PM

Electric Vehicle Smart Charging

MASTER'S DEGREE PROJECT

Dino Gonçalves Vasconcelos

MASTER IN ELECTRICAL ENGINEERING - TELECOMMUNICATIONS

UNIVERSIDADE da MADEIRA

A Nossa Universidade

www.uma.pt

June | 2020

Electric Vehicle Smart Charging

MASTER'S DEGREE PROJECT

Dino Gonçalves Vasconcelos

MASTER IN ELECTRICAL ENGINEERING - TELECOMMUNICATIONS

ORIENTATION

João Dionísio Simões Barros

CO-ORIENTATION

Amâncio Lucas de Sousa Pereira

Electric Vehicle Smart Charging

Report of Dissertation

Master in Electrical Engineering – Telecommunications

Dino Gonçalves Vasconcelos

June 2020

Orientation: Professor Doutor João Dionísio Simões Barros

Co-Orientation: Doutor Amâncio Lucas de Sousa Pereira

(This project has received funding from the European Union´s Horizon 2020 Reasearch and Innovation program under the grant agreement no. 731249 and from LARSyS – FCT Plurianual funding 2020-2023)

H2020-LCE-2016-2017
EUROPEAN COMMISSION
Innovation and Networks Executive Agency
Grant agreement no. 731249

Abstract

In recent years, the number of electric vehicles (EVs) has been increasing. It will play more and more important role in the power grid operation because of its storage features, where charging and discharging control strategies and construction of the charging facilities are the priorities to be solved in this field. At the same time, the availability of the charging infrastructures is still limited and provides a promising application area for communication and control research.

This thesis focuses on the control of the energy status offered by the buildings/houses providing a power management across the EV charging session. It also describes the development of a system to dynamically control the charging of EVs and maintain the operation of the power system by knowing the available power. To charge EVs a communication between the EV and the charging station is needed. The communication is based on the standard IEC61851, which specifies a pulse width modulation (PWM) signal that is sent to the EV to define the charging current.

The system hardware consists of two modules, a charging station and a coordinator both with communication capabilities, while the power management algorithms are in a server, which then calculates the available power based on the power consumption behaviour and uses it to assign the charge of the EV. The system aims to control the EV charging session considering the power consumption to control their charging current. For the case of multiple EVs, the system will manage the charging session based on a priority level prioritizing the EV how started to charge first.

The results shows the feasibility of the charging system approach to control the EV charging station considering the system power consumption and the introduction of the priority level for multiple EVs.

Keywords: Electric vehicles, power management, charging session, charging station and control.

Resumo

Nos últimos anos, o número de veículos elétricos está aumentando. Eles desempenham um papel cada vez mais importante na operação da rede elétrica por causa do seu recurso de armazenamento, onde as estratégias do controlo de carregamento e descarregamento e a construção das instalações de carregamento são as prioridades a serem resolvidas neste campo. Ao mesmo tempo, a disponibilidade das infraestruturas de carregamento ainda é limitada e fornece uma área de aplicação promissora para pesquisa em comunicação e controlo.

Esta tese concentra-se no controlo do estado de energia oferecido pelos edifícios/casas fornecendo uma gestão de energia durante a sessão de carregamento de veículo elétrico. Também descreve o desenvolvimento de um sistema para controlar dinamicamente o carregamento dos veículos elétricos para manter a operação do sistema de energia, conhecendo a energia disponível. Para carregar os veículos elétricos, é necessária uma comunicação entre o veículo elétrico e a estação de carregamento. A comunicação é baseada na norma IEC61851, que especifica um sinal de modulação por largura de pulso enviado ao veículo elétrico para definir a corrente de carregamento.

O *hardware* do sistema consiste em dois módulos, uma estação de carregamento e um coordenador ambos com capacidade de comunicação, enquanto os algoritmos de gestão de energia estão num servidor que calcula a energia disponível com base no comportamento do consumo de energia e usa-o para atribuir a carga do veículo elétrico. O sistema tem como objetivo controlar a sessão de carregamento do veículo elétrico, considerando o consumo de energia para controlar a corrente de carregamento. No caso de vários veículos elétricos, o sistema irá gerir a sessão de carregamento com base em um nível de prioridade, priorizando o veículo elétrico que iniciou o primeiro carregamento.

Os resultados mostram a viabilidade da abordagem do sistema de carregamento para controlar a estação de carregamento do veículo elétrico, considerando o consumo de energia do sistema e a introdução do nível de prioridade para vários veículos elétricos.

Palavras chave: Veículo elétrico, gestão de energia, sessão de carregamento, estação de carregamento e controlo.

To my girlfriend, brother, sister and parents

"Imagination is more important than knowledge. Knowledge is limited. Imagination encircles the world.

Imagination is the highest form of research.

I have no special talent. I am only passionately curious."

Albert Einstein

Acknowledgements

A special thank you to my advisor, professor Dionísio Barros, for all the support, availability, motivation and knowledge transmitted, not only during this project but over my academic journey.

The Horizon 2020 European Union Funding for Research & Innovation program, under the scope of project SMILE (Smart Island Energy Systems), and LARSyS – FCT Plurianual how funded the development of this project.

To my co-advisor, Lucas Pereira, for all the support and motivation, and to all Madeira Island SMILE partners for helping and making this project possible.

I am grateful to the Electricity Company of Madeira for providing the facilities to implement and test the smart charging system, and workers, Diogo Vasconcelos, Henrique Correia, Adriano Silva and Aires Henriques for the support, availability and help provided during this project.

LARSyS | ITI and PRSMA colleagues, Wilson Santos, Luísa Barros, Jonathan Cavaleiro, Filipe Quintal, Sabrina Scuri and Mary Barros for the support and help provided during this project.

Many thanks to my colleagues, Íuri Viveiros, Rui Martins, Pedro Nunes, Rafael Velosa, Sérgio Rodrigues, Duarte Alves, David Inácio, Rodrigo Teixeira, among others, for the support and good moments passed during this project.

A thank you to the University of Madeira and teachers for providing me all the conditions necessary to complete this journey.

Table of Contents

List of Tables

List of Figures

Glossary

AC	*Alternating Current*
ASCII	*American Standard Code for Information Interchange*
BEV	*Battery Electric Vehicle*
CP	*Control Pilot*
DC	*Direct Current*
EV	*Electric Vehicle*
GPIO	*General Propose I/O*
HEV	*Hybrid Electric Vehicle*
HTTP	*Hyper Text Transfer Protocol*
ISM	*Industry Scientific and Medical*
IEEE	*Institute of Electrical and Electronics Engineers*
ICE	*Internal Combustion Engine*
ICEV	*Internal Combustion Engine Vehicle*
IEC	*International Electrotechnical Commission*
ISO	*International Organization for Standardization*
PV	*Photovoltaic*
PHEV	*Plug-in Hybrid Electric Vehicle*
PP	*Proximity Plug*
PWM	*Pulse Width Modulation*
RTU	*Remote Terminal Unit*
SAE	*Society of Automotive Engineer*
SoC	*State of Charge*
SQL	*Structured Query Language*
TCP	*Transmission Control Protocol*
URL	*Uniform Resource Locator*
V2B	*Vehicle to Buildings*
V2G	*Vehicle to Grid*
V2H	*Vehicle to Home*
V2V	*Vehicle to Vehicle*

Chapter 1

Introduction

The EVs engineering is merged with the automotive and electrical engineering by including a motor, a power electronic converter, a controller, a battery and an energy management system. Each module have to be able to work together and have the best performance to achieve the required driveability at the maximum energy efficiency.

EVs development has progressed tremendously during the last few years that led to a steady increase in EVs adoption [1]. This increase brings new challenges and the electrical grid may need to increase the power capacity, or infrastructure, in order to accommodate charging of large numbers of EVs. The main problem is the extra energy required to charge EVs and the consequent peak load of charging. The simultaneous charging of several EVs will lead to a considerable additional load that can overload the grid.

This thesis focuses on the development of a smart EV charging system to manage the EV charging session by using the IEC61851 standard, which is based on a PWM signal that is sent through a dedicated communication line to control the EV charging. This system allows the EV to charge depending on the power consumption status.

This chapter presents the motivation behind this project, the main objectives and the way that this work is organized.

1.1. Motivation

EV are now gaining some attraction in Madeira Island and given the recent advances, as well as, the recent proposals in some countries for totally removing fuel power vehicles, it is expected that in the next few years EVs will become very prominent in Madeira Island. Ultimately, while EVs come with the promise of a bright future, this will result in a significant increase in the power consumption demand, which have to be met by the local grid.

Most of the buildings/houses are already nearing the power capacity or have already reached it, meaning no new electrical infrastructure can be added without costly upgrades. Without that infrastructure, EV chargers cannot be added. As the number of EVs on the road grows this becomes an untenable solution and the grid cannot produce unlimited power.

Building and houses need a way to tap into the grid without adding new infrastructure and still future-proofing themselves for even more EVs in the coming years. Giving every building/house the ability to intelligently allocate power is important. Power management makes it possible to charge more EVs by dynamically sharing existing power across more charging spots to charge more EVs.

A smart charging solution is the ideal way to meet the EV demand for electricity. This solution is essential to manage the EV charging. It is also motivated by the relevance and importance of the problem, which arises from the fact that the EV market is and will continue to grow resulting in a grid impact.

1.2. Objectives

The project aims to implement an EV charging system in buildings or houses, with the goal of providing a dynamic charging system to manage the EV power consumption considering the building or house power consumption.

It is necessary to analyse the challenges caused in the power grid with the increase of the EVs power consumption and the opportunities made by the integration of the EVs.

Investigate the EV charging infrastructure, such as standards, charging schemes, sockets/plugs and charging modes, and what types of charging stations are in the market for buildings, houses and public environment, as well as, the most common charging strategies implemented in research.

Check the EV and charging station specifications to implement the appropriate hardware to establish the communication between the charging station and the EV.

Design the EV charging system according with the specifications and requirements necessary to perform the control of the EV power consumption considering the power consumption.

Simulate charging scenarios to control the EV power consumption to compare results and implement the ideal scenario on the developed charging system.

Last but not least, test the developed charging system and implement the charging scenario to compare with the simulation results.

1.3. Thesis outline

This thesis is divided into six chapters. Following this introduction chapter, where the project outline, motivation and objectives are present, is the literature review chapter, where is presented the EV background, grid integration, charging infrastructure, chargers in the market, charging strategies, charging system specifications and a short discussion with conclusions.

Chapter 3 presents the design of the EV charging system prototype, to be able to remotely control the EV charging session. The first subchapter presents the coordinator implemented to aggregate all the data and be responsible to exchange the messages between the charging station and the server. The second subchapter presents the charging station custom hardware developed to allow the EV connection and control of the power consumption. The last subchapter presents the server with web page interface to visualize the EV power consumption and schedule the EV charging session, as well as, the available power calculation, according with a priority system, and the PWM duty cycle calculation, according with the available power.

Chapter 4 presents the simulations of EV charging scenarios and the comparison of the results to choose the best scenario.

Chapter 5 presents the experimental verification of the EV charging system, the charging scenario applied and a comparison with the simulation results.

Chapter 6 concludes the project and gives recommendations for further work.

Chapter 2

Literature review

This chapter presents the EV charging infrastructure, grid integration, chargers in the market, charging strategies, the EV charging system, the charging station specifications, the EV designation and the overall conclusion of this chapter.

The EV designation has the different types of EV, the most common batteries constitution, the different types of EVs depending on the electricity source supply, the history, the advantages and disadvantages of the EVs compared to the internal combustion vehicles.

To connect the EV into the power grid, it is necessary to study the grid integration challenges and opportunities. However, the first step to allow the EV connection, is to know the EV sockets connections, standards, communication, charging modes and schemes, as well as, the type of EV charger.

The charging station is the interface that the EV has to access the electric power grid and it is necessary to know the charging station requirements to allow the EV connection, to be able to stablish a communication with the EV, as well as, know the EV hardware and how the power consumption can be control.

2.1. Electric vehicles

Due to increasing concerns on environmental issues, clean energy economic factors, EVs have attracted more and more the attention of governments and industries. EVs are regarded as one of the most effective strategies to reduce the oil dependence and gas emission [1].

In the past years, the EV technologies focused on individual components or systems in EVs, such as electric machines, drive systems, batteries, fuel cells, on-board renewable energy, and so on. However, with the emerging concept of the smart-grid, or micro-grid, EVs are playing a new role as energy exchange with the power grid because they are capable of not only drawing the energy from the power grid, but also delivering the energy back to the grid via a bidirectional charger [2].

The rapid development and major accomplishments for the production of advanced batteries have also resulted in significant interest toward the development of EVs. Battery powered electric vehicles (BEVs) produce no direct emissions. Electric motors are more efficient than internal combustion engines (ICE) and can take advantage of energy saving techniques, such as regenerative braking which recovers some of the energy that would otherwise be lost as heat and friction [3].

Mass production EVs capable of traveling longer distances results in a need for electric service stations that can satisfy the requirements for a significant amount of power provided in a time duration similar to that of filling a car with oil based fuel. These vehicles could pull into the station and need a large amount of power delivered over a short period of time for the rapid recharging of batteries which serve as their “fuel tank” [4].

The overall EV engineering philosophy essentially is the integration of automobile and electrical engineering. System integration and optimization are prime considerations to achieve good EV performance at an affordable cost. Since the characteristics of electric propulsion are fundamentally different from those of engine propulsion, a new design approach is needed for EV engineering. Advanced energy sources and intelligent energy management are key factors to enable EVs competing with internal combustion engines vehicles (ICEVs) [5].

2.1.1. History

Since the invention of electric motor, EV was considered among the earliest automobile a well ahead of combustion engine. It dominated the vehicle registration comparing to gasoline vehicles in the late 1920s to 1930s and held most of the land vehicle performance record in early 1900s. It was a major transportation tool and widely used in the society for local transportation improved from horse carriages [6].

Until 1930, EV leadership was overtaken by gasoline vehicle. Infrastructure improvement and demand of inter-city travel required a longer travel distance that was never able to be exploited by EV before. The widely discovery of gasoline and ready availability of cheap fuel also contributed the spread of gasoline vehicle because it could be carried around by a container which enabled and extended the mobility of owning a vehicle [7].

EV development was dominated by EV1 who produced by General Motors for fleet application, Ford developed EV Ranger pick-up truck, Toyota provided Rav4 EV and Honda had an EV available as well during late 1990s and early 2000s. Figure 2.1 shows this manufactures vehicles and estimate range [8].

Figure 2.1 - The General Motors, Honda, Ford and Toyota EVs [8].

Unfortunately, this EVs availability did not made into commercial production because of politics, economic and technology that includes vehicle production cost and safety concerns. EV1, Ranger, Rav4 and Honda EV were intended for fleet test only, almost all the vehicles have been discontinued, destroyed and recycled [8].

At the beginning of the 21st century, combined with environmental concern, the interest in electric has increased the EV technology. Since 2010, combined sales of all electric cars and utility vans achieved 1 million units delivered globally in September 2016 and combined global sales of light duty all electrics and plug-in hybrids passed 5 million in December 2018 [9].

2.1.2. Types

Electric vehicles are divided into three main types:

- Hybrid electric vehicles (HEVs) uses a small electric battery to supplement internal combustion engines (ICEs). The battery is recharged by the ICE and regenerative braking [10].
- Plug-in hybrid electric vehicles (PHEVs) is also a dual fuel car in which both the electric motor and the ICE can propel the car. Compared to the HEV, it has a larger battery-pack and it can be charged from the power grid [11].
- Battery electric vehicles (BEVs) have no ICE and must be plugged into the electric power grid for charging its batteries. Compared to the PHEV, it has a larger battery-pack [12].

2.1.3. Battery

Battery is the key component of the EV offering the role of energy exchange between the EV and the power grid. An electric battery is a combination of two or more electric cells connected together. The cells consists of positive and negative electrodes in an electrolyte. The chemical reaction between these two generates DC electricity. In the case of rechargeable batteries, the chemical reaction can be reversed by reversing the current which will make the battery return to a charged state [13], [14].

