Zur Erlangung des Grades

Bachelor of Science

im Studiengang Elektrotechnik – Elektronik – Informationstechnik

Fachrichtung Mikroelektronik

an der Technischen Fakultät, Departement EEI,

der Friedrich-Alexander-Universität Erlangen-Nürnberg

Entwicklung, Aufbau und Test einer Ladeeinrichtung für Elektrofahrzeuge nach IEC 62196

Bachelorarbeit

Vorgelegt von Pascal Thurnherr[k1]

Matrikel-Nr. 22379926

Erlangen, 31.08.2020

Betreuung: Prof. Dr.-Ing. Thomas Moor

Lehrstuhl für Regelungstechnik

Friedrich-Alexander-Universität Erlangen-Nürnberg

Eidesstattliche Erklärung

Ich versichere, dass ich die Arbeit ohne fremde Hilfe und ohne Benutzung anderer als der angegebenen Quellen angefertigt habe und dass die Arbeit in gleicher oder ähnlicher Form noch keiner anderen Prüfungsbehörde vorgelegen hat und von dieser als Teil einer Prüfungsleistung angenommen wurde. Alle Ausführungen, die wörtlich oder sinngemäß übernommen wurden, sind als solche gekennzeichnet.

Erlangen, der 31.08.2020

Pascal Thurnherr

Inhalt

Eidesstattliche Erklärung 2

Abbildungsverzeichnis 5

Tabellenverzeichnis 6

Quellenverzeichnis 6

1. Einleitung 8

2. Theorie und Hintergrund 10

2.1 Einführung 10

2.2 SAE J1772 und IEC/DIN 62196 11

2.3 Combined Charging System (CCS) 15

2.4 Technische Umsetzung der SAE J1772-Signalisierung 15

2.5 Mikrocontroller und eingebettete Systeme 16

3. Anforderungen 18

4. Elektronikentwicklung 21

4.1 Konzept und Werkzeuge 21

4.2 Schaltungsdesign 22

4.2.1 Bauteile und Verschaltung 22

4.2.2 Pinbelegungen 27

4.3 Leiterplattenlayout 30

4.4 Prototypenbau 32

4.4.1 Bauteilbeschaffung 32

4.4.2 Bestückung 32

4.4.3 Revisionsliste 33

5. Firmwareentwicklung 35

5.1 Konzept 35

5.1.1 Bootloader 35

5.1.2 Anwendungsfirmware 37

5.2 Entwicklungsumgebung 38

5.3 Funktionale Beschreibung 40
5.3.1 Main 40
5.3.2 UART 42
5.3.3 CMD 43
6. Test und Validierung 45
6.1 Testaufbau 45
6.1.1 Labortests 45
6.1.2 Praxistest 46
6.2 Ergebnisse 47
7. Fazit 47
Anhang A – Schaltpläne 49
Anhang B – Materialstückliste 52

Abbildungsverzeichnis

Abbildung 1: Beispielhafte Verschaltung von EV und Ladestation für Mode-3-Wechselstromladen ..... **Fehler! Textmarke nicht definiert.**
Abbildung 2: Beispielschaltung für die technische Implementierung der SAE J1772-Signalisierung 16
Abbildung 3: Simulation der Messschaltung für Stromwandler-Signale 26
Abbildung 4: Pinbelegung des ATmega4808 27
Abbildung 5: Testaufbau der Ladestation 46
Abbildung 6: FGCCS-Ctrl22-Schaltplan, Seite 1: Netzspannungsanschlüsse, Halbleiterrelais und Spannungsversorgung 49
Abbildung 7: FGCCS-Ctrl22-Schaltplan, Seite 2 – Mikrocontroller, WLAN-Modul und Peripherie 50
Abbildung 8: FGCCS-Ctrl22-Schaltplan, Seite 3 – Analogschaltung und Signalterminals 51

Tabellenverzeichnis

Tabelle 1: Ladezustände nach SAE J1772 12
Tabelle 2: Ladekabel-Kodierung der maximalen Stromstärke nach IEC 61851 13
Tabelle 3: Verfügbarer Ladestrom in Abhängigkeit der Pulsweite nach IEC 61851-1 14
Tabelle 4: Pinbelegung des ESP32-WROOM-Moduls 29
Tabelle 5: Serielles Kommunikationsprotokoll zur Fernsteuerung der Ladestation 43
Tabelle 6: Materialstückliste 53

Quellenverzeichnis

Bernstein, H. (2015). *Mikrocontroller.* Wiesbaden: Springer Vieweg.

Brünner, G. (13. Dezember 2009). *Mikrocontroller.net*. Abgerufen am 19. August 2020 von https://www.mikrocontroller.net/topic/159753

Carius, F. (10. Dezember 2017). *www.msxfaq.de*. Abgerufen am 19. August 2020 von https://www.msxfaq.de/sonst/stromer/ladetechnik.htm

Elektrotechnik, Deutsche Kommission. (Januar 2020). *www.vde.com.* (B. D. VDE, Hrsg.) Abgerufen am 19. August 2020 von Verband der Elektrotechnik Elektronik und Informationstechnik e.V.: https://shop.vde.com/de/der-technische-leitfaden-ladeinfrastruktur-elektromobilit%C3%A4t

Espressif Systems. (September 2019). *ESP32-WROOM-32 Datasheet.* Abgerufen am 19. August 2020 von https://www.espressif.com/sites/default/files/documentation/esp32-wroom-32_datasheet_en.pdf

Hüning, F. (2019). *Embedded Systems für IoT.* Berlin: Springer Vieweg.

IEC (Hrsg.). (03. Februar 2011). *A step forward for global EV roll-out*. Abgerufen am 19. August 2020 von https://www.iec.ch/newslog/2011/nr0411.htm

IEC (Hrsg.). (2014). *IEC 60309-1:1999/AMD1:2005/COR1:2014.* Genf: IEC.

IEC (Hrsg.). (2014). *IEC 62196-1:2014.* Genf: IEC.

IEC (Hrsg.). (2016). *IEC 62196-2:2016.* Genf: IEC.

IEC (Hrsg.). (2017). *IEC 61851-1:2017.* Genf: IEC.

IEC (Hrsg.). (2020). *IEC TS 62196-3-1:2020.* Genf: IEC.

Kampker, A., Valleé, D., & Schnettler, A. (. (2018). *Elektromobilität, 2. Auflage.* Berlin: Springer Vieweg.

Microchip Technology. (06. Dezember 2019). *Application Note TB3216: Getting Started with USART.* Abgerufen am 19. August 2020 von https://www.microchip.com//wwwAppNotes/AppNotes.aspx?appnote=en609135

Microchip Technology. (Januar 2020). *ATmega4808.* Abgerufen am 19. August 2020 von http://ww1.microchip.com/downloads/en/DeviceDoc/ATmega4808-09-DataSheet-DS40002173B.pdf

Nationale Plattform Elektromobilität. (kein Datum). *Elektromobilität: So funktioniert's*. Abgerufen am 19. 08 2020 von http://nationale-plattform-elektromobilitaet.de/anwendung/privat-laden/

Optiboot Project. (16. August 2020). *www.github.com/Optiboot*. Abgerufen am 19. August 2020 von https://github.com/Optiboot/optiboot

SAE (Hrsg.). (2017). *J1772_201710.* Warrendale: SAE.

SmartEVSE Project. (26. Juni 2020). *www.github.com/SmartEVSE/*. Abgerufen am 19. August 2020 von https://github.com/SmartEVSE/smartevse

Weemaes, G. (kein Datum). *GoingElectric Wiki*. Abgerufen am 19. August 2020 von Typ2 Signalisierung und Steckercodierung: https://www.goingelectric.de/wiki/Typ2-Signalisierung-und-Steckercodierung/

Wikipedia. (5. Oktober 2019). *SAE J1772*. Abgerufen am 19. August 2020 von https://de.wikipedia.org/wiki/SAE_J1772

Wikipedia. (21. April 2020). *IEC 62196*. Abgerufen am 31. August 2020 von Wikipedia: https://de.wikipedia.org/wiki/IEC_62196#Typ_3:_EV_Plug_Alliance

Wikipedia. (02. Juli 2020). *IEC 62196 Typ 2*. Abgerufen am 19. August 202 von https://de.wikipedia.org/wiki/IEC_62196_Typ_2

1. Einleitung

Diese Bachelorarbeit im Studiengang Elektrotechnik-Elektronik-Informationstechnik entsteht im Frühjahr/Sommer 2020 am Lehrstuhl für Regelungstechnik der FAU in Erlangen, betreut von Prof. Dr. Ing. Thomas Moor. Dabei wurde auf Basis der Standards IEC/DIN 62196 und SAE J1772 die Elektronik für eine Elektrofahrzeug-Ladestation entwickelt. Hard- und Firmware dieses Projekts sind in großen Teilen an das Open-Hardware-Projekt SmartEVSE[1] angelehnt. Die hier vorgestellte Elektronik soll als Grundstein für ein geplantes Netzwerk von öffentlichen Ladestationen dienen, welche mittels eingebauter WiFi-Module von einem zentralen Server gesteuert und verwaltet werden können. Außerdem soll das Ladestations-Modul auch ohne Netzwerkanbindung als Einzelinstallation betrieben werden können, z.B. als private Wandladestation (sog. „Wallbox“ [k2]) in der heimischen Garage für Elektrofahrzeug-Betreiber[2].

Nach einem kurzen Überblick über die verschiedenen Konzepte und Begriffe wird im Abschnitt „Theorie und Hintergrund“ zunächst die Funktion des CCS-Ladestandards, wie er aktuell in Europa verwendet wird, erläutert.

In Kapitel 3 „Anforderungen“ [k3]werden die technischen Anforderungen an das fertige System beschrieben und die daraus resultierenden Designentscheidungen dargelegt.

Vor dem Hintergrund des SmartEVSE-Projekts werden an der Elektronik und Firmware diverse Anpassungen und Erweiterungen vorgenommen, welche im Detail in den Kapiteln 4 und 5 erläutert werden. Der Abschnitt zur Elektronikentwicklung umfasst die Zeichnung des Schaltplans, die Auswahl und Beschaffung der benötigten Bauteile sowie das Erstellen des Leiterplattenlayouts für den zu testenden Prototypen.

Mit dem bestückten Prototyp wird zum einen die Funktionalität der Hardware überprüft, zum anderen die Firmware für den Testbetrieb an einem Elektrofahrzeug geschrieben und validiert. Die vorliegende Arbeit konzentriert sich auf die Entwicklung der Hardware und deren Test. Die [k4]Programmierung der WLAN-[k5]Module und des Verwaltungs- und Steuerungsservers finden im Rahmen einer weiteren Bachelorarbeit am Lehrstuhl für Regelungstechnik statt.

Der abschließende Praxistest findet an einem BMW i3 statt, wobei die Funktionalität und Betriebssicherheit im Einzelbetrieb ohne Netzwerkanbindung getestet werden. Erkenntnisse

[1] (SmartEVSE Project, 2020)
[2] (Nationale Plattform Elektromobilität, kein Datum)

aus Funktionstests, Laborversuchen und dem Praxistest werden gesammelt, ausgewertet und zum Abschluss der Arbeit in eine erste Revision der Hardware umgesetzt.

Alle Schaltpläne, Designunterlagen, Code und kompilierte Firmware für die in dieser Arbeit beschriebenen Tests der Ladestation sowie Informationen zur Programmierung des Mikrocontrollers sind unter https://github.com/dreadnomad/FGCCS-Ctrl22 veröffentlicht. Lizenzbedingungen für die Weiterverwendung der Arbeit sind in der Datei LICENSE.md erläutert.

2. Theorie und Hintergrund

2.1 Einführung

Das Laden von batteriebetriebenen Elektrofahrzeugen[k6] (engl. „Electric vehicle, im Folgenden mit EV abgekürzt) [k7]beruht bei aktuellen Serienfahrzeugen auf einem von zwei Prinzipien. Zum einen bietet der Großteil der erhältlichen Fahrzeuge die Möglichkeit, den Akkumulator unter Verwendung eines eingebauten Ladegeräts mit ein- oder dreiphasigem Wechselstrom[3] zu laden. Dabei können Ladeleistungen von bis zu 43kW (63A bei 3 x 400V) erreicht werden.[4] Verbreitete Standard-Ladeleistungen sind 3.7kW (16A, einphasig), 11kW (16A, dreiphasig) und 22kW (32A, dreiphasig). In diesem Fall übernimmt die eingebaute Ladeelektronik die Regelung des Stromflusses und die Überwachung des Akkus, was die Anforderungen an die technische Komplexität der Ladestation überschaubar macht und die Installation von Ladeinfrastruktur in Privatgaragen oder ähnlichen Räumen erlaubt. Das Kommunikationsprotokoll zwischen EV und Ladestation beschränkt sich auf das „Aushandeln“ [k8]der verfügbaren und maximal erlaubten Ladeströme und kann mit geringem Hardware- und Software-Aufwand implementiert werden.

Da höhere Ladeleistungen auch die in den Fahrzeugen verbauten Ladegeräte aufwändiger und damit schwerer und teurer machen, wurden für diesen Fall andere Konzepte entwickelt. Für Ladeleistungen bis 150kW und mehr wird das Ladegerät mit Gleichrichtern [k9]und zugehöriger Elektronik in die Ladestation ausgelagert und der Fahrzeug-Akku mittels Gleichstrom geladen[5]. Da in diesem Fall der Akku direkt von der Ladestation mit hohen Strömen gespeist wird, muss ein komplexes Kommunikationsprotokoll zur Regelung des Leistungsflusses und der Akkuüberwachung implementiert werden. Gleichstrom-Ladestationen sind daher deutlich aufwändiger in Design und Betrieb als ihr Wechselstrom-Äquivalent, und dementsprechend für Privathaushalte oder für flächendeckende Installation zu kostenintensiv.

[3] (Elektrotechnik, Deutsche Kommission, 2020), S. 8
[4] (Kampker, Valleé, & Schnettler, 2018), S.36
[5] (Elektrotechnik, Deutsche Kommission, 2020), S. 8

2.2 SAE J1772 und IEC/DIN 62196

Die dominierende Norm für die Ladeanschlüsse von Elektrofahrzeugen in Europa ist derzeit IEC 62196, welche die verwendeten Steckertypen und Lademodi definiert. In Deutschland ist die Norm als DIN EN 62196 gültig. In Teil 1 der Norm, erstmals publiziert am 28. April 2003, werden die allgemeinen Anforderungen an die Steckersysteme und Ladevorrichtungen definiert, sowie vier verschiedene Lademodi spezifiziert. Die Spezifikation deckt Wechselspannungen bei 50/60 Hz von bis zu 690V bei 250A Nennstrom sowie Gleichspannungen von bis zu 600V bei 400A Nennstrom ab. Die vier Lademodi werden wie folgt definiert[6]:

- Mode 1: Ladung an Haushaltssteckdosen mit Schutzkontakt (in Deutschland: Schuko-Stecker), maximaler Nennstrom: 16A
- Mode 2: Ladung an ein- bis dreiphasigen Anschlüssen mit fest kodiertem Ladeprotokoll, maximaler Nennstrom 32A
- Mode 3: Ladung an eigens vorgesehenen Stecksystemen mit zwei Pilotleitungen, maximaler Nennstrom: 250A
- Mode 4: Schnelle Gleichstromladung mit externem Ladegerät bis maximal 400A Nennstrom

Zunächst wurde bezüglich der Steckverbinder auf die in IEC 60309 spezifizierten Steckertypen verwiesen, welche umgangssprachlich zumeist als CEE-Drehstromsteckverbinder bezeichnet werden und im Industriebereich für Mehrphasen-Drehstromsysteme verwendet werden. Diese existieren in zwei Varianten: Der rote Dreiphasen-Steckverbinder mit fünf Leitern (Drei Phasen/Außenleiter und Neutral- sowie Schutzleiter) und der blaue Einphasen-Verbinder mit drei Leitern für Phase, Neutral- und Schutzleiter.[7]

Mit Veröffentlichung von Teil 2 der Norm als IEC 62196-2, erstmals publiziert am 13. Oktober 2011, wurde eine Liste von Ladesteckern definiert, welche zusätzlich die für das Laden nach Mode 3 nötigen Pilotkontakte aufweisen[8]:

- **Typ 1:** Einphasiger Ladestecker mit zwei Pilotkontakten, ursprünglich in Nordamerika vom SAE-Normengremium im Jahr 2001 festgelegt und in die IEC-Norm übernommen. Maximaler Nennstrom ist mit der Revision 2009 auf 80A festgelegt.

