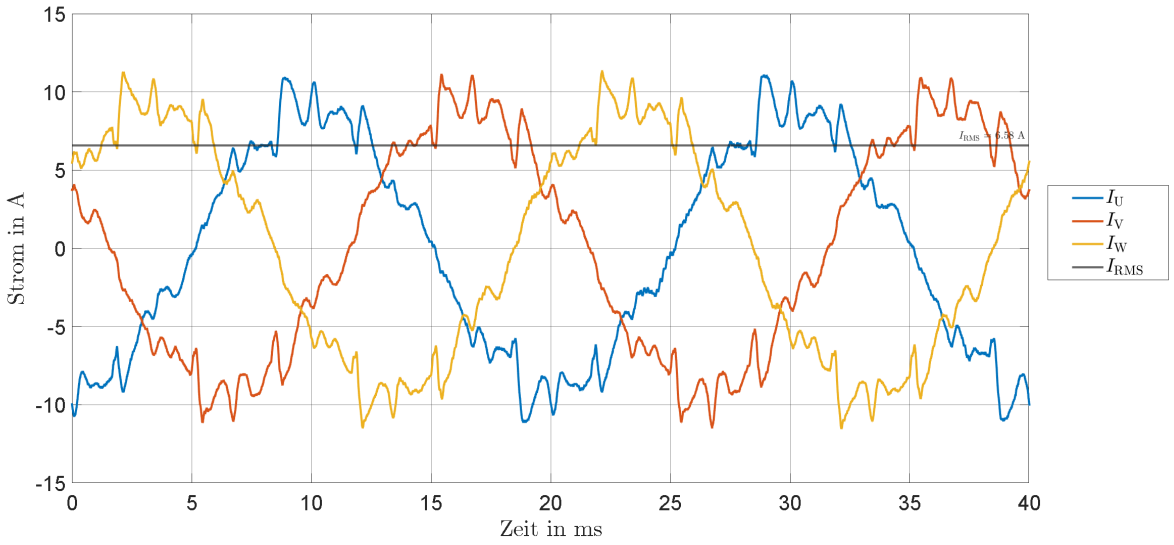



**Abb. C.11:** Ladekurven des Renault Zoe während des dreiphasigen Ladens mit einem vorgegebenen Ladestrom von 6 A und einem $I_{RMS}$ von 6,58 A

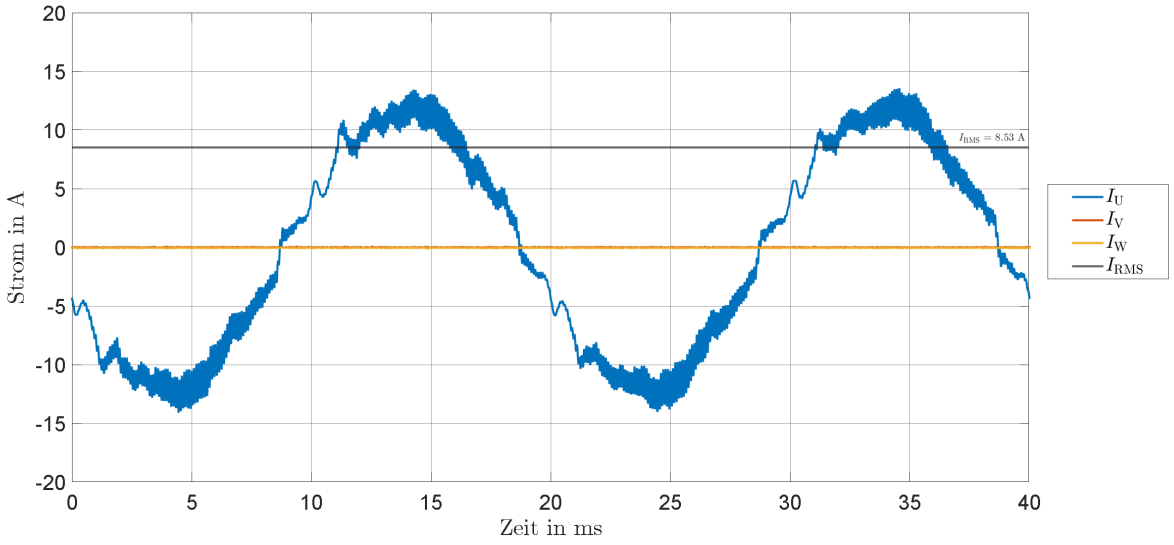



**Abb. C.12:** Ladekurven des Renault Zoe während des einphasigen Ladens mit einem vorgegebenen Ladestrom von 10 A und einem $I_{RMS}$ von 8,53 A