The main battery selection criteria is: type, size (energy density - Wh/kg or Wh/l), weight, cost, life cycle and efficiency. Table 2.1 lists three main types of batteries, namely, the lead–acid (Pb–acid), the nickel–metal hydride (Ni–MH), and the lithium-ion (Li–ion). The Pb– acid type has the lowest energy density and cycle life. Furthermore, it has the memory effect for charging which needs to fully charge each time and is not ideal for modern EVs. On the contrary, the Ni–MH and Li–ion batteries have much better performances, hence favouring modern EVs [14] - [15].

Table 2.1 - Pb-acid, Ni-MH and Li-ion energy densities and cycle life's [16].

Type	Pb-acid	Ni-MH	Li-ion
Specific energy (Wh/kg)	30 ~ 45	60 ~ 120	90 ~ 160
Specific power (W/kg)	200 ~ 300	150 ~ 400	250 ~ 450
Cycle life	400 ~ 600	600 ~ 1200	1200 ~ 2000

Since there are many cells stacked in series to achieve the required voltage for the EV, the battery reliability depends on the reliabilities of several hundred cells [16]. Table 2.2 compares the cell and battery reliabilities, adopting many batteries to achieve the voltage level of about $312\ V$. It can be seen that the cell reliabilities of these batteries are all over 97 %. However, when the cells are connected in series, only Li–ion can offer the battery reliability over 90 %.

Table 2.2 - Pb-acid, Ni-MH and Li-ion batteries characteristics [16].

Type	PB-acid	Ni-MH	Li-ion
Cell voltage (V)	2	1.25	3.6
Number of cells	156	250	87
Battery voltage (V)	312	312.5	313.2
Cell reliability (%)	98.45	97.53	99.14
Battery reliability (%)	85.55	77.9	91.69

With these comparisons most of EVs use lithium-ion batteries. They have higher energy density, longer life span and higher power density than most other practical batteries. However, Li-Ion batteries have their disadvantages as well. They are expensive to manufacture, at least for the time being, and subject to aging, even if not in use. The batteries require a protection circuit to limit voltage and current, otherwise if a Li-Ion battery is run fully empty, the battery might become useless. Also high temperature, air humidity and fast charging shorten the lifetime of the Li-Ion battery. Safety forms are a very important aspect in battery technology as it might explode when short-circuited or overheated while charging, so a proper cooling system is important [15] - [18]. Table 2.3 presents the typical charging time of an 24 kWh EV battery depending on the power supplied.

Table 2.3 - Typical charging time of an 24 kWh EV battery [19].

Charging time for a typical 24 kWh battery capacity	Power supplied (kW)	Voltage	Maximum current (A)
10.4 hours	2.3	230 VAC	10
8.3 hours	3	230 VAC	13
6.5 hours	3.7	230 VAC	16
3.2 hours	7.4	230 VAC	32
1.6 hours	14.5	230 VAC	63
1.04 hours	23	230 VAC	100
29 minutes	50	400 – 500 VDC	100 – 400
15 minutes	100	400 – 500 VDC	100 – 400

2.1.4. Description

An EV uses one central electric motor, or several electric motors, directly connected to the wheels of the vehicle and drives its electric power from a pack of batteries that are charged from an external power source through an electric socket and also uses regenerative braking to charge the batteries (Figure 2.2) [20], [21].

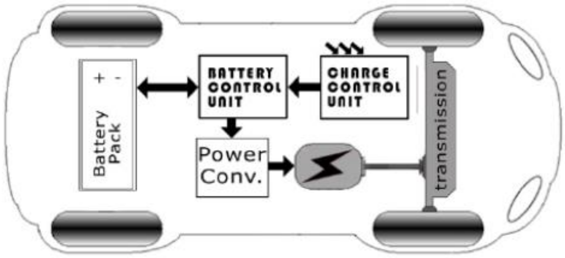

Figure 2.2 - Simple EV description [16].

The speed of the motor varies by changing the voltage across the motor. The vehicle incorporates a protective system in case of a fault, a current limiter to limit the charging current and a thermistor to sense the motor winding temperature to act if the system temperature exceeds a pre-set safe value. Microprocessors are used in speed controllers for optimal efficiency control and self-diagnostics. The advances in semiconductor device technology, particularly the availability of high power transistors and high power gate turn off thyristors, make the use of the AC drives a realistic possibility by controlling the energy transfer between the EV and the external power source [20], [21].

2.1.5. Advantages and disadvantages

Compared with ICE, electric motors are, mechanically, very simple providing a smooth operation and consequently produce less noise, whether at rest or in motion. EVs emit no tailpipe emissions, which makes them suitable for use in urban and suburban areas. The conversion of ICEVs into EVs has the potential to reduce emissions even considering the increase of CO2 emissions in electric power plants. In large power plants, where the power generation mix is based in renewable or nuclear energy, an EV generates less CO2 than a conventional ICEV [22].

EVs allows energy independence by using domestically generated electricity rather than relying on foreign oil or power plants and can be generated using natural renewable resources, such as solar or wind power that helps to minimize pollution [23].

The EVs are impacting both economy and ecology. EVs are more efficient and consume less energy than ICEV. However, a main drawback of EV, compared to those with ICE, is lower autonomy due to the capacity storage of electric energy and long charging periods. The limitation of autonomy demands an optimal charging network for EVs. This means that the number and spatial distribution of charging stations must provide charging ability to travel on longer distances. Charging locations should be set in such way that when demand for charging occurs, the vehicle can be recharged at nearest charging point. In addition, the electric battery needs long charging periods to fully charge, from about thirty minutes to over six hours [23], [24].

2.2. EV grid integration

One of the great challenges in power system engineering is making sure that the amount of power supplied to the loads is in line with expectations. Since the active power has to be consumed at the exact time as it is produced, the amount of active power consumed plus losses should always be equal to the active power produced and supplied from energy storage within the system [25].

To integrate EVs into the grid, an intelligent and sustainable manner is needed. Depending on the level of sophistication of the vehicle charging process, the EVs may be considered from the view of the distribution system as [25]:

- A simple load that drawing a continuous current independently from discrete network nodes;
- Flexible load from an aggregation of vehicles with coordinated charging;
- Generation units where EVs are using their storage devices to inject power into the grid according to available resources.

There are three emerging concepts of grid connected EV technologies, being: Vehicle to Home (V2H), Vehicle to Vehicle (V2V) and Vehicle to Grid (V2G) [15]. V2H refers to the power exchange between the EV battery and home power network. In this case, EV battery can work as energy storage providing the backup energy to the home electric appliances and to the home renewable energy sources. V2V is a local EV community that can charge or discharge EV battery energy among them. V2G utilizes the energy from the EV battery and trades energy to the grid through the management of the local aggregator [26].

2.2.1. Challenges

Distribution grids are planned to be able to accommodate all the loads in the system, given a certain simultaneous factor, for the peak hour. The integration of a large number of EVs into the electric power system is a major challenge, since it requires an assessment and observation in terms of economic impacts, operation and control benefits at optimal conditions. Many existing literatures analysed the impact of the EVs on the distribution power system, while others analysed the different application on how to make the EV adoption into the electric power system [27].

The majority of the EV charging are conceived to be undertaken at home and the load peak hour consumption in a residential area coincide with home arrival of a large number of EVs, following the common traffic patterns. Therefore, the EVs will be plugged-in by that time and charging will start with no control, using a free policy charging, aggravating the previously existent operating conditions from the power grid [28].

EV charging is also foreseen to be taking place in the public, commercial or working place charging stations. Since these vehicles will require the use of batteries with high energy storage capacity and with large electric load charging requirements, a large deployment of this concept will provoke considerable impacts in the electric power system design and operation. These impacts may limit the growth of EV penetration if no additional measures are adopted. With the increase in load consumption at peak hours, due to the EVs presence, will require generation levels to increase leading to a subsequent reduce of efficiency, require additional generator starts, voltage deviations, harmonics distortion, substation transformer loading and incurred new investments of power distribution facilities [29] - [30].

2.2.2. Opportunities

The adoption of the EVs can extensively add value in the electric grid in terms of performance, efficiency and power quality improvements. This is possible if the EVs integration is well planned and technically reorganized to conform to the power system operational standards [31].

The EVs penetration in the power grid have opened up the possibility of the V2G implementation. V2G refers to the control and management of EV loads. The EV battery could be seen as an extra storage device when plugged-in, but, more importantly, as a flexible and highly controllable load that can be manage by the power utility, or aggregators via the communication between vehicles and the power grid, in order to achieve desired benefits. In most cases, the objectives of the V2G management are to maximize profit, reduce emissions and improve the grid power quality [31], [32].

Many studies focus on smart charging control of the EVs and it becomes clear that there are many different philosophies. Smart charging could be implemented to avoid distribution level grid congestion, peaks in demand, voltages below a certain value by active power control, low voltages by reactive power injection that could be reduce by the power factor for vehicles charging and EVs could be instructed to avoid charging [32].

The integration of renewable energy sources into the existing power grid suffers from unpredictable supply of the electricity, especially wind and PV solar energies. The electric power production from these renewable energy sources can be very high, more than the power demand, or very low, less than the power demand, depending on the available energy sources. EV batteries can be aggregated and act as an energy storage system that can balance the integration of the renewable energy sources into the power market as dynamic energy storage device, which can absorb the excessive power generated through different charging schemes, or can deliver power to the grid in the lower power generation scenarios and level the grid operations. The EVs will be acting as energy buffer for the grid regulations and ancillary services [33], [34].

2.3. Charging infrastructure

A charging infrastructure is an important entity of the power grid. Availability of this infrastructure is a key factor of the EVs. In addition to the physical charging facilities, the charging infrastructure is a central communication interface among EV, charging station, power grid and energy suppliers. The data returning from EV through the charge point can be used for monitoring, scheduling, energy distribution controlling and managing the energy consumption [35].

2.3.1. Standards

Reliability, availability, safety and efficiency are important factors in charging an EV and the widespread acceptance of EV depends on these factors. Different EV related standards are provided by the International Organization for Standardization (ISO), International Electrotechnical Commission (IEC), Society of Automotive Engineers (SAE) and other standard development organizations to address the connectors, safety, communication, charging topology and interoperability [36], [37]. Table 2.4 summarizes some of these standards.

Table 2.4 - Standards of EVs infrastructure [37].

Measure	Standards
Connector	SAE J1772, IEC62196, GBT20234
Safety	IEC60529, IEC60364, ISO6469, ISO17409, SAE J1766, SAE J2344, SAE J 2929, SAE J2578, SAE J2464, SAE J2380
Communication	SAE J2931, SAE J2847, ISO/SAE 15118, IEC61851, IEC61850, IEEE 80211P, GBT 27930
Charging topology	SAE J2836, IEC61851, GBT20234, IEEE P2030, IEEE P1547, IEEE P1901.2, IEC61980
Interoperability	SAE J2953, IEEE P2030

Standards beginning with SAE were developed in the United States of America, GBT were developed in China and IEC/ISO were developed internationally and are mainly used in Europe.

The main centre of activity in standardization development appears to be in United States of America, Japan and China with slower progress in the European Union.

Figure 2.3 presents the main standards between the EV and the charging station.

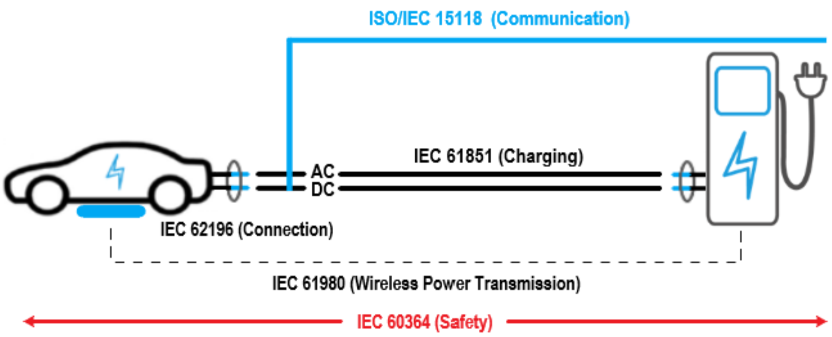

Figure 2.3 - Sematic of charging station and EV standard protocols [38].

2.3.2. Chargers

EV chargers can be categorized into on-board, off-board, unidirectional, bidirectional, conductive and inductive chargers, with support for different power charging types [39], [40].

The on-board charger is located inside the EV and allows it to plug-in to any suitable power source, but this charger limits the power that the battery can charge. An off-board charger does not have this limitation, since it is installed on a charging infrastructure, and it is used for higher power rate charging [39], [40].

The EV charger can be unidirectional or bidirectional power flow. Unidirectional is the traditional method of charging and its simplicity makes it relatively easy for a utility to manage heavily loaded feeders due to multiple EVs. Unidirectional chargers can only charge the EV and they are not able to inject energy back to the power grid. Although, bidirectional chargers can inject energy back to the power grid, but additional hardware is necessary to provide this ability [39], [40].

In conductive charging, there is a physical wired connection between the EV and the charger providing AC or DC power. A specific cable is needed considering the high voltage and current to charge an EV. However, inductive charging is based on magnetic contactless power transfer and there is no wired connection between the EV and the charger [36] [39], [40].

2.3.3. Plugs

There are four main different types of plugs defined in the IEC62196 standard [36]:

- Type 1 (Figure 2.4) – Is a single-phase plug and can provide a charging power up to 7.4 kW. This plug is mainly used in United States of America and Asia [41].

Figure 2.4 - Type 1 plug/socket, also known as J1772 [42], [43].

- Type 2 (Figure 2.5) – Enables a single or three-phase power system and can provide a charging power up to 55 kW. However, a charging power of 22 kW is more common. This plug is mainly used in Europe and most of public station are equipped with type 2 plug [41].

Figure 2.5 - Type 2 plug/socket, also known as Mennekes [42], [43].

- Type 4 (Figure 2.6) – The quick charging system was develop in Japan and allows a charging power up to 50 kW [41].

Figure 2.6 - Type 4 plug/socket, also known as CHAdeMO [43], [44].

- Domestic (Figure 2.7) – Enables a charging power up to 3.7 kW, but a charging power of 2.3 kW is more common. Most of EVs manufactures provide a charging cable adaptor for this plug [41].

Figure 2.7 - Domestic plug and a charging cable example with the in-cable box controller adaptor [43], [45].

2.3.4. Charging modes

The EV charging speed can be roughly split into slow and fast charging, that can be further divided into sub-categories defined by IEC62196 standard. In the standard the sub-categories are: mode 1 and 2 correspond to a slow charging process, mode 3 to a fast charging process and mode 4 to a rapid charging process [36], [46].

The charging modes are referenced in IEC62196 standard:

- Mode 1 (Figure 2.8) – Provides basic charging capabilities for domestic use, such electrical plugs and outlets that can be used. The current in this mode is at most 10 A, meaning that for a single phase connection the charge power is limited to approximately 2.3 kW. This mode corresponds to a slow charging process [46].

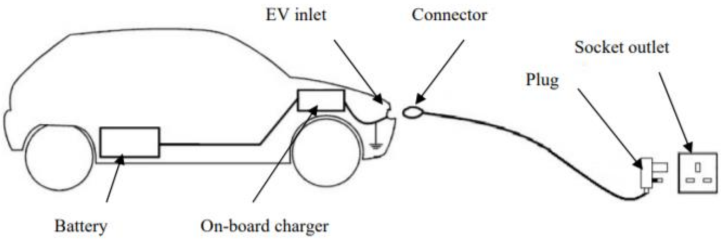

Figure 2.8 - Charging mode number 1 [47].

- Mode 2 (Figure 2.9) – Allows a higher charging current than mode 1, but imposes additional safety measures on the EV inlet. A control pilot (CP) pin in the charging cable between the in-cable control box and the EV charger connector is used to indicate the maximum charge current, supported by the cable or installation. This mode corresponds to a slow charging process [46].

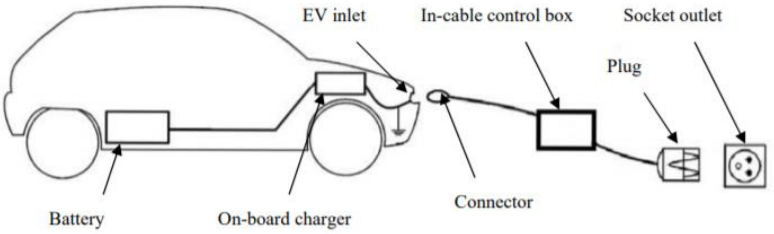

Figure 2.9 - Charging mode number 2 [47].

