

Charging Station for ISO / IEC 15118 Protocol

*Building working smart networked charging station with support for Both ISO 15118 and IEC 61851*

Bachelors Project

**presented by**

Jiztom Kavalakkatt Francis

Nivas Gokul Manimurugesan

Raphael Scholz

**supervisor**Prof. Dr.-Ing. Ansgar Meroth, Heilbronn University

Heilbronn University

Faculty of Mechanics and Electronics

Automotive Systems Engineering

# Contents

[Contents III](#_Toc476173417)

[Directory of Tables V](#_Toc476173418)

[List of Figures V](#_Toc476173419)

[1 Introduction 1](#_Toc476173420)

[1.1 Short version 1](#_Toc476173421)

[1.2 task 2](#_Toc476173422)

[1.3 Chapter Overview 3](#_Toc476173423)

[2 State of the art 4](#_Toc476173424)

[2.1 IEC 62196: vehicle plug 4](#_Toc476173425)

[2.2 IEC 61851: Charging 6](#_Toc476173426)

[2.3 ISO / IEC 15118 8th](#_Toc476173427)

[2.3.1 Open V2G project 9](#_Toc476173428)

[2.4 Powerline Communication 10](#_Toc476173429)

[2.5 state machine 11](#_Toc476173430)

[3 hardware 13](#_Toc476173431)

[3.1 Requirements / Features 13](#_Toc476173432)

[3.2 EVAchargeSE 14](#_Toc476173433)

[3.2.1 construction 15](#_Toc476173434)

[3.2.2 Create serial\_Programming directory 16](#_Toc476173435)

[3.3 Communication between computer and EVAchargeSE 18](#_Toc476173436)

[3.3.1 Flashing a program 18](#_Toc476173437)

[3.3.2 Compile and run a program 20](#_Toc476173438)

[3.4 Problems encountered and measures for troubleshooting 22](#_Toc476173439)

[3.5 Directories used commands 23](#_Toc476173440)

[4 software 25](#_Toc476173441)

[4.1 Communication within the Boards 27](#_Toc476173442)

[4.2 IP-based communication structure between EV and EVSE 28](#_Toc476173443)

[4.3 Creating an ISO / IEC 15118 Stacks 30](#_Toc476173444)

[4.3.1 Extension to a state machine 32](#_Toc476173445)

[5 Results 34](#_Toc476173446)

[6 Summary & Outlook 37](#_Toc476173447)

[7 attachment 38](#_Toc476173448)

[7.1 AC charging procedure 38](#_Toc476173449)

[7.2 DC charging procedure 40](#_Toc476173450)

[7.3 Flowcharts the charging station 42](#_Toc476173451)

[7.4 Flowcharts of the electric vehicle 46](#_Toc476173452)

[8th bibliography 48](#_Toc476173453)

# Directory of Tables

Table 2.1: Contact names 5

Table 2.2: states CP-PE (Wiki\_Stecker, 2016) 7

Table 3.1: Description of the parameters contained in the frames 16

Table 3.2: Description of Services for communication between i.MX28 and KL02 17

Table 3.3: I2SE commands in Terminal 23

Table 3.4: Linux terminal commands (shell commands, 2016) 24

# List of Figures

[Figure 2.1: charging plug systems (Mültin, 2014) 5](#_Toc454570580)

[Figure 2.2: Voltage CP-PE (Lewandowski, 2014) 7](#_Toc454570581)

[Figure 2.3: Assignment of the communication content to the different charging status 8th](#_Toc454570582)

[Figure 2.4: Program flow inside the V2A codes (OpenV2G, 2016) 9](#_Toc454570583)

[Figure 2.5: modulated PWM signal 10](#_Toc454570584)

[Figure 2.6: Example of a state machine (Bustamante, 2014) 11](#_Toc454570585)

[Figure 2.7: Implementation of a state machine 12](#_Toc454570586)

[Figure 3.1: EVAchargeSE board (I2SE, 2016) 14](#_Toc454570587)

[Figure 3.2: Connection of data lines 15](#_Toc454570588)

[Figure 3.3: Message frames between i.MX and KL02 (I2SE, 2016) 16](#_Toc454570589)

[Figure 3.4: FileZilla 19](#_Toc454570590)

[Figure 3.5: Successful SSH connection 20](#_Toc454570591)

[Figure 3.6: Compiling and running a program 21](#_Toc454570592)

[Figure 3.7: Display of another power line communication system 22](#_Toc454570593)

[Figure 4.1: Flow of software programming 25](#_Toc454570594)

[Figure 4.2: Typical socket communication (Large, 2015) 28](#_Toc454570595)

[Figure 4.3: IP addresses of EVAchargeSE boards 30](#_Toc454570596)

[Figure 4.4: State Machine EVSE Part 1 of 3 32](#_Toc454570597)

[Figure 4.5: State Machine EVSE Part 2 of 3 32](#_Toc454570598)

[Figure 4.6: State Machine EVSE Part 3 of 3 32](#_Toc454570599)

[Figure 5.1: Hardware configuration 33](#_Toc454570600)

[Figure 5.2: Display of Linux terminals to start serial\_Programming 33](#_Toc454570601)

[Figure 5.3: Level of Control pin for charge status A 34](#_Toc454570602)

[Figure 5.4: Level of Control pin for charge status B 34](#_Toc454570603)

[Figure 5.5: Level of Control pin for charge status C 35](#_Toc454570604)

[Figure 7.1: News Summary for AC charging cycle (1 of 2) 37](#_Toc454570605)

[Figure 7.2: News Summary for AC charging cycle (2 of 2) 38](#_Toc454570606)

[Figure 7.3: News Summary for DC charging cycle (1 of 2) 39](#_Toc454570607)

[Figure 7.4: News Summary for DC charging cycle (2 of 2) 40](#_Toc454570608)

[Figure 7.5: Program flow chart of charging stations Page 41](#_Toc454570609)

[Figure 7.6: Program flow chart of the function call State B1 Communication 42](file:///C:\Users\Melanie\Desktop\HSHN\_Projektarbeit\abgabedok\Dokumentation.docx#_Toc454570610)

[Figure 7.7: Program flow chart of the function call State C Communications 43](file:///C:\Users\Melanie\Desktop\HSHN\_Projektarbeit\abgabedok\Dokumentation.docx#_Toc454570611)

[Figure 7.8: Program flow chart of the function call State B2 Communications 44](#_Toc454570612)

[Figure 7.9: Program flow chart of the electric vehicle (Part 1) 45](file:///C:\Users\Melanie\Desktop\HSHN\_Projektarbeit\abgabedok\Dokumentation.docx#_Toc454570613)

[Figure 7.10: flow chart of the electric vehicle (part 2) 46](file:///C:\Users\Melanie\Desktop\HSHN\_Projektarbeit\abgabedok\Dokumentation.docx#_Toc454570614)

# Introduction

In this chapter a short summary of this work is described. In addition, the task description as well as to overview Of the following chapters and Their contents are given.

## Short version

The present project work serves to set up a working model for the current standardization of the vehicle-column communication gemäß to ISO / IEC 15118 alongwith the HMI display.

The built-up working model besteht of two interconnected boards and a Raspberry Pi for teh display. Each communication user is Represented by a circuit board.

The software used is derived from at existing stack and angepasst for a defined application of ISO / IEC 115118th

.According To the ISO variable parameters are listed as macros and Documented, so did a change is possible at any time. If parameters are selected so dass die Resulting requirements can no longer be met by the respectivement other subscriber, a fault message is output.

## task

The project is expected to result in a structure did demons trates the communication staple of ISO / IEC 15118th

For this purpose on existing stack for the ISO is put into operation. The freely available stack is written so far for demonstration purposes of the software. For use at a demonstrator, it must be separated, reordered and modularized so did a stack for the vehicle as well as the load side column is created. The INITIALLY defined load parameters are defined as macros, so did a change is possible at any time. Further More, incorrect error messages are to be output in the event of incorrect inputs or not supporting functions of the charging column or the vehicle.

At the sametime, a development environment to be used for demonstration purposes is to be established. First of all, hardware requirements must be defined. From the Resulting specifications, Approaches for implementation have to be found. The Resulting results are Evaluated and selected gemäß to the Previously defined viewpoints.

As soon as the selected hardware is ready for surgery, it must be commissioned and all functions required for commissioning must be checked.

Finally, the functional stack is flashed to the development environment and tested for functions on.

The Resulting project results are defined as follows:

* Integration into vehicle charging columns Communication gemäß to ISO / IEC 15118
* Separation of an existing stack in the vehicle and loading column side
* Commissioning and demonstration of the separate stack
* Selection and commissioning of a hardware suitable for demonstration
* Start up the demonstrator with the Previously created communication pack
* documentation of the results

## Chapter Overview

In Chapter 1 be shown a summary of this work and the task and this Chapter overview.

*chapter* 2provides an overview of previous work at the university as well as documents that have contributed to the incorporation into the subject. Further used components of hardware and software are described.

An overview of the hardware requirements and the commissioning is in Chapter 3to find. Furthermore, an overview of the work with the Linux terminal is given. This is important for the communication with the board, particularly in the field of data management and for compiling and starting a program.

In Chapter 4the various parts of the created and edited the software will be described. This includes newly created files for communication within the board and program components accepted for communication in accordance with ISO / IEC 15118th Newly created components contain features which 61851 are important in particular for the signal generation according to IEC. For data exchange, the communication participant according to ISO / IEC 15118 also a connection is necessary, which is brought also shown in this chapter.

The results achieved by this work are in Chapter 5reproduced. Last is a summary of the work and an outlook of the project in Chapter6 reproduced.

Parts of ISO / IEC 15118, together with program schedules in Chapter 7shown. In both cases, this is done in order to obtain an overview of the requirements according to DIN / ISO and the resulting programs.

# State of the art

This chapter provides an overview of work and information to which the project is worked up. These include, inter alia, the former way of loading a vehicle as well as the previous exchange of information, different vehicle used connector for loading of electronic vehicles. Further information, for this work are a study work, which is to describe the 15118 ISO accurate and a dissertation of Dr. Marc Mültin which is engaged in the electric vehicle as a "flexible consumers and energy storage in the smart home".

## IEC 62196: vehicle plug

Connector types and Lademodis of electric vehicles are defined by the International Electrotechnical Commission in IEC 62196 (Wiki\_Stecker, 2016),

The second part of the standard was published in 2011 and includes different types of connectors. This includes three of the most popular at this time charging plug.