[6] (IEC (Hrsg.), 2011)
[7] (IEC (Hrsg.), 2014), Kap. 7
[8] (IEC (Hrsg.), 2016), Kap. 18 & 19

- **Typ 2:** Dreiphasiger Ladestecker mit zwei Pilotkontakten, zuerst in Deutschland als Derivat der CEE-Steckverbinder entwickelt und von der DKE/VDE genormt. Der Typ 2-Stecker ist kleiner als der CEE-Stecker und durch Abflachung an einer Seite mechanisch gegen Verpolung gesichert. Der Typ 2-Stecker unterstützt mit demselben Gehäuse Lademodi von einphasig/16A bis dreiphasig/63A, also von 3.7kW bis 43kW, und damit nicht das gesamte Spektrum nach IEC 62196-1 Mode 3.
- **Typ 3:** Aufgrund gesetzlicher Regelungen in einigen europäischen Ländern, u.a. Frankreich und Italien, in welchen für Ladestecker ein mechanischer Berührungsschutz gefordert wurde, gab es Bestrebungen seitens eines Konsortiums französischer und italienischer Firmen, einen dritten Steckertyp zu normen. Der für bis zu dreiphasigen/32A-Lademodi spezifizierte Typ 3C-Steckverbinder wurde in die Norm aufgenommen, im Jahr 2015 wurde die Produktion aber endgültig eingestellt, nachdem die Europäische Kommission im Januar 2013 den Typ 2-Stecker als Standard für die europäische Ladeinfrastruktur festgelegt hatte.[k10][9]

Die Funktion der Pilotkontakte in den Steckertypen von IEC 62196-2 wurde 2001 in SAE J1772 beschrieben und entsprechend in die IEC-Norm 61851[10] übernommen. Damit verwenden Typ 1 und Typ 2-Stecker dasselbe Kommunikationsprotokoll, was die Standardisierung der Ladesysteme unterstützt. Es kann damit für nordamerikanische und europäische Ladestationen, unabhängig von Netzspannungspegel und Stromstärken,[k11] dieselbe Steuerungs- und Kommunikationselektronik verwendet werden.

Die SAE J1772-Signalisierung (größtenteils identisch zur Signalisierung nach IEC 61851), erstmals beschrieben im Jahr 2001, basiert auf einer Abfolge von Zuständen gemäß Tabelle 1.[11]

Zustand	A	B	C	D	E	F
Spannung CP-PE	+12V ±0.4V	+9V ±1V	+6V ±1V	+3V ±1V	±0V	-12V
Ladefreigabe	Kein EV	EV erkannt	Ladebereit	Ladebereit mit Belüftung	Kurzschluss od. PE defekt	Fehler

Tabelle 1: Ladezustände nach SAE J1772

Im Standby-Modus ohne angeschlossenes Fahrzeug wird von der Ladestation über einen 1kΩ-Widerstand eine Spannung von 12V ±0.4V dauerhaft an den Control Pilot (CP)-Kontakt angelegt.

[9] (Wikipedia, 2020)
[10] (IEC (Hrsg.), 2017)
[11] (Wikipedia, 2019)

Diese Spannung liegt ebenfalls am Proximity Pilot (PP)-Kontakt an. Wenn ein EV über das Ladekabel an die Station angeschlossen wird, wird der CP-Stromkreis über eine Diode und einen 2.7kΩ-Widerstand auf den Schutzleiter (PE) zurückgeführt. Über einen Widerstand zwischen PP und PE wird die maximale Strombelastbarkeit des Ladekabels inkl. Steckverbinder gemäß Tabelle 2 kodiert.[12]

Widerstand PP-PE	1.5kΩ	680Ω	220Ω	100Ω
Max. Ladestrom	13A	20A	32A	63A

Tabelle 2: Ladekabel-Kodierung der maximalen Stromstärke nach IEC 61851

Diese Kodierung war in der SAE J1772-Norm nicht vorgesehen, dort dient der PP-Kontakt nur zur Signalisierung einer geöffneten Steckerverriegelung, damit die Ladestation bei einer Trennung der Verbindung die Ladung unterbrechen und die Kontakte stromlos schalten kann.[13]

Durch die Verbindung des Ladekabels zum EV via Diode und Widerstand fällt die Spannung am CP-Kontakt auf 9V ±1V, was von der Ladestation als Signal für ein angeschlossenes Fahrzeug interpretiert wird. Daraufhin wird stationsseitig[k12] ein Signalgenerator eingeschaltet, welcher eine 1-kHz-Rechteckschwingung mit einem Signalpegel von ±12V an die Pilotkontakte anlegt. Die Pulsweite des Rechtecksignals wird entsprechend der verfügbaren Ladeleistung variiert, wobei die Beziehung Pulsweite-Stromstärke für Pulsweiten zwischen 10% und 85% über folgende Formel definiert ist[14]:

$$Verfügbare\ Stromstärke\ [A] = Pulsweite\ [\%] * 0.6\ A$$

Bei Pulsweiten über 85% wird folgende Formel angesetzt[15]:

$$Verfügbare\ Stromstärke\ [A] = (Pulsweite\ [\%] - 64) * 2.5A$$

Die maximale Stromstärke, welche mit dieser Signalisierung angezeigt werden kann,[k13][k14] liegt bei 80A mit einer Pulsweite von 97%. Es ergibt sich mit oben genannten Formeln beispielhaft die Tabelle 3.

Pulsweite	Ladestrom
97%	80A
90%	65A
80%	48A

[12] (Weemaes, kein Datum)
[13] (Wikipedia, 2019)
[14] (Wikipedia, 2020)
[15] (Wikipedia, 2020)

50%	30A
40%	24A
25%	15A
10%	6A

Tabelle 3: Verfügbarer Ladestrom in Abhängigkeit der Pulsweite nach IEC 61851-1

Ist das EV ladebereit, z.B. durch die Betätigung eines Schalters im Auto oder auch automatisch bei Erkennen des Rechtecksignals, wird fahrzeugseitig ein 1.3kΩ-Widerstand parallel zu den bereits vorhandenen 2.7kΩ geschaltet. Durch den daraus resultierenden Gesamtwiderstand CP-PE von 880Ω wird die Spannung an CP auf +6V ±1V reduziert, was für die Ladestation das Signal zum Starten des Ladevorgangs darstellt.[k15] Bei Fahrzeugen, welche bei Erwärmung des Ladegeräts oder des Akkus eine Belüftung aktivieren, kann stattdessen über einen 270Ω-Widerstand die Spannung an CP auf +3V ±1V gesenkt werden, worauf die Ladestation allfällig vorhandene Belüftungsvorrichtungen in Innenräumen (z.B. Garage) einschalten kann, oder falls diese nicht vorhanden sind, den Ladevorgang abbrechen. Dies dient der Vermeidung von übermäßiger Erhitzung der Ladeinfrastruktur. Grundsätzlich sind in der Regel sowohl im Fahrzeug als auch in der Ladestationen Temperatursensoren verbaut, welche den Ladevorgang bei Überhitzung abbrechen.[16]

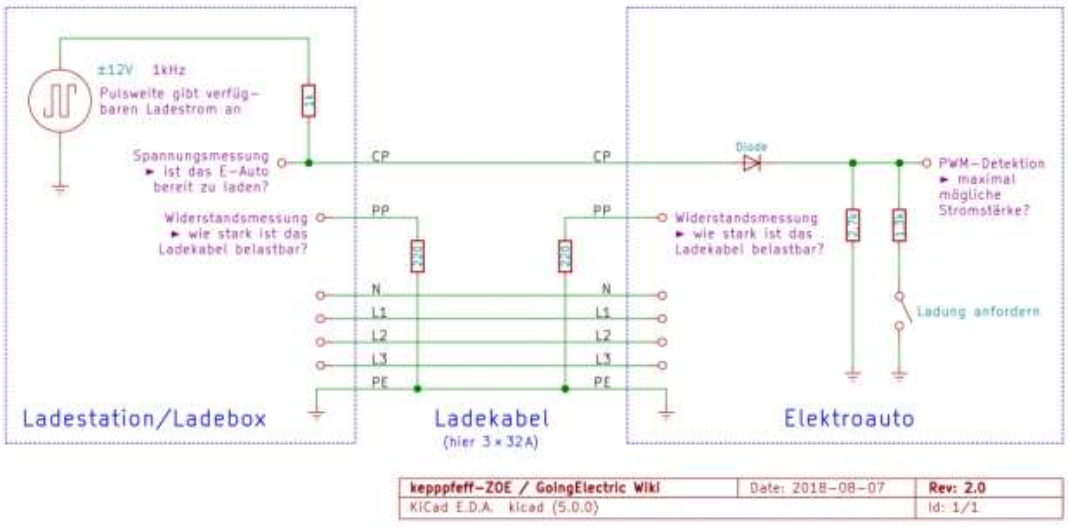

Abbildung 1: Beispielhafte Verschaltung von EV und Ladestation für Mode-3-Wechselstromladen

Sinkt die Spannung an CP unter +2V liegt ein Fehlerfall vor und die Ladestation bricht den Ladevorgang ebenfalls ab. Der Ladevorgang ist beendet, wenn seitens des Fahrzeugs die Spannung an CP wieder auf +9V angepasst wird. Aktuelle Steckersysteme nach IEC 62196 Typ-2

[16] (Carius, 2017)

verfügen über elektromechanische Verriegelungssysteme, welche das Abziehen des Ladekabels im stromführenden Zustand verhindern. Die Verschaltung von Ladestation und EV ist in **Fehler! Verweisquelle konnte nicht gefunden werden.** beispielhaft dargestellt.

2.3 Combined Charging System (CCS)

Als Weiterentwicklung des ausschließlich für Wechselstrom-Laden geeigneten Typ 2-Steckers wurde 2011 das Combined Charging System als Gemeinschaftsprojekt von Steckverbinder-Herstellern und deutschen Automobilkonzernen vorgestellt. Es besteht aus einer fahrzeugseitigen Buchse und dem entsprechenden Stecker, welche sowohl Wechselstrom- als auch Gleichstromladen unterstützen. Damit kann die Installation mehrerer Buchsen in Fahrzeug und Ladestation vermieden werden. Dazu wurde der Typ 2-Stecker nach IEC 62196-2 um zwei zusätzliche hoch belastbare Gleichstromkontakte erweitert, wobei aber der Standard-Typ 2-Stecker dieselbe Buchse verwenden kann.

Das CCS wurde von den nordamerikanischen und europäischen Normungskommissionen gemeinsam entwickelt und in IEC 62196-3 ab 2011 genormt. Für den nordamerikanischen Markt existiert eine Abwandlung des Typ 1-Steckers mit zusätzlichen Gleichstromkontakten.[17]

Die in der vorliegenden Arbeit entwickelte Ladestation ist nicht für Gleichstromladen ausgelegt, es könnte aber problemlos eine CCS-Buchse für den EV-Anschluss verwendet werden, um die Kompatibilität für neuere Fahrzeuge mit Ladesystemen nach CCS-Standard zu gewährleisten. Die Gleichstromkontakte würden in diesem Fall nicht angeschlossen.

2.4 Technische Umsetzung der SAE J1772-Signalisierung

Um das Kommunikationsprotokoll gemäß SAE J1772/IEC 61851 technisch zu implementieren, [k16]muss eine Schaltung zur Messung der entsprechenden Pegel auf den Pilotleitungen und zur Generierung des 1-kHz-Rechtecksignals entworfen werden. Da die Pilotsignale vollständig zeit- und wertkontinuierlich (analoge Signale) sind, ist keine aufwändige Digitalelektronik nötig[k17]. Sämtliche Verarbeitungs- und Generatorfunktionen kann ein kostengünstiger und einfach

[17] (IEC (Hrsg.), 2020)

programmierter Mikrocontroller übernehmen, während einerseits ein als Spannungsfolger beschalteter Operationsverstärker das Eingangssignal auf der CP-Leitung puffert und andererseits ein als Komparator beschalteter OPV das vom Mikrocontroller generierte PWM-Signal auf den geforderten Pegel von ±12V verstärkt. Eine Beispielschaltung ist in Abbildung 2 dargestellt.

Der Spannungsteiler aus R1 und R2 teilt die Ausgangsspannung des Puffer-OPV von 12V auf 3.27V herunter und bringt das Signal damit in den Spannungsbereich der Mikrocontroller-Eingänge. Der Spannungsteiler aus R4 und R5 stellt am invertierenden Eingang des Komparators eine Referenzspannung von VDD/2 bereit. Damit wird das vom Mikrocontroller generierte PWM-Signal mit Pegeln von 0V und 3.3V in eine Rechteckschwingung von ±12V umgesetzt.

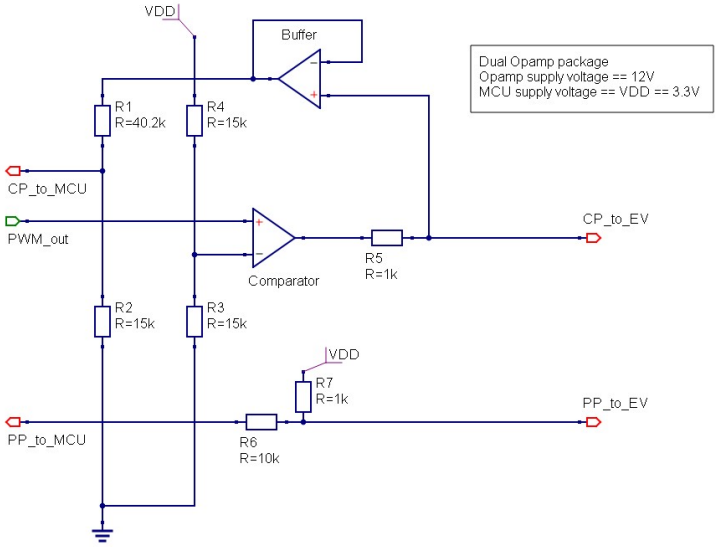

Abbildung 2: Beispielschaltung für die technische Implementierung der SAE J1772-Signalisierung

Die Spannung am PP-Eingang des Mikrocontrollers variiert je nach parallel geschaltetem Widerstand zwischen PP und PE/Masse, wodurch die Kapazität des Ladekabels berechnet werden kann. So wird die Dekodierung der Signalisierung auf das Auslesen und Interpretieren der Analog-Digital-Umsetzer des Mikrocontrollers beschränkt.