- Mode 3 (Figure 2.10) – Defines a charging connection up to 55 kW and requires the use of a charging point. The requirement of a proximity pin (PP) ensures that a sudden disconnection or interruption is detected and the cable becomes unpowered. This mode corresponds to a fast charging process and is the most common charging mode implemented [46].

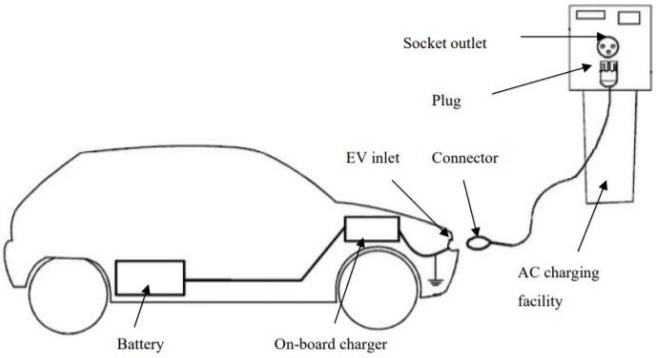

Figure 2.10 - Charging mode number 3 [47].

- Mode 4 (Figure 2.11) – Defines DC charging by converting AC power to DC power in the charging station, without needing the EV on-board charger. The charging capacity varies from 50 to 100 kW of power. This mode corresponds to a rapid charging process [46].

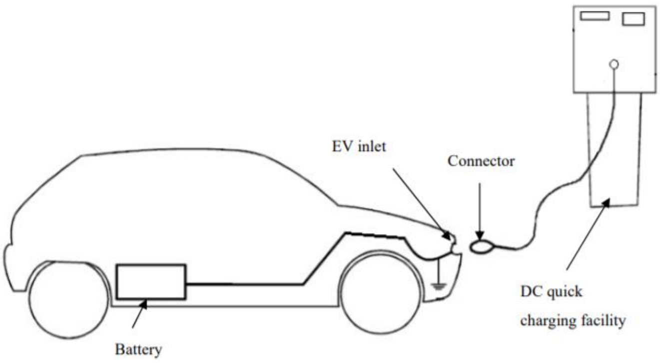

Figure 2.11 - Charging mode number 4 [47].

2.3.5. Communication

All plugs and sockets, with the exception of the domestic plug, are designed to allow communication between the EV and the charging station:

- Charging station to EV – The IEC61851 standard defines a low level signalling protocol over the CP from the charging station to the EV using a $\pm$ 12 V 1 kHz PWM signal, which the duty cycle is varied to indicate the current capabilities of the charging station. The minimum current available is 6 A and the maximum is 80 A [47].

- EV to charging station – The EV can also send state information to the charging station by switching load impedances between the CP and the earth pin. The EVs charger can indicate whether it is ready for charging or that ventilation is required during the charging process [47].

Also higher layer protocols exists to allow applications related to identification, payment and value added services to be implemented in the charging station.

2.3.6. Charging schemes

The impact of the EVs on the power grid will depend on the number of EVs and the charging scheme. There are four main different schemes for charging EVs:

- Plug and charge – It is the most conventional charging scheme. The EV is plugged-in and charged as any other load [48].
- Smart charging – It is when the power grid allows the EV to charge. To make this charging scheme possible, communication between the power grid and the EV is needed. The smart grid concept with advanced metering infrastructure facilitates this application [48].
- V2G – It is the most challenging charging scheme and can be considered as an extension of smart charging. In addition to the functions of the smart charging, it also allows the energy stored in the EV batteries to be used for grid support through a bidirectional power flow [16].
- Vehicle to building (V2B) – It is similar to V2G, but instead of communicating with the power grid, the vehicle communicates with the building smart energy aggregator [16].

2.4. EV chargers

The majority of the EVs are used for transportation purposes and can be charged at different locations. Any electric car that uses batteries needs a charging system to recharge the batteries. The charging system has two main objectives:

- Charge the batteries as quickly as they allow it;
- Avoid damaging the batteries during the charging session.

Charging an EV is a big issue for EV owners. Without enough charge, they are not able to complete a planned journey. Essentially, there are two main options when it comes to charging an EV, either charging it at home or in a public charging point.

2.4.1. House and building

Home charging is related to the charging mode 2 and mode 3, which uses AC voltage that can take 4 to 11 h to fully charge a typical 24 kWh EV battery (Table 2.3). A mode 3 dedicated home charging point is the fastest possible charging speed, has built-in safety features and, if it is Wi-Fi enabled, access to additional smart features, like energy monitoring. The mode 3 chargers has a cable attached to the charging station, or a type 2 female socket, to plug-in to the EV. EVs can also be plugged into a standard 3-pin plug at home (charging mode 2), however it takes longer to charge and the sockets do not have the same safety features of a mode 3 dedicated charger [49], [50].

It is highly convenient to charge the EV at the driver workplace, because it is often parked for an extended period of time. The workplace chargers typically offer the same charging speed as home charging does and normally it has type 2 plugs (Figure 2.5). Depending on the company preference, the EV charge may be started by simply plugging in or by using an RFID swipe card or a smartphone app. The company can install higher power chargers up to 50 kW, but given the cost of high power chargers, these would usually be installed for companies that uses a fleet of EVs rather than employee cars [49], [50].

The Figure 2.12 and Figure 2.13 presents examples of wall boxes that are available to buy for domestic or building implementation.

Figure 2.12 - WALLBOXOK [49].

Figure 2.13 - KEBA 98125-98150 [50].

2.4.2. Public

The EV when parked in public locations, like retail parks, town centre car parks, etc., it is not necessary to fully charge the EV battery, but frequent charge up to a comfortable level by its user to permit the next trip, or not to wait so long until fully charge form an empty battery.

Public charging stations usually offers 22 kW of charging power and the customer needs to download a smartphone app to start charging the EV, although in some cases it is as simple as just plugging in and others charging points require a radio frequency identification card to start charging.

On long distance journeys, sometimes the remaining range in the EV battery would not be enough to complete the journey. In this scenario EV drivers can use the rapid chargers (mode 4) that offers 50 kW to 100 kW of charging power to reduce the waiting time in a charging station and to not disrupt the original travel plan. The use of this chargers is expensive, because it dispenses a lot of energy in a short period of time and the cost of such specialized charger is higher than a standard public charger, as its installation involves changes in the power infrastructure requiring new transmission, sub transmission and distribution lines. The rapid charger have, usually, a type 4 socket and if the EV does not have this type of plug, the EV are not allowed to charge [51], [52].

The Figure 2.14 and Figure 2.15 presents examples of public charging stations and cost.

Figure 2.14 – CIRCUITOR URBAN T22 [51].

Figure 2.15 - MC-QCA50 [52].

2.5. Charging strategies

There are several charging strategies that analyse how the growing number of EVs affect the electricity demand. In [53] the authors analysed the effects of the EV charging in unbalanced, residential and distribution systems. They compared uncontrolled and smart charging schemes and simulated different scenarios throughout a day, and therefore, uncontrolled EV charging adversely affects the power grid, so the controlled chargers, through smart charging approaches, is highly recommend.

Researchers from [54] develop a smart EV charging strategy based on a photovoltaic (PV) system by determining the optimal schedule for EV charging, relying on the prediction of PV power and the energy consumption. Authors assessed their approach by simulating 12 EVs with a typical battery capacity of 24 kWh, presenting three different initial SoC (state of charge) profiles (20, 30 and 40 %) and a target SoC of 80 %. Additionally, all the vehicles have a fixed charging period from 8 a.m. to 7 p.m. Their proposal is able to reduce the charging cost compared to the plug & charge scheme.

Other researchers applied an EV charging management method based on fuzzy logic controller [55]. The controller did not need communication with the infrastructure and it was used to control the EV charging rate depending on two inputs, the voltage at the point of connection and SoC of EV battery. Advantages from this approach is that the controlled charging reduces the total power demand, the maximum transformer loading during the day and improves the voltage profile.

Authors from [56] proposed an intelligent charging system aimed at minimizing the cost of the recharges. To evaluate their proposal, they simulated a single vehicle recharging at home during a week and compared it with the plug and charge scheme. However, the EVs do not charge only at 3 kW and it is going to take hours to fully charge an EV at that power rate.

Other studies promoted charging schedules to rearrange EV charges to the overnight power demand valley. In [57] is proposed a decentralized charging control strategy, specially designed for large populations of EVs with two battery capacities, 10 kWh and 20 kWh, and a maximum charging rate of 3 kW. The main goal is to reduce power generation costs by promoting charges overnight. However, EVs are not limited to a power rate of 3 kW and EV drivers will not wait 8 h to fully charge in a public station.

In [58] is proposed a distributed protocol for managing day-ahead EVs charging schedules. The scheduling horizon is from 20: 00 h to 9: 00 h and the EV can charge until reaches 10 kWh of power. Authors performed a one-day simulation considering the average residential load profile in the service area of South California, and they evaluated their proposal with 10, 20, and 40 vehicles, and assumed that an EV can be charged at any rate from 0 to 3.3 kW. However, EVs do not allow low power rates, being the minimum 2.3 kW, and do not only charge at 10 kWh.

2.6. EV charging system

The EV internal components is essential to understand how the EV charging works and to design a charging station. This is accomplished by utilizing an existing standard, known as IEC61851, which specify the EV internal components and communication, as well as, the charging station specifications that needs to be followed. The EV communication is based on a ± 12 V 1 kHz PWM signal and the duty cycle of this signal is linked to the charging current, which must not be exceeded. Therefore, the control of the EV charging is possible through the PWM signal [59].

2.6.1. Control

Nearly all EVs that has rolled off the assembly line in the last decade supports the IEC61851 standard to allow the connection between the charging station and the EV [59].

The control pilot pin (Figure 2.5) is used to communicate a series of EV states. When the EV is connected to the charging station, this enters in the charging state B, meaning that the connection has been accepted. The EV enters in the charging state C state when charging is needed or enters in a D state when auxiliary exterior ventilation is required. If an error occurs the EV enters in the charging state E or F, meaning that an error occur in the charging session. These charging states are in Table 2.5 [59].

Table 2.5 - EV states [59].

EV state	Description
A	EV Not connected
B	EV connected
C	EV charge
D	EV charge (area ventilation required)
E	Error
F	Unknown / Error

Nearly all battery management systems in the EVs have charging algorithms designed to charge the battery as quickly as possible. The charging power throughout most of the charging session will be at the maximum permitted and, because of this, the limit essentially becomes a control signal, which according to the IEC61851 standard is the duty cycle from the PWM signal. This signal can control how much power rate the EV can consume from the power grid. The EV has a maximum of 3 s to respond a PWM duty cycle change event and the Table 2.6 shows the relationship with the charging current [59], [60].

Table 2.6 - Relationship between the PMW duty cycle and the available current specified in the IEC61851 standard [60], [61].

PWM duty cycle (tolerance $\pm$ 1 %)	**Charging current**
duty cycle (%) < 3 %	0 A (charging not permitted)
3 % ≤ duty cycle (%) ≤ 7 %	Indicates that digital communication is being used to control a DC voltage charger outside the vehicle or to transmit the available current value to a charger inside the vehicle. Digital communication can also be used with other pulse duty factors. Charging is only permitted with digital communication. 5 % pulse duty factor should be used if the pilot cable is used for digital communication.
7 % < duty cycle (%) < 8 %	0 A (charging not permitted)
8 % ≤ duty cycle (%) < 10 %	6 A
10 % ≤ duty cycle (%) ≤ 85 %	$Charge\ current\ (A) = duty\ cycle\ (\%) \times 0.6$
85 % < duty cycle (%) ≤ 96 %	$Charge\ current\ (A) = (duty\ cycle\ (\%) - 64) \times 2.5$
96 % < duty cycle (%) ≤ 97 %	80 A
duty cycle (%) > 97 %	0A (charging not permitted)

2.6.2. Hardware

The Figure 2.16 shows a simple circuit diagram to understand the EV hardware interface.

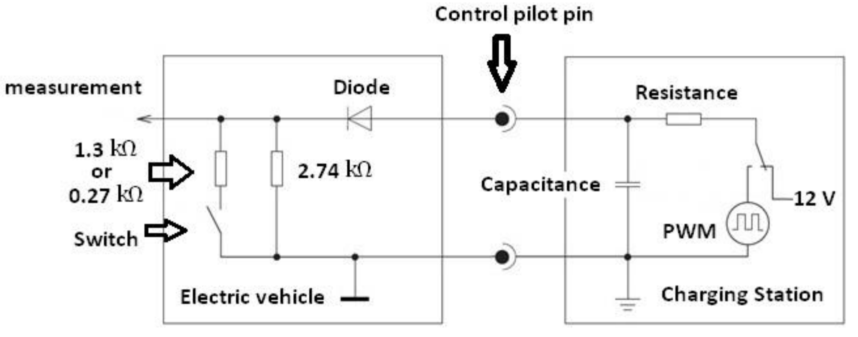

Figure 2.16 - Simplified circuit diagram of the EV hardware [62].

The charging station generates a 12 VDC signal throughout the CP pin (Figure 2.5) and when a connection is detected, this signal is changed into a ±12 V 1 kHz PWM signal. On the EV side, a load of 2.74 kΩ pulls down the $+12$ V to $+9$ V, meaning that a safe connection was establish and the EV is in state B. There is an additional load rated at 1.3 kΩ or 0.27 kΩ connected to a switch that when it closes, the EV is ready to charge and depending on the value of the load, the $+9$ V signal is pulled down to either $+6$ V or $+3$ V. This voltage indicates that the EV is in state C or D and whether or not ventilation is required during the charging session. When an error occurs the signal is 0 V or higher that -12 V, representing the EV states E or F, respectively. Table 2.7 summarises the PWM signal amplitudes, frequency and the loads in the each EV state [63].

Table 2.7 - Vehicle parameters according to IEC61851 standard [63].

EV state	Vehicle connected	Vehicle Switch	Charging possible	Pilot high (V)	Pilot low (V)	Frequency (kHz)	Resistance (kΩ)
A	No	Open	No	+12	-	-	1
B	Yes	Open	No	+9	-12	1	2.74
C	Yes	Closed	Vehicle ready	+6	-12	1	1.3
D	Yes	Closed	Vehicle ready (area ventilation required)	+3	-12	1	0.27
E	Yes	Open	No	0	0	-	-
F	Yes	Open	No	-	-12	1	-

2.7. Charging station

A charging station is an element that supplies power to charge an EV. Figure 2.17 shows a simple diagram of the charging station main components present in its design, which are a power metering device (1), a control module (5), a switching device (4) and power protection devices such as a circuit breaker (3) and a ground fault circuit breaker (2).

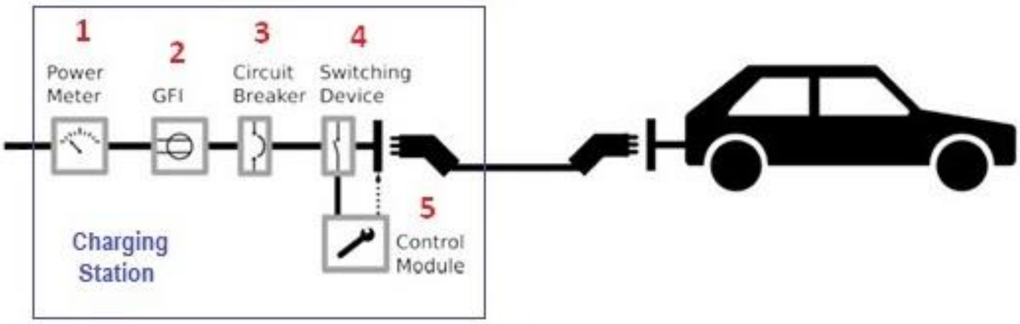

Figure 2.17 - Diagram of the charging station main components [63].

To permit the EV communication a specific plug type is required, such as the type 2 plug (Figure 2.5). With this, the only charging mode permitted is the mode 3 (Figure 2.10) and the power delivery is through AC.