The Type 1 charging plug, which inFigure 2.1is shown, takes its specification of the SAE J1772. This was first published in 1996 by the Society of Automotive Engineers and has since been expanded and maintained by this. The disadvantage of this connector type is found in the contacts, since these do not allow a three-phase charging with alternating current.

Type 2 of the standard charging plug is the currently the most built-up type of charging plug systems and also in Figure 2.1to find. The plug finds its origins through a collaboration of the connector manufacturer Mennekes with the power company RWE and the carmaker Daimler. The naming of the Mennekes plug thus receives this by its manufacturer.

The third recorded in the standard connector type, the EV Plug Alliance, was created by a consortium under the leadership of French and Italian companies. Due to the low demand, the additional production of the plug has been set.

For all defined types of connectors as defined in Type 1 Signal contacts CP (Control Pilot) and PP are (Proximity pilot) included. Which allow charging to IEC 61,851th

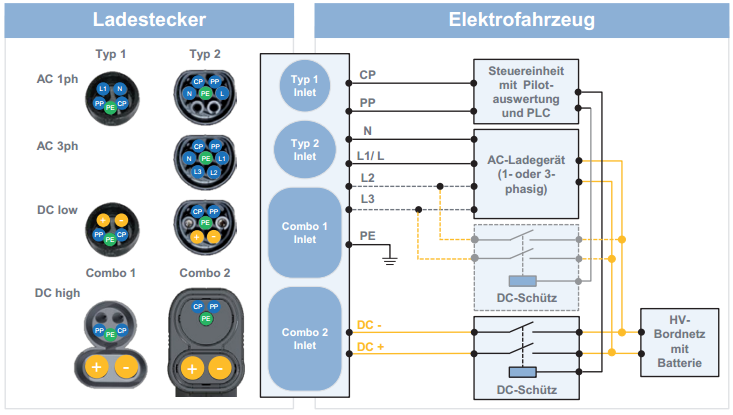


Illustration 2,1: Charging plug systems(Mültin, 2014)

In the Figure 2.1 Signal contacts shown are defined as follows:

|  |  |  |
| --- | --- | --- |
| abbreviation | Contact | function |
| CP | Control pilot | Control signals charging station🡪electric vehicle |
| PP | Proximity pilot | Check the presence of a charging cable |
| N | Neutral | For AC charging |
| L1, L2, L3 | Current-carrying phases | For AC charging with a (L1 / L) or three (L1, L2, L3) Phases |
| PE | Protective Earth | protective conductor |
| DC +/- | Current-carrying phases | For DC charging |

table 2,1: Contact names

## IEC 61851: Charging

Prior to the definition of a charging process according to ISO / IEC 15118 necessary charge parameters have been defined by means of a PWM signal in accordance with IEC 61851 for the charging process. To determine the required parameters for charging the signals of the in chapter2.1 described contacts Control pilot (CP), Protective Earth (PE) and proximity pin (PP) is required.

To charge the vehicle initially both communication participants are connected. Then, a 1kHz signal is generated with 12V on the CP-contact from the side of the loading column. The pulse width of the signal indicates which maximum power can be provided by the charging station available. It applies to 10% max. 10A, 25% 16A 50% max. 32A and 90% quick charge(Wiki\_Stecker, 2016),

Resistances between CP and PE or PP and PE are the vehicle side switched in the further course. By correspondingly different switchable level of tension between the CP and PP ContactFigure 2.2different charge states are thereby indicated. It should be noted that the negative voltage value is permanently -12V, and change only the positive values. is a definition of each statesTable 2.2,

Last specifies the maximum charging current of the electric vehicle is a vehicle-side resistance between the PP and the PE contact. Where the greater the resistance used is selected, the lower is the maximum charging current. Concretely this means for a 1,5kΩ- resistance a maximum charging current of 13A, 20A maximum at a resistance of 680Ω, 220Ω at the maximum charging current 32A and 63A at 100Ω.

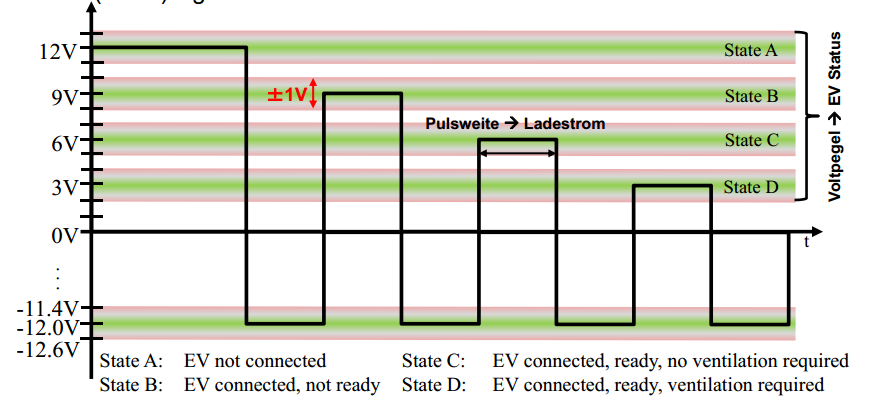


Illustration 2,2: Voltage CP-PE(Lewandowski, 2014)

|  |  |  |
| --- | --- | --- |
| level | State | condition Description |
| 12 ± 1 V | State A | Electric vehicle is not connected |
| 9 ± 1 V | State B | connected electric vehicle, not charging Ready |
| 6 ± 1 V | State C | connected electric vehicle, ready to charge |
| 3 ± 1 V | State D | connected electric vehicle, ready for loading, ventilation needed |
| 0 ± 1 V | State e | Network problem, PP Short to earth |
| -12V | State F | Vehicle unavailable Error |

table 2,2: States CP-PE(Wiki\_Stecker, 2016)

## ISO / IEC 15118

The International Organization for Standardization (ISO) and the International Electronic Commission (IEC) in 2009 started to describe the standardization of a "digital IP-based communication protocol" between electric vehicle and charging station(Mültin, 2014), This should be a "plug-and-charge" mechanism for authentication, authorization, accounting, and for load control, so that needed to load enable parameters are stored in the vehicle and the user both communication parties must connect only. The individual communication Content will be the level of tension control pin signal from Chapter2.2 correspondingly Figure 2.3 assigned.

Illustration 2,3: Assignment of the communication content to the different charging status

The full schedule of communication stacks for AC or DC charging an electric vehicle according to ISO / IEC 15118 can be found in Figure 7.1 to Figure 7.4, An overview of the variables contained in the messages within the AC communication stack is described together with an overview of the ISO / IEC 15118 in a previous study, work(Barth, 2015),

### Open V2G project

An already far ahead paced example for implementing a communication stack to ISO / IEC 15118 was initiated already by the support of Siemens Corporate Technology as an open source project (OpenV2G, 2016), Both Ladesäulen- and side of the vehicle are shown in a program code, and according to the newsFigure 2.4successively generated, tested and the next message is generated. On the current status (version 0.9.3) can be from the sequence of the individual requests and responses, and the Message Content AC and DC charging derive very good. It is one of the objectives of the present work, this code split so that each communication participant present a program.

****

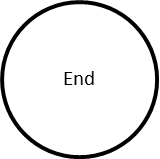
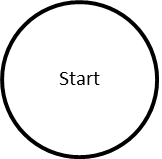
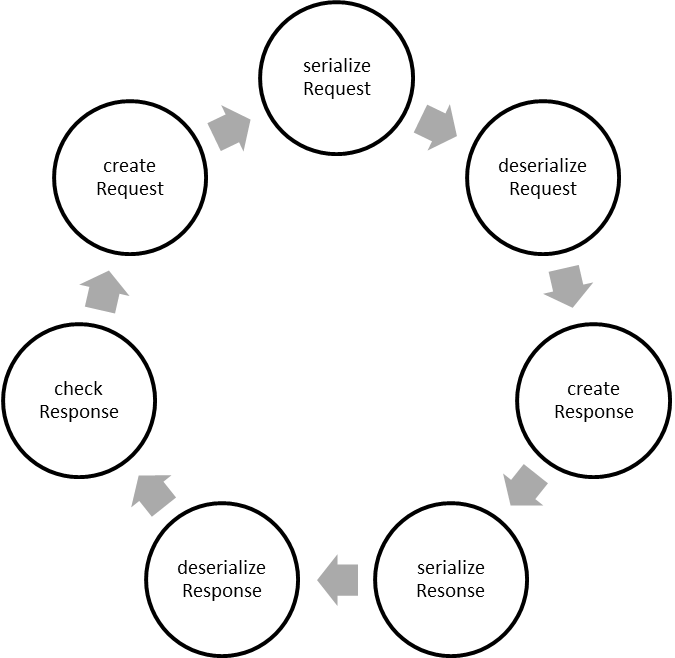
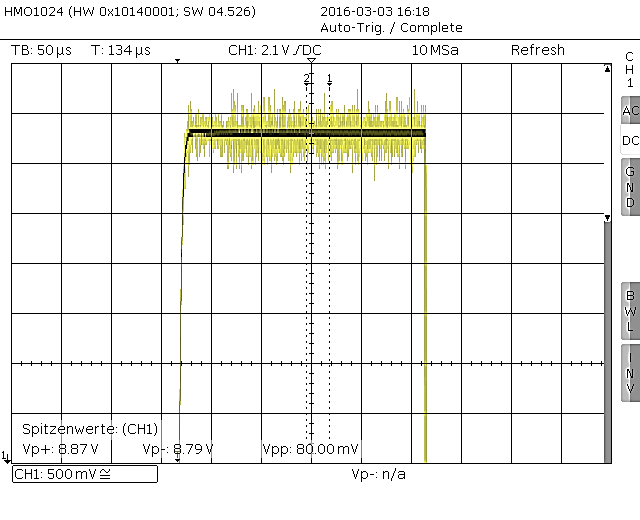


Illustration 2,4: Program flow inside the V2A codes (OpenV2G, 2016)

## Powerline Communication

General power line communications (PLC) is described as "transmission of data over the power cord" (El-Ko, 2016), It this type of signal transmission is used for some time, for example in Intercom systems. By using this communication variant no more pins and wires must be defined and retrofitted into the existing charging plugs and cables.