2.5 Mikrocontroller und eingebettete Systeme

Zur Steuerung elektronischer Systeme werden heutzutage meist Mikrocontroller verwendet. Nach Bernstein kann [k18]der prinzipielle Aufbau eines Mikrocontrollers folgendermaßen zusammengefasst werden: „Der Mikrocontroller ist ein hochintegrierter Baustein, der, neben

einem Standard-Mikroprozessor, einen oder mehrere serielle und parallele Portbausteine, Datenspeicher, Programmspeicher und eventuell noch einige andere Sonderfunktionen an Hardware auf einem einzigen Chip vereinigt."[18]

Durch die fortschreitende Miniaturisierung in der Halbleiterfertigung und der damit einhergehenden Erhöhung von Speicherkapazitäten, Taktfrequenzen und Funktionsumfang von integrierten Schaltkreisen werden heute kostengünstige Mikrocontroller mit hohen Rechenleistungen angeboten, welche für verschiedenste Steuerungsaufgaben in eingebetteten Systemen geeignet sind. Das Einsatzspektrum moderner Mikrocontroller geht von einfachen Anwendungen wie Haushaltsgeräten (Waschmaschine, Kaffeeautomat[k19]), über tragbare Elektronik wie Digitaluhren bis hin zu komplexen Steuerungen von Kraftfahrzeugen (Einspritzsysteme, Klimaanlagen, Anti-Blockiersysteme uvm.).[19]

Ein eingebettetes System in diesem Sinne, [k20]ist ein System, welches in einem größeren technischen System die Aufgabe der zentralen Rechen- und Steuerungseinheit übernimmt, wobei nach außen hin die Recheneinheit nicht direkt in Erscheinung tritt (wie z.B. in einem herkömmlichen PC), sondern nur dessen zur Verfügung gestellte Funktionalität, wie beispielsweise die Bordelektronik eines Kraftfahrzeuges.[20]

Moderne Mikrocontroller werden zumeist in der Programmiersprache C programmiert, welche als Hochsprache die effiziente Programmierung auch komplexer Software ermöglicht, aber trotzdem einen sehr direkten Zugriff auf die Hardware des Prozessors und seiner Peripherie bereitstellt, was bei der hardwarenahen [k21]Programmierung von Mikrocontrollern ohne gesondertes Betriebssystem von großem Vorteil ist.[21]

[18] (Bernstein, 2015), S.2
[19] (Bernstein, 2015), S.2
[20] (Hüning, 2019), S. 11
[21] (Bernstein, 2015), S.180

3. Anforderungen

Das Anforderungsprofil für die im Rahmen dieser Arbeit entwickelte Ladestation ergibt sich aus der gewünschten Funktionalität als Wechselstrom-Ladestation im Einzelbetrieb (Wallbox), sowie aus der zusätzlich zu implementierenden Möglichkeit, die Station als Teil eines größeren Netzwerkes von öffentlichen Ladestationen zu betreiben (Mesh[TP22]-Betrieb). Der beispielhafte Einsatzfall für das System ist eine leistungsstarke Photovoltaikanlage, deren erzeugte Leistung direkt zum Aufladen von EV-Akkus verwendet werden soll. Dazu muss es möglich sein, die Ladestation über ein drahtloses Netzwerk von einem zentralen Server aus zu steuern und die verfügbare Gesamtladeleistung auf alle aktiven Ladestationen im Netzwerk zu verteilen. Dieser Vorgang des Lastausgleichs wird auch als Load Balancing bezeichnet.[k23]

Aus diesen grundlegenden Bedingungen lassen sich folgende technischen Anforderungen ableiten:

1) **Zentrale Mikrocontroller-Einheit zur Steuerung des lokalen Ladevorganges**
   Dieser Controller bildet das Herzstück des Systems und soll so ausgewählt werden, dass sämtliche Funktionalität mit einer minimalen Anzahl von zusätzlichen, externen Bauelementen implementiert werden kann.
2) **Zusätzliches, optionales WLAN-Modul auf Basis eines Mikrocontrollers**
   Dieses Modul dient zur Bereitstellung von Kommunikations- und Fernwartungsfunktionalität via drahtloser Netzwerke, bevorzugt über Wireless LAN (WLAN). Das System soll so konzipiert sein, dass für den Einzelbetrieb auf das WLAN-Modul verzichtet werden kann, also keine Bestandteile der Ladestation davon abhängig sind.
3) **Ansteuerung von bis zu dreiphasigen Wechselstromnetzen**
   Über drei getrennte Schütze (eines pro Phase) soll die Ladestation Ladeleistungen bis 43kW ermöglichen, wobei die Phasen sowohl gemeinsam als auch einzeln geschalten werden können.
4) **Messung der lokal verbrauchten Ladeleistung**
   Mittels geeigneter Sensorik (Stromwandler an den drei Phasen) soll der lokale Stromverbrauch andauernd gemessen und zu Log-Zwecken sowie für Load Balancing über mehrere verteilte Stationen weiterverarbeitet werden.
5) **Einfach zu wartende und erweiterbare Firmware**

Um Software-Updates via WLAN zu ermöglichen und für die allfällige spätere Erweiterung des Funktionsumfanges der Ladestation soll ein Startprogramm oder Bootloader [k24]verwendet werden, welches über eine kabelgebundene oder drahtlose (via das WLAN-Modul) serielle Schnittstelle angesprochen werden kann.

6) **Implementierung des SAE J1772 Kommunikationsprotokolls**

   Dazu dient eine gemischte Analog-Digital-Schaltung zur Generierung und Verarbeitung der benötigten Signale, welche über Pilotleitungen die Kommunikation zwischen Ladestation und EV herstellen.

7) **Alternative Möglichkeit zur Fernsteuerung des Systems**

   Diese wird über eine kabelgebundene RS485-Schnittstelle implementiert, welche für erhöhte Robustheit gegenüber elektromagnetischen Störungen in störungsbelasteten Umgebungen anstelle des WLAN-Moduls eingesetzt werden kann.

8) **Elektronische Ansteuerung gängiger Verriegelungssysteme für Typ-2-Ladestecker**

   Die in der Praxis eingesetzten Verriegelungssysteme basieren entweder auf elektromechanischer Verriegelung über eine Spule oder auf kleinen Gleichstromaktoren, zu deren Bedienung kurzfristig hohe Ströme von der Ladestation abgegeben werden müssen.

9) **Möglichkeit zur elektronischen Trennung und Wiederverbindung der Pilotleitungen**

   Bei EV, welche unter gewissen Umständen in einen Standby-Modus verfallen können, soll dieser durch ein simuliertes Aus- und wieder Einstecken des Ladekabels beendet werden.

10) **Unkomplizierte Bedienung**

    Die Bedienung der Ladestation soll für den Anwender sehr minimalistisch gehalten werden, entsprechend wird ein einzelner Taster mit integrierter LED vorgesehen, mithilfe dessen der Ladevorgang gestartet und gestoppt werden kann.

11) **Formfaktor passend für DIN-Hutschienengehäuse**

    Um die Ladestation zusammen mit Fehlerstrom-Schutzschalter, Sicherungen, Wechselstrom-Schützen und der Verkabelung in standardisierten Installationskästen unterbringen zu können, soll auch der Formfaktor der Elektronik so angepasst sein, dass diese in einem 6TE-Hutschienengehäuse Platz findet.

12) **Open Source Hard- und Software**

    Sämtliche Schaltpläne, Leiterplattendesigns, Firmware-Quellcodes und alle Dokumentation des Entwicklungsprozesses sollen offen zugänglich sein, um interessierten Privatpersonen und Institutionen die Möglichkeit zu geben, das System

weiterzuentwickeln und für Ihre Zwecke anzupassen. Dazu wird unter Berücksichtigung der Software-Lizenzen von hier verwendeten Bibliotheken und Code-Fragmenten sämtliches Material, welches erforderlich ist um oben genanntes Ziel zu erreichen, unter einer freien Lizenz veröffentlicht.

Das bestehende und getestete Open-Hardware SmartEVSE-Projekt erfüllt zwar viele dieser Anforderungen, ist aber nicht ohne Weiteres[k25] für den Mesh-Betrieb [k26]umzurüsten. Dementsprechend mussten ausgehend vom SmartEVSE-Referenzdesign diverse Anpassungen und Erweiterungen vorgenommen werden. Diese Anpassungen, die Auswahl der Bauelemente und das gesamte Schaltungsdesign werden in Kapitel 4 genauer erläutert.

4. Elektronikentwicklung

4.1 Konzept und Werkzeuge

Die zu entwerfende Schaltung wurde entsprechend der in Kapitel 3 ermittelten Anforderungen geplant. Als Entwicklungswerkzeug wurde die freie ECAD-Software Kicad in Version 5.1.5 verwendet. Diese Software-Lösung enthält verschiedene Programme, darunter den Schaltplan-Editor eeschema, pcbnew für das Leiterplatten-Layout sowie diverse Hilfsprogramme zur Bearbeitung von einzelnen Bauteilen und deren Leiterplatten-Footprints. Mit der Software wird eine umfangreiche Bibliothek von Bauteilen ausgeliefert, wobei fehlende Teile unkompliziert importiert oder von Hand erstellt werden können. In Kicad ist kein Simulationsprogramm enthalten, es besteht aber die Möglichkeit, das freie Simulationswerkzeug Ngspice einzubinden. Da diese Schnittstelle aktuell noch nicht fertig implementiert und daher nur eingeschränkt nutzbar ist, wurde für die Simulation einzelner Schaltungskomponenten das frei verfügbare Programm QUCS eingesetzt.

Das grundlegende Schaltungskonzept beruht auf der Trennung der Schaltungsteile nach Spannungspegel. Funktionsbedingt beinhaltet die Ladestation auf derselben Platine sowohl Bereiche von 230V Netzspannung, als auch einen großen 12V und 3.3V Niederspannungsteil. Diese Abschnitte wurden von Beginn weg getrennt geplant und nur an ihren Schnittstellen (AC-DC-Schaltnetzteil und Halbleiterrelais) verbunden. Drei Halbleiterrelais dienen dazu, drei externe Schütze zu schalten, um das EV zu laden. Im Niederspannungsteil wiederum kann unterschieden werden zwischen einer Vielzahl von integrierten Schaltungen mit den jeweils dazugehörigen passiven Bauelementen einerseits, und einer Operationsverstärker-basierten Analog- und Digitalschaltung zur Verarbeitung der Pilot- und Messsignale. Daraus folgt, dass Hochspannungs- und Niederspannungsanschlüsse getrennt auf zwei Schraubterminals an gegenüberliegenden Seiten der Leiterplatte gelegt werden.

Die Spannungsversorgung der Ladestation und aller Ihrer Komponenten soll über ein einfaches AC-DC-Schaltnetzteil hergestellt werden, welches[k27] zunächst eine 12V-Gleichspannung erzeugt. Diese 12V werden benötigt, da das in SAE J1772 definierte Kommunikationsprotokoll einen Spannungspegel von +/- 12V für die PWM-Signale festlegt. Daher muss aus der primären Spannungsversorgung über einen Inverter-Baustein zusätzlich eine Spannung von -12V bereitgestellt werden, um den Operationsverstärker und Komparator zu versorgen. Außerdem

werden handelsübliche Verriegelungssysteme für Typ-2-Ladestecker ebenfalls bei 12V betrieben. Der Mikrocontroller, das WLAN-Modul und der RS485-Transceiver hingegen benötigen eine 3.3V-Spannungsversorgung. Diese wird aus der 12V-Schiene durch einen Pegelwandler gewonnen, welcher genügend Leistung für diese Komponenten zur Verfügung stellen muss.

Für die Programmierung des Mikrocontrollers und des WLAN-Moduls müssen die entsprechenden Pins über Pfostenleisten zugänglich sein, zusätzlich zur RS485-Schnittstelle. Allfällige unbenutzte Pins des Mikrocontrollers sollen zu Debugging- und Testzwecken ebenfalls auf Pfostenleisten geführt werden.

Die gesamte Schaltung soll so ausgelegt werden, dass die Bestückung des WLAN-Moduls optional bleibt, dass also die Programmierung des Mikrocontrollers auch kabelgebunden möglich ist, wie auch der Zugang über die serielle Schnittstelle.

4.2 Schaltungsdesign

Da mit dem SmartEVSE-Projekt bereits eine getestete und funktionierende Basis für eine Ladestation vorlag, wurde zunächst dessen Schaltplan in Version 2.2 untersucht. Von primärem Interesse war dabei, welche Teile davon wiederverwendet werden konnten, und an welchen Teilen der Schaltung Änderungen vorgenommen werden mussten. Zusätzlich musste aufgrund fehlender Dokumentation die Funktion einzelner Schaltungsteile ermittelt und durch Recherche in den Datenblättern der Bauteile sowie Simulation einiger Funktionseinheiten genauer bestimmt werden.

Aufgrund der Anforderungen konnte direkt das im SmartEVSE-Projekt vorgesehene LCD-Display und Menü weggelassen werden, da diese Funktionalität nicht benötigt wird. Außerdem konnte so die benötigte Anzahl Pins des Mikrocontrollers stark reduziert, bzw. konnten die freigewordenen Pins anderweitig genutzt werden[k28].

4.2.1 Bauteile und Verschaltung

Der vollständige Schaltplan des Systems findet sich in Anhang A, alle hier genannten Bauteilbezeichnungen finden sich in den Schaltplänen wieder.

Zur Spannungsversorgung der Schaltung dient U4, ein Mean Well IRM-10-12 AC/DC-Wandler mit 12V Ausgangsspannung bei 0.85A Ausgangsstrom, für insgesamt 10W Ausgangsleistung, wodurch die Schaltung über genügend Leistungsreserven verfügt. Um für die an 3.3V betriebenen Bauteile die Versorgungsspannung bereitzustellen, wurde als Spannungsregler U6 für die 3.3V-Schiene der AP-5100-Schaltspannungsregler von Diodes Inc. eingesetzt, welcher bei geeigneter Außenbeschaltung nach Datenblatt bis zu 1.2A Ausgangsstrom liefern kann. Damit verbleiben auch bei Lastspitzen rund 5W für die an 12V betriebenen Bauteile, was sich als ausreichend herausstellte. Die Spannungsversorgung erfolgt netzseitig [k29]über eine der drei Wechselstrom-Phasen, wobei bei Installation mehrerer Ladestationen diese gleichmäßig über die drei Phasen verteilt werden können.

Der Schaltspannungsregler U5 funktioniert dabei als Stromquelle, deren Ausgangsstrom durch die Außenbeschaltung mit den Widerständen R1 und R2 in eine Spannung gewandelt wird. Der FB-Pin von U5 koppelt die Ausgangsspannung auf den Eingang zurück, der Schaltregler nutzt diese Rückkopplung zur Regelung des Ausgangsstromes regelt. Die Spule L1 dient zur Filterung von hochfrequenten Störungen, welche durch die 1.4MHz-Schaltfrequenz des Reglers am Ausgang auftreten. Die Kondensator-Bänke an Ein-und Ausgang des Reglers haben den Zweck, [k30]die Ausgangsspannung puffern und zu glätten.

Die Wechselstrom-Schütze der Ladestation werden alle mit derselben Phase gesteuert, wobei für jedes Schütz ein separates Halbleiterrelais (U1 – U3) vom Typ AQH1213AX eingesetzt wird. Diese Relais verfügen über einen Nulldurchgangsdetektor, wodurch Abrissfunken beim Schalten vermieden werden.