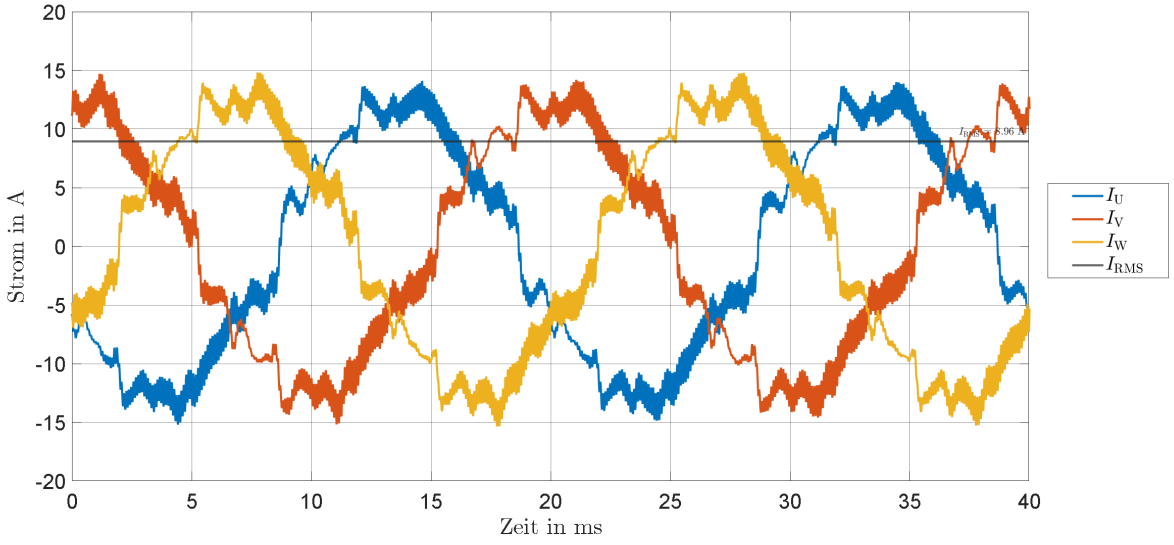



**Abb. C.13:** Ladekurven des Renault Zoe während des dreiphasigen Ladens mit einem vorgegebenen Ladestrom von 10 A und einem $I_{RMS}$ von 8,96 A

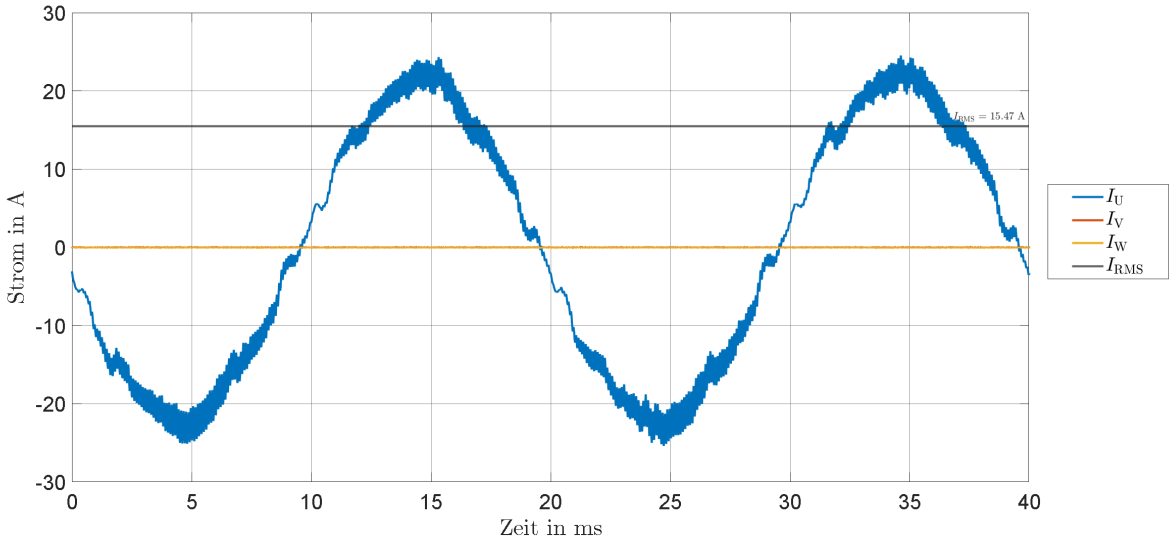



**Abb. C.14:** Ladekurven des Renault Zoe während des einphasigen Ladens mit einem vorgegebenen Ladestrom von 16 A und einem $I_{RMS}$ von 15,47 A