One of the important aspects of the charging station control module is to follow the IEC61581 standard that has requirements to allow an EV communication. One of that requirements is the PWM signal specification, which allows the charging station to communicate with the EV. The other one is the cable requirements to charge an EV regarding the charging station maximum power delivery.

2.7.1. Communication requirements

The communication with the EV is based on a ± 12 V 1 kHz PWM signal that is sent over a dedicated communication line called CP shared by the charging station and EV. According to the IEC61851 standard, the communication signal has requirements to be follow such as amplitude, frequency levels and the equivalent source resistant, as well as the capacitance [64]. Table 2.8 shows these requirements.

Table 2.8 - EV communication requirements according the IEC61851 standard [64].

Parameter	Unit	Nominal value	Maximum value	Minimum value
Voltage parameters				
Positive open circuit	V	+12.00	+11.40	+12.60
Positive voltage state B	V	+9.00	+9.56	+8.36
Positive voltage state C	V	+6.00	+6.49	+5.48
Positive voltage state D	V	+3.00	+3.25	+2.62
Negative voltage state B, C, D, F and open circuit	V	-12.00	-12.60	-11.40
Resistance and capacitance parameters				
Equivalent source resistance	Ω	1000	1030	970
Total equivalent charging station capacitance without cable	pF	n.a.	n.a.	300
Total equivalent charging station capacitance including cable	pF	n.a.	3100	n.a.
Frequency parameters				
Frequency	Hz	1000	1050	950
Pulse width	µs	Value (between 0 – 100)	Value (between 0 – 100) + 25	Value (between 0 – 100) + 25
Rise time	µs	n.a.	2	n.a.
Fall time	µs	n.a.	2	n.a.
Settling time	µs	n.a.	3	n.a.

With the PWM signal requirements, the charging station can control the power delivery to the EV through the PWM duty cycle (Table 2.6) by changing it. This control can be made remotely by an external command using, for example, an internet connection or locally.

2.7.2. Cable requirements

The IEC61851 standard specify as well the charging cable power capacity to be checked by the charging station [63]. Table 2.9 shows the cable capacity ratings.

Table 2.9 - Cable current carrying capacity according to IEC61851 standard [63].

Resistance value (Ω)	Cable current carrying capacity (A)	Cable power capacity (kW)	
		Single - phase system	Three - phase system
< 100	0	-	-
100	63	14.4	43.4
220	32	7.3	22
680	20	4.6	13.8
1500	13	2.9	8.9
> 1500	0	-	-

The cable is accepted if the rating is equal or superior to the maximum power capability of the charging station, otherwise charging is not permitted. The cable rating is checked through the resistance value that can be verified in the PP pin (Figure 2.5) [63].

2.8. Discussion and conclusions

The primary function of the EV is to provide the mobility service expected by its user. This means that EVs has to obtain energy from the grid and store it to use while in motion.

It is known that, at high penetration levels, uncontrolled EV charging can congest the grid operation, obstructing the lines, transformers and cause voltage deviations in the distribution system. Similarly, at low penetration levels, such as household level, the uncontrolled charging can lead to power outages due to the high power demand of the EV charge. Smart chargers can impose the power rate and be responsible to control the EV charge, leading to an improvement of the charging congestion problems.

The most common charging scheme for charging EVs is the plug and charge. The charge process starts immediately when the user arrives at the charging station and plugs-in the EV to charge as a fixed load. However, this scheme is not really efficient, since it does not rely on any energy efficiency or cost reduction parameters (e.g. electricity pricing, the current state of the power grid or building/house and the current battery level).

By virtue of being equipped with a battery, a layer of communication exists between the charging station and the EV. The EVs are potentially capable of having their power requirements being served according to a pre-established schedule or value, or even injecting energy back into the power grid, in cases where the energy system demand is higher.

Depending on the charging infrastructure, the EV charging station can allow conductive or inductive charging. The conductive charging is the most common method used in the charging stations. Depending on the charging mode, the EV can charge from 2.3 kW to 100 kW. The most common charging station implements the charging mode 3, representing fast charging with a power delivery up to 50 kW, and a type 2 connector, which is the most connector used in Europe.

The EV charging stations on the market for building and house do not consider energy management algorithms to control the EV charging. The charging stations only monitors the energy consumption and schedule the EV charging. To provide an EV charging solution that considers the power consumption status, a charging station has to be developed.

The IEC61851 standard does not dictate that the EV should charge at the maximum power rate permitted by the charging station. The EVs are capable of having their power requirements served according to a power rate value that is defined by the PWM duty cycle, which is sent to the EV by the charging station.

Overall, most of the charging strategies regarding EVs focused on: improving the charging capabilities, optimizing the SoC, reducing power demand peaks and exploiting the overnight demand valley filling, as well as the effects of EVs increasing power demand. Additionally, and similarly to other researches, most of these charging strategies relied on simulations to assess their proposals that does not meet with the EV charging session requirements and specifications.

Chapter 3

Design of the EV charging system

This chapter presents the EV charging system to handle the EV charging based on a server connection with a web interface. This system has a charging station, a coordinator module and a server.

3.1. System description

The system is composed by a charging station, which is responsible for the EV communication, a coordinator module, which is responsible to send and receive data from the server and the charging station, and a server, which is responsible to manage the EV charging session by applying energy management algorithms. Figure 3.1 shows the charging system.

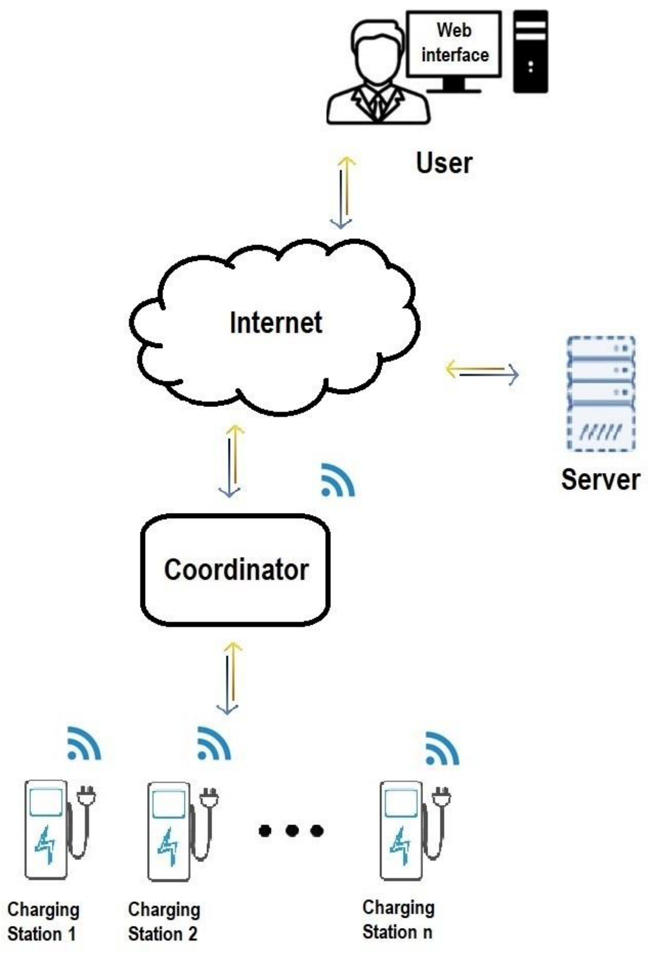

Figure 3.1 - Diagram of the EV charging system.

All the processing data is performed by the server to control the EV charging, which is managed by the PWM duty cycle, and take into account the maximum power that the charging station can supply to the EV.

The charging system coordinator coordinates the connectivity between the charging station and the server by aggregating all the power information data and send it to the server to receive all the PWM duty cycles to be applied on the charging stations.

The main module in the charging station is the control unit, which is responsible to interact with all the hardware modules and follow the EV communication standard protocol IEC61851 as part of the prototype requirements.

3.2. Coordinator

The coordinator (Figure 3.2) is responsible to aggregate all the charging station data and send it to a server to receive the PWM duty cycle to send it back to the charging station. All of this interactions are made by a Raspberry Pi 3 B+, which is a small computer capable to use a wireless communication and store all the data in a local database.

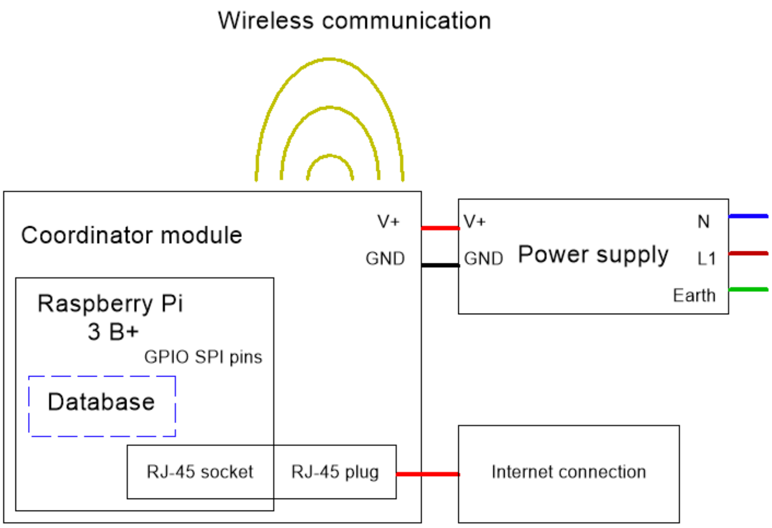

Figure 3.2 - Diagram of the coordinator components.

Every minute an internet connection is establish to send and receive data from the server to execute on the charging station. The data sent contains the EV state, EV connectivity, charging station identification number and power information variables.

Figure 3.3 shows the flowchart of the coordinator operating cycle and the implemented code is in Attachment A.

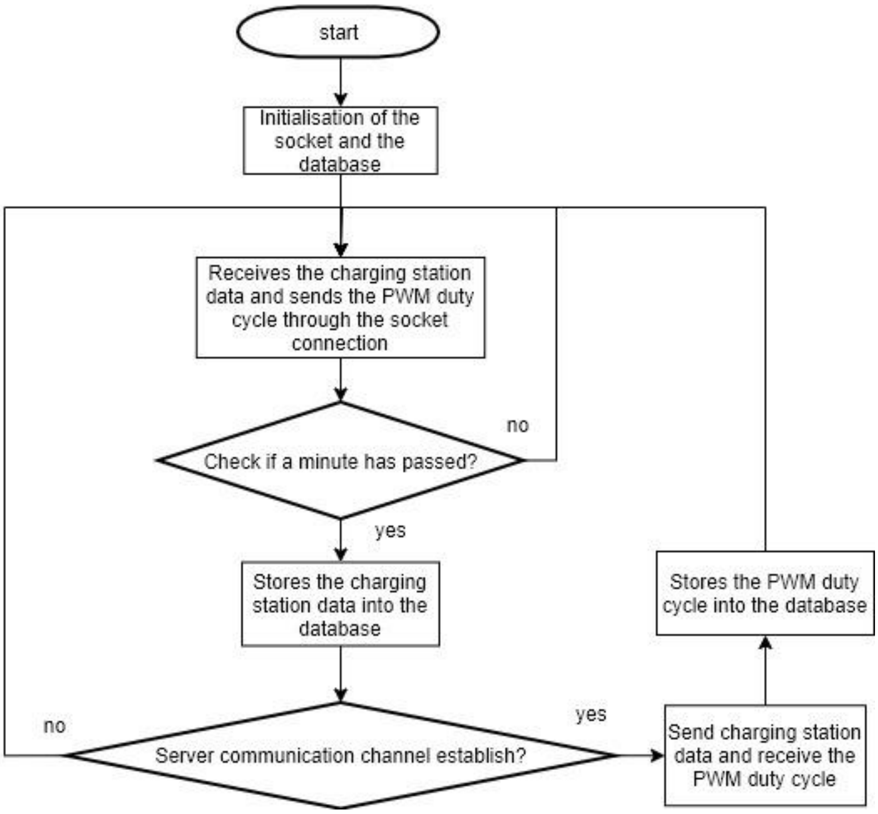

Figure 3.3 - Flowchart of the coordinator operating cycle.

3.2.1. Database

On the Raspberry Pi 3 B+ a local database was implemented called MariaDB, which is a derivation of a MySQL database. MySQL is an open source database management system that uses the SQL (Structured Query Language) language as interface. MariaDB is, as well, developed as an open source software turning data into structured information in an array [65].

The database has two tables, one for the EV connectivity, EV state and power information variables, and the other has the PWM duty cycle values. To access each table is by the charging station identification number and they are updated when the coordinator receives data from the server or the charging station. However, when the EV connectivity, EV state and power information variables are sent, the data is deleted to prevent the overlay of data.

3.2.2. Wireless communication

The wireless communication is accomplish by using the Raspberry Pi 3 B+, which is capable to send and receive data through a socket connection establish in the local area network created with the charging station. A socket is a two-way communication link between two programs running on a computer network. One socket listens on a particular port and IP address, while another socket connects to achieve communication.

The way that sockets send data are controlled by two properties. The address family, which determines the network layer protocol used and the socket type, which determines the transport layer protocol. Depending on the transport layer protocol, sockets can either be for message-oriented datagram transport, often associated with the user datagram protocol, or stream-oriented transport, often associated with the transmission control protocol [66].

On the Raspberry Pi 3 B+ was installed a library named socket to create a socket connection using the user datagram protocol. From this library six functions were used:

- socket (socket.AF_INET, socket.SOCK_DGRAM) – Create the user datagram socket;
- socket.timeout (float number) – Set a socket timeout to prevent a socket block,
- socket.bind ((IP, port number)) – Bind the socket to the port number;
- socket.setsockopt (socket.IPPROTO_IP, sockey.IP_ADD_MEMBERSHIP, struct.pack (‘4sL’, socket.inet_aton (IP), socket.INADDR_ANY)) – Set on all interfaces of the operating system the socket connection;
- socket.sendto (data, (IP, port number)) – Send data to the device;
- socket.recvfrom (maximum bytes to read) – Receive the data from the device.

To implement the socket library a function named “socket_connection ()” was created to send and receive data. When called, this function creates the socket to send the PWM duty cycle to the charging station and receive its identification number, EV state, EV connectivity and power information variables to store it in the database and then closes the socket. Figure 3.4 shows the operating function

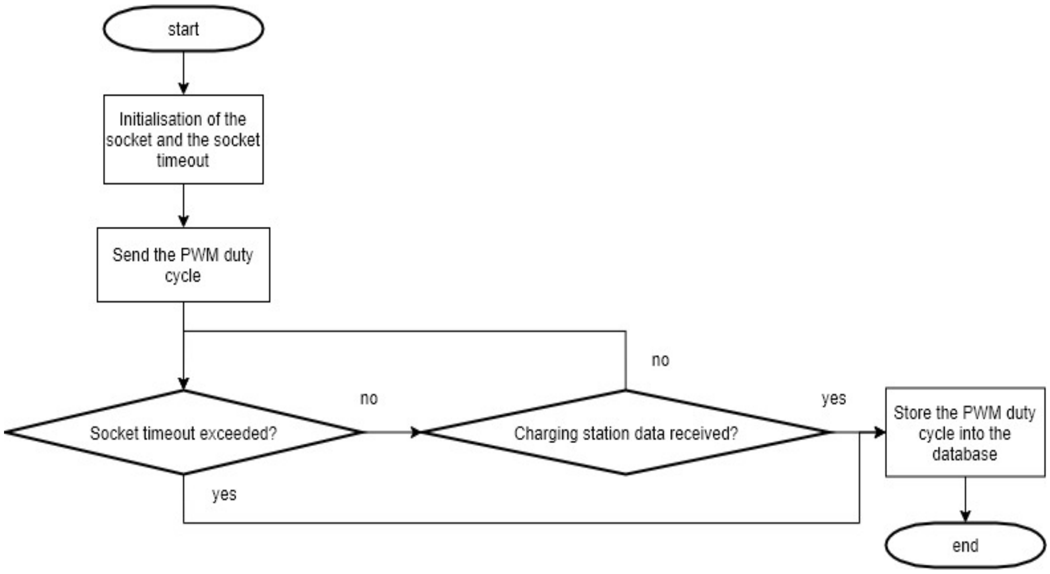

Figure 3.4 - Flowchart of the wireless communication operating function of the coordinator.