Die Bereitstellung der -12V-Spannungsschiene für die Operationsverstärker der Analogschaltung erfolgt durch den TC7660SEOA-Schaltspannungsregler U5 von Microchip Technology, welcher mit passender Beschaltung eine 12V-Versorgungsspannung invertiert am Ausgang abgeben kann. Der maximale Ausgangsstrom ist dabei auf 20mA begrenzt, was aber in der vorliegenden Schaltung ausreichend ist, da die Operationsverstärker nur als Komparator und Spannungsfolger eingesetzt werden und keine nennenswerte Leistungsverstärkung liefern müssen.

Für die Verriegelung von Typ 2-Steckverbindern werden in der Regel einfache elektromechanische Verriegelungsaktoren verwendet, welche zum Öffnen oder Schließen der Verriegelung für kurze Zeit einen Spulenstrom von bis zu 3A bei 12V benötigen. Um diese

Stromspitzen zu bewältigen und die Verriegelung durch die 3.3V-Ausgänge des Mikrocontrollers zu steuern, wird der MOSFET-Gate-Treiber U8 (FAN3214TMX von ON Semiconductor) eingesetzt. Mithilfe des Reservoir-Kondensators C12 von 10000µF können so kurzzeitig hohe Ströme abgegeben werden. Ein Widerstandsnetzwerk von vier parallel geschalteten 100Ω-Widerständen an der Versorgungszuleitung zum Gate-Treiber begrenzt im Fehlerfall den Dauerstrom durch das Bauteil.

Das RS485-Kommunikationsprotokoll bedient sich der differentiellen Signalübertragung, wobei ein serielles Signal vom USART-Modul des Mikrocontrollers in zwei Signale A und B gewandelt wird, welche zueinander invertiert sind. Dies erhöht die Gleichtaktunterdrückung am Empfänger und ermöglicht eine sichere Datenübertragung,[k31] auch in Umgebungen mit starken elektromagnetischen Störungen. Diese Pegelwandlung übernimmt U7, ein TC7660SEOA-Baustein von Texas Instruments.

Als Mikrocontroller U10 für die Steuerung des Ladevorgangs wurde der Atmel ATmega4808[22] im 28-Pin SOIC-Package ausgewählt. Der ATmega4808 ist ein 8-Bit AVR-Prozessor der neuesten Generation (Mega-0-Series) von Microchip Technology Inc., er verfügt über einen internen 20-MHz Oszillator, 48kB Flash-Speicher und 6kB SRAM. Bei einer Versorgungsspannung von 3.3V kann der Mikrocontroller mit bis zu 13MHz Takt betrieben werden. Außerdem verfügt der ATmega4808 über sämtliche benötigten Peripheriemodule für dieses Projekt: drei USART-Schnittstellen, Analog-Digital-Umsetzer, mehrere Zähler/Zeitmesser mit PWM-Funktionalität und[k32] eingebauter Temperatursensor. Zudem ist mit AVR-GCC ein leistungsfähiger, plattformübergreifender und weit verbreiteter Open-Source-Compiler für C-basierte Firmware verfügbar, sowie Atmel Studio 7 als integrierte Entwicklungsumgebung unter Windows. Der Controller ist auch in größeren Gehäusen mit bis zu 32 Pins verfügbar. Für [k33]dieses Projekt stellten sich aber 28 Pins als ausreichend heraus.

Um das Modul U9 für die WiFi-Funktionalität bereitzustellen, wurde das Espressif ESP32-WROOM-32-Modul[23] ausgewählt. Dieses Modul basiert auf dem ESP32 32-Bit-Mikrocontroller mit LX6 Zwei-Kern-Prozessoren von Tensilica und integrierter WLAN- und Bluetooth-Unterstützung. Der ESP32 kann mit bis zu 240MHz Taktfrequenz betrieben werden und verfügt über 520kB SRAM. Im ESP32-WROOM-Modul sind zusätzlich diverse Peripheriemodule, 4MB Flash-Speicher und eine Funkantenne fest integriert. Es [k34]sind aber auch Module mit Anschlussmöglichkeit für eine externe Antenne verfügbar.

[22] (Microchip Technology, 2020)
[23] (Espressif Systems, 2019)

Es wäre möglich, die gesamte Ladestation nur über den ESP32 zu steuern, da dieser Mikrocontroller genügend Ressourcen und zugängliche Pins verfügt. Im Interesse der Modularität und der Zugänglichkeit wurde entschieden, den Betrieb des Systems auch ohne das WLAN-Modul zu ermöglichen und die Kernfunktionen im kleineren und weniger komplexen ATmega4808 zu implementieren. Dies[k35] erhöht auch die Zugänglichkeit für Einzelpersonen und Institutionen, die sich mit der Weiterentwicklung oder Anpassung des Systems beschäftigen wollen, da die AVR-Architektur über weite Verbreitung [k36]und eine große internationale Gemeinschaft verfügt

Da der ATmega4808 einen integrierten Temperatursensor besitzt[k37], konnte auf einen externen Sensor verzichtet werden, was einen weiteren Pin am Mikrocontroller verfügbar machte.

Die Messung der abgegebenen Leistung wird mittels dreier induktiver Stromwandler realisiert, welche sekundärseitig [k38]einen zur Primärseite proportionalen Strom abgeben. Dieser wird mithilfe einer Messschaltung bestehend aus C23, R28 und R31 (für CT0) in eine Spannung gewandelt, welche dann am Analog-Digital-Umsetzer des Mikrocontrollers gemessen und weiterverarbeitet wird. Da es sich um eine Wechselstrommessung handelt, wird eine Offsetspannung aus einem Spannungsteiler aus R26 und R27 zwischen 3.3V und Masse gewonnen, wodurch die Messung um 1.65V pendelt. Zur Überprüfung der Bauteilwerte und der Funktionalität dieser Messschaltung wurde eine Simulation durchgeführt. In Abbildung 3 sind die Ergebnisse der Simulation dargestellt, zusammen mit der simulierten Schaltung, welche den Stromwandler-Eingängen auf Seite 3 des Schaltplans (siehe Anhang A) entspricht. Simuliert wurde mit dem maximalen Eingangsstrom von mA, was einem primärseitigen Strom von 100A entsprechen würde, solche Ströme sind unter IEC/DIN 62196 nicht vorgesehen, der maximale Ladestrom liegt hier bei 80A[24]. Da bei 50mA Eingangsstrom die maximale Spannung am Mikrocontroller-Eingang bei unter 2.8V liegt, könnte der Messwiderstand auch vergrößert werden, um bei niedrigeren Strömen eine höhere Messauflösung am ADU-Eingang des Controllers zu erreichen.

Die Pilotsignale und das für die Signalisierung benötigte PWM-Signal werden gemäß der Beispielschaltung aus Abschnitt 2.4 verarbeitet. Ein dualer Operationsverstärker, LM7332MA, dient als Pufferverstärker U11A und Komparator U11B, während das Rechtecksignal vom Mikrocontroller an Pin 24 mit der gewünschten Pulsweite generiert wird.

[24] (Wikipedia, 2020)

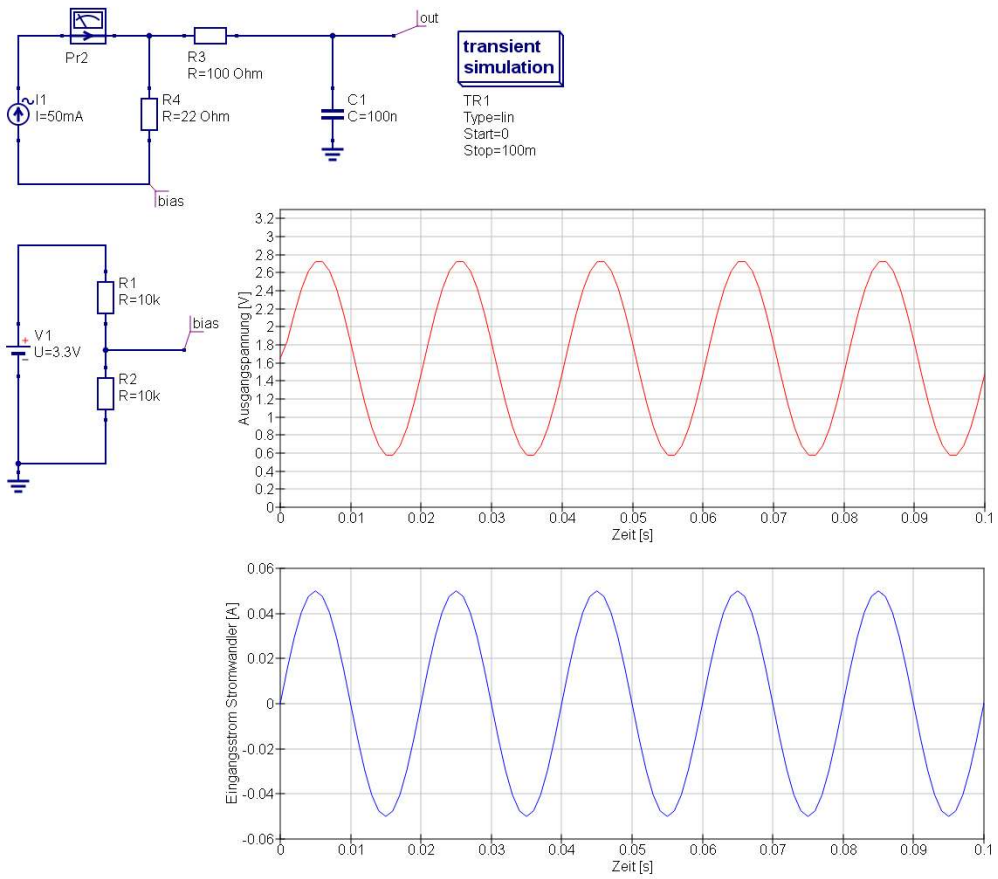

Abbildung 3: Simulation der Messschaltung für Stromwandler-Signale

Für die elektronische Trennung und Wiederverbindung der Pilotleitungen kommen zwei PhotoMOS-Relais U12 und U13 zum Einsatz, welche als Öffner wirken und von Pin 25 des Mikrocontrollers gesteuert werden. Der Taster zum Starten und Beenden des Ladevorganges sowie die darin integrierte LED werden extern am Gehäuse der Ladestation montiert und schalten von 12V nach Masse. Entsprechend [k39]wird der Treiber-MOSFET Q1 benötigt, um die LED über den Pin 9 des Mikrocontrollers zu schalten[k40], während [k41]der Taster über den strombegrenzenden Widerstand R24 und die Schottky-Diode D7 mit Pin 10 des Mikrocontrollers verbunden wird.

Zusätzlich zu den über die Terminals J1, J2, J6 und J7 nach außen geführten Anschlüssen wurden die drei 2.54mm-Pfostenleisten J3, J4 und J5 vorgesehen, um für die Inbetriebnahme und Tests die Programmier-Pins und die serielle Schnittstelle zugänglich zu machen.

Der über das Niederspannungs-Schraubterminal J6 nach außen geführte 12V-Anschluss wird mit der selbstrückstellenden 0.2A-Sicherung F1 abgesichert. Die hochspannungsseitig [k42]an J2

angeschlossene 230V-Phase wird direkt am Eingangs-Terminal von der trägen 1.25A-Sicherung F2 abgesichert.

Alle Mikrocontroller-Eingänge, deren Signale den erlaubten Bereich (0-3.3V) über- oder unterschreiten könnten, werden mit BAT54S-Doppel-Schottky-Dioden zwischen 3.3V und Masse geschützt. Der Eingang des Pilotsignal-Spannungsfolgers wird zusätzlich mit der parallel liegenden ESD-Schutzdiode D6 gegen Spannungsspitzen geschützt.

4.2.2 Pinbelegungen

Die Pinbelegungen des ATmega4808 wurden gemäß Abbildung 4 festgelegt. Es wurde darauf geachtet, dass die zwei zuletzt noch freien Pins auf PF0 und PF1 gelegt wurden, so dass bei Bedarf eine zusätzliche serielle Schnittstelle verfügbar wäre. Wird diese nicht verwendet, können diese beiden Pins als allgemeine I/O-Pins verwendet werden, z.B. um zusätzliche LED anzuschließen.

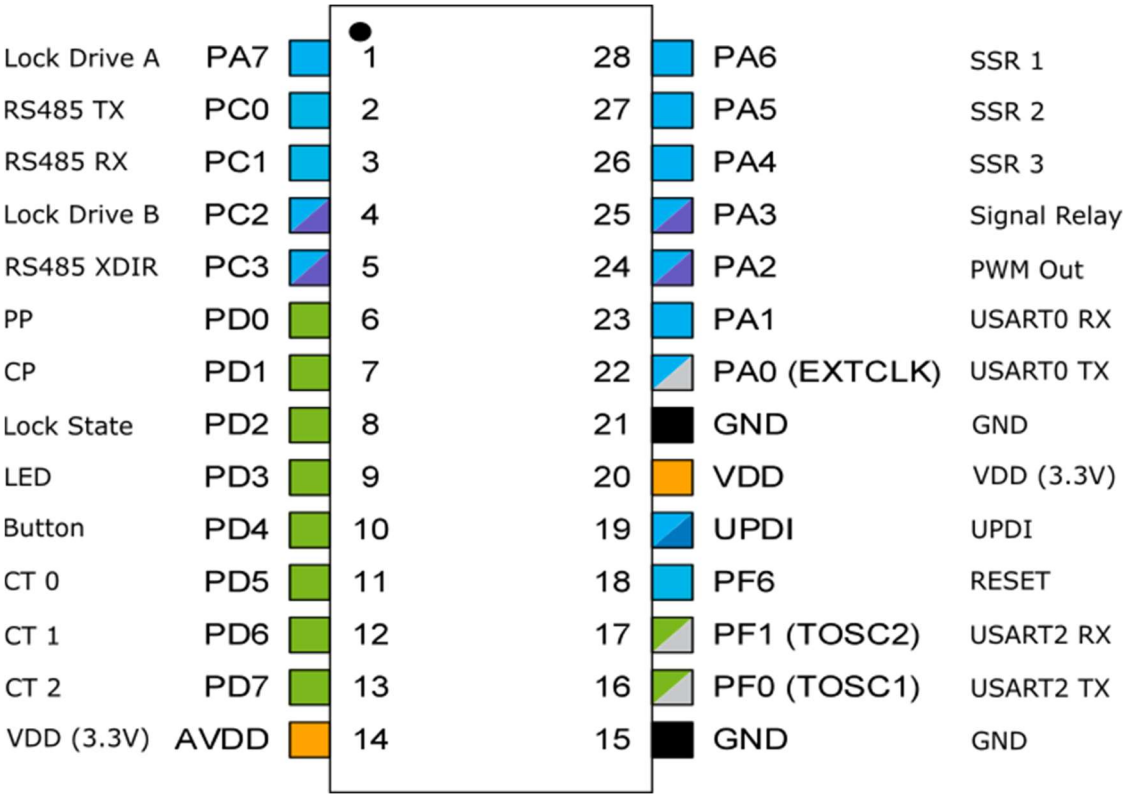

Abbildung 4: Pinbelegung des ATmega4808[25]

[25] (Microchip Technology, 2020)

Die Pins 1 – 5 werden als digitale Ausgänge genutzt, wobei Lock Drive A und B die Steuersignale für den Gate-Treiber zur [k43]Ver- und Entriegelung des Ladesteckers dienen. Pin 2 sendet serielle Nachrichten an den RS485-Transceiver, Pin 3 ist der Empfänger. Pin 5 steuert Sende- und Empfangsrichtung am Halb-Duplex-Transceiver. An CP & PP liegen die Pilotsignale an, beides sind Eingänge des Analog-Digital-Umsetzers.