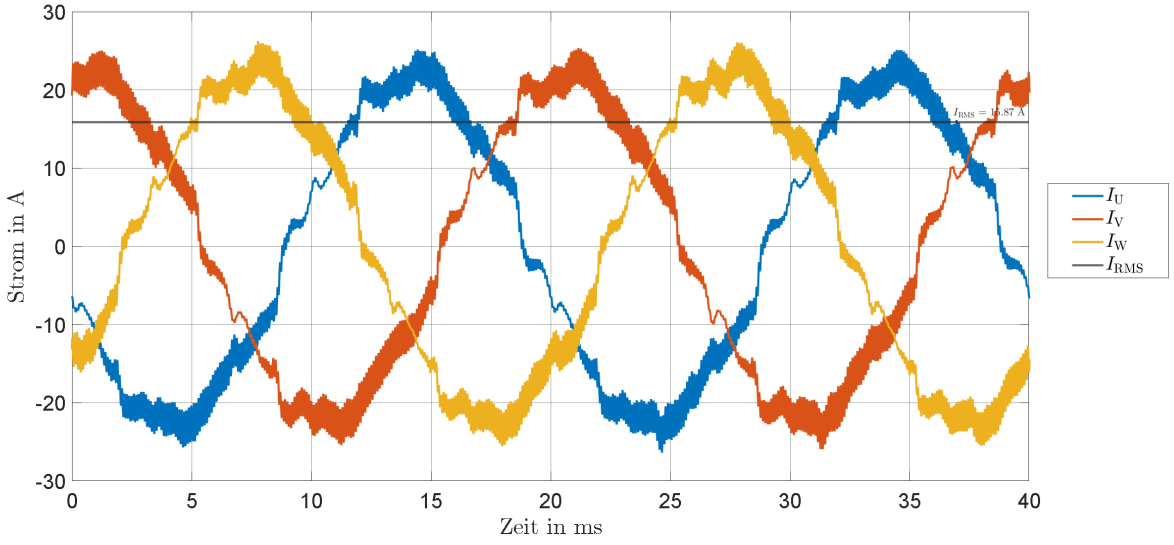



**Abb. C.15:** Ladekurven des Renault Zoe während des dreiphasigen Ladens mit einem vorgegebenen Ladestrom von 16 A und einem $I_{RMS}$ von 15,87 A

# D Literatur


[1] P. Komarnicki, J. Haubrock und Z. A. Styczynski. *Elektromobilität und Sektorenkopplung*. Springer Berlin Heidelberg, 2018. DOI: 10.1007/978-3-662-56249-9.

[2] B. K. Sovacool, J. Axsen und W. Kempton. „The Future Promise of Vehicle-to-Grid (V2G) Integration: A Sociotechnical Review and Research Agenda“. In: *Annual Review of Environment and Resources* 42.1 (Okt. 2017), S. 377–406. DOI: 10.1146/annurev-environ-030117-020220.

[3] *Raspberry Pi 4 Computer Model B*. 1. Aufl. Raspberry Pi (Trading) Ltd. Mai 2020.

[4] P. Schnabel. *Raspberry Pi: GPIO beschalten*. URL: https://www.elektronik-kompendium.de/sites/raspberry-pi//2006031 (besucht am 22. 09. 2020).

[5] *BCM2711 ARM Peripherals*. 1. Aufl. Raspberry Pi (Trading) Ltd. Feb. 2020.

[6] *Solid State Relay G3MB Low cost Subminiature PCB mounting 2 amp Single in-line package (SIP) SSR*. OMRON ELECTRONICCOMPONENTS LLC. 55 E. Commerce Drive, Suite BSchaumburg, IL 60173, Juni 2009.

[7] DIN EN 62196-1:2015-06, Stecker, Steckdosen, Fahrzeugkupplungen und Fahrzeugstecker - Konduktives Laden von Elektrofahrzeugen - Teil 1: Allgemeine Anforderungen (IEC 62196-1:2014, modifiziert).

[8] DIN EN IEC 61851-1 VDE 0122-1:2019-12 Konduktive Ladesysteme für ElektrofahrzeugeTeil 1: Allgemeine Anforderungen (IEC 61851-1:2017).

[9] *7E.78.8.400.0112 Dreiphasiger, MID geeichter, Energiezähler (80 A) für 3 oder 4 Leiteranschluss*. FINDER S.p.A. sole proprietorship. URL: https://findernet-cms-s3.s3.eu-west-1.amazonaws.com/app/uploads/2020/09/11050900/IB7E78_0112DE.pdf (besucht am 07. 10. 2020).

[10] DIN EN 62053-31:1999-04 Einrichtungen zur Messung der elektrischen Energie (AC) - Besondere Anforderungen - Teil 31: Impulseinrichtungen für Induktionszähler oder elektronische Zähler (nur Zweidrahtsysteme) (IEC 62053-31:1998).

[11] *A9C21844 Installationsschütz iCT 40A 4S 220/240V 50Hz*. Schneider Electric GmbH. Gothaer Straße 2940880 Ratingen, Okt. 2020. URL: https://www.se.com/de/de/product/download-pdf/GC4040M6 (besucht am 13. 10. 2020).