In the power line communication to be transmitted data is modulated at RF frequencies on the power cord and demodulates the receiver side. In the case of the ISO / IEC 15118, the modulation and demodulation of the communication protocol is carried out according toFigure 2.5 on the PWM signal of the CP-conductor.



**Illustration 2,5: Modulated PWM signal**

## state machine

A state machine consists of states, state transitions and actions. These tools allow the control of a system is to be implemented, which takes into account events of the past, the current time and the future. Each state are assigned to actions that are executed when leaving or whose. It must be defined at any point during the term of the system state. A state transition, however, describes the links between the various states to one another as well as the event which occurrence has to switch between states.

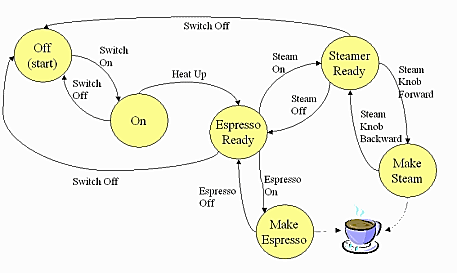


Illustration 2,6: Example of a state machine (Bustamante, 2014)

To illustrate such a state machine, a coffee machine is used in accordance withFigure 2.6, In this case, the state machine begins with the start state, which is the off state of the coffee machine in the present example. Here is a state change only by the transition "Switch on" possible. Depending on the user's input, the machine may switch to "Espresso ready", "Steamer ready" or "OFF" move.

Particularly, this example shows the involvement of different tenses. had to be ready the coffee machine be switched in advance and brought to a defined temperature. Which state is assumed in the further course depends on unforeseeable events. It is also clearly shown that the machine can not activate the individual states at will at any time. To assume a certain state, this must be a state transition from the current state of continuously connected to the. Thus, no coffee can be dispensed as long as the coffee machine in the "ON" state in the given example.

The programming of a state machine can be implemented with the switch case function. An initial state is defined in advance. Once an event that could cause a change of state occurs, the function is started. Here, the currently defined state is queried and defined in the relevant case the state change in order to reach a new state can. An example to the coffee machineFigure 2.6 shows Figure 2.7,

switch (current\_state)

case OFF: if (switch\_OFF) current\_state = SWITCH\_ON; break;

case ON: if (Heat\_up) current\_state = Espresso\_ready; break;

case Espresso\_ready: if (Steam\_on) current\_state = Steamer\_ready;

else if (Espresso\_on) current\_state = Make Espresso;

break;

...

...

Illustration 2,7: Implementation of a state machine

# hardware

One component of this work is to develop a demonstrator. For hardware, a board is for this reason each vehicle and charging stations side needed which generate a signal in accordance with IEC 61851 and at the same time build a powerline based communication.

## Requirements / Features

thus a total of two boards are required for the realization of the demonstrator. The aim is one for the construction of a Ladesäulendemonstrators and the second used for the electric vehicle. Both boards will need to demonstrate that in Chapter2.3 pins required for communication processing according to ISO / IEC 15,118th Furthermore must lines for controlling the power supply as well as the response of the locking of the charging connector to prevent withdrawal during the charging process, be present.

For communication in accordance with IEC 61851 must charging posts side there are the possibility to generate a PWM signal and to change the positive level of the latter. Furthermore, this communication node must be able to measure the level of the other pins. The side of the vehicle required to determine the current level of the positive voltage and the pulse width of this to the maximum charging current to determine. Further to resistances between the CP, PP and PE contacts can be switched to the feedback.

An important feature of this project is also the possibility of powerline communications. This must be supported by Ladesäulen- and electric vehicle side to 15118 modulate messages in accordance with ISO / IEC on the signal of the control pins.

## EVAchargeSE

The Board of EVAchargeSE I2SE was developed for communication in accordance with ISO / IEC 15118th The advantage to the board is that it can be used for both vehicle and charging stations other. This two identical boards can be used with different configurations for the demonstration building.

On the board a i.MX28 is installed with a Linux operating system. The two needed for communication GPIO's CP, PP and PE are brought out to a connector. Programs for communication stack can be transferred using the existing Ethernet interface. These are stored in the 4GB Memory and started from there.

A contained on the board KL02 microcontroller can be addressed by the i.MX28 via UART. In the master / slave connection of KL02 is addressed as a slave and is 61851 responsible for generating and measuring the signals in accordance with IEC. Furthermore, it is used to GPIO's to address, which are to start a locking mechanism of the charging cable.

Furthermore, the Board is EVAchargeSE PLC-capable. The messages to be sent via TCP / IP sockets are shipped. A QCA chip 7000 modulates the data sent in the further course of the PWM signal of KL02.

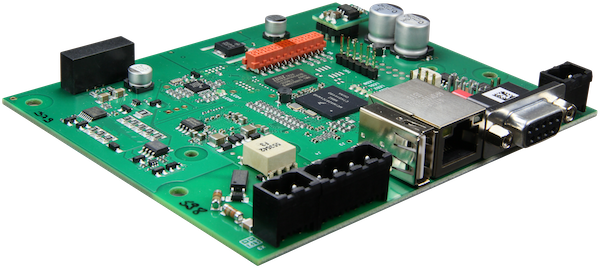


Illustration3,1: EVAchargeSE board(I2SE, 2016)

### construction

To start up the boards, they are first built in the laboratory. The supply voltage amounts to 12V at 0.3A. For communication between the boards EVAchargeSE the terminals of CP, PP and PE are correspondinglyFigure 3.2connected with each other. Further, the structure shows an Ethernet connection to a router and a computer with Linux operating system. This is an access to the Linux systems that are located on the boards, allows. At this time, this access useful because this allows programming in an environment like Eclipse on a computer. The programs are initially started using Eclipse on the computer to check the runnability. Later in the programs for vehicle and charging station side are flashed on the respective boards, compiled and started from there. Detailed instructions for this is in Chapter3.3 to find.

router

EV

EVSE

Linux

Eth

Eth

Eth

PE CP

PP CP

CP CP

CP

PP CP

PE CP

Illustration 3,2: Connecting the information lines

### Create serial\_Programming directory

The serial\_Programming- directory is responsible for communication between the i.MX 28 and the KL02, which is responsible inter alia for the generation of the PWM signal. Thus, the settings of the microcontroller are important in distinguishing between electric vehicle and charging station. As noted in Chapter3.2mentioned, the communication between the two microcontrollers via UART. Here, the i.MX 28 is declared as the master and the KL02 as a slave. This means that the KL02 waits for a command prompt of the ARM processor, and in this reply. The contents of the transmitted messages are divided into frames. This has the advantage of low data to be transmitted crowd with it. A frame is in this case in accordanceFigure 3.3both vehicle and charging station side together. The parameters contained therein are inTable 3.1 more precisely defined.

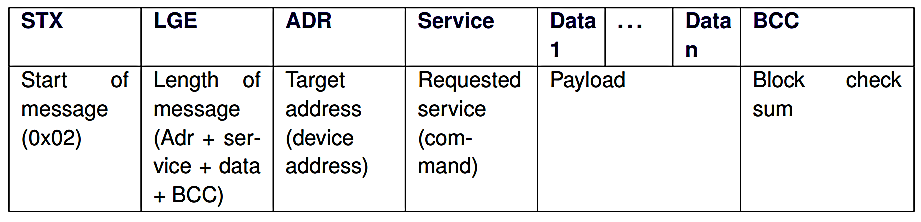


Illustration 3,3: Message frames between i.MX and KL02 (I2SE, 2016)

|  |  |
| --- | --- |
| Designation in the frame | description |
| STX | Indicates the start of a new frame |
| LGE | Specifies the number of bytes contained in ADR, Service, Data and BBC |
| ADR | Number of the target device. |
| service | Describes the service to be carried out in accordance Table 3.2 |
| Data 1 ... n | Contains the required to implement the Services parameter and returns Requested values |
| BCC | Checksum: To avoid transmission errors.  Is calculated by setting all bytes contained in the frame are interconnected by a logical XOR. |

table 3,1: Description of the parameters contained in the frames

|  |  |
| --- | --- |
| Service ID | service Description |
| 0x01 | Test the connection. Has return value software and hardware version of EVAchargeSE Boards |
| 0x04 | Another quick test. Does the return value, among other things the last reason a reset. A list of possible reasons can be found in the datasheet. |
| 0x10 | Measures the frequency of the PWM signal |
| 0x11 | Changes the frequency of the PWM signal |
| 0x12 | Defines whether a PWM signal is generated.  This setting is important for distinguishing between electric vehicle and charging station, since the signal may only be charging stations side generated. |
| 0x14 | Measures both positive and negative level of CP-Pins |
| 0x15 | Changes the positive level of the CP signal. This service may only be called charging stations side, since this parameter the vehicle side is important for determining the current level. |
| 0x17 / 0x18 | Speaks pins on the control of motors for charging cable lock |
| 0x1A | Indicates the current status of the motors for charging cable lock |
| 0x20 | Enables an automatic measurement and return of the PWM signal |
| 0x31 | Ensures a break in the connection |
| 0x33 | Initiates a reboot of KL02 |
| 0x50 | Defines the value of the resistor between PP and PE |
| 0x51 | Enables the resistance between PP and PE. This service may only be the vehicle side is activated. Characterized the maximum charging current of the electric vehicle is defined. |
| 0x52 | Measures the tension between PP and PE |

table 3,2Description of Services for communication between i.MX28 and KL02

## Communication between computer and EVAchargeSE

On the board a Debian operating system of the manufacturer of the boards I2SE installed. This "uses the Linux operating system kernel, but the most basic OS tools come from the GNU project"(Debian, 2016), In order to facilitate a communication structure is used for working with the EVAchargeSE only the Linux system Ubuntu. Through the terminal, the user is after a short training period allows fast access, exchange information and data on other subscribers connected the router.

### Flashing a program

Before a program can be started, it must be on the microcontroller, on which it is to be compiled and carried out at a later stage, be flashed.

For programs with few files could be developing the Secure File Transfer Protocol (SFTP). This protocol runs on the basis of Secure Shell and establishes a secure connection to transfer data. Where applicable, data can use SFTP to EVAchargeSE Boards transmitted are received by using a Linux terminal.