An Pin 8 liegen bei verriegeltem Stecker 3.3V an, ansonsten 0V. Über Pin 9 wird die LED des Tasters geschalten, Pin 10 liest diesen aus. An Pins 11 – 13 werden die Analogsignale der Stromwandler gemessen, Pin 14 kann bei sensiblen Schaltungen als getrennte Analog-Versorgungsspannung genutzt werden, hier war dies nicht nötig, entsprechend liegt der Pin an 3.3V.

An Pin 16 und 17 kann optional eine dritte serielle Schnittstelle genutzt werden, im Normalfall sind diese Pins ungenutzt. Am UPDI-Pin (19) wird ein Programmieradapter angeschlossen und der Mikrocontroller über das UPDI-Protokoll programmiert.

Pins 22/23 dienen als primäre serielle Schnittstelle und sind im Normalbetrieb direkt mit der seriellen Schnittstelle des ESP32-Moduls verbunden, wodurch drahtlose serielle Kommunikation via WLAN ermöglicht wird. An Pin 24 wird das 1-kHz-Rechtecksignal ausgegeben, welches für die Kommunikation mit dem angeschlossenen [k44]EV verwendet wird. Über Pin 25 werden die PhotoMOS-Relais zur Trennung der Pilotleitungen angesteuert, während an Pin 26 - 28 die Halbleiterrelais zur Steuerung der Wechselstrom-Schütze angeschlossen sind.

Die Pinbelegung des ESP32-Moduls wird gemäß Tabelle 4 festgelegt. Die Pins IO12, IO26 und IO17 werden dabei mit den Pins 18, 22 und 23 des ATmega verbunden, sodass zwischen den beiden Mikrocontrollern serielle Kommunikation stattfinden und das ESP-Modul den ATmega4808 über den RESET-Pin neu starten kann.

VDD / Pin 2	3.3V
GND-Pins	Masse
TXD0 / 35	USART0 TX
RXD0 / 34	USART0 RX
IO0 / 25	Programmierung
IO12 / 14	ATmega RESET
IO16 / 27	USART2 RX
IO17 / 28	USART2 TX
EN / 3	3.3V

Tabelle 4: Pinbelegung des ESP32-WROOM-Moduls

Die weiteren Pins werden auf die Stiftleiste J5 gelegt, zusammen mit dem UPDI-Pin des ATmega, diese Leiste dient als Programmierschnittstelle für die beiden Controller.

Bei Nichtverwendung des WLAN-Moduls steht an der Stiftleiste J4 die serielle Schnittstelle UART0 des ATmega zur Verfügung, um den Mikrocontroller kabelgebunden zu programmieren.

4.3 Leiterplattenlayout

Für das Leiterplattenlayout waren folgende zentralen Anforderungen zu berücksichtigen:

- Leiterplattenabmessungen müssen für DIN-Hutschienengehäuse ausgelegt werden. Durch die zusätzlichen Bauteile gegenüber dem SmartEVSE-Projekt muss für die Ladestation ein größeres Gehäuse gewählt werden. Die Wahl fiel auf ein 6 Teilungseinheiten breites Standard-Hutschienengehäuse von Axxatronic, wodurch die Platinenabmessungen auf 86.5mm x 103mm festgelegt wurden.
- Bereiche mit 230V-Netzspannungen sollen deutlich gekennzeichnet und vom Niederspannungsteil (12V und 3.3V) räumlich getrennt sein.
- Im Netzspannungsbereich ist auf ausreichende Abstände zwischen den Leiterbahnen und ausreichend dimensionierte Leiterbahnbreiten zu achten.
- Anschlüsse für Signalleitungen und jene für Versorgungs- und Netzspannungsleitungen sind an gegenüberliegenden Seiten der Platine auszuführen.
- Nach Möglichkeit sind SMD-Bauteile zu verwenden, um den begrenzten Platz auf der Leiterplatte optimal auszunutzen. Allerdings sollen alle Bauteile von Hand zu löten sein.
- Die Leiterplatte soll zweilagig ausgeführt werden, da mehr Kupferlagen zusätzliche Kosten verursachen würden, und für eine Schaltung dieser Komplexität unnötig sind.

Das Layout wurde mit dem in Kicad 5.1.5 enthaltenen Programm pcbnew erstellt. Dadurch konnte der zuvor mit eeschema erstellte Schaltplan direkt übernommen werden. Für die meisten verwendeten Bauteile waren die benötigten Footprints (Ätzschablonen für die Leiterplattenfertigung) bereits in der Kicad-Bibliothek enthalten. Im Footprint werden Kupferpads, Pin-Bohrungen und Bestückungsdruck für das Bauteil festgehalten. Für gewisse Teile, beispielsweise die Schraubterminals und die Pin-Header, mussten spezielle Footprints mit Legende für die Anschlussbelegung erstellt werden. Dies ist mit dem in Kicad integrierten Footprint-Editor einfach möglich. Für die meisten Bauteile stellen Lieferanten oder Hersteller Footprints für die meisten gängigen Layout-Programme zur Verfügung.

Bei der Erstellung des Layouts werden die im Schaltplan logisch angeordneten und verschalteten Bauelemente, bzw. ihr Abbild als Footprint physisch auf der Leiterplatte platziert und mit Kupferbahnen verbunden. Hier war speziell auf die Trennung von Hoch- und Niederspannungsbereichen zu achten, sowie auf die räumliche Anordnung der Bauelemente mit Bezug zum geplanten Hutschienengehäuse. Die SMD-Bauteile des Analog- und Digitalteils der Schaltung wurden dabei auf der Unterseite der Leiterplatte angeordnet, während räumlich

ausgedehnte Bauelemente wie Schraubterminals, Stiftleisten, Elektrolykondensator, AC/DC-Netzteil und weitere auf die Oberseite gesetzt wurden. Soweit die begrenzte Ausdehnung der Leiterplatte dies zuliess, wurde darauf geachtet, die Bauteile möglichst als Funktionsblöcke (ähnlich der Anordung im Schaltplan) zu gruppieren. Diese Vorgehensweise erleichtert beim Test der Hardware die Fehlersuche und das Auffinden einzelner Bauteile. Speziell bei kleinen SMD-Widerständen und -Kondensatoren ist das hilfreich. Um die SMD-Bauteile von Hand löten zu können, wurde bei der Auswahl der Footprints darauf geachtet, jeweils die „HandSoldering"-Varianten zu verwenden, welche über etwas großzügiger ausgelegte Pads verfügen.

Da die Schaltung nicht für empfindliche Hochfrequenzanwendungen ausgelegt werden musste, gab es bei der Planung der Kupferbahnen keine entsprechenden Anforderungen. Dadurch konnten die Leitungen effizient auf der Leiterplatte verteilt werden.

Die Fertigung der Leiterplatten wurde bei AISLER B.V.[26] in Auftrag gegeben, zunächst wurden sechs Prototypen der Revision 1 bestellt. Nach den ersten Praxistests wurde eine Revision des Layouts vorgenommen, diese Version ist funktionsfähig und kann als Grundlage für die weiter Firmwareentwicklung verwendet werden.

Der Bestellprozess für Kleinserien von Leiterplatten bei AISLER B.V. ist sehr geradlinig gehalten, ein großer Vorteil ist die Möglichkeit, die Layoutdateien direkt als Kicad-Projekt hochzuladen. Dazu wird über die Schaltfläche "Projekt importieren" die *.kicad_pcb-Datei des Projekts (zu finden im Github-Repository gemäß Abschnitt 1) ausgewählt. Die für die Fertigung benötigten Layout-Dateien (Gerber-Dateien, Dateiendung .GBR) werden dann automatisch generiert. Ein digitales Abbild der zu produzierenden Leiterplatte kann dann im Detail inspiziert werden, um allfällige Fehler noch zu finden. Es kann zwischen verschiedenen Kategorien gewählt werden, wobei die Prototypen als zweilagige „Beautiful Boards HD" bestellt wurden. Die Leiterplatten können in Vielfachen von drei Exemplaren bestellt werden, wobei die Kosten für die Platinen nur von Abmessungen und Anzahl der Kupferlagen abhängt, nicht von der Anzahl und Art der Bohrungen oder ähnlichen Fertigungsaspekten.

Obige Ausführungen sollen den Bestellvorgang für die Prototypen-Leiterplatten illustrieren, und keinesfalls als Werbung verstanden werden. Selbstverständlich können die im Repository enthaltenen Gerber-Dateien auch für die Fertigung der Platinen bei beliebigen anderen Lieferanten verwendet werden. Details zu den benötigten Gerber-Dateien kann der jeweilige Fertiger zur Verfügung stellen.

[26] https://aisler.net

4.4 Prototypenbau

4.4.1 Bauteilbeschaffung

Sämtliche Bauteile für das Projekt wurden bei Mouser Inc.[27] in den USA bestellt, da dieser Lieferant über eine äußerst umfangreiche Auswahl an Bauelementen verfügt und darauf geachtet wurde, möglichst alle benötigten Teile aus einer Hand zu beziehen. Eine vollständige Materialstückliste ist in Anhang B zu finden. Für die meisten verwendeten Bauteile sind auch pinkompatible Alternativen erhältlich, so dass auch andere Lieferanten für die Bauteilbeschaffung in Frage kommen könnten. Allerdings wäre in diesem Fall unbedingt das jeweilige Datenblatt des Herstellers zu konsultieren und alle relevanten Parameter zu vergleichen.

4.4.2 Bestückung

Die Prototypen wurden schrittweise bestückt, zunächst einmal nur mit 3.3V Spannungsregler, ATmega4808 und ESP32-WLAN-Modul sowie deren passive Außenbeschaltung, um die Funktion dieses Schaltungsteils zu testen. Nach ersten Erkenntnissen bezüglich falsch montierter Bauteile konnte diese Teilschaltung in Betrieb genommen und erfolgreich getestet werden.

Die Bestückung der SMD-Bauteile stellte sich als durchaus von Hand machbar heraus, einzelne Bauteile wie der Spannungsregler U6 im SOT26-6-Gehäuse erforderten jedoch einiges Fingerspitzengefühl.

Bei allen hier verwendeten SMD-Bauteilen ist die Vorgehensweise prinzipiell dieselbe: Zunächst wird ein Pad auf der Leiterplatte verzinnt, dann das Pad erneut erhitzt und das Bauteil aufgelegt. Dann können bei zweipoligen oder dreipoligen Bauelementen die restlichen Pads nacheinander verlötet werden. Bei integrierten Schaltkreisen in SOT- oder SOIC-Gehäusen müssen nach dem Fixieren des Bauelementes an einem Pad die restlichen Pads mit viel Flussmittel verlötet werden, allfällige Lötbrücken können dann mit Flussmittel und Entlötlitze wieder entfernt

[27] https://www.mouser.de

werden. Mit dieser Vorgehensweise konnte auch der ATmega4808 im SOIC-28-Gehäuse problemlos bestückt werden.

Für eine serienmäßige Produktion der Ladestationen, oder auch eine größere Anzahl an Einzelstationen, würde sich auf jeden Fall die Beschaffung einer SMD-Schablone beim Leiterplattenfertiger empfehlen, um die Bauteile unter Verwendung von Lötpaste in einem Lötofen zu bestücken. Beim Handlöten unumgänglich ist die Möglichkeit, eine Heissluft-Lötstation zu benutzen, um Fehler bei der Bestückung von integrierten Bausteinen mit hoher Pin-Anzahl korrigieren zu können.

Die zweite bestückte Leiterplatte wurde vollständig bestückt, abgesehen vom AC/DC-Netzteil U4, stattdessen wurden die sekundärseitigen [k45]Anschlüsse von U4 mit Stiftleisten versehen, um die Schaltung über ein 12V-Labornetzteil mit Spannung zu versorgen. Dadurch konnten Tests der Hardware ohne direkte Anbindung an Netzspannung in einer sicheren Arbeitsumgebung durchgeführt werden.

Erst nach erfolgreichen Tests aller Funktionseinheiten wurde U4 bestückt und die Schaltung an eine einphasige Wechselstromleitung angeschlossen. Damit konnten die Funktion der Halbleiterrelais U1-U3, das Schalten der Wechselstromschütze, sowie die Spannungsversorgung der gesamten Schaltung überprüft werden, bevor der Prototyp zum finalen Praxistest an eine dreiphasige Drehstrom-Leitung angeschlossen wurde.

4.4.3 Revisionsliste

Bei den Tests der Revision 1 konnte die Funktionalität aller Hardware-Funktionseinheiten bestätigt werden, bis auf die Beschaltung der 12V-Schiene beim Gate-Treiber U8, hier wurde durch das 100Ω-Widerstandsnetzwerk der maximale Strom zu stark begrenzt, wodurch keine zuverlässige Verriegelung des Ladesteckers gewährleistet werden konnte. Weitere Untersuchungen im Labor ergaben, dass eine Anpassung des Widerstandsnetzwerkes auf einen resultierenden Widerstand von 25 Ohm durch Parallelschaltung von R3-R6 dieses Problem behob.

Weitere, für die unmittelbare Funktion der Schaltung unkritische Verbesserungen wurden wie folgt vorgenommen:

- Bei der Erstellung des Leiterplattenlayouts war mir bei der Auswahl der Footprints für J4 und J5 ein Fehler unterlaufen, wodurch die Pins in der falschen Reihenfolge belegt

wurden. Außerdem stellte sich heraus, dass für J4 eine einreihige, vierpolige Stiftleiste ausreichend war. Für J5 wurde die Belegung leicht angepasst, anstelle eines redundanten 3.3V-Pins (Pin 7) wurde ein zusätzlicher IO-Pin des ESP32-Moduls (IO15) zugänglich gemacht. Die 2x4 Stiftleiste wurde durch eine verpolungssichere Variante ersetzt, um die Konfektionierung und den Einsatz eines Programmiersteckers für die Schaltung zu vereinfachen.

- Zur Erhöhung der mechanischen Stabilität verfügen die verwendeten 6-TE-Hutschienengehäuse über Trennstege auf halber Länge. Die ursprüngliche Ausführung der Schraubterminals als durchgehende Blöcke geriet damit in Konflikt, worauf die Anschlussterminals J1 und J2 in der Mitte halbiert und etwas versetzt werden mussten.
- Um die empfindlichen Analogeingänge des ATmega4808 ($-0.5V \leq U_{Pin} \leq U_{Versorgung} + 0.5V$) vor allfälligen Spannungsspitzen zu schützen, wurden die Eingänge der Stromwandler (CT0 – CT2) und der Pilotleitung CP mit BAT54S-Schottkydioden versehen.
- Zur einfacheren Verdrahtung der Ladestation mit den Wechselstrom-Schützen wurde am netzseitigen Schraubterminal der Neutralleiter für jedes Schütz separat zugänglich gemacht.
- Zuletzt wurden das Symbol und der Footprint für den Tantal-Kondensator C16 so angepasst, dass dessen polare Natur deutlich zu erkennen ist, was die Chance für eine Fehlbestückung stark verringert.

5. Firmwareentwicklung

5.1 Konzept

Die Firmware des ATmega4808 soll folgende Aufgaben erfüllen:

1) Steuerung des Ladevorganges nach der Verbindung eines EV mit der Ladestation
2) Bereitstellung einer seriellen Schnittstelle als Grundlage der Kommunikation mit dem WLAN-Modul, über die RS485-Schnittstelle oder direkt mit einem PC/Laptop über den seriellen Port
3) Kommandozeilen-Interpreter, über welchen sich sämtliche Aspekte des Ladevorganges steuern lassen, und über welchen[k46] sich interne Parameter der Ladestation auslesen und verändern lassen.