3.2.3. Server communication

A communication channel was created, between the coordinator and the server, based on the hypertext transport protocol (HTTP) protocol using the POST request method, which is used to send data to a server. Its body is a block of data sent with the request. There are extra headers to describe the message body, such as the Content-Type, which is an application of a uniform resource locator (URL) encoded form, and the Content-Length, which gives the length of the URL encoded form data [67].

On the Raspberry Pi 3 B+ was installed a library named request to access the HTTP communication channel. From this library a function was used:

- POST (server URL, data, headers = headers, timeout = 10) – Send the data to the server.

This function is called every minute to make a request to the server to send all the charging station data information present in the database. When the request is accepted, the data is sent and the response is the PWM duty cycle that is store in the database to be sent to the charging station.

3.3. Charging station

The charging station allows 22 kW inductive charging using a type 2 plug connection (Figure 2.5), meaning that the only charging mode allowed is 3 (Figure 2.10). In side of it sits an energy meter, Raspberry Pi 3 B+, electrical protections, a control unit, a power supply and a contactor, as showed in Figure 3.5.

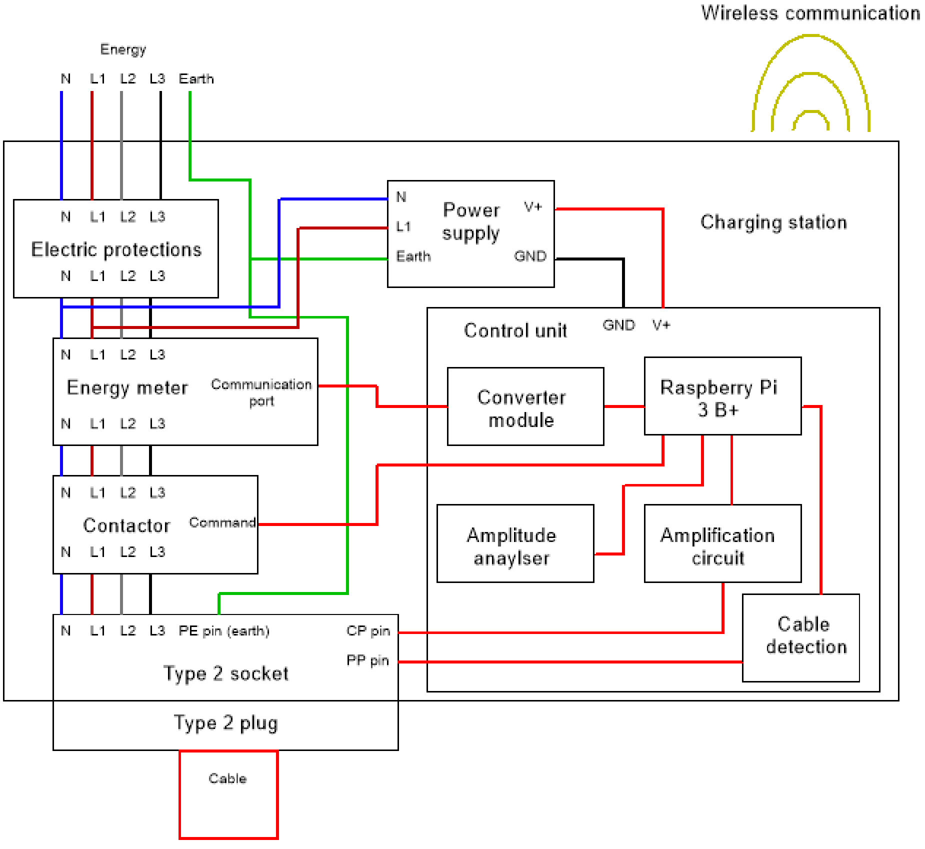

Figure 3.5 - Charging station diagram.

The control unit was design to support the different communication protocols used to communicate with the different equipment's. To generate the IEC61851 standard PWM signal a Raspberry Pi 3 B+ was used, which provides a set of GPIO (general purpose input/output) pins to manage the EV connection by sending the charging station identification, EV connectivity and power information through a socket connection establish in the local area network created with the coordinator to receive the PWM duty cycle.

The EV communicates a series of states through the PWM signal, if it is connected, ready to charge or an error occur. Depending on the EV state, a relay is needed to allow or not the power transfer between the EV and the charging station. However, the EV charging it is only allowed if the charging cable meets the IEC61851 standard requirements for a 22 kW charging station, which can supply a maximum current of 32 A.

Figure 3.6 shows the flowchart of the operating cycle of the charging station and the implemented code is in Attachment B.

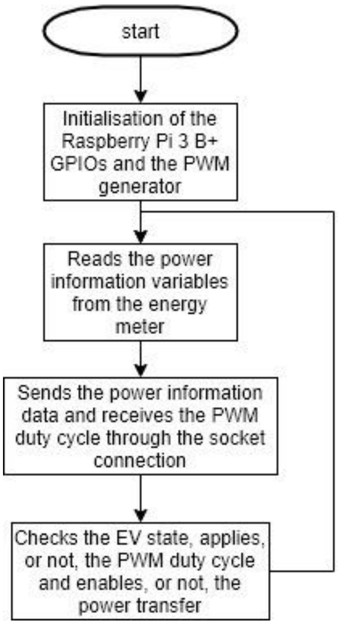

Figure 3.6 – Flowchart of the charging station operating cycle.

3.3.1. Socket connection

To be able to communicate with the coordinator, a socket connection similar to the coordinator was implemented on the charging station. When the energy meter provides the power information, the control unit sends it through the socket connection, as well as the charging station identification, EV connectivity and state. The data socket received is applied on the PWM signal to control the EV charging.

On the Raspberry Pi 3 B+ was installed a library named socket to implement a socket connection using the user datagram protocol. From this library the following functions were used:

- socket (socket.AF_INET, socket.SOCK_DGRAM) – Create the datagram socket;
- socket.timeout (float number) – Set the socket timeout to prevent a socket block,
- socket.setsockopt (socket.IPPROTO_IP, sockey.IP_MULTICAST_TTL, struct.pack (‘b’, 1)) – Set the time-to-live for messages to 1 to get all the messages so they do not go past the local network;
- socket.sendto (data, (IP, port number)) – Send data to the device;
- socket.recvfrom (maximum bytes to read) – Receive the data from the device;
- socket.close () – Close the socket.

To implement the socket library a function named “socket_connection ()” was created to send and receive data. When called, it sends the charging station id, EV state, EV connectivity and power information to receive the PWM duty cycle to be applied on the PWM signal. Figure 3.7 shows the operating function.

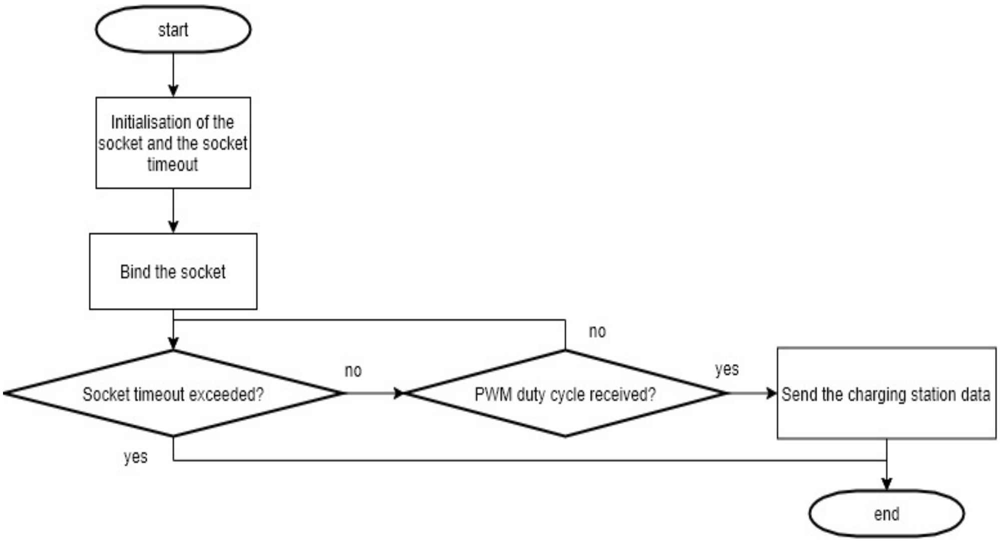

Figure 3.7 – Flowchart of the wireless communication operating of the charging station.

3.3.2. Cable detection

When connecting an EV charging cable on the charging station type 2 plug, a cable detection circuit, as showed in Figure 3.8, was design to check if the cable meets the IEC61851 requirements (Table 2.9).

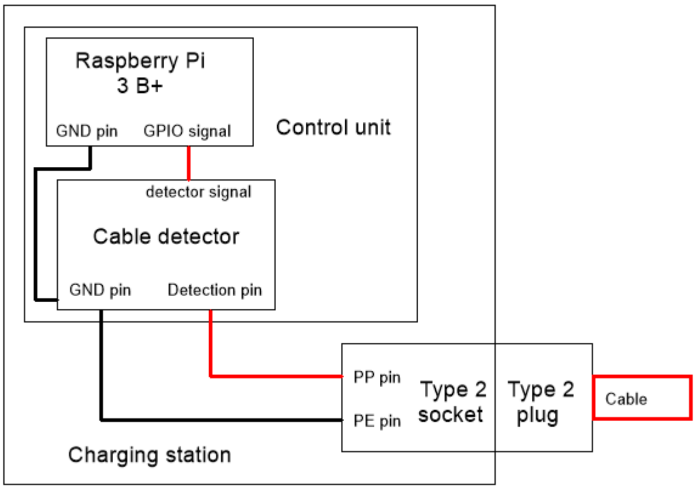

Figure 3.8 - Cable detection diagram.

The cable detector circuit detects if the cable can support a current carrying capacity between the 32 A and 63 A. The Raspberry Pi 3 B+ receives a signal through the GPIO if the cable is accepted or not. Only with the cable acceptance the charging station can start the EV charging session. Figure 3.9 shows the cable detector diagram.

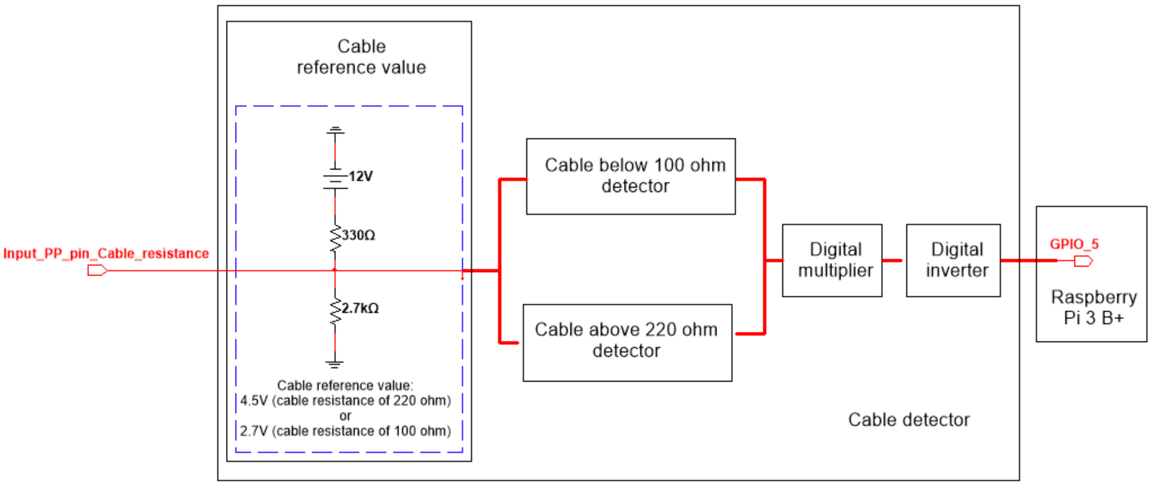

Figure 3.9 - Cable detector diagram.

The cable type 2 plug (Figure 2.5) has the proximity plug pin connected to the earth pin through a resistor. This resistor was converted into a voltage reference value by connecting the proximity plug pin to a voltage divider to set a threshold value for the cable acceptance. The cable detector circuit has two cable detectors, one for 100 Ω resistor and other for 220 Ω resistor. On the first one (Figure 3.10), the cable reference value (2.7 V) is compared to a reference value of 2.1 V. If the cable value is below the 2.1 V, the compare circuit sends a digital LOW signal (0 V) throughout the GPIO, otherwise sends a digital HIGH signal (3.3 V).

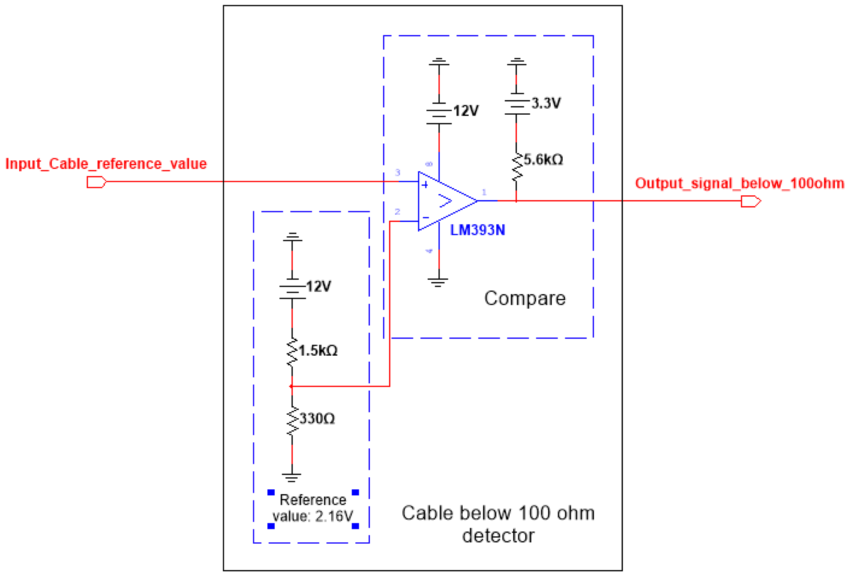

Figure 3.10 - Cable detector circuit for a 63 A current carrying capacity.

On the second one (Figure 3.11), the cable reference value (4.5 V) is compared to 5.33 V. If the cable value is above the 4.5 V, the compare circuit sends a digital LOW signal throughout the GPIO, otherwise sends a digital HIGH signal.

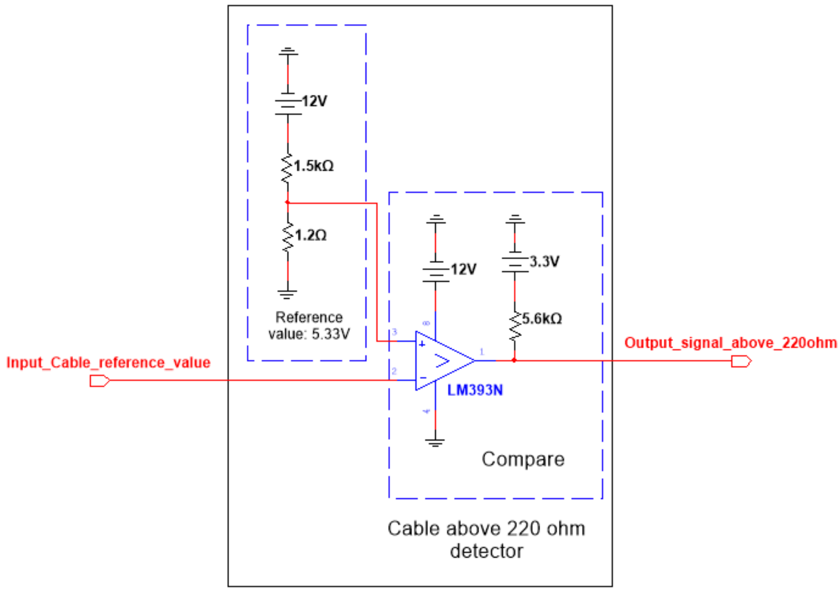

Figure 3.11 - Cable detector circuit for a 32 A current carrying capacity.