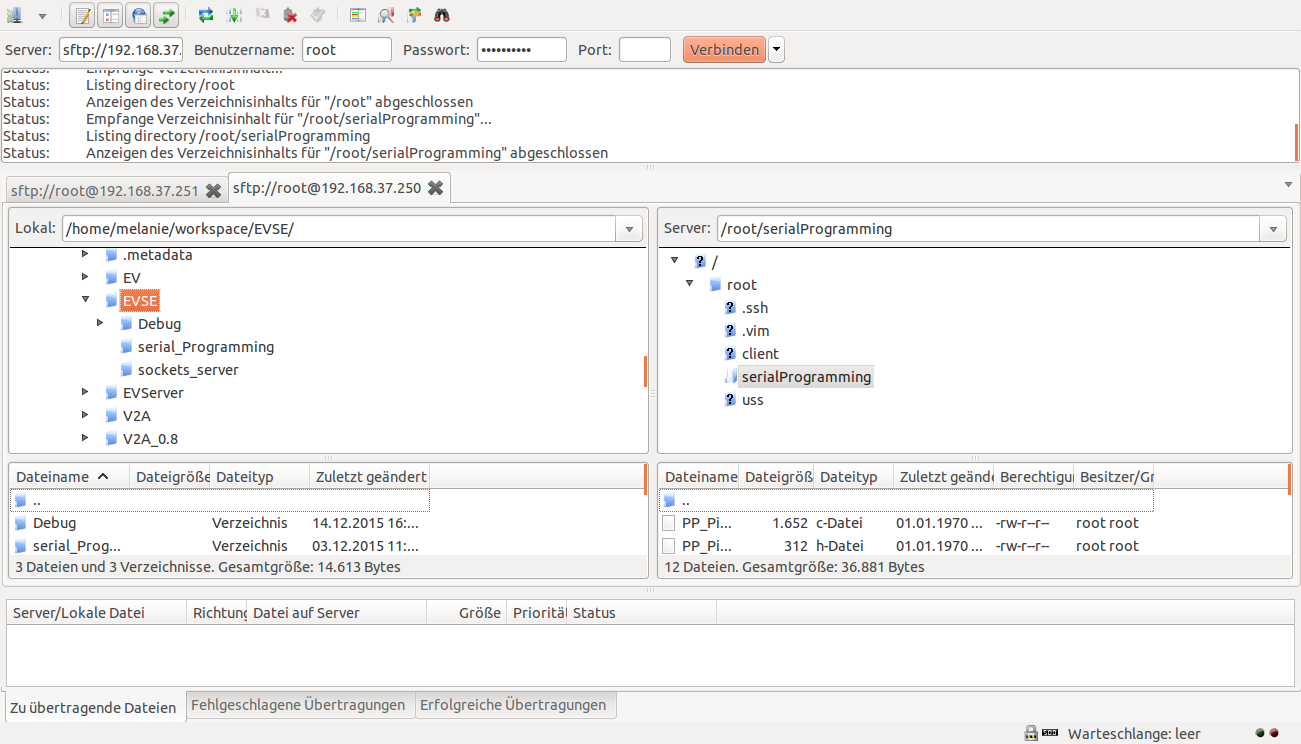
To start an SFTP connection the command is "sftp <username> @ <IP address>" used. If a subscriber with the identification data accessing a password entry is required. Once this is properly initiated in the terminal commands are executed by the user of the device to the specified IP address.

Thus Example White can through the "cd" command the current directory to be changed on the board. If, however, be called to another directory on the Linux system of the computer, the command is provided with an exclamation point in front of the command ( "! cd").

Use the command "put <filename>" can be copied from the Linux machine to the board files. A copy of an existing on-board file can be written on the computer directory by the command set "get <filename>".

Due to the increasing number of files, such a procedure for transmitting the individual files is very time consuming. For this reason, data is transferred in the course of the program using FileZilla. The FTP client enables easy and clear transmission of multiple data files. At the same time the user is presented by the graphical user interface, the ability to obtain an overview of the existing folder structures and directories. This allows local files as well as the related communication partner are found and transferred from various folders in a short time.

To transfer data, the program is started and the IP address, username, password, and port of EVAchargeSE Boards accordingly Figure 3.4entered in the appropriate fields. Once a connection is made such for the computer connected to the Board appears next to the folder structure on the local machine. By dragging and dropping files can be exchanged in both directions of the link(FileZilla, 2016),



IP address

192.168.37.250

192.168.37.251

User name

root

Password:

zebematado

port

22

Tabs for multiple connections

Local folders

EVAchargeSE

Illustration 3,4: FileZilla

### Compile and run a program

To compile and run a program initially a link between local computer and each development board has to be prepared. In order to locally can make a remote command line available, a Secure Shell (SSH) is used. This is a network protocol that establishes an encrypted network connection(Wiki\_SSH, 2016),

The function call is to invoke such a compound similar to that in Chapter 3.3, First, the Secure Shell is built with the call "ssh <username> @ <IP address>" in the Linux terminal. For backup purposes here in the next step, a password prompt.

Here identical usernames are both Board (root) and password (zebematado) used. The However, IP addresses must have differences (192.168.37.250 and 192.168.37.251). The password is correct the current date and the last logged in is correspondinglyFigure 3.5displayed. The Linux Terminal, was where this call is made now represents a command line of EVAchargeSE boards.

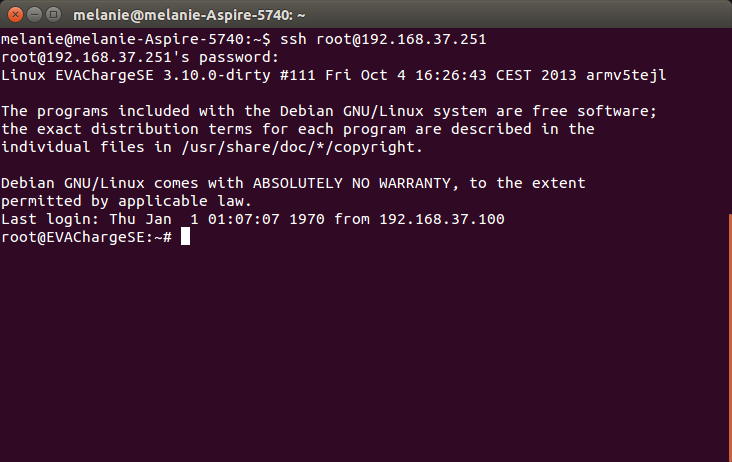


Illustration 3,5: Successful SSH connection

To compile the previously transmitted C-code must first be coordinated in the appropriate directory. Overview of the current directory path and navigation in other directories, the commands "pwd" and "cd" are correspondinglyTable 3.4used. In order for the Board EVAchargeSE compiled later in the present in the directory source files, the command "gcc -o <NAME> \* .c" is called. The method used here variable <NAME> can be named anything, and includes the start-up file. To start the Compiled codes now the previously defined boot file is called by "./ <NAME>". A Linux terminal, which compiled in the serial\_Programming this directory and Starts inFigure 3.6 to see.

If an executable file, this can be stopped by using the key combination "Ctrl" and "C" are performed. can When the work is carried out in the terminal of the Development Boards the SSH connection by "exit" to close.

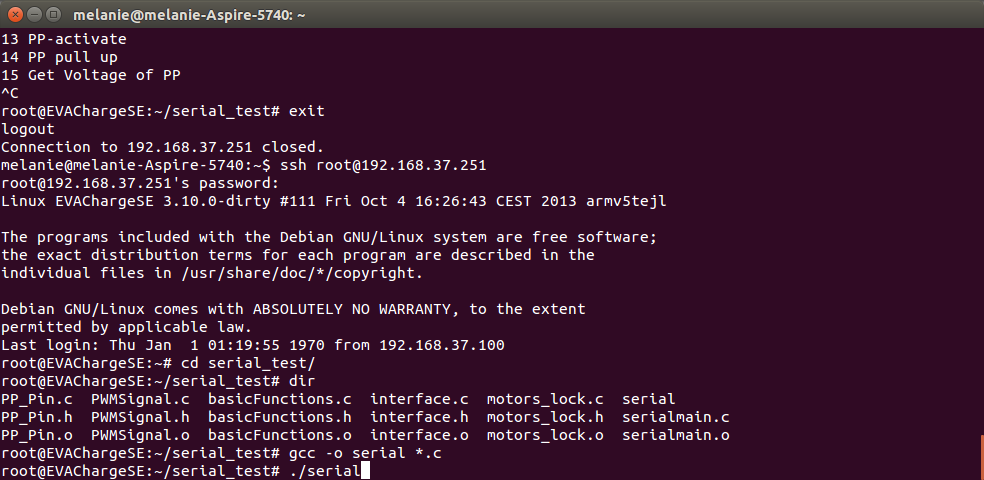


Illustration 3,6Compile and run a program

## Problems encountered and measures for troubleshooting

During commissioning of the hardware used, problems have arisen that can not be described in the data sheet of the boards. Follow up and error prevention, the most important are described in subsequent sections.

In the first step to start up the boards a connection between the PC and a board is established. The structure for this will be in accordanceFigure 3.2realized with only one board. accordingly When attempting other participants connected to the router using the "ping" commandTable 3.4to address, the Linux computer does not receive a response from the board. The reason for this can be found in the IP router and EVAchargeSE-boards. Since the configuration of the router provides an IP address in a different subnet. By changing the IP address of the router to the network participants can recognize and the "ping" command of the computer receives a response from the board.

Another problem can be found in an undefined Response Service serial communication. If an error occurs in the transmission of frames from master to slave this is a response back to not Documented response service 0x99. It can be detected in the data block 0x43 for an error of checksum or 0x44 for an unknown error as cause of the error.