Zusätzlich zur eigentlichen Anwendungsfirmware auf dem ATmega-Mikrocontroller muss ein Bootloader entworfen oder[k47]gefunden werden, welcher die Möglichkeit bietet, über die serielle Schnittstelle Firmware-Updates auf den Mikrocontroller zu laden.

5.1.1 Bootloader

Ein Mikrocontroller-Bootloader ist im Prinzip eine einfache Firmware, welche als Startprogramm nach dem Reset des Controllers als erstes geladen und ausgeführt wird. Dieses Programm startet dann,[k48] je nach Situation, entweder eine vorhandene Anwendung, oder nimmt Befehle zum Beschreiben des Flash-Speichers mit einer neuen Anwendung im Binärformat, bei AVR-Prozessoren in der Regel im Intel .HEX-Format[k49], entgegen. Ein Bootloader kann auch weitere Funktionalität, beispielsweise eine Debugging-Konsole oder Hardware-Testroutinen beinhalten. Im vorliegenden Fall soll der Bootloader ausschließlich zum Starten und Aktualisieren der Firmware dienen.

Der Bootloader soll eine einfache serielle Schnittstelle zur Verfügung stellen, so dass neue oder aktualisierte Firmware ohne den Umweg über einen Programmieradapter (wie ATMEL-ICE o.ä.) via WLAN-Modul [k50]auf den Mikrocontroller geladen werden kann. Dadurch kann bei bereits installierten Ladestationen auf physischen Zugang und eine Öffnung des Gehäuses verzichtet

werden, sofern das WLAN-Modul und der Bootloader bei der Inbetriebnahme des Systems bereits programmiert wurden.

Diese geforderte Funktionalität wird vom unter der GPLv2 veröffentlichten Optiboot-Bootloader[28] implementiert. Optiboot ist auf minimalen Speicherbedarf und Ausführungsgeschwindigkeit optimiert, indem nur ein kleiner Teil des Atmel STK500-Kommunikationsprotokolls implementiert wird. Der Ablauf des Bootprozesses sieht folgendermaßen aus:

1. Nach jedem Zurücksetzen des Mikrocontrollers wird Optiboot gestartet und liest den Grund für den Reset aus dem entsprechenden Register der CPU. Für alle Resetgründe außer [k51]„External Reset“ wird automatisch die Anwendung gestartet.
2. Falls ein externer Reset erkannt wird, versucht der Bootloader neue Anwendungsfirmware zu laden.
3. Dazu blinkt [k52]eine konfigurierbare LED als Signal für den Start des Bootloaders, während die serielle Schnittstelle gemäß der im Vorfeld angepassten Konfiguration (Port, Baudrate) initialisiert wird, neben einem Watchdog Timer (WDT) mit einer Frist von einer Sekunde.
4. Über die serielle Schnittstelle versucht Optiboot, Befehle zu empfangen. Gültige Befehle setzen den WDT zurück und werden anschließend ausgeführt.
5. Wenn keine gültigen Befehle empfangen werden, oder wenn der Programmiervorgang abgeschlossen ist, setzt der WDT die CPU zurück, worauf die Anwendung, wie unter Punkt 1 erläutert, gestartet wird.

Für die neueren Mega-0-Serie Mikrocontroller, zu welchen auch der hier verwendete ATmega4808 gehört, wurde aufgrund der leicht veränderten CPU-Architektur eine überarbeitete Version des Bootloaders entwickelt, welche unter der Bezeichnung OptibootX firmiert. Diese Version erfüllt alle Anforderungen dieses Projekts und wurde dementsprechend mit angepasster Konfiguration übernommen.

[28] (Optiboot Project, 2020)

5.1.2 Anwendungsfirmware

Die Firmware des Mikrocontrollers, als eigentliche Anwendungssoftware, soll die Steuerung der Hardware während des Ladevorgangs übernehmen und Kommunikationsfunktionen für die Fernsteuerung und –Wartung des Systems bereitstellen. Dazu wird die Firmware in drei Module aufgeteilt:

- Main: Hauptmodul, beinhaltet die eigentliche Anwendung und alle Subroutinen zur Kontrolle des Ladevorgangs
- UART: Implementierung der seriellen Schnittstelle
- CMD: Implementierung des Kommandozeileninterpreters und zugehöriger Hilfsfunktionen

Die Firmware soll bewusst so einfach und platzsparend wie möglich entworfen werden, da sämtliche Netzwerk- und Lastverteilungs-Funktionalität für den Mesh-Betrieb in die ESP32-Firmware bzw. in die Software des zentralen Servers ausgelagert werden. Diese Aspekte des Gesamtsystems fallen aber nicht in den Umfang dieser Arbeit.

Als Basis für das Hauptmodul wurde die für den PIC18-Mikrocontroller entworfene Firmware des SmartEVSE-Projekts wiederverwendet, angepasst auf die verwendete Hardware und portiert für die AVR-Architektur des ATmega4808. Dieses Modul ist im Prinzip als Abfolge von `if()…else()`–Abfragen zur Ermittlung des aktuellen und des Folgezustandes gemäß Tabelle 1 aufgebaut, mit zusätzlichen Klauseln zur Ermittlung von Tastendrücken, Zeitmessung, Strommessungen, Nachrichtenempfang und Kommandozeilen-Verarbeitung. In einem weiteren Schritt könnten diverse dieser Aufgaben in Interrupt-Service-Routinen ausgelagert werden, aus Gründen der Komplexität wurde hier aber darauf verzichtet.

Das UART-Modul baut auf den Application Notes von Atmel auf und befasst sich vornehmlich mit der Initialisierung der USART-Schnittstelle des ATmega4808 und einer Anzahl von Hilfsfunktionen für das Senden und Empfangen von Nachrichten.

Das CMD-Modul bedient sich beim AVR_CMD_INTERFACE-Projekt von Gerhard Brünner[29], erweitert um die Möglichkeit zur Parameter-Abfrage und –Bearbeitung sowie einige Vereinfachungen. Damit wird ein einfaches, menschenlesbares und dennoch maschinentaugliches Kommunikationsprotokoll implementiert, welches die Fernsteuerung des

[29] (Brünner, 2009)

gesamten Ladevorganges ermöglicht. Das verwendete Protokoll wird in Abschnitt 5.3.3 genauer erläutert.

5.2 Entwicklungsumgebung

Die Firmware-Entwicklung für dieses Projekt wurde vollständig in C ausgeführt. Diese Programmiersprache ist aufgrund ihrer Hardware-Nähe und der frei verfügbaren GNU Compiler Collection (GCC)-Toolchain bestens für die Firmware-Entwicklung von eingebetteten Systemen geeignet. Für AVR-Mikrocontroller existiert mit AVR-GCC ein plattformübergreifender C-Compiler, aktuell in Version 5.4.0.

Die Entwicklung und der Test der Firmware wurde unter Microsoft Windows 10 vorgenommen, da für diese Plattform mit Atmel Studio 7.0 eine umfangreiche, kostenlose und immer aktuell gehaltene Entwicklungsumgebung von Atmel/Microchip zur Verfügung steht. Atmel Studio basiert auf Microsoft Visual Studio, enthält aber diverse Werkzeuge zur erleichterten Firmware-Entwicklung für Atmel AVR-Prozessoren. Darunter sind ein CPU-Simulator, welcher auf Basis von Hardware-Modellen den Programmablauf simulieren kann, umfangreiche Möglichkeiten zur Inspektion von Registern und Speicher des Mikrocontrollers und Unterstützung für den Atmel-ICE-Programmieradapter und Debugger. Dieser Adapter diente für die Entwicklung als zentrale Schnittstelle zwischen PC und Mikrocontroller/Ladestation.

Die in Atmel Studio 7.0 enthaltene GCC-Toolchain beinhaltet bereits sämtliche prozessorspezifischen Header-Dateien und die AVR-Libc-Standardbibliothek für AVR-Mikrocontroller. In dieser Bibliothek sind diverse Standardfunktionen wie Warteschleifen (util/delay.h), vordefinierte C-Präprozessor-Makros für AVR-Chips (avr/io.h) und Definitionen für Interrupt-Vektortabellen (avr/interrupt.h) enthalten. Regelmäßig werden von Atmel sogenannte Device Packs veröffentlicht, welche aktualisierte Definitionen für neu erschienene Prozessoren enthalten. Diese können direkt aus Atmel Studio heraus aktualisiert werden. Des Weiteren wird für die erfolgreiche Kompilierung der Ladestations-Firmware die C-Standardbibliothek Libc benötigt, welche in der GCC-Toolchain mit enthalten ist.

Zur Kommunikation über die serielle Schnittstelle wurde die Software Hterm 0.8.5 verwendet. Dabei handelt es sich um[k53] ein kostenloses Programm zum Versenden, Empfangen und Analysieren von seriellen Nachrichten. Bei PCs ohne hardwareseitig vorhandene serielle

Schnittstelle kann mittels eines einfachen Serial-to-USB-Adapters die Kommunikation zur Ladestation aufgebaut werden.

Da die gesamte AVR-GCC-Toolchain auch für Linux-Betriebssysteme zur Verfügung steht, ist grundsätzlich eine Entwicklung der Firmware unter Linux möglich, allerdings ist die Konfiguration der aktuellen Device Packs von Atmel etwas aufwändiger, da Atmel Studio 7.0, welches unter Windows die Installation der Device Packs vereinfacht, für Linux nicht verfügbar ist.

5.3 Funktionale Beschreibung

5.3.1 Main

Das Hauptmodul besteht aus zwei Dateien, der Headerdatei main.h und der zugehörigen Quellcode-Datei main.c.

Die Header-Datei enthält sämtliche Makro-Definitionen für die konfigurierbaren Parameter wie maximale Strombelastbarkeit der Zuleitungen, verfügbare Leistung, Temperaturgrenzwerte, Kabelkonfiguration und viele mehr. Einige Definitionen müssen vor der Kompilierung der Firmware vorgenommen werden, wie beispielsweise die Angabe, ob ein fest installiertes Kabel oder eine Typ-2-Buchse an der Ladestation installiert wird. Zur Laufzeit veränderliche Parameter[k54], wie maximale und minimale Ladeströme, können auch über die serielle Schnittstelle, je nach Bedarf, [k55]angepasst werden. Außerdem werden in der Header-Datei sämtliche globalen Variablen und global aufrufbaren Funktionen des Hauptmoduls deklariert.

In der Quellcode-Datei werden die im Header deklarierten Variablen und Funktionen definiert, sowie die Datenstrukturen für Kommandozeilen-Befehle und –Parameter initialisiert. Dadurch sind diese Strukturen bereits beim Kompilieren der Firmware bekannt, was die Speicherplatzeffizienz des Programms erhöht.

Der Quellcode für das Hauptmodul enthält die `main()`-Funktion, welche den Kern des Anwendungsprogramms darstellt. In `main()` befindet sich eine Endlosschleife, welche wiederum aus einer Abfolge von `if()`−Abfragen besteht. Diese `if()`−Blöcke behandeln die drei implementierten Zustände A-C. Zustand D (Laden mit Belüftung, siehe Tabelle 1) ist derzeit nicht vorgesehen, kann aber bei Installationen mit Belüftungseinrichtungen zusätzlich implementiert werden.

Nach dem Start des Systems befindet sich die Ladestation in Zustand A, hier sind alle Schütze offen geschaltet, der Ladestecker ist spannungsfrei. An der CP-Leitung liegen konstant +12V an, das Rechtecksignal ist deaktiviert. Nun wird über die Funktion `readCP()` die Spannung an CP gemessen, um ein angeschlossenes EV zu erkennen. Falls das Ladekabel in der Buchse verriegelt ist, wird die Verriegelung deaktiviert. Nach drei Versuchen (konfigurierbar mit `MAX_UNLOCK_ATTEMPTS`) wird die Entriegelungsfunktion beendet, der Mechanismus muss dann manuell geöffnet werden.

Wenn die Pilotleitung weiterhin bei +12V liegt, verbleibt die Ladestation in Zustand A. Falls die Spannung an CP durch den bei angeschlossenem Ladekabel parallel liegenden 2.7kΩ-Widerstand auf +9V absinkt, wechselt die Ladestation nach einer kurzen Wartezeit (um Schwankungen der Spannung zu filtern) in den Zustand B, nachdem über das Auslesen der Spannung an der PP-Leitung mithilfe der Funktion `readPP()` die kodierte Strombelastbarkeit des Ladekabels bestimmt wurde. Der Zustandswechsel findet nur dann statt, wenn zuvor die Variable `access` auf 1 gesetzt wurde. Dies geschieht je nach Wert des `SWITCH`-Makros entweder automatisch (`SWITCH = 0`, automatisches Laden nach Herstellung der Verbindung) oder nach Betätigen des Tasters (`SWITCH = 1`, manueller Start/Stopp des Ladevorganges).

Nach dem Wechsel in Zustand B wird das 1-kHz-Rechtecksignal aktiviert und die Pulsweite gemäß der ermittelten Strombelastbarkeit des Ladekabels eingestellt. Falls die Kabelkapazität über dem maximal verfügbaren Strom der Ladestation liegt, wird stattdessen dieser Maximalstrom eingesetzt. In Zustand B wird regelmäßig die Spannung auf der CP-Leitung überprüft, um eine getrennte Verbindung zu EV (U_{CP} = +12V) oder den Beginn des Ladevorganges (U_{CP} = +6V) zu erkennen. Da die Spannung an CP aber durch das Rechtecksignal zwischen U_{CP} und -12V schwankt, muss diese Messung jeweils zum Beginn der 1ms-Periode des Rechtecksignals vorgenommen werden.

Sobald durch das Fahrzeug / den Fahrer der Ladevorgang gestartet wurde, wird noch einmal überprüft, ob keine Fehlermeldungen vorhanden sind und ob die Diode im Ladestecker korrekt funktioniert. Sind diese Bedingungen erfüllt, werden die Schütze geschaltet und der Ladestrom beginnt zu fließen. Damit wechselt die Ladestation in Zustand C. Zum Start des Ladevorganges wird der Ladestecker verriegelt. Falls [k56]nach drei Versuchen (konfigurierbar mit `MAX_LOCK_ATTEMPTS`) die Verriegelung nicht erfolgt ist, wird der Ladevorgang abgebrochen.

In diesem Zustand verbleibt das System so lange, bis entweder eine Fehlspannung (0V oder -12V) auf der Pilotleitung gemessen wird, der Ladestecker entfernt wird oder das Ladegerät des Fahrzeugs durch Auskopplung des zweiten parallelen Widerstands U_{CP} wieder auf +9V erhöht, was das reguläre Ende des Ladevorgangs anzeigt

Im[k57] weiteren Verlauf der main-Endlosschleife werden einerseits die Systemzeit in ms-Schritten hochgezählt, und jede Sekunde einmal sowohl die Temperatur[TP58] als auch die Ladeströme der drei Phasen gemessen[TP59]. Die Temperaturmessung benutzt dazu den im ATmega4808 integrierten Temperatursensor, der über einen eigens dafür vorgesehenen Kanal des ADC-Peripheriemoduls angesteuert wird. Da die Ausgangsspannung des Sensors

produktionsbedingt stark variieren kann, sind im Festwertspeicher des Mikrocontrollers Korrektur- und Kalibrationswerte hinterlegt.[30] Falls eine zu hohe Temperatur, verglichen mit der Makro-Konstante `MAX_TEMP`, gemessen wird, wird der Ladevorgang unterbrochen. Erst nach erfolgter Abkühlung des Systems unter die Obergrenze wird das Laden fortgesetzt.