The output signals from the cable detectors circuits are inputs in a digital multiplier circuit (Figure 3.12). This circuit has a transistor in a logic NAND configuration, meaning that the output is going to be the negative multiplication of the digital inputs signals. If the current cable requirements are between the 32 A (cable resistance of 100 Ω) and 63 A (cable resistance of 220 Ω), the output signal is a digital LOW signal, otherwise is a digital HIGH signal.

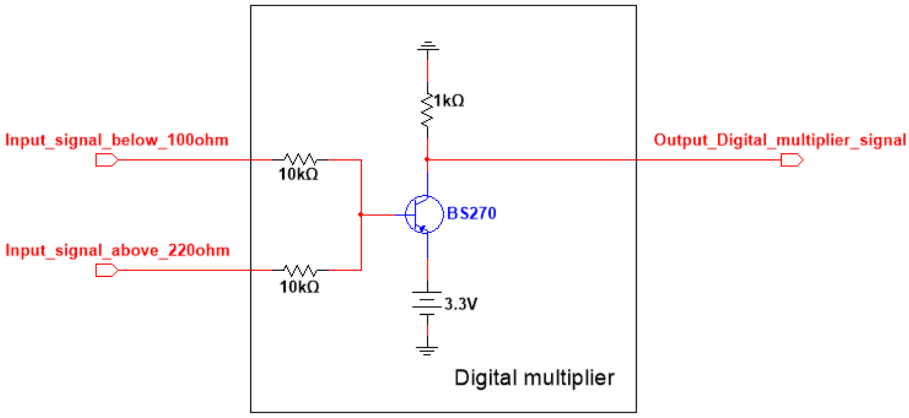

Figure 3.12 - Digital multiplier circuit to check the 32 A and 63 A current cable requirements.

However, the digital multiplier signal is connected to a digital inverter circuit (Figure 3.13), which has a transistor in a logic NOT configuration to invert the output signal. When the digital multiplier signal provides a digital LOW, the output is a digital HIGH, and vice-versa, meaning that the Raspberry Pi 3 B+ GPIO is a digital HIGH signal when the current cable requirements are between 32 A and 63 A, otherwise is a digital LOW signal when the cable does not meet the requirements.

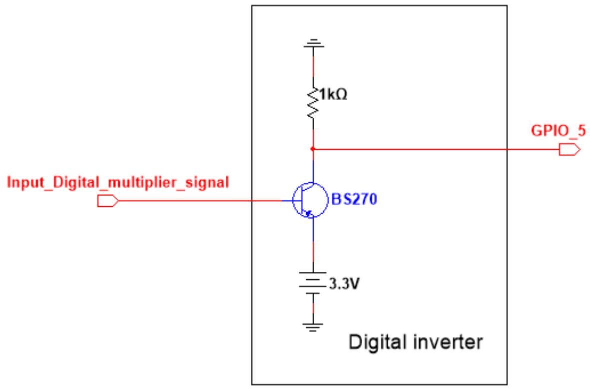

Figure 3.13 - Digital inverter circuit to invert the input signal.

3.3.3. Charging control

The EV communication is based on the IEC61851 standard and to be able to control the EV charging the Raspberry Pi 3 B+ has to provide all the interactions needed and Figure 3.14 shows all the modules needed for that interactions.

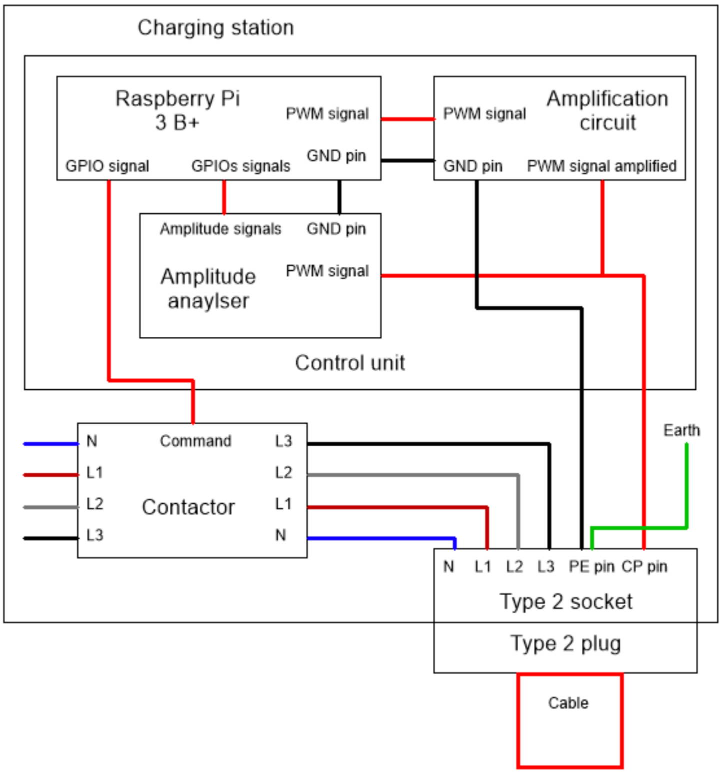

Figure 3.14 - Charging station control unit diagram.

To be able to connect to the EV a ±12 V 1 kHz PWM signal is required and the Raspberry Pi 3 B+ cannot provide it directly, meaning that adding an amplification circuit, the signal can be made with the IEC61851 standard specifications. To know which state the EV is, it was design an amplitude analyser to analyse the amplitude of the PWM signal shared between the amplification circuit and the EV to act according to each state. If the EV state changes from B to C, it means that the charging can start and the control unit sends a command signal to the relay circuit to allow the power transfer. The state C, means that the

EV is charging and the power consumption can be control by the PWM duty cycle (Table 2.6). However, if the EV state changes from C to B, it means that the EV does not need to charge anymore and a command signal is sent to the relay circuit to shut off the power transfer.

The power transfer can be cancelled when the EV is in state C by the PWM duty cycle. If the duty cycle is between 0 to 3 %, 7 to 8 % or 97 to 100 %, the EV is not allowed to charge and the EV state is going to change to B, meaning that the control unit is going to shut down the power transfer by sending a command signal to the relay circuit. This signal is sent, as well, when the EV state is E and F because an error occur in the charging station or in the EV during the EV charging session.

3.3.3.1. PWM signal

On the Raspberry Pi 3 B+ was installed a library named *pigpiod* to generate the PWM signal. From this library the following functions were used:

- pigpio () – Enable the PWM generator;
- hardware_PWM (GPIO number (int), frequency in Hz (int), duty cycle in µs x 10^4 (int)) – Generates the PWM signal.

However, the signal generated was a +3.4 V 1.000 kHz PWM signal (Figure 3.15), meaning that the frequency is on the 1000 Hz and rise, fall and settling time does not need to be verified according to the Table 2.8. However, the amplitude of the signal does not meet the IEC61851 standard requirements and needs to be amplified to ±12 V.

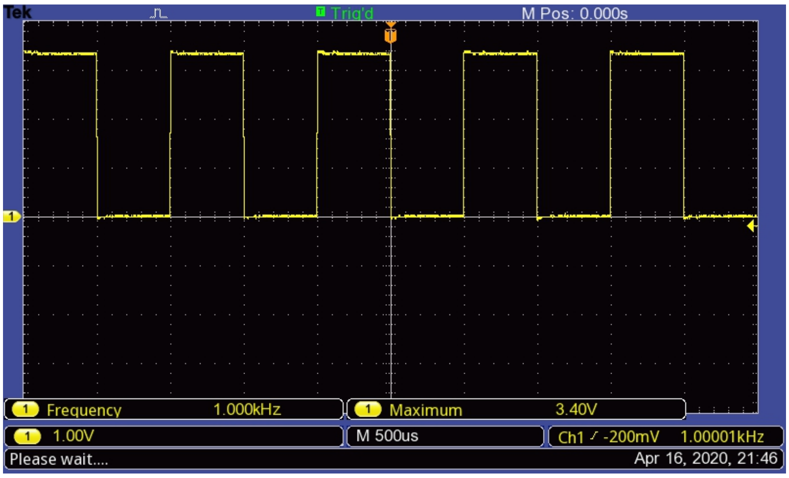

Figure 3.15 - Raspberry Pi 3 B+ PWM signal.

3.3.3.2. Amplification circuit

The amplification circuit was design to receive the Raspberry Pi 3 B+ 3.4 V 1 kHz PWM signal and amplify the signal to a ±12 V 1 kHz PWM signal with an equivalent source resistance of 1 kΩ, according to the IEC61851standard (Table 2.8).

Figure 3.16 shows the amplification circuit where the input PWM signal was amplified to ±15 V. However, the voltage levels were too high and a set of diodes were introduce to pull down the voltage to ±12 V.

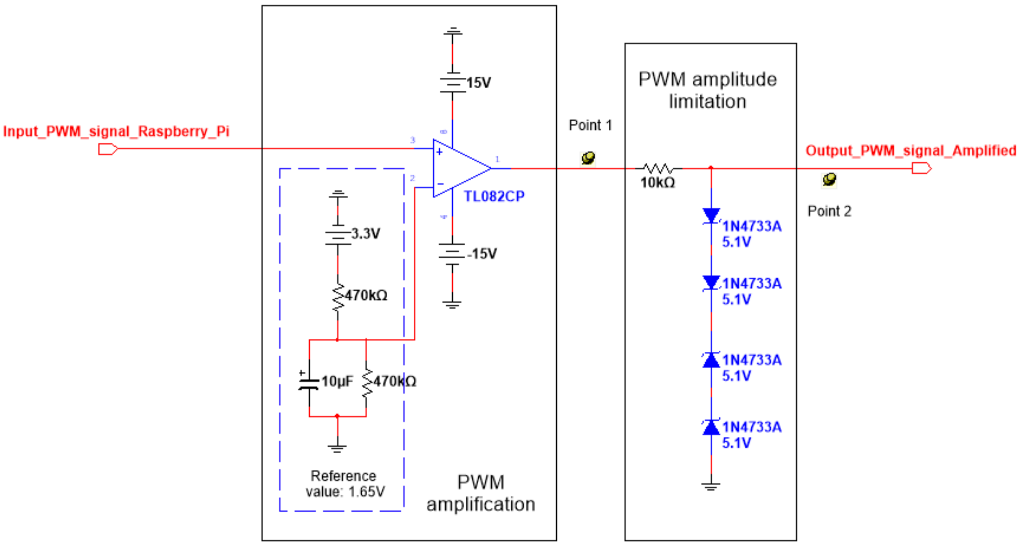

Figure 3.16 - PWM amplification and limitation circuit.

Because of the amplitude limitation, the circuit has an equivalent resistance of 10 kΩ and, to meet the standard, an isolation circuit was added to provide the 1 kΩ equivalent source resistance, as showed in Figure 3.17.

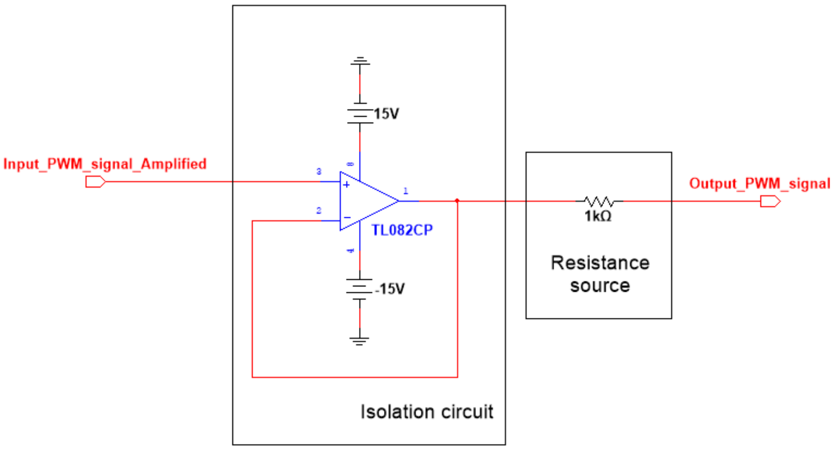

Figure 3.17 - Isolation circuit with the 1 kΩ source resistance.

The circuit was implemented and the resistance use has 1% of error, meaning that the source resistance is between 990 Ω and 1010 Ω. With this, the circuit capacitance does not need to be consider and the output signal (Figure 3.18) provided by the circuit was a +12.4/−11.6 V 1 kHz PWM signal, which is according with the IEC61851 standard shown in Table 2.8.

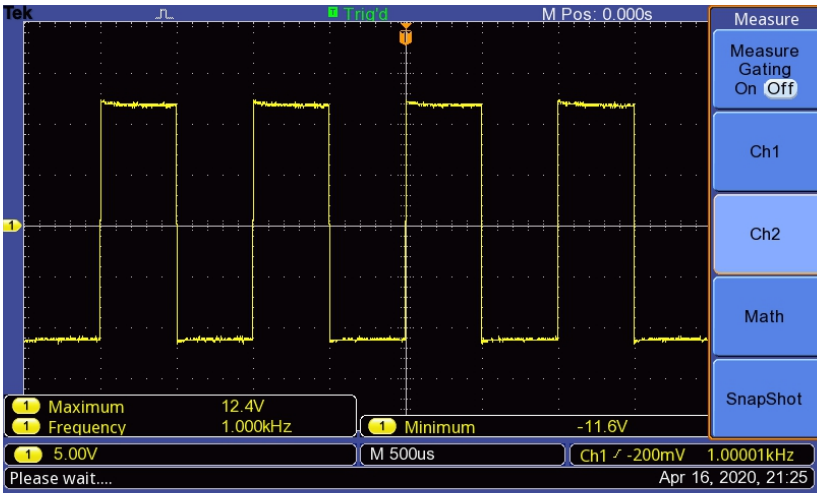

Figure 3.18 - Control unit PWM signal generated.

3.3.3.3. Amplitude analyser

To know the EV state an amplitude voltage analyser circuit was design to connect to the GPIOs of the Raspberry Pi 3 B+ and communicate the EV state. This circuit was implemented after the 1 kΩ source resistance with another isolation circuit to prevent the source resistance change, as showed in Figure 3.19.

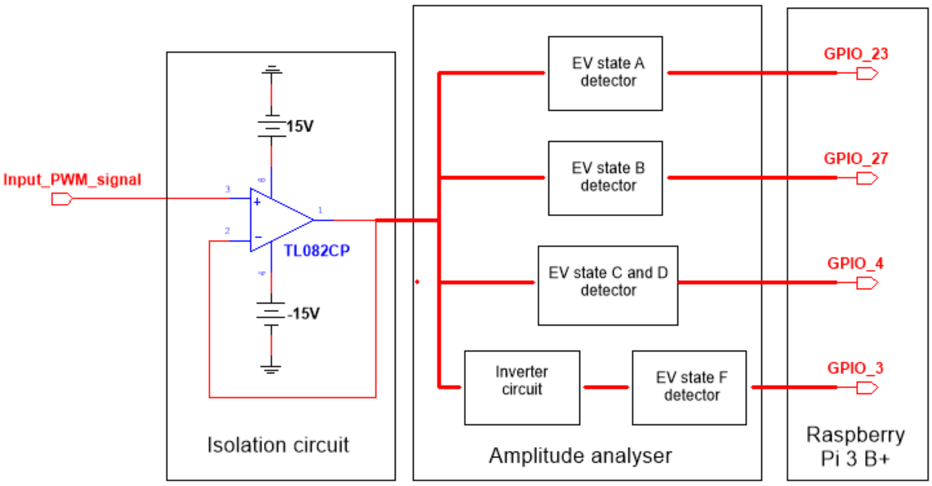

Figure 3.19 - Amplitude analyser diagram.

Figure 3.20 shows the EV state A detector circuit, which detects the state A (+12 V). When the positive PWM amplitude is converted into a DC signal by the filters, the signal is compared with a reference value of 10.66 V. If the signal is above the reference value, the amplitude compare sends a digital HIGH signal throughout the GPIO, meaning that the EV state is A, otherwise sends a digital LOW signal, meaning that the EV is in other state.