To communicate the boards together mutual ping is initiated via the powerline communications interface in the further course. Since the two communication participants could not find the Council of I2SE was consulted. By inTable 3.3described command "PLCSTAT -i qca0 -t" was a recognition not possible either. By reading the parameter information block both Boards could be determined that both boards have been initialized with an identical MAC address. A change in the address is also inTable 3.3described and fixes this bug. This communication partner with a renewed call for display of all available Powerline an answer accordinglyFigure 3.7,

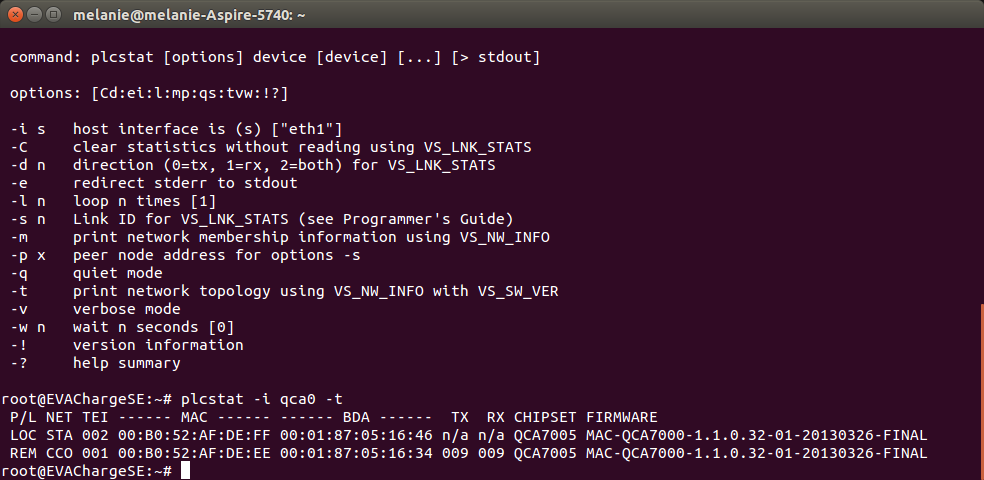


Illustration 3,7: Display of another power line communication system

## Directories used commands

As noted in Chapter 3.2.2above, there are various, defined by I2SE shell commands. These are especially needed for configuration and display of the powerline connection. By communicating with I2SE while editing the project overview was inTable 3.3to be created. It will always be the three main commands "PLCSTAT", "PLCTool" and "modpib" used. The command "PLCSTAT" is responsible for network settings between the two communication partners of the powerline connection. These include settings and display of parameters. "PLCTool" is responsible for the internal settings of the powerline connection. By "modpib", defined parameters are read and changed a pib file.

|  |  |
| --- | --- |
| I2SE commands |  |
| PLCSTAT | PLC Network Settings |
| PLCTool | Local Settings of the PLC communication |
| PLCSTAT -I -t qca0 | Recognition of available Powerline connections |
| PLCTool -i qca0 -p / tmp / local pib | Changing the MAC address |
| modpib -M <MAC> -v / tmp / pib |
| plcwait -R -I qca0 |
| / Usr / local / bin / plcboot -N /root/Mac-QCA7000-v1.1.0-01-X-FINAL.nvm -P / tmp / pib -S |
| /root/NvmSoftloader-QCA7000-v1.1.0-01-FINAL.nvm -I -F qca0 |
| PLCTool -I qca0 -p / tmp / local pib | Show the QCA settings |
| modpib -v / tmp / pib |

table 3,3: I2SE commands in Terminal

To navigate within the Linux terminal commands are in Table 3.4Shell commands most commonly used during processing together. The detailed description of a command can also be issued by "man <command>" from the terminal. More support for programming with Nutshells can be found in the bibliography(Hekman, 1998),

|  |  |
| --- | --- |
| General commands Nutshell |  |
| ssh [root@192.168.37.250](mailto:root@192.168.37.250) | Building a Secure Shell connection (Chapter3.3.2) |
| sftp[root@192.168.37.250](mailto:root@192.168.37.250) | Connect an SFTP protocol (sh. Chapter 3.3) |
| Ifconfig | Configuration and status display of all available network interfaces |
| cd <folder name> | Change Directory: Change to a subdirectory of the current folder |
| cd .. | Go to parent directory |
| cd / home / user | Switch to File deposit / home / user |
| vi <filename> | Open a file in the vi editor. To get out of the editor back into the terminal press "ESC", "q" and confirm with Return. |
| gcc -o <NAME> \* .c | Compiling all source files corresponding to Chapter3.3.2, |
| ./ <NAME> | Launching a program in accordance with Chapter 3.3.2, |
| ping <IP address> | Sends data packets to an IP address in order to check the presence of a compound |
| ping -I qca0 <IP address> | Ping command via network interface qca0 |
| to you | Shows all the current directory of the folders and files. |
| pwd | Is the current folder path |
| rm <file> | Deleting a file |

table 3,4: Linux Terminal Commands(Shell commands, 2016)

# software

As noted in Chapter1.2mentioned it is an object of the project is technically Software to split an existing communication stack to the ISO / IEC 15118 in vehicles and charging stations side into two parts and to implement the EVAchargeSE boards for demonstration run. For this reason, the establishment of a register for generating the PWM signal, and the IP-based communication structure must be considered additionally.

Figure 4.1 shows this approach chosen White during the project work.

while communication between the i.MX 28 and KL02 must first be prepared for the use of the functions of serial\_Programming directory. The contents of the messages which are communicated in this connection, include in Chapter3.2.2 defined.

Subsequently, the modulation of the IP-based messages on the PWM signal is realized. are sockets for the implementation of such a communication method in accordance with Chapter4.1used.

The final work package is divided into vehicle and charging station, as already mentioned, the existing stack. By connecting all the software packages of the communication stack according to ISO / IEC to be realized 15118th In addition to the split stack belongs to generate or read out of the PWM signal and the communication via TCP / IP.

**Illustration 4,1: Expiration of software programming**

## Communication within the Boards

For communication between the i.MX 28 and the KL02 a UART interface is used. Since the i.MX 28 controls the program flow in the further course, it is defined as master. For this reason, the KL02 is already taught as slave programming. This means that it only to requests in accordance with Chapter3.2.2 responds.

To establish the connection to the settings according to the "Board Support Package" document (I2SE, 2016)used. Therein a baud rate of 57600Bd at 8 data and 1 stop bit are defined. Further, the modem device via which the data are to be transferred must be known. When you open the ports, the connection is secured in a file handle, so even with several existing compounds a differentiation can take place. For the initialization of a communication protocol such sources already exist, which are used to change the configurations(Sweet, 2016),

After initialization of the UART interface messages can be transmitted. To send a message, it is written in an array. Sending <write (filehandle, array, number bytes to be transferred)> the array is sent via UART. The return value of the function indicates how many bytes were sent. By <read (filehandle, array, number to be received bytes)> the data to be received is written to the previously defined array. In both cases, must be known, expected or sent as many bytes. In the3.2.2 defined frames are therefore always written in an array, shipped and received again.

## IP-based communication structure between EV and EVSE

To establish a TCP communication widespread LAN interface for network applications, a socket is used. The in3.3.1 and 3.3.2described features SFTP and SSH are applications that use a socket interface. With this protocol, the client-server principle applies, the charging station is assigned to the server side(ITWissen, 2016),

In order to devices that are connected to a network reach, and be able to distinguish, an IP address is assigned to them. Because some protocols can be used by multiple users at the same time simultaneously, port numbers to distinguish the processes used in addition. To establish a connection between the two boards EVAchargeSE the port number used both sides must be known. In addition, the client side needs to know the IP address of the server to be accessed.

the socket functions are accordingly for the construction, communication and the termination of the TCP connection Figure 4.2 used.

The socket () is responsible for the establishment of a communication end point on. This includes parameters defining the address family, the type of transmission and the protocol to use for transmission, specify.

After both sides will have built a socket communication endpoint of the server port number by using the command bind () assigned. This port number must be known to the client, in order to build a future connection.

On the server side, all settings required for communication are known. For this reason, this communication node is ready to listen () for a connection request on the port previously defined.

Once the client tries to establish a connection using the call connect (), the server side must accept this with accept () to enable communication.

To send or receive now the functions send () and recv () are used. As with the UART connection size needs to be transmitted or the data to be received to be known Pact here in both functions. For this reason structs are used in a given case, their content and content sizes have been defined in advance.

Recently must terminate the communication to be terminated the socket connection to. This is done by calling Close (). This is the port used for communication is free and can again with the listen () - wait function to a connection request.

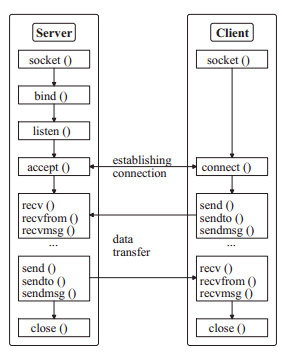


Illustration 4,2Typical socket communication(United, 2015)

## Creating an ISO / IEC 15118 Stacks

To have to build a stack to build a demonstrator not completely new, is built on a program for ISO / IEC 15118 from an open source project(OpenV2G, 2016), This especially the message contents and their data types are described. Further exemplary processes for loading are presented with direct and alternating current.

In previous versions of the communication stack both vehicle and charging stations page are simulated by a program. This brings a large number of files with them, which additionally contain up to 20,000 lines of code. In addition, the final status of the project is to include two C programs, which enables integration into a demonstrator with two communication subscribers. This makes it imperative for each vehicle and charging stations side to implement a program and thus divide the existing stack.

To validate the two programs they need to communicate. For this reason, in the course in Chapter4.2created library for IP-based communications involved. For the socket connection, the IP address 127.0.0.1 is used during initialization of the client. This leads to a local host which send messages to your own computer by this address is allowed. This is a program start Example White Eclipse allows on a computer, without having to make changes to the IP address. A distinction is made using the macro "CODE\_EXAMPLE". If this parameter specifies the value of the defined macro "CODE\_EXAMPLE\_SOFTWARE", the IP address 127.0.0.1 is used. For establishing communication with the server must "CODE\_EXAMPLE\_HARDWARE" be specified. In this project, this is the IP address of the charging station. When communicating the EVAchargeSE boards a powerline communication is being established. therefore not the previous IP address of the Ethernet connection is used for this connection. The board, which has been addressed for connection to a computer with the IP address 192.168.37.250, uses the IP address 192.168.38.32 for powerline communications. The other board will tend to powerline connecting the IP address 192.168.38.33. This means that when flashing the boards must be paid to the IP addresses. If a server program flashed on the board with the IP address 192.168.37.250, has this to powerline communications address 192.168.38.32. It follows that the Board 192.168.37.251 is the client side using the IP address 192.168.37.33. When initializing the socket connection now has on the client side, the IP address of the server, so there indicated 192.168.37.32. serves for a better overview in this regardFigure 4.3,

After the communication stack in accordance with ISO / IEC 15118 is running in the development environment, this also needs to be implemented as hardware in the last step. In order to realize the necessary functions from the serial\_Programming library must be involved. It is important to be careful to distinguish between the vehicle and charging station. These functions are protected by the macro "CODE\_EXAMPLE" activated or deactivated. Such disconnection is the possibility to start the source files in a development platform and to functions that require a return value of other communication partners to bridge.