Die Messung der Ladeströme erfolgt in einem Intervall von einer Sekunde, wobei jeweils in einem Durchgang 512 Messwerte des Analog-Digital-Umsetzers gesammelt werden, und mithilfe eines Filteralgorithmus der Gleichspannungsanteil herausgefiltert wird. Anschließend werden die Messwerte quadriert, aufsummiert und zum Schluss durch die Anzahl der Messwerte geteilt, um so den Effektivwert der Ladeströme zu erhalten.

Bei jedem Durchlauf der Schleife wird der Zustand des Tasters abgefragt[k60] und entsprechend der Konfiguration der Ladevorgang gestartet oder gestoppt. In regelmäßigen Abständen wird innerhalb der Schleife überprüft, ob sich eine vollständige Nachricht im Empfangspuffer der seriellen Schnittstelle befindet[k61] und in diesem Fall die Nachricht mithilfe der Funktion cmd_parse() verarbeitet und allfällige Befehle ausgeführt.

5.3.2 UART

Im UART-Modul, bestehend aus Header uart.h und Quellcode-Datei uart.c wird die serielle Schnittstelle des ATmega4808 initialisiert und werden die entsprechenden Hilfsfunktionen deklariert bzw. definiert. Da der Mikrocontroller über drei UART-Schnittstellen verfügt, werden diese entsprechend in der `uart_init()` –Funktion konfiguriert, wobei das Makro `UART_SEL` in main.h dazu dient, zwischen der primären Schnittstelle und der sekundären RS485-Schnittstelle zu wählen. Die dritte serielle Schnittstelle wird im Normalfall nicht konfiguriert, kann aber bei Bedarf durch Definition des Makros `AVR_UART2` aktiviert werden.

Durch die Definition der Streams `uart0_stream` und `uart1_stream` kann je nach Wert von `UART_SEL` die Ausgabe der `printf()`-Bibliotheksfunktion auf eine der beiden Schnittstellen umgeleitet werden, wodurch die Ausgabe von formatierten Zeichenketten stark vereinfacht wird.

Das UART-Modul basiert zu großen Teilen auf der Application Note TB3216[31].

[30] (Microchip Technology, 2020), S. 402
[31] (Microchip Technology, 2019)

5.3.3 CMD

Der Kommandozeilen-Interpreter wird im CMD-Modul mit Header cmd.h und Quellcode-Datei cmd.c implementiert. In der Header-Datei werden zunächst zwei Datentypen definiert, `cmd_table_t` und `param_table_t`. Mit `cmd_table_t` wird eine Datenstruktur geschaffen, welche eine Kombination aus Name (als Zeichenkette) und Funktion (als Funktionszeiger) für jeden implementierten Kommandozeilenbefehl speichert. Ein Feld solcher Datenstrukturen dient dann als Befehlsliste für die Verarbeitung von eingehenden Kommandozeilen. Die Datenstruktur von `param_table_t` ist ähnlich aufgebaut, musste aber zur Behandlung von unterschiedlichen Datentypen der verschiedenen Parameter (es werden Parameter-Variablen von 8-, 16- und 32-Bit Breite verwendet) um ein weiteres Feld für die Größe der Daten erweitert werden. So kann für jeden Parameter ein `void`−Zeiger, der von der tatsächlichen Größe des Datentyps unabhängig ist, verwendet werden[k62]. Dieser Zeiger wird beim Zugriff auf die Variable gemäß der hinterlegten Größe in den korrekten Zeigertypen umgewandelt.

Das Kommunikationsprotokoll für die Kommandozeileneingabe ist nach Tabelle 5 aufgebaut. Damit lassen sich zum einen Funktionen direkt ausführen, wie beispielsweise die Verriegelung des Steckers, oder die Schaltung der Schütze, zum anderen können Parameter wie maximaler Ladestrom, Systemzeit und andere abgefragt und gegebenenfalls angepasst werden. Optional wäre es möglich, für gewisse Parameter einen Schreibschutz zu implementieren, um die Veränderung kritischer Parameter zu verhindern. Für die Testzwecke im Rahmen der vorliegenden Arbeit wurde darauf verzichtet.

`<Befehl>`	Die Funktion, auf welche der Eintrag mit Name <Befehl> zeigt, wird ausgeführt
`<Parameter>?`	Der aktuelle Wert der Variable <Parameter> wird ausgegeben
`<Parameter>=<Wert>`	Die Variable <Parameter> erhält den neuen Wert <Wert> zugewiesen
`??`	Eine Liste aller definierten Befehle und Parameter wird ausgegeben

Tabelle 5: Serielles Kommunikationsprotokoll zur Fernsteuerung der Ladestation

Die Hilfsroutinen `cmd_exec()`, `param_get()` und `param_set()` dienen dazu, je nach Art der empfangenen Kommandozeilen die entsprechende Operation auszuführen. Verpackt ist die Verarbeitung in die Funktion `cmd_parse()`, welche das eigentliche Protokoll nach Tabelle 5 mithilfe der C-Standardbibliothek für Zeichenketten (definiert in string.h) umsetzt. Der Kommandozeilenbefehl `??` dient als Abkürzung für den Befehl `status`, welcher sämtliche in den beiden Datenstrukturen `cmd_table` und `param_table` enthaltenen Befehle und Parameter ausgibt. Damit kann beispielsweise ein zentraler Server im Mesh-Betrieb auf einfache Art und Weise einen Überblick über sämtliche Ladestationen im Netzwerk und deren Zustand, Firmware-Version o.ä. erhalten.

6. Test und Validierung

6.1 Testaufbau

6.1.1 Labortests

Die Labortests der fertiggestellten Hardware wurden [k63]in drei Schritten durchgeführt. Wie in Abschnitt 4.4.2 beschrieben, wurden die Platinen schrittweise bestückt und zunächst auf Layout- und Bestückungsfehler überprüft. Diese Prüfung fand mit Digital-Multimeter und Oszilloskop statt, wobei zuerst der Digitalteil der Schaltung (ATmega4808 und ESP32-Modul mit Außenbeschaltung) direkt mit 3.3V aus einem Labornetzteil versorgt wurde. Zur Funktionsüberprüfung der beiden Bauteile wurde ein einfaches LED-Blink-Testprogramm geschrieben, welches fehlerfrei ausgeführt werden konnte.

In einem zweiten Schritt wurde die restliche Bestückung vorgenommen, und die einzelnen Funktionsblöcke mit Hilfe einer Testfirmware auf fehlerfreie Funktion überprüft. Den Test des ESP32-Moduls und der seriellen Kommunikation zwischen ESP32 und ATmega4808 übernahm dabei Prof. Thomas Moor. [k64]

Zum Test der einzelnen Schaltungsteile wurde die bestückte Ladestation über Jumper-Kabel mit einem Steckbrett verbunden, auf welchem die fahrzeugseitigen Widerstandsnetzwerke, LED und Taster sowie der Ladestecker-Verriegelungsmechanismus aufgebaut wurden. Für jeden Funktionsblock wurde ein kleines Testprogramm geschrieben, dessen Code nach erfolgreichem Test in die Produktions-Firmware übernommen wurde. Somit konnten Hard- und Firmware parallel getestet und entwickelt werden.

Nach Abschluss der Tests am ESP32-Modul konnte auf diesem ein WLAN-Access-Point implementiert werden, wodurch der Zugriff auf die serielle Schnittstelle UART1 über das telnet-Protokoll ermöglicht wird. Damit eröffnet sich die Möglichkeit, den Praxistest ohne kabelgebundene Kommunikation und stattdessen über einen per WLAN verbundenen Laptop durchzuführen, was aus Sicherheitsaspekten von großem Vorteil ist.

Zuletzt wurde das 230V AC/DC-Netzteil bestückt und das System an Netzspannung getestet, als Vorbereitung für die finalen Tests an einem realen Elektrofahrzeug, einem BMW i3. Dabei wurde

als Last ein 500W Halogenstrahler eingesetzt, welcher mithilfe eines Testaufbaus zur Steuerung der Pilotsignale angesteuert und ein-/ausgeschaltet wurde.

6.1.2 Praxistest

Die Ladestation wurde für diesen Test in ein stabiles Kunststoffgehäuse eingebaut, gemeinsam mit Fehlerstrom-Schutzschalter für alle drei Phasen, zusätzlichem Sicherungsautomaten für die Spannungsversorgung der Elektronik sowie drei Stromwandlern zur Strommessung und dem Verriegelungsmechanismus für das EV-Ladekabel.[k65] In Abbildung 5 ist der Aufbau dargestellt.

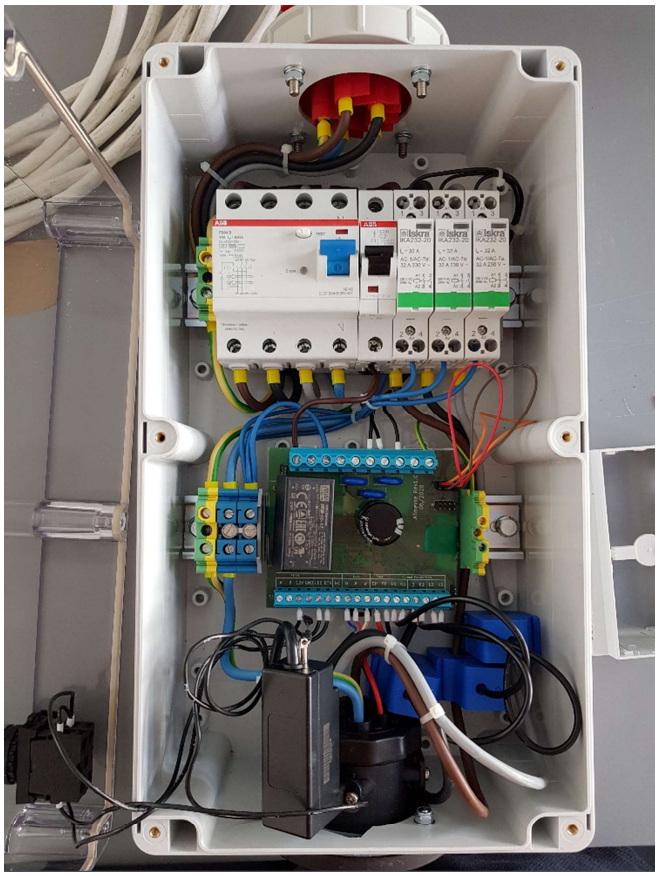

Abbildung 5: Testaufbau der Ladestation

Der Praxistest wurde an einem BMW i3 durchgeführt, welcher fahrzeugseitig einen maximalen Ladestrom von 3x16A bei Wechselstrom zulässt. Dazu wurde die Ladestation an eine 11kw/3x16A-CEE-Steckdose angeschlossen und der Ladevorgang via Laptop und WLAN-Zugang überwacht.

6.2 Ergebnisse

Der Test verlief erfolgreich, das EV konnte mit maximal 11kW geladen werden. Änderungen des maximalen Ladestroms seitens der Ladestation wurden vom Fahrzeug korrekt erkannt und der Strombedarf vom Ladegerät entsprechend angepasst. Der Ladestrom wurde von 6A bis 16A pro Phase variiert, was für dieses Fahrzeug und die zur Verfügung stehende 3x16A-CEE-Steckdose die obere und untere Grenze darstellt. Mittels der SAE J1772-Signalisierung konnte diese Strombegrenzung korrekt an das EV kommuniziert werden.

Alle grundlegenden Hardware-Funktionen der Ladestation funktionierten mit der verwendeten Test-Firmware (beschrieben in Abschnitt 5.3) ohne Probleme. Der Widerstand zwischen PP und PE wurde korrekt gemessen, ebenso die Spannung an CP. Der elektromechanische Verriegelungsmechanismus des EV-Ladekabels und die elektronische Trennung der Pilotleitungen durch die PhotoMOS-Relais erfüllten ihre Funktion wie geplant.

Softwareseitig traten kleinere Probleme auf. Zum einen stellte sich die Effektivwertberechnung (siehe Abschnitt 5.3.1[TP66]) der Ladeströme pro Phase als unzuverlässig heraus. Zwar wurden an allen drei Phasen Werte gemessen, welche proportional zum tatsächlichen Ladestrom waren, die[k67]aber über der [k68]Zeit stark variierten und nicht die realen Effektivwerte (parallel mit Benning CM 9-Stromzange gemessen) abbildeten. Hier muss softwareseitig [k69]weiter getestet und kalibriert werden[TP70].

7. Fazit

Im finalen Praxistest konnte ein Elektrofahrzeug mit bis zu 11kW (3x16A) geladen werden. Das Projekt konnte damit hardwareseitig [k71]zu einem erfolgreichen Abschluss gebracht werden. [k72]Die Hardware in Revision 1.1 behebt sämtliche in den Tests erkannten Probleme und Fehler und kann als funktionsfähig betrachtet werden.

Im Rahmen dieser Arbeit wurde die Firmware des Systems in erster Linie unter dem Gesichtspunkt der Hardware-Verifikation entwickelt, der aktuelle Stand der Firmware ist keinesfalls ausgereift. Insbesondere im Bereich der Strommessung sind weitere Tests und Versuche nötig, um mithilfe der eingesetzten Stromwandler ein zuverlässiges Messergebnis zu erhalten. Hierbei sind die Entwicklung und Implementierung effizienter Algorithmen zur Verarbeitung der Messwerte wichtige Ziele.[k73]

Die gesamte Hard- und Software, welche im Rahmen dieser Arbeit entstanden sind, werden unter einer Open-Source-Lizenz zur freien Verwendung und Weiterentwicklung veröffentlicht, so dass Einzelpersonen, Organisationen und Unternehmen das System nach Ihren Bedürfnissen weiterentwickeln und verbessern können. Insbesondere die geplante Weiterentwicklung hin zu einem Netzwerk von Ladestationen ist vielversprechend. Durch die Einbindung der entwickelten Ladestation in ein solches Netzwerk, welches über einen zentralen Server gesteuert wird, kann die verfügbare Anschlussleistung an Orten wie Großparkplätzen, Tiefgaragen oder Parkplätzen von Mehrfamilienhäusern gezielt auf sämtliche angeschlossenen Elektrofahrzeuge verteilt werden. Solche Konzepte bergen großes Potential bei der Bereitstellung von Lademöglichkeiten im öffentlichen Raum, besonders auch in Kombination mit variablen Anschlussleistungen wie beispielsweise bei Photovoltaikanlagen oder Windkraftturbinen.[k74] Zusätzlich können untereinander vernetzte Ladestationen auch mögliche Netzüberlastungen durch das gleichzeitig stattfindende Laden vieler Fahrzeuge abgemildert und verhindert werden, indem die Ladeleistung zeitlich und pro Anschluss gerecht verteilt wird.