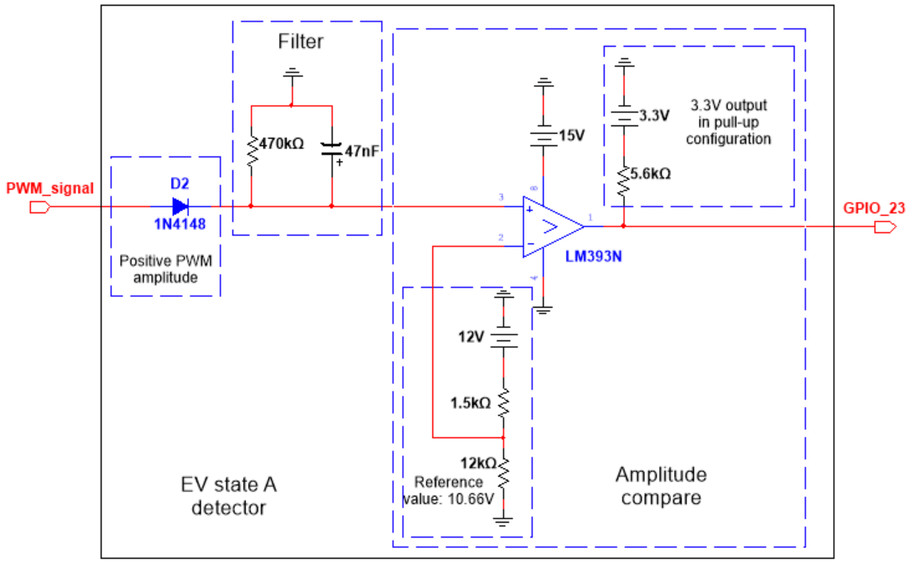

Figure 3.20 - EV state A detector circuit.

To detect the others EV states, the EV state A detector circuit was replicated with a different reference value and Raspberry Pi 3 B+ GPIO. The Figure 3.21 shows the EV states PWM signals with the reference values in red colour. When the positive PWM amplitude is above 7.2 V the EV is in state B, meaning that the amplitude compare sends a digital HIGH signal throughout the GPIO. The same thing happens to detect the C and D state, if the signal is above 1.56 V.

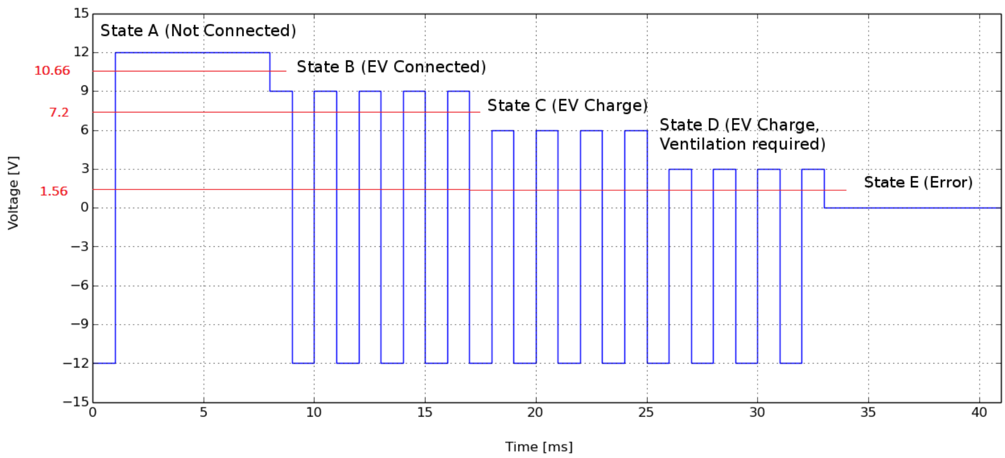

Figure 3.21 - EV state amplitudes and levels detection [62].

The amplitude detection of the EV state F is not possible by the amplitude compare and filter circuit, because it only compares positive PWM amplitude. Adding an inverter circuit (Figure 3.22) before the EV state F detector to invert the negative amplitude (−12 V) of the PWM signal into a positive amplitude (+12 V) to be compared makes it possible to detect the negative amplitude and the reference value is the same as the state A detector, meaning that if the signal amplitude is above the 10.66 V the amplitude compare sends a digital HIGH signal throughout the GPIO.

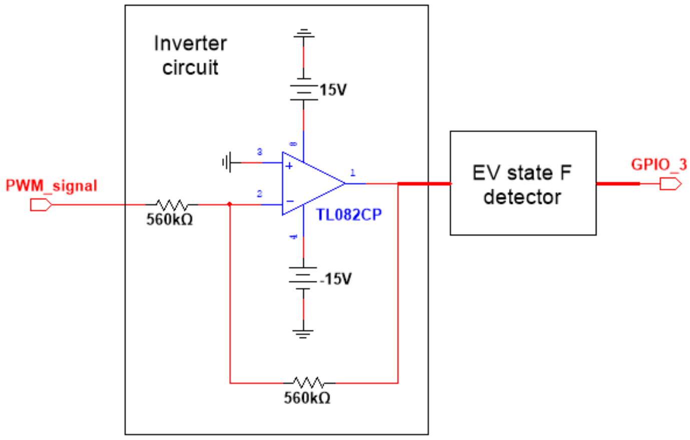

Figure 3.22 - Inverter circuit and the EV state F detector.

On the Raspberry Pi 3 B+ a function named “EV_State ()” was created to verify the EV state and, depending on the state, apply the PWM duty cycle to allow, or not, the power transfer. Figure 3.23 shows the operating function.

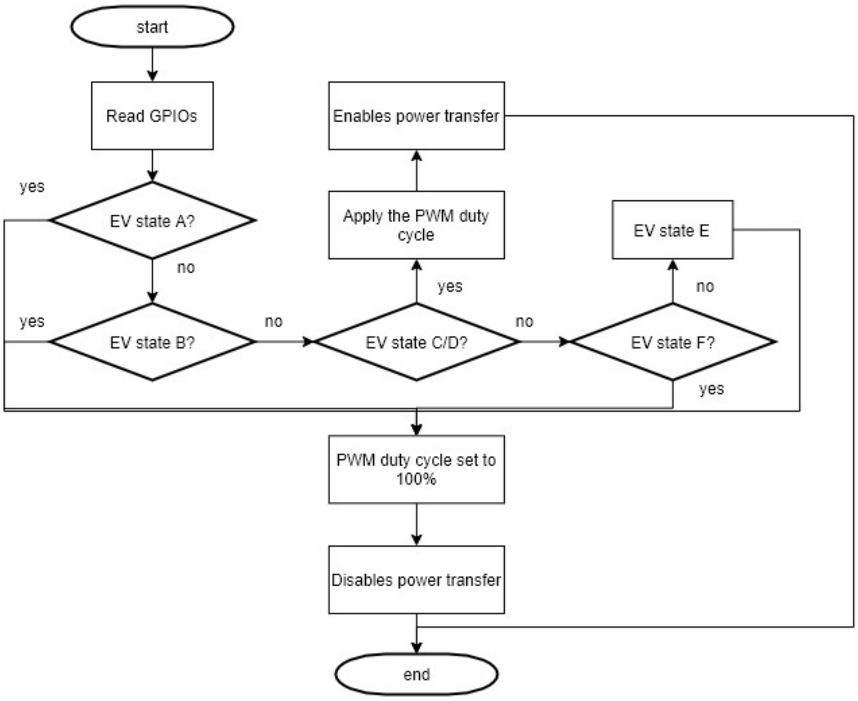

Figure 3.23 – Flowchart of the EV state detection.

3.3.4. Relay circuit

To allow the power transfer between the EV and the charging station, a relay circuit (Figure 3.24) was designed to send a 230 VAC command signal to the contactor (Figure 3.25). This circuit has the Raspberry Pi 3 B+ GPIO 17 connected to a transistor, acting as a switch, to activate, or not, the contactor. When activated the contactor enables the power transfer, otherwise is deactivated.

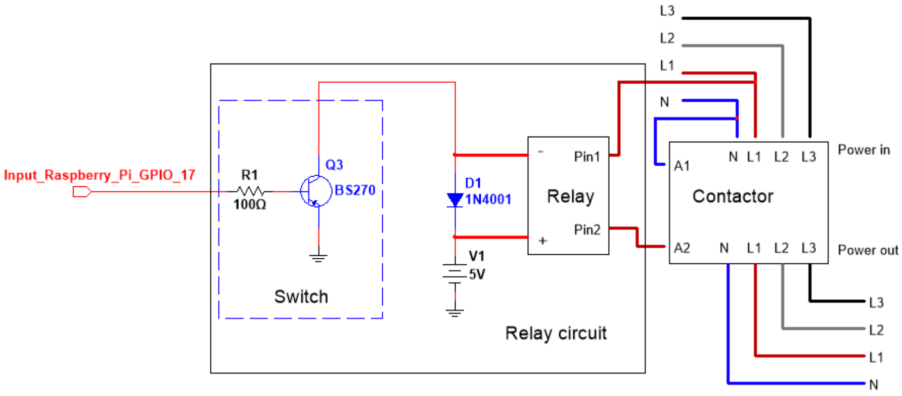

Figure 3.24 - Relay circuit and contactor.

Figure 3.25 - EUC3-63-4P contactor [68].

3.3.5. Energy meter

In the charging station has an energy meter named EM340 (Figure 3.26). Upon receiving a control unit command to retrieve the power information, the meter returns its voltage, current, active power, reactive power, apparent power, power factor and frequency, using the Modbus communication protocol.

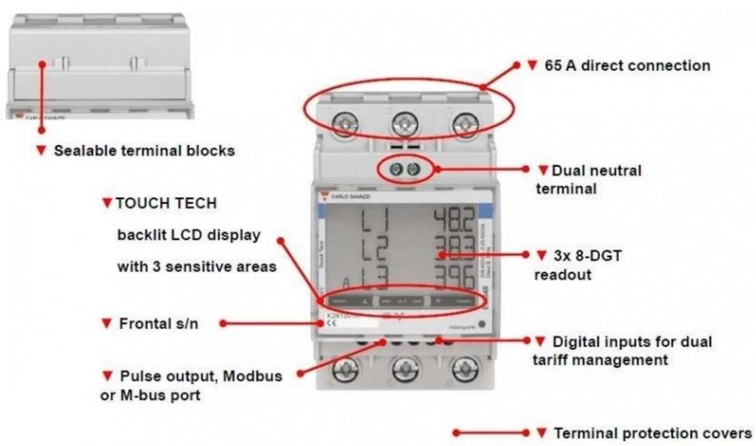

Figure 3.26 - EM340 energy meter [69].

Modbus is an open source protocol that has become a standard communications protocol in the electronics industry and is used to transmit signals, or data gathering, from instrumentation and control devices back to a main controller. The Modbus messages correspond to a simple operation of read and write 16 bit words and binary registers. Versions of the Modbus protocol exist for serial communication (Modbus RTU and ASCII) and for Ethernet communication (Modbus TCP). In the Modbus ASCII the messages are exchanged as lines of hexadecimal codes. On the other hand, in the Modbus RTU the messages are exchanged directly as binary frames [70].

The communication protocol supported by the EM340 is the Modbus RTU. However, this protocol does not specify any physical connection and to access the meter power information is throughout the RS485 interface, which is a physical layer that specifies the electrical characteristics for the connection [71].

The connection (Figure 3.27) between the energy meter and the Raspberry Pi 3 B+ is made by a converter module named J4H-HV-TRM-RTC-485 (Figure 3.28), which converters the Raspberry Pi serial connection into a RS485 interface to be able to access the meter power information.

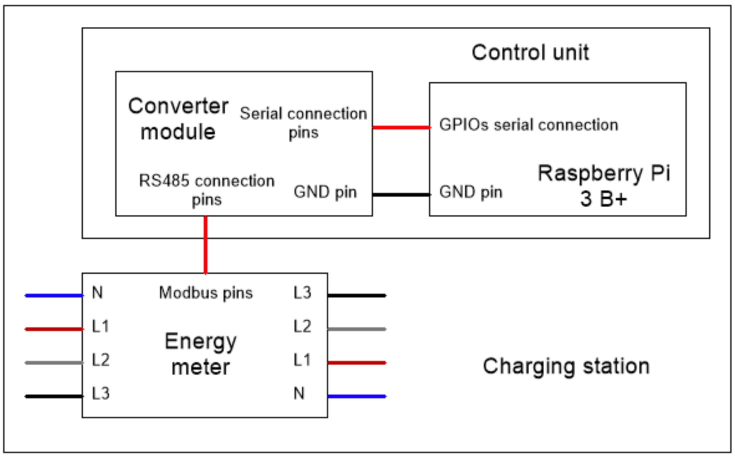

Figure 3.27 – Control unit diagram with the Energy meter.

Figure 3.28 – J4H-HV-TRM-RTC-485 converter module [72].

To access the power information variables, such as voltage, current, active power, power factor and frequency, on the EM340, it is necessary to know the decimal number of the physical address of each variable, as showed in the Table 3.1.

Table 3.1 – EM340 corresponding physical address to each power information variable [71].

Decimal	Physical address	Length (words)	Variable	Data format	Notes
0	0000h	2	V L1-N	INT32	$Volt \times 10$
2	0002h		V L2-N		
4	0004h		V L3-N		
12	000Ch		A L1		$Ampere \times 1000$
14	000Eh		A L2		
16	0010h		A L3		
18	0012h		W L1		$Watt \times 10$
20	0014h		W L2		
22	0016h		W L3		
40	0028h		W ∑		
42	002Ah		VA ∑		$VA \times 10$
44	002Ch		VAR ∑		$var \times 10$
46	002Eh	1	PF L1	INT16	$PF \times 1000$ (positive value correspond to imported active power and negative value to exported active power)
47	002Fh		PF L2		
48	0030h		PF L3		
51	0033h		Hz		$Hz \times 10$

With this, on the Raspberry Pi 3 B+ was installed a library named pymodbus to access the power information variables through the corresponding physical address decimal number. From the library the following functions were used:

- ModbusClient (method = "rtu", port = (int), baudrate = 9600) – Set the Modbus communication;
- Read_holding_registers (decimal number of the physical address (int), number of registers to read (int) +1, unit = meter id (int)) – Request the variable data present in the meter and returns the data as a pack of two bytes, which the first byte contains the high order bits and the second the low order bits;
- BinaryPayloadDecoder (packed data ([int]), byteorder = Endian.Big, wordorder = Endia.Little) – Decode the byte order to represent the packed data as values of integers or floating numbers;
- decode_16bit_int () – Decode data format INT16;
- decode_32bit_int () – Decode data format INT32.

A function named “Smart_meter ()” was created to implement the pymodbus library functions. When it is called, returns an array with all the requested values of the power information variables. Figure 3.29 shows the operating function.

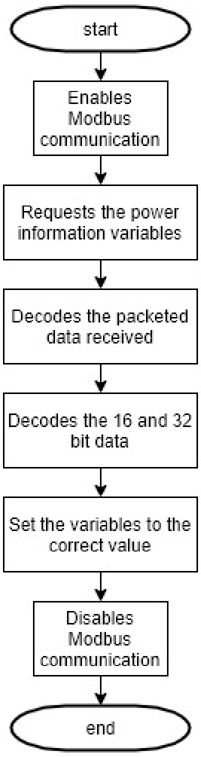

Figure 3.29 – Flowchart of the power information request to the energy meter.

The Figure 3.30 shows the implemented circuit between the energy meter and the Raspberry Pi 3 B+.

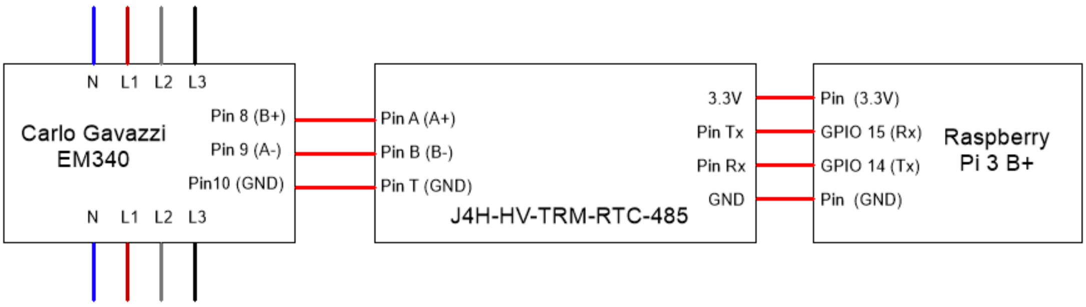

Figure 3.30 - EM340, J4H-HV-RTC-485 and Raspberry Pi 3 B+ connections.