The resulting program flowchart of the electric vehicle is annexed by Figure 7.9 and Figure 7.10 shown.

Powerline IP

192.168.38.33

**router**

**client**

Powerline IP

192.168.38.32

**server**

Powerline IP

192.168.37.250

Powerline IP

192.168.37.251

Powerline connection for communication in accordance with ISO / IEC 15118

SSH SFTP connection to flash, Compile

and starting programs

Illustration 4,3: IP addresses of EVAchargeSE boards

### Extension to a state machine

As noted in Chapter 2.5describes a state machine between its states changes due to entering events. Des Weitern are previous and future states involved in the implementation with. Moreover, such a machine, such asFigure 2.6shows are clearly displayed. This further brings with it the advantage to identify missing elements of the state machine.

Particularly in the present project, the implementation of such a state machine on the side of the charging station is advantageous. The communication requests are sequentially started by the electric vehicle, as indicated byFigure 7.9 and Figure 7.10you can see. Since the charging station in a large part of the available states have several options of a subsequent state, provides a state machine for reasons of clarity, a good overview of the different branches. Furthermore, the individual states, which can take in this project such a machine, already in the news content inFigure 7.1 to Figure 7.4 to recognize.

A state transition is thereby triggered by a signal received from the electric vehicle message. If this message is composed of information that is not defined in any of the state transitions to the next state in accordance with ISO / IEC 15118, an error condition is assumed and the communication is terminated. Last to be considered that a distinction between AC and DC load is necessary from the time of the first level change in charging status C.

When implementing a function first "handshake" is called. This figure already contains the message "supported app Protocol". After the function has been successfully executed, the variable "next\_state" the first state "Session Setup" preset.

An overview of the state machine is in Figure 4.4. Figure 4.5 and Figure 4.6to see. To start the states inFigure 4.4 and Figure 4.5should the level of the CP signal the charging status B match. For a start the states inFigure 4.5 however, charging status C.

The program flow chart of the charging station and the functions in which the state machine is realized, can be found in the appendix starting with Figure 7.5,



Illustration 4,4: State machine EVSE Part 1 of 3

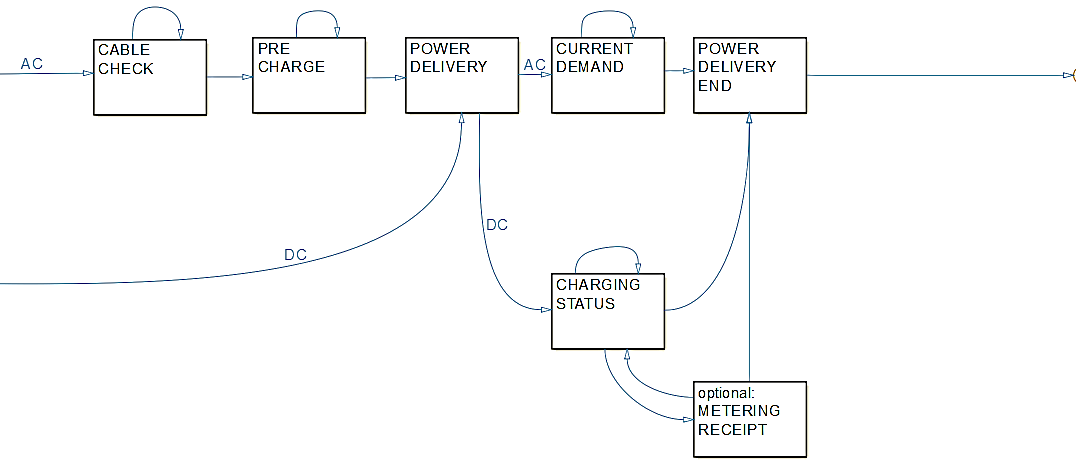


Illustration 4,5: State machine EVSE Part 2 of 3



Illustration 4,6: State machine EVSE Part 3 of 3

# Results

The boards of I2SE were accordingly Figure 5.1 constructed and the pins which correspond to those of the charging connector connected to the twisted pair links.

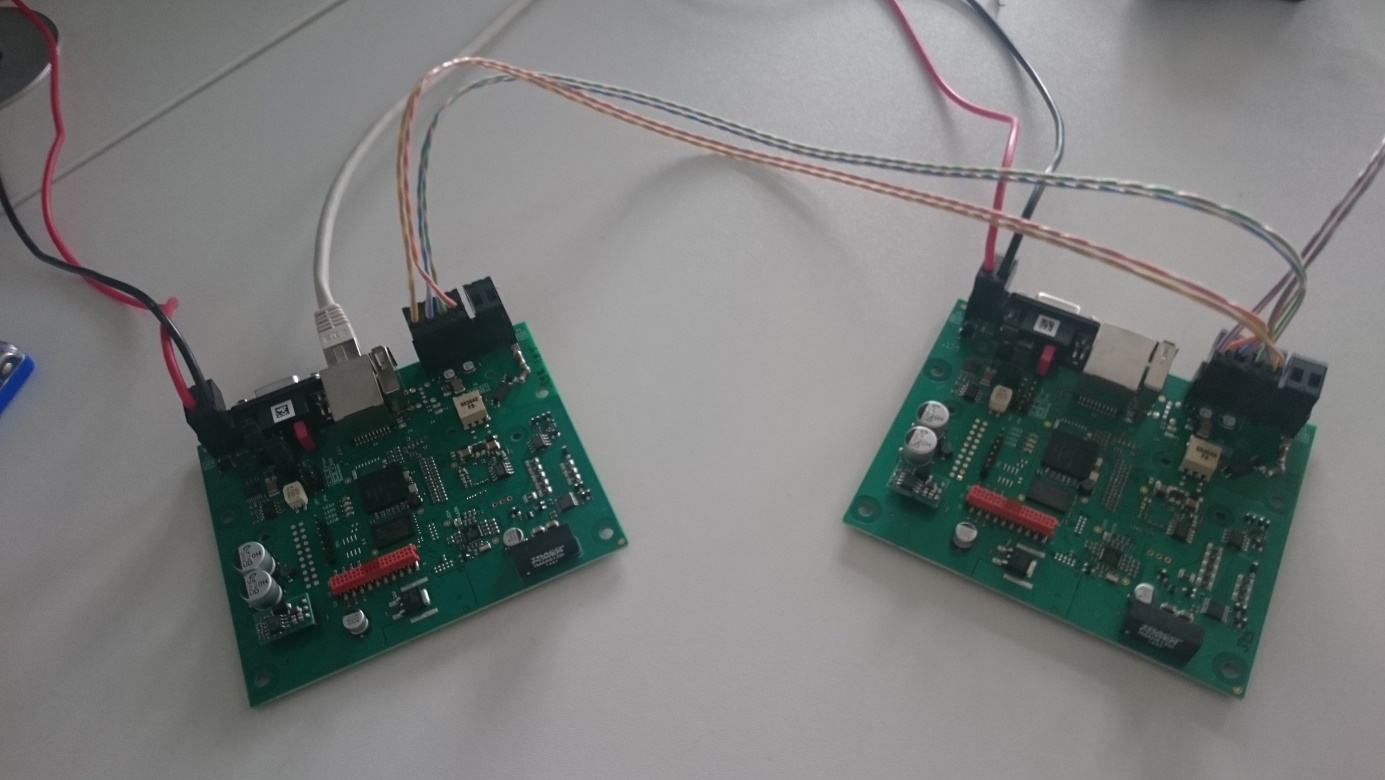
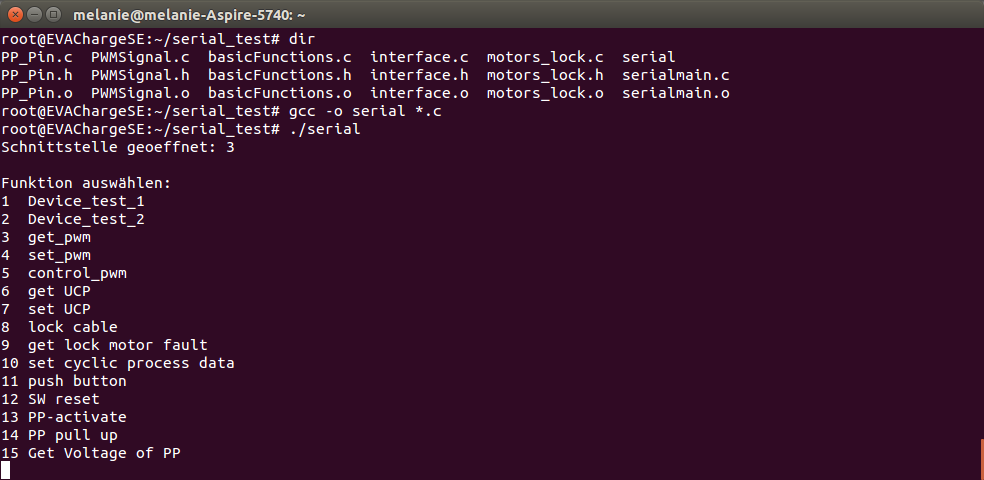


Illustration 5,1Hardware configuration

Furthermore, a setting up of serial\_Programming library could be achieved. Once a board with a Linux computer is connected, it can be started using a Linux terminal. After starting a menu accordinglyFigure 5.2 displays the available functions can be started by that.



**Illustration 5,2: Display of the Linux terminal to start serial\_Programming**

The functions of the ISO / IEC 15118 required from the resulting library are located in the same named subdirectory of the final program. This results in signals on the line of control pins according to the following pictures.