Anhang A – Schaltpläne

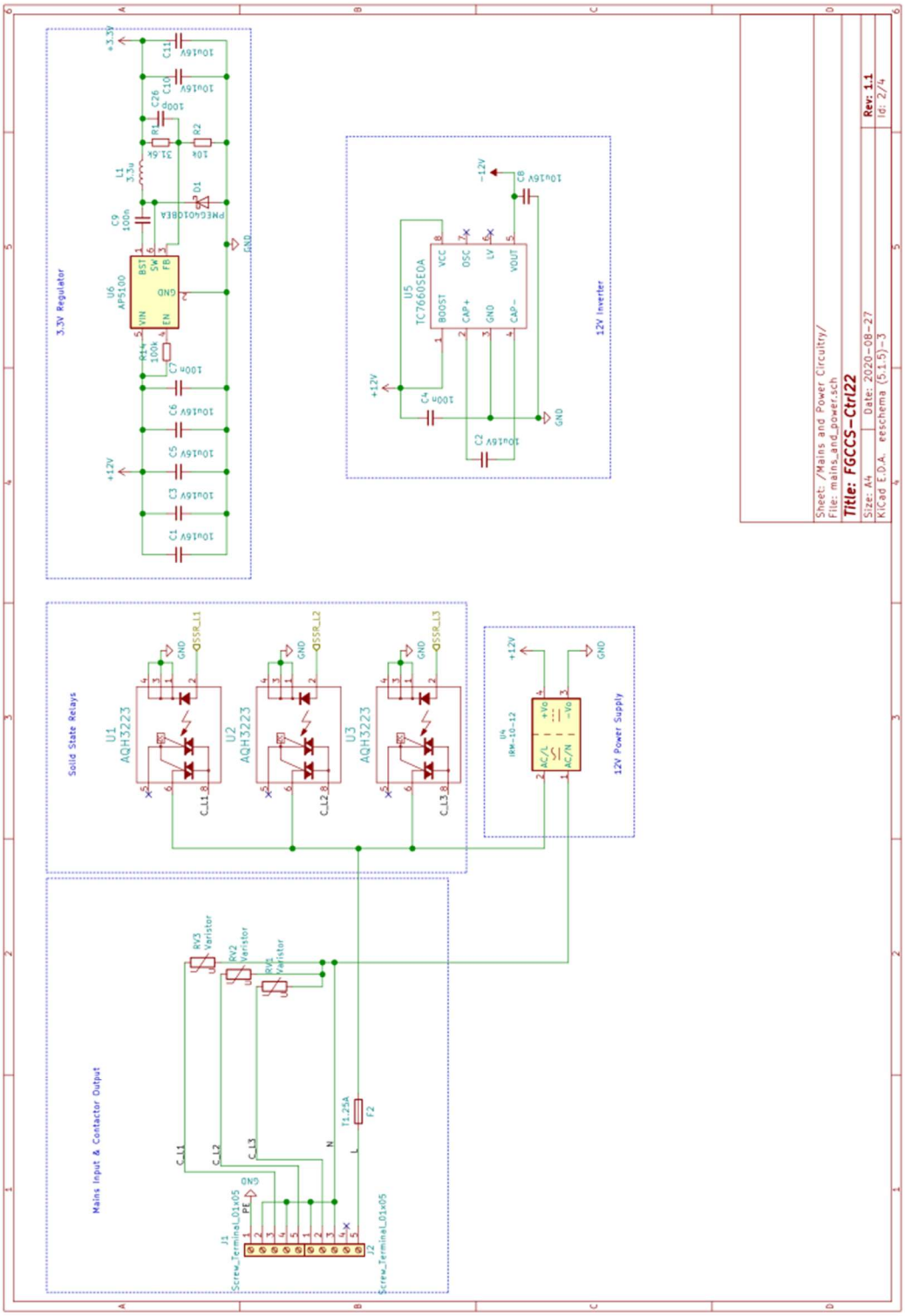

Abbildung 6: FGCCS-Ctrl22-Schaltplan, Seite 1: Netzspannungsanschlüsse, Halbleiterrelais und Spannungsversorgung

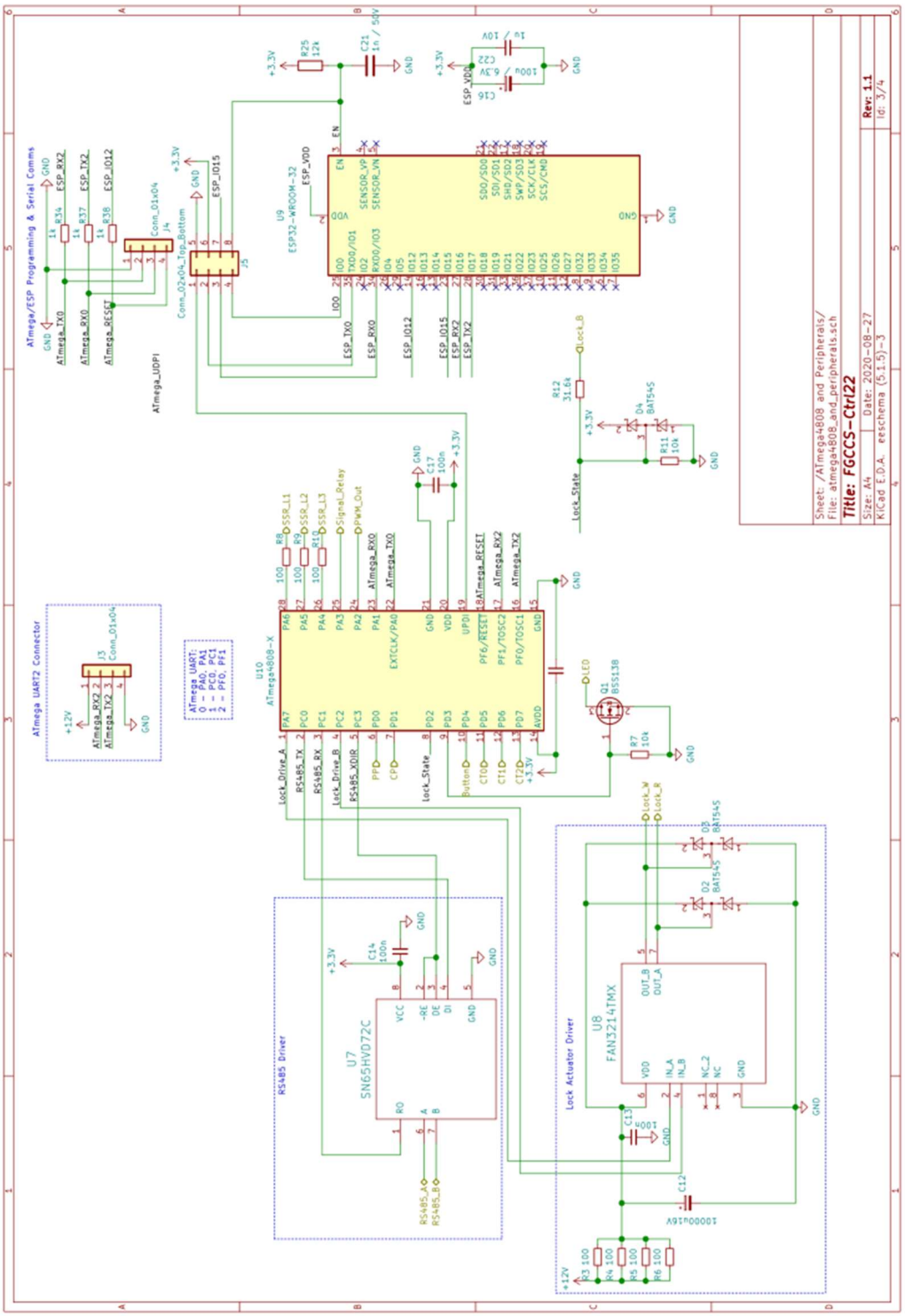

Abbildung 7: FGCCS-Ctrl22-Schaltplan, Seite 2 – Mikrocontroller, WLAN-Modul und Peripherie

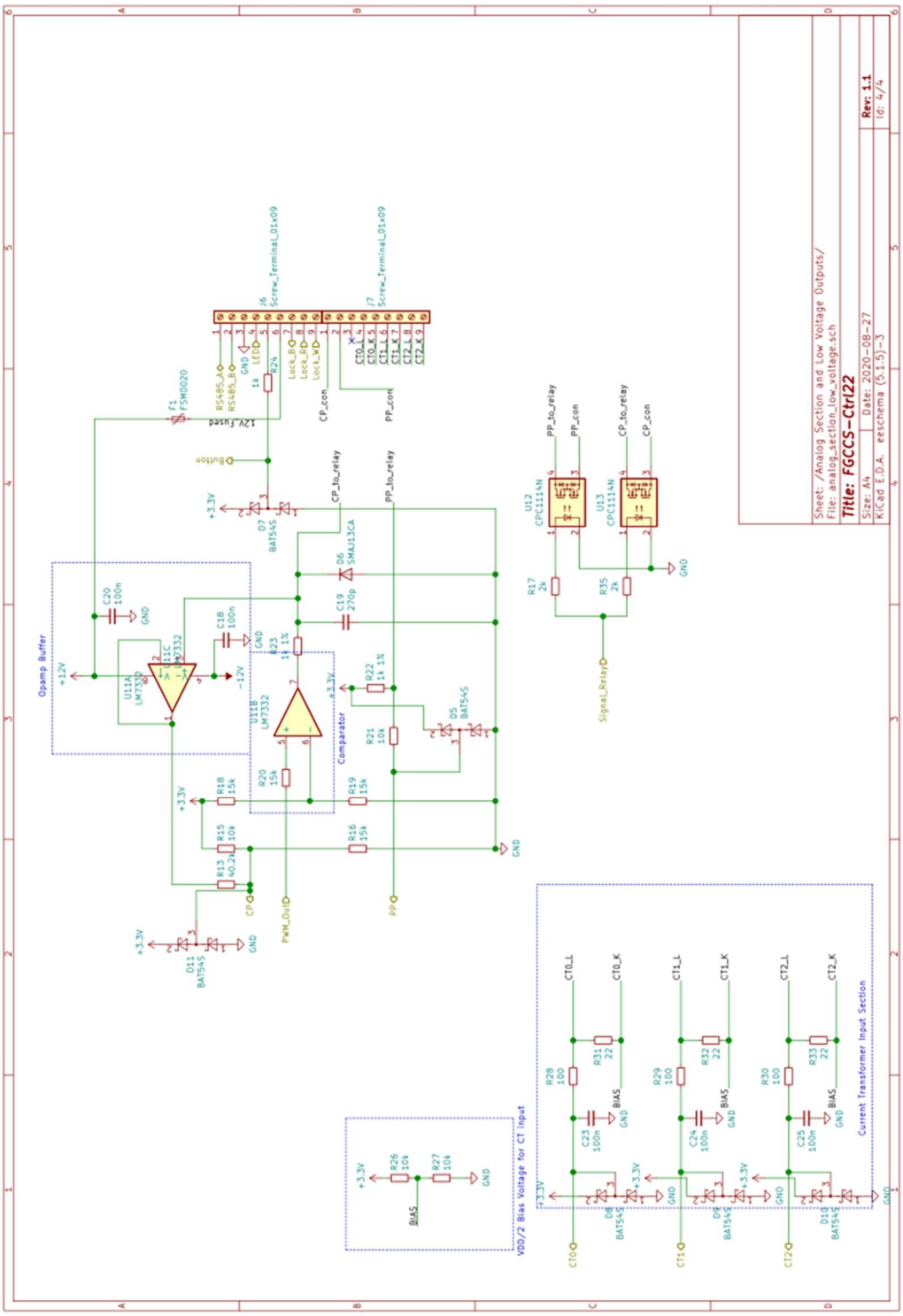

Abbildung 8: FGCCS-Ctrl22-Schaltplan, Seite 3 – Analogschaltung und Signalterminals

Anhang B – Materialstückliste

Stückzahl	Herst.- Nr.	Hersteller	Beschreibung
1	ATMEGA4808-XFR	Microchip	8-Bit Mikrocontroller - MCU 20MHz, 48KB, SOIC
1	ESP32-WROOM-32D	Espressif	WLAN-Modul (802.11) SMD
1	LM7332MA/NOPB	Texas Instruments	Dualer Operationsverstärker
1	TC7660SEOA	Microchip	12V Schaltspannungsregler/Inverter
1	SN65HVD72D	Texas Instruments	RS-485-IC-Schnittstelle
1	FAN3214TMX	ON Semiconductor	Zweikanal 4A Gate-Treiber
3	AQH1213AX	Panasonic	Halbleiterrelais - AC 600 V
1	AP5100WG-7	Diodes Incorporated	3.3V Schaltspannungsregler 1.2A Step-Down
2	CPC1114N	IXYS	Halbleiterrelais - Öffner OptoMOS Relay
3	MCT06030C2209FP500	Vishay	22 Ohm, 100mW, 1%, SMD 0603 Widerstand
1	MCT06030C1202FP500	Vishay	12k Ohm, 125mW, 1%, SMD 0603 Widerstand
6	MCT06030C1001FP500	Vishay	1k Ohm, 100mW, 1%, SMD 0603 Widerstand
4	MCT06030C1502FP500	Vishay	15k Ohm, 100mW, 1%, SMD 0603 Widerstand
2	MCT06030C2001FP500	Vishay	2k Ohm, 100mW, 1%, SMD 0603 Widerstand
1	MCT06030C1003FP500	Vishay	100k Ohm, 100mW, 1%, SMD 0603 Widerstand
1	MCT06030C4022FP500	Vishay	40.2k Ohm, 100mW, 1%, SMD 0603 Widerstand
10	MCT06030C1000FP500	Vishay	100 Ohm, 100mW, 1%, SMD 0603 Widerstand
7	MCT06030C1002FP500	Vishay	10k Ohm, 100mW, 1%, SMD 0603 Widerstand
2	MCT06030C3162FP500	Vishay	31.6k Ohm, 100mW, 1%, SMD 0603 Widerstand
1	BSS138	ON Semiconductor	Logik-Mosfet, SOT23
1	61304011121	Wurth Elektronik	Stiftleiste, 2.54mm, 1x40 Pins
3	B72214S0271K551	EPCOS / TDK	Varistor 275VAC 10% 14mm
1	39211250000	Littelfuse	Sicherung Träge 250V 1.25A
1	1812L020PR	Littelfuse	Rückstellende Sicherung, 0.2 A, SMD 1812
1	PMEG4010BEA,135	Nexperia	Gleichrichterdiode 1A
9	BAT54S	Rectron	Duale Schottky-Diode, 0.2A, 30V, SOT23
1	SMAJ13CA	Bourns	ESD-Entstörer/TVS-Diode 13volts 5uA
8	EMK212ABJ106MG-T	Taiyo Yuden	10uF/16V, 20%, SMD 0805 Keramikkondensator
12	EMF212B7104KGHT	Taiyo Yuden	100nF, 10%, SMD 0805 Keramikkondensator
1	VJ0805A271KXJPW1BC	Vishay	270pF/16V, 10%, SMD 0805 Keramikkondensator
1	VJ0805A102KXAAC	Vishay	1nF/50V, 10%, SMD 0805 Keramikkondensator
1	VJ0805Y105KXARW1BC	Vishay	1uF, 10%, SMD 0805 Keramikkondensator
1	VJ0805Y101KXACW1BC	Vishay	1pF/50V, 10%, SMD 0805 Keramikkondensator
1	TLCR107M006RTA	AVX	100uF/6.3V, 20%, SMD 0805 Tantalkondensator
1	B41231A4109M000	EPCOS / TDK	10000uF/16V, 20%, 6mm Raster Alu-Elektrolytkondensator
1	LQH32CN1R0M23L	Murata	1uH, 20%, 800mA, SMD 1210 Festinduktivität
1	IRM-10-12	MEAN WELL	AC/DC-Schaltnetzteil 12V 0.85A 10.2W 85-264V

2	TB005-762-05BE	CUI Devices	Schraubterminal, 7.62mm Raster, 5-polig
2	TB004-508-09BE	CUI Devices	Schraubterminal, 5.08mm Raster, 9-polig
1	61200821621	Wurth Elektronik	Stiftleiste, 2.54mm, 2x4 Pins, verpolungssicher
1	61200823021	Wurth Elektronik	IDC-Stecker, 2.54mm, 2x4 Pins, verpolungssicher

Tabelle 6: Materialstückliste