[12] *22.32 Installationsschütze 25 A*. FINDER S.p.A. sole proprietorship. Juni 2019. URL: https://findernet-cms-s3.s3.eu-west-1.amazonaws.com/app/uploads/2020/09/11030217/IB2232DE.pdf (besucht am 13. 10. 2020).

[13] *Einzelrelais - REL-MR-230AC/21-21AU - 2961480*. Phoenix Contact GmbH. Ada-Christen-Gasse 41100 Wien. URL: https://www.phoenixcontact.com/at/produkte/2961480 (besucht am 14. 10. 2020).

[14] *Mean Well 20W Single Output Industrial DIN Rail Power Supply*. Mean Well Enterprise. Juni 2019. URL: https://www.meanwell.com/webapp/product/search.aspx?prod=MDR-20 (besucht am 14. 10. 2020).

[15] *PyCharm - The Python IDE for Professional Developers*. https://www.jetbrains.com/pycharm/. Aufgerufen: 12.01.2021.

[16] https://www.jetbrains.com/help/pycharm/quick-start-guide.html. Aufgrufen: 13.01.2021.

[17] M. Weigend. *Raspberry Pi programmieren mit Python*. ger. 4. Aufl. mitp-Verlag. ISBN: 9783958459137. URL: http://www.content-select.com/index.php?id=bib_view&ean=9783958459137.

[18] *Flask - web development, one drop at a time*. https://flask.palletsprojects.com/. Aufgerufen: 12.01.2021.

[19] P. Barry. *Python von Kopf bis Fuß*. O'Reilly Verlag, 2017. ISBN: 9783960101352.

[20] D. R. Brooks. *Programming in HTML and PHP*. Springer, 2017. ISBN: 978-3-319-56973-4.

[21] P. W. Jeremy McPeak. *Beginning JavaScript*. John Wiley & Sons, Inc., 2015. ISBN: 978-1-118-90333-9.

[22] DIN EN 81346-1:2010-05 Industrielle Systeme, Anlagen und Ausrüstungen und Industrieprodukte - Strukturierungsprinzipien und Referenzkennzeichnung - Teil 1: Allgemeine Regeln (IEC 81346-1:2009).

[23] DIN VDE 0100-722:2019-06 Errichten von Niederspannungsanlagen - Teil 7-722: Anforderungen für Betriebsstätten, Räume und Anlagen besonderer Art - Stromversorgung von Elektrofahrzeugen (IEC 60364-7-722:2018).

[24] DIN VDE 0100-410:2018-10 Errichten von Niederspannungsanlagen - Teil 4-41: Schutzmaßnahmen - Schutz gegen elektrischen Schlag (IEC 60364-4-41:2005, modifiziert + A1:2017, modifiziert).

[25] https://bulma.io/. Aufgerufen: 19.01.2021.

[26] https://wallbox-info.de/. Aufgerufen: 05.02.2021.

[27] https://blog.renault.de/neues-von-renault-elektroauto-zoe-aufladen-reichweite-erweitern/. Aufgerufen: 09.02.2021.

[28] *Operator's ManualWaveRunner 8000 Oscilloscopes*. Teledyne LeCroy, Inc. Jan. 2019. URL: http://cdn.teledynelecroy.com/files/manuals/waverunner-8000-operators-manual.pdf (besucht am 16. 01. 2021).

[29] *Instruction Manual TT-SI 9101 100MHz Active Differential Probe*. Testec Elektronik GmbH. URL: https://www.testec.de/assets/pdf/TT-SI/TT-SI-9101_Manual_EN.pdf (besucht am 16. 01. 2021).

[30] *AC/DC Current Measurement Systems*. Tektronix. Mai 2019. URL: file:///C:/Downloads/TCPA-Amplifiers-and-TCP300-400-Probes-Datasheet-60W1645812.pdf (besucht am 16. 01. 2021).

[31] *WaveStation Function/ArbitraryWaveform Generators*. Teledyne LeCroy, Inc. Apr. 2015. URL: http://cdn.teledynelecroy.com/files/pdf/wavestation_datasheet.pdf (besucht am 16. 01. 2021).

[32] *4 Kanal Leistungsmessgerät LMG450*. ZES ZIMMER Electronic Systems GmbH. März 2011.

[33] *PulsAres Betriebsanleitung EV SimpleCharge*. 6. Aufl. Pulsares GmbH. Steinbreite 3 31688 Nienstädt, Aug. 2020. URL: https://www.pulsares.shop/evsimplecharge.html.

[34] *Darstellung der Stiftleiste des Raspberry Pi (Pinout)*. URL: https://www.raspberrypi.org/documentation/usage/gpio/images/GPIO-Pinout-Diagram-2.png (besucht am 23. 09. 2020).