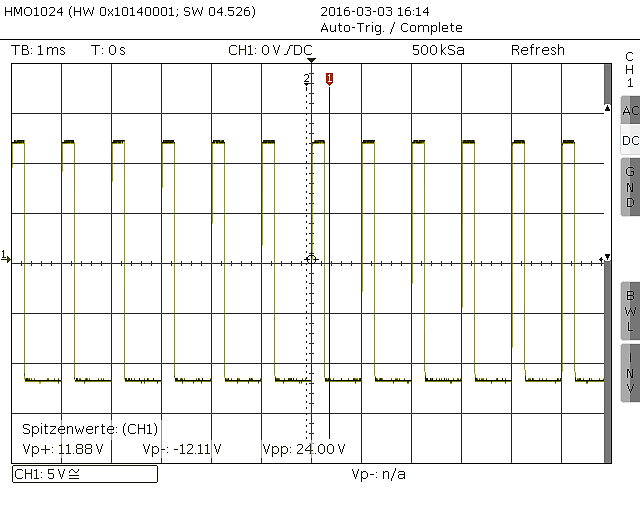


Illustration 5,3: Level of the control pin in charging status A

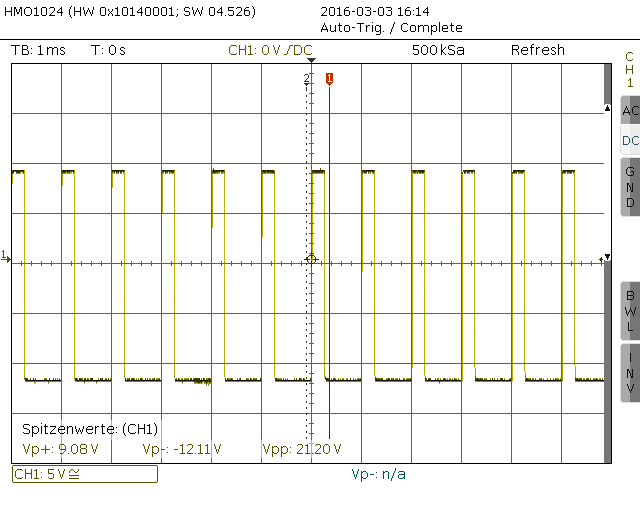


Illustration 5,4: Level of control pin for charge status B

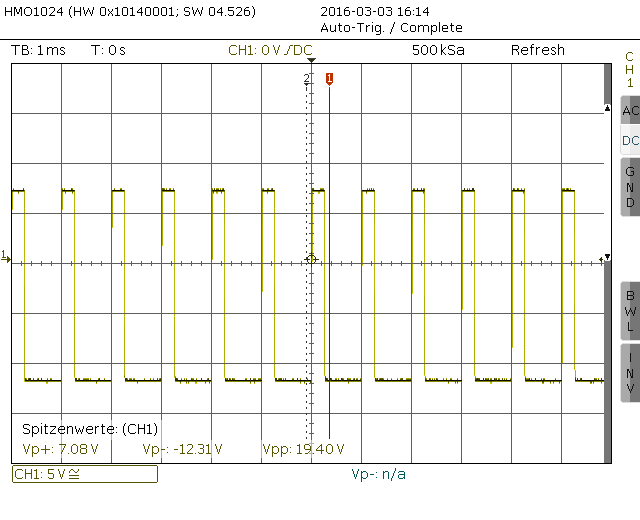


Illustration 5,5: Level of the control pin in Charge status C

It shows Figure 5.3the charging status A, which has a level of ± 12 ± 1 volt. According to IEC 61851, this corresponds to an unaffiliated electric vehicle. In the given case, however, the resistors can be set accordingly, so that the levels shown can be achieved while maintaining a connection between the CP, PE and PP consists of the boards.

In Figure 5.4the level is indicated at bandaged, but not Ladebereitem electric vehicle. As noted in Chapter2.2 mentioned, the negative level -12 ± 1V must be maintained while B is a range of 8 ± 1V is predetermined in the positive range in the present state of charge.

Figure 5.5indicates a limit of the tolerance zone during the charging status C, whereby a load-ready compound is indicative. The tolerance zone for the level with min -12 ± 1V and max. 6 ± 1V indicates the positive peak is however 7,08V. Since the measured value for any further calculations benötig is considered merely one decimal place for determining the level.

# Summary & Outlook

The resulting outcome of the present project is a demonstrator of the ISO / IEC 15118 in hardware and software. For a corresponding hardware has been defined and put into operation and split a communication stack on vehicles and charging stations page. Furthermore, an important part of the construction of the PWM signal and the implementation of IP-based communications between EV and EVSE.

In the course of building a "wallbox" is realized within the scope of a study work. An electric car charger of this type is widespread in private use, as it offers no payment option(Wiki\_Wallbox, 2016), Furthermore, an app with the protocol to be connected, so that a start of the communication protocol and the definition of the parameters contained in it can be transmitted via smartphone.

# attachment

## AC charging procedure



Illustration 7,1: News Summary for AC charging cycle (1 of 2)

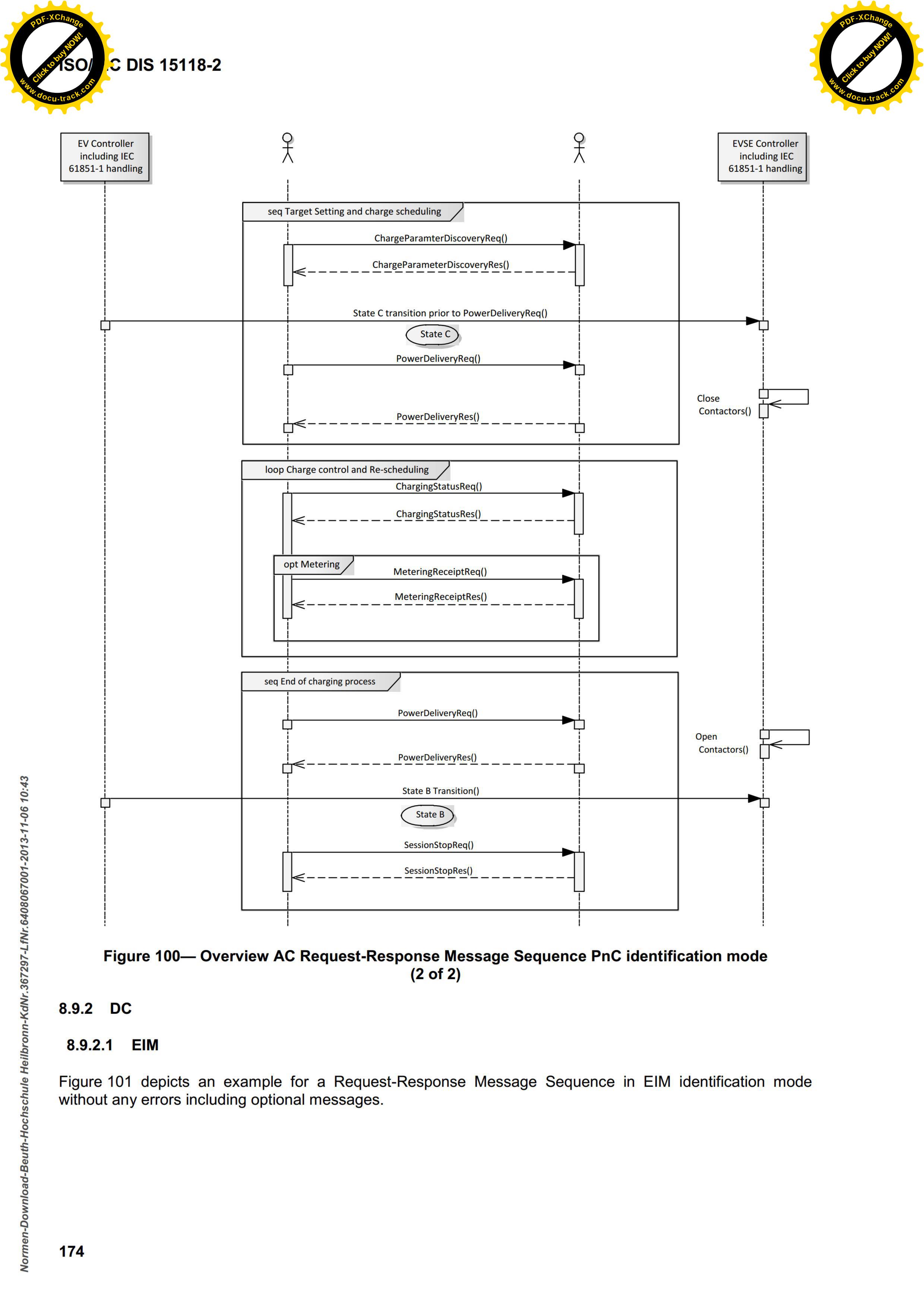


Illustration 7,2: News Summary for AC charging cycle (2 of 2)

## DC charging procedure

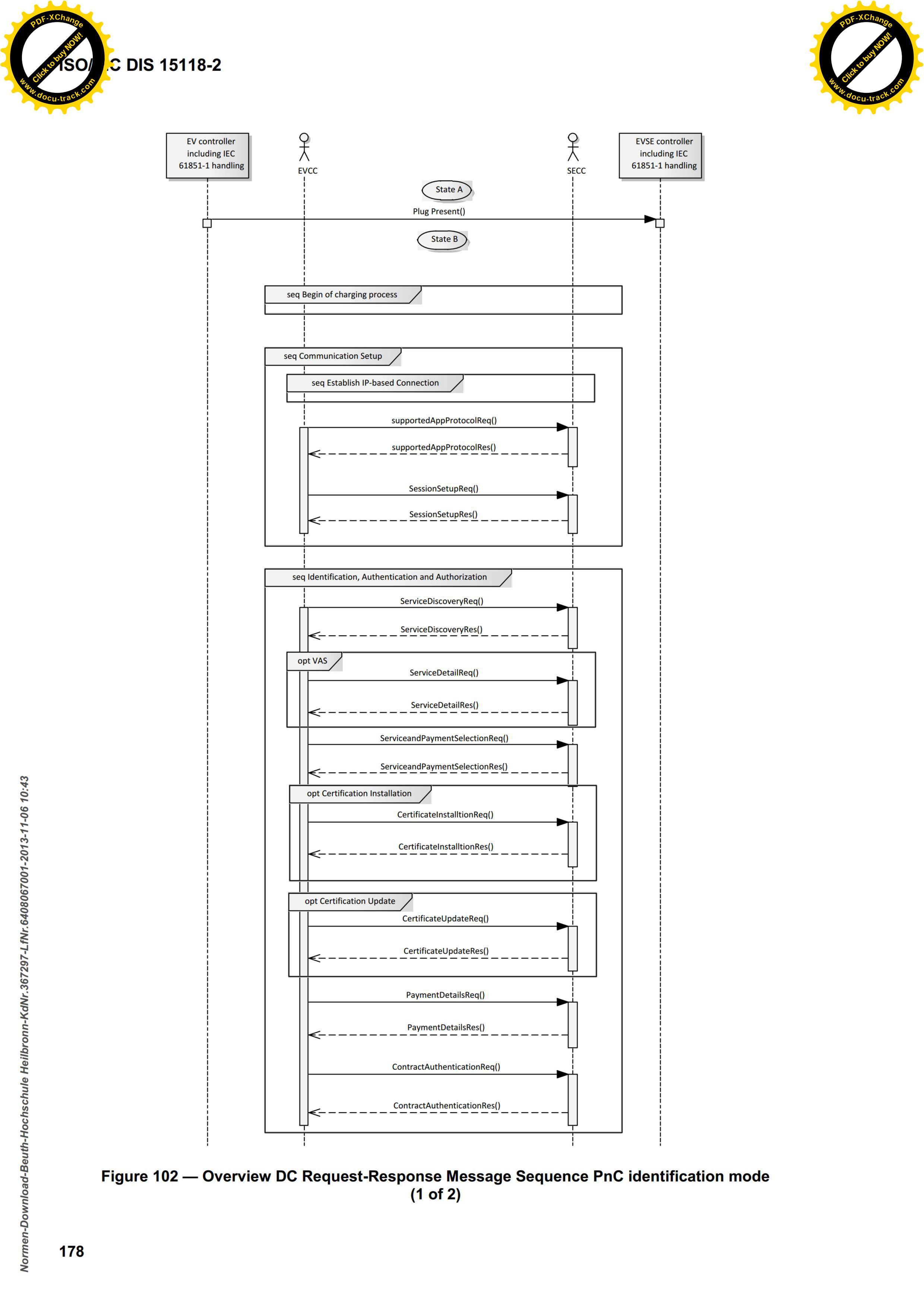


Illustration 7,3: News Summary for DC charging cycle (1 of 2)

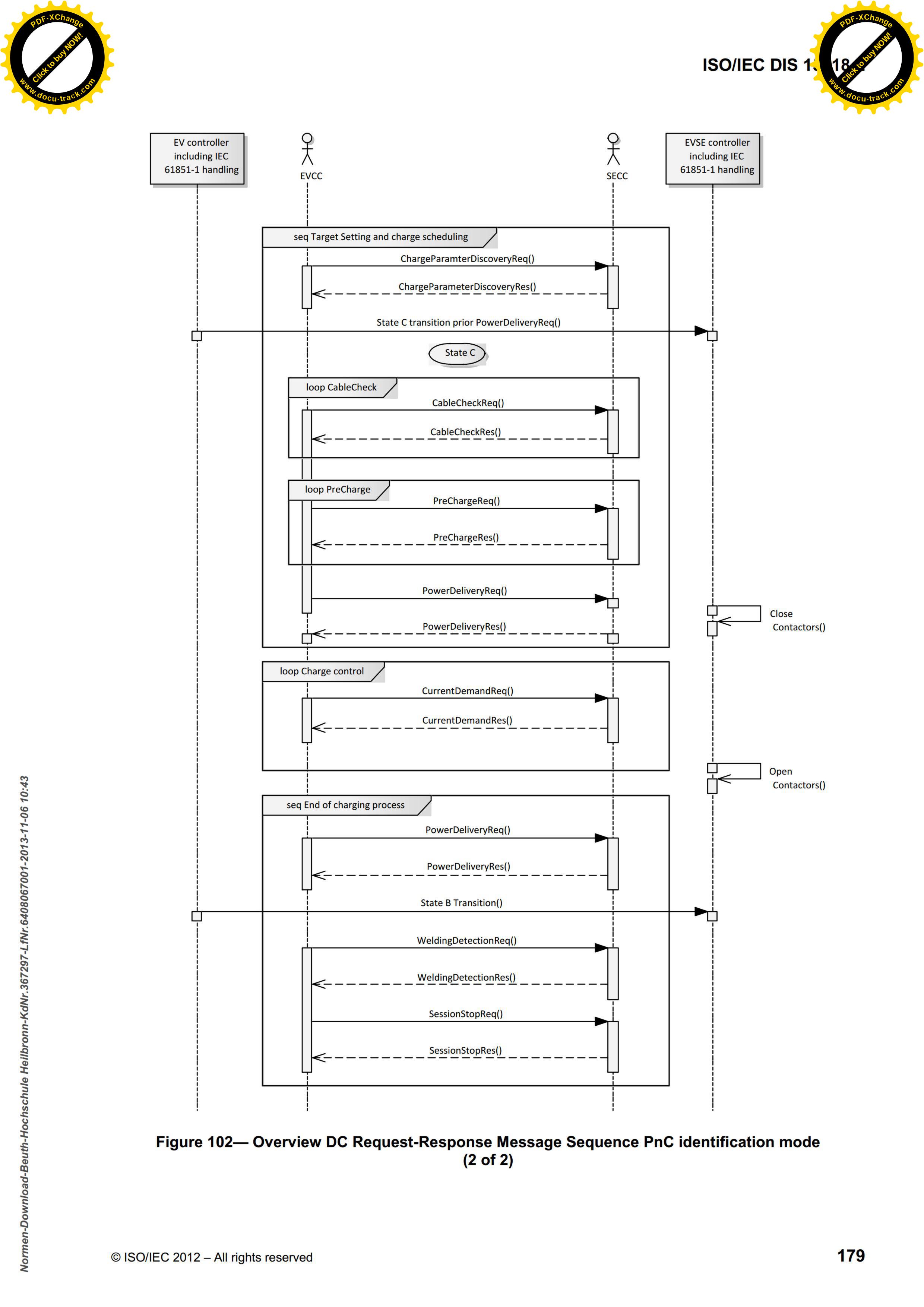


Illustration 7,4: News Summary for DC charging cycle (2 of 2)

## Flowcharts the charging station



Illustration 7,5: Flow chart of charging stations Page



Illustration 7,6: Flow chart of the function call State B1 Communication



Illustration 7,7: Flow chart of the function call State C Communications



Illustration 7,8th: Flow chart of the function call State B2 Communications

## Flowcharts of the electric vehicle



Illustration 7,9: Flow chart of the electric vehicle (Part 1)



Illustration 7,10: Flow chart of the electric vehicle (part 2)

# bibliography

|  |  |
| --- | --- |
| (Barth, 2015) | **Alexander Barth, Michael Röhrich**  Studienarbeit ISO 15118  (E-mail dated 13.05.2015) |
| (Bustamante, 2014) | **Michele Leroux Bustamante**  Finite State Machines, Wizards, and the Web  <https://msdn.microsoft.com/en-us/library/aa478972.aspx>  (Published February 2004) |
| (Debian, 2016) | **What is Debian GNU?**  Debian User's Manual  <http://debiananwenderhandbuch.de/wasistdebiangnu.html>  (Accessed on 22.04.2016) |
| (El-Ko, 2016) | **Electronics Compendium**  Powerline Communication  (Accessed on 13.04.2016) |
| (FileZilla, 2016) | **FileZilla**  <https://filezilla-project.org/>  (Accessed on 25.04.2016) |
| (United, 2015) | **Prof. Dr. Siegmar United**  Script parallel processing  Cape. 4 The socket interface  Fulda University, Department of Applied computer science  (Accessed on 21.06.2015) |
| (Hekman, 1998) | **Jessica Perry Hekman**  Linux in a Nutshell  O'Reilly  ISBN 3-930673-57-6  1. reprint January 1998 |
| (I2SE, 2016) | **I2SE GmbH**  EVAchargeSE SE board  <https://www.i2se.com/product/evacharge-se/>  (Accessed on 17.04.2016) |
| (ITWissen, 2016) | **ITWissen**  socket  <http://www.itwissen.info/definition/lexikon/Socket-socket.html>  (Accessed on 03.01.2016) |
| (Lewandowski, 2014) | **Christian Lewandowski.**  .Entwurf And performance evaluation of a communications architecture for the provision of balancing power by electric vehicles  Technical university Dortmund  (Published on 10/17/2014) |
| (Mültin, 2014) | **Marc Mültin**  Dissertation: The electric vehicle as flexible consumers and energy storage in the smart home  KIT Faculty of Economics at the Karlsruhe Institute of Technology (KIT)  (Accessed on 08.04.2015) |
| (Opatz, 2006) | **Felix Opatz**  Network programming with BSD sockets  revision 1:15  http://www.zotteljedi.de/socket-buch/socket-buch.pdf  (Published on 08/07/2006) |
| (OpenV2G, 2016) | **www.Sourceforge.net**  Open source project to implement a complete ISO 15118 communication protocol  http://openv2g.sourceforge.net/index.php/home.html  (Accessed on 07.04.2016) |
| (Shell commands, 2016) | **Shell commands**  Overview of the most common shell commands  http://www.shellbefehle.de/befehle/  (Accessed on 27.04.2016) |
| (Shrestha, 2015) | **Nerad Shrestha**  10 sFTP Command Examples to transfer files on the remote server in Linux  http://www.tecmint.com/sftp-command-examples/  (Updated 01/03/2015) |
| (Sweet, 2016) | **Michael R. Sweet**  Serial Programming Guide  5th Edition, Revision 3re  <https://support.dce.felk.cvut.cz/pos/cv5/doc/serial.html#config>  (Accessed on 05.01.2016) |
| (Ubuntu\_SSH, 2016) | **Wiki.Ubuntuusers**  SSH  <https://wiki.ubuntuusers.de/SSH/>  (Accessed on 22.04.2016)) |
| (Wiki\_SSH, 2016) | **Wikipedia**  Secure Shell  https://de.wikipedia.org/wiki/Secure\_Shell  (Accessed on 25.04.2016) |
| (Wiki\_Stecker, 2016) | **Wikipedia**  Connector types and charging modes for electric vehicles  <https://de.wikipedia.org/wiki/IEC_62196>  https://de.wikipedia.org/wiki/IEC\_62196\_Typ\_2  (Accessed on 08.03.2016) |
| (Wiki\_Wallbox, 2016) | Wikipedia  Wall charger  <https://de.wikipedia.org/wiki/Wandladestation>  (Accessed on 06.05.2016) |