3.3.6. Electric power protections

The objective of a power system protection is to have a safe availability of electrical power without any interruption to every load connected to it. To ensure a safe power transfer, the charging station has electric protection, such as circuit breaker and ground fault circuit breaker to detect abnormal conditions, for example electric faults, and act on it. The function of the circuit breaker is to protect the cable by shutting off the electric power when the current is above the cable current rating. The purpose of the ground fault circuit breaker is to detect the imbalance between the outgoing and incoming current, which can be provoked by an electric fault such as a short circuit, equipment malfunction, insulation failure or an electric shock provoked by people, and act on it by shutting down the charging station power. The circuit breaker and the ground fault circuit breaker used in the charging station protections are present in the Figure 3.31

Figure 3.31 – Hager 4P 40 A circuit breaker (1) and Hager 4P 40 A 30 mA ground fault circuit breaker (2) [73], [74].

Other protection that is included is an actuator that is present in the type 2 socket (Figure 2.5) with the main function of locking the plug of the EV charging cable. This locking system prevents an unwanted cable disconnection to ensure a safe power transfer between the charging station and the EV.

3.4. Server

A server is a computer program or device that provides a service to another computer program and its user, also known as the client. This architecture is called client-server model, meaning that a server program awaits and fulfils requests from clients programs. Severs are often categorized in terms of their purpose, for example, in a web server the web browser acts as client. In a file server a client is responsible for the central storage and management of data files so that other clients on the same network can access them [75].

To send the PWM duty cycles to the coordinator, the server waits until the coordinator makes a POST request, which is made every minute. If the POST request is made, receives data and sends the PWM duty cycles.

The data is visualized on a web page, which is a document for the world wide web that is identified by a URL and can be accessed and displayed on a monitor, or mobile device, through a web browser. The web page interface provides inputs parameters to manage the EV charging session and if the user does not provide any input the EV will start charging according to the system power availability. The system available power can be calculated and converted to PWM duty cycle and sent to the charging station through the coordinator. The serve checks the connected EVs that are charging and those how needs charging and divides equally among all the available power to allow a fare charging.

All the data receive by the server is store on a database named MongoDB, which is an open source database, to prevent the loss of data and to access it when needed [76]. The database has two tables, one for the EV connectivity, EV state and power information variables, and the other has the PWM duty cycle values. To access each table is by the charging station identification number.

Figure 3.32 shows the server operating cycle and implemented code is in Attachment C.

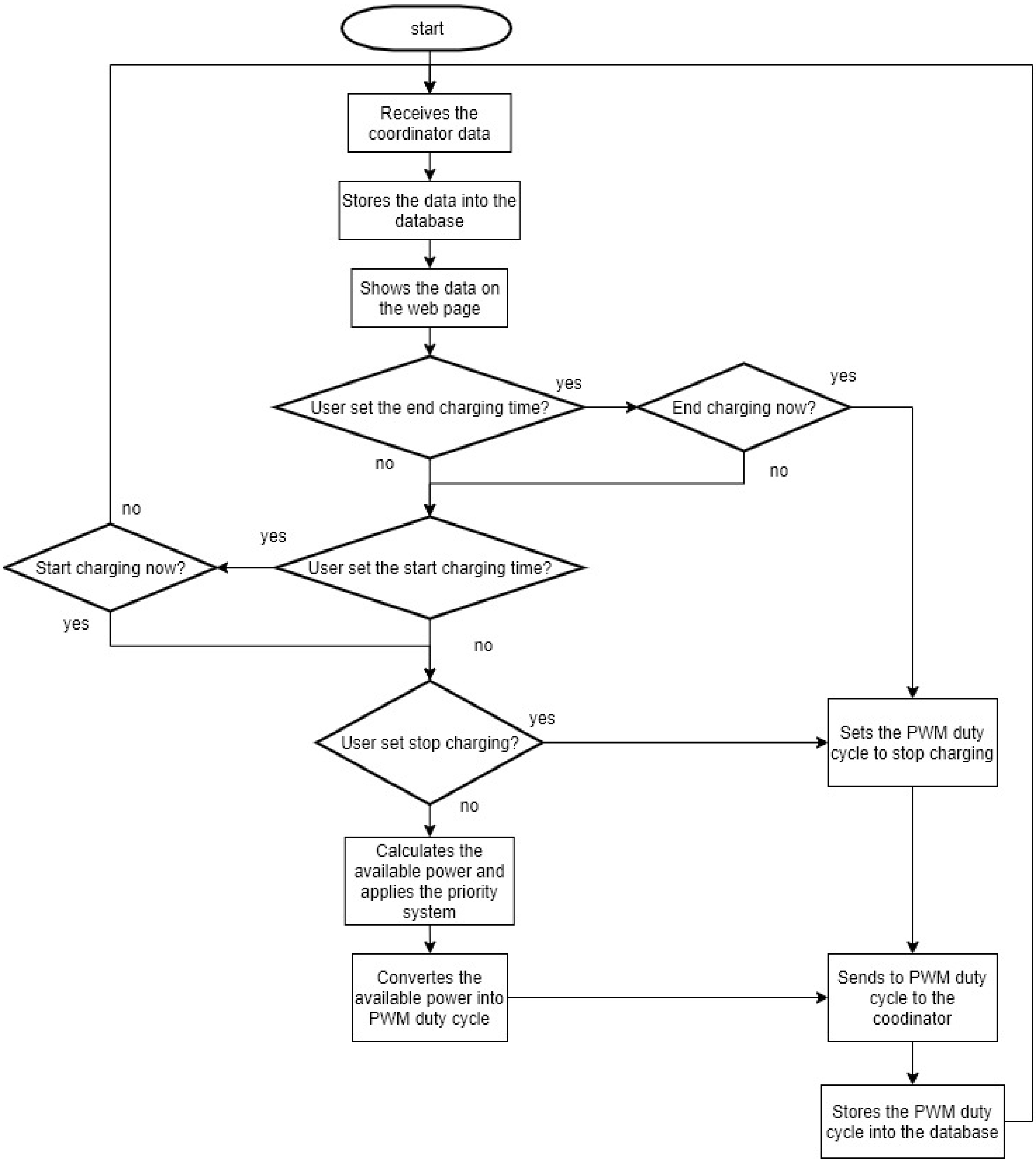

Figure 3.32 - Flowchart of the server operating cycle.

3.4.1. Web interface

To be able to remotely control the EV charging session, the web page interface, showed in Figure 3.33, provides input parameters to allow a user interaction:

- Finish – Terminates the EV charging;
- Schedule – Set the charging session start and end time.

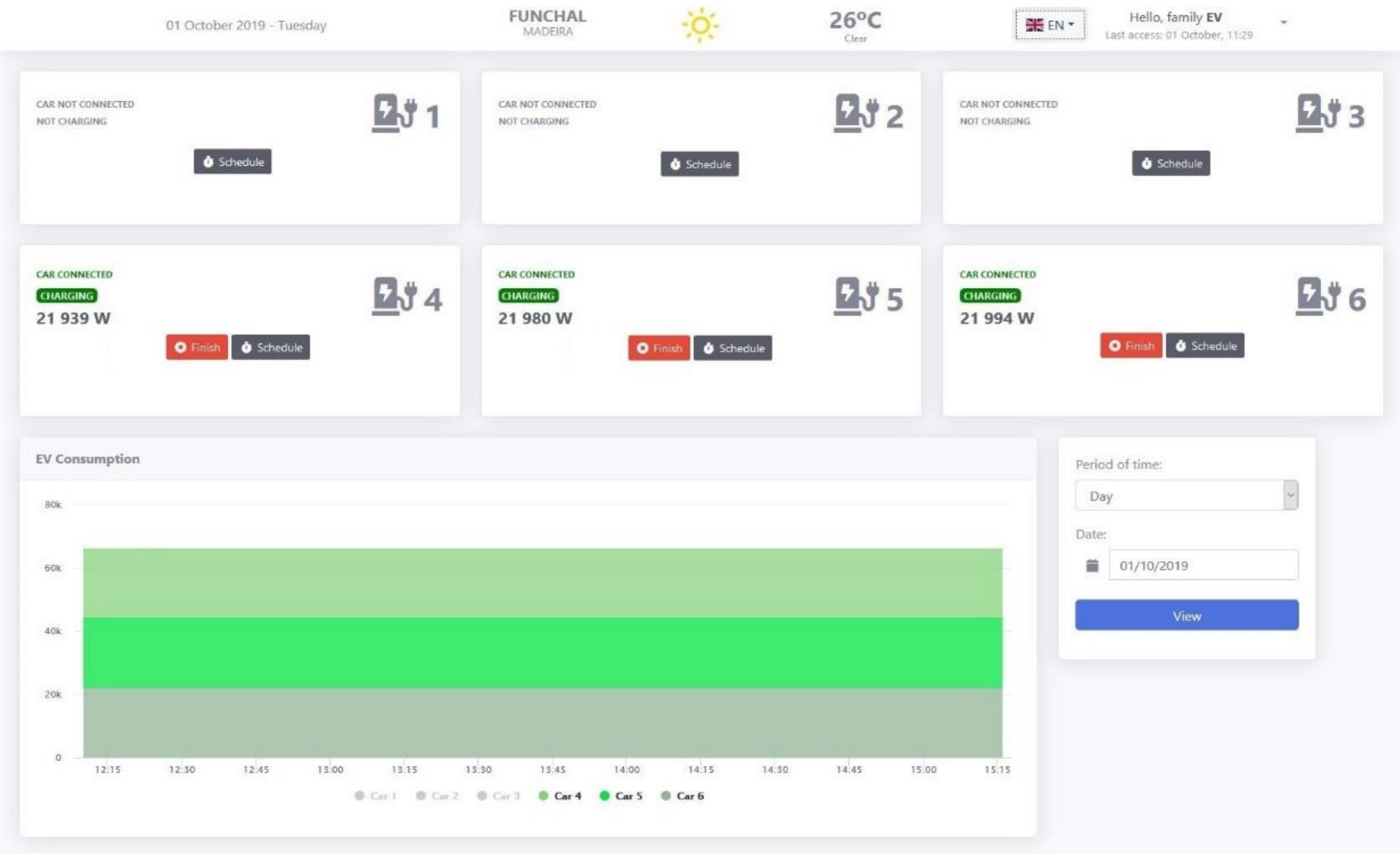

Figure 3.33 - Web page interface.

After any EV connection to the charging station, the charging schedule start parameter has to be set by the user, otherwise is not available throughout the EV charging session, only the end parameter is available, because if the EV is connected it means that the EV is going to start its charging session and there is no need to set the start parameter. The charging schedule has a start and end time inputs as showed in Figure 3.34.

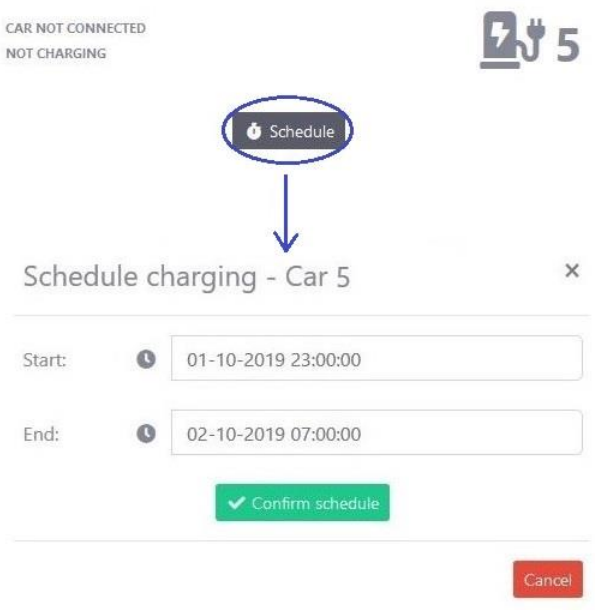

Figure 3.34 – EV charging schedule setup.

3.4.2. PWM duty cycle calculation

In order to control the EV power consumption, a relationship between the available power and the PWM duty cycle has to be made. The available power is:

$$available\ power\ (kW) = \frac{V(V) \times I(A) \times PF \times charging\ typology}{1000} \tag{3.1}$$

where $V\ (V)$ is the voltage, $I\ (A)$ is the current, PF is the power factor and the $charging\ system\ typology$ is whether the EV uses three-phase or single-phase charging. The value 1 is for single-phase and value 3 for three-phase.

The PWM duty cycle is related to the charging current, according to the IEC61581 standard (Table 2.6) and Figure 3.35 shows that relationship where the charging current can be obtain with the following equation:

$$I\ (A) = PWM\ duty\ cycle\ (\%) \times 0.6 \tag{3.2}$$

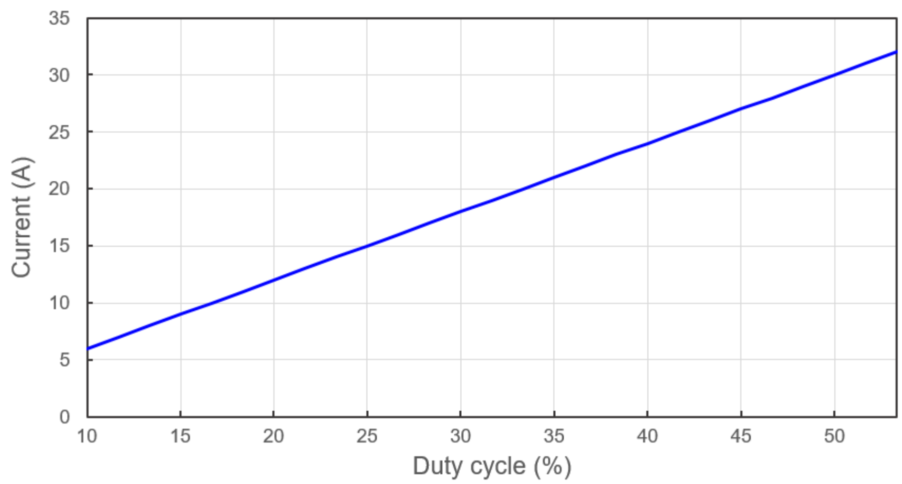

Figure 3.35 - Relationship between the charging current and the PWM duty cycle [60].

Adding the PWM duty cycle on the Equation 3.1, the relationship between the available power and the PWM duty cycle is made:

$$PWM\ duty\ cycle\ (\%) = \frac{available\ power\ (kW) \times 1000}{V\ (V) \times PF \times 0.6 \times charging\ typology} \tag{3.3}$$

considering an ideal power system, the $V = 230$ V and $PF = 1$:

$$PWM\ duty\ cycle\ (\%) = \frac{available\ power\ (kW) \times 1000}{138 \times charging\ typology} \tag{3.4}$$

If the EV uses a single-phase charging system the $charging\,typology$ is 1 and the $available\,power$ needs to be higher than 1.38 kW, otherwise is 0 kW and the $PWM\,duty\,cycle$ is set to 100 %, because the minimum current permitted to start charging an EV is 6 A and $230\ V \times 6\ A = 1380\ W = 1.38\ kW$. If the $available\,power$ is higher than $7.36\ k$W, it means that the charging station cannot supply more than 32 A ($230\ V \times 32\ A = 7360\ W = 7.36\ kW$) of current because its maximum power delivery and the $available\,power$ is limited to that value. If the EV uses a three-phase charging system the $charging\,typology$ is 3 and the $available\,power$ needs to be higher than $4.14\ k$W ($230\ V \times 6\ A \times 3 = 4140\ W = 4.14\ kW$), otherwise is 0 kW, and cannot surpass the 22 kW ($230\ V \times 32\ A \times 3 = 22000\ W = 22\ kW$), because the charging station cannot supply more than its maximum power delivery, so the $available\,power$ is limited to that value.

3.4.3. Available power calculation

In order to send the PWM duty cycle to the charging station to be applied on the EV, the available power needs to be obtain. The available power depends on the building/house power status, meaning that is has to be considered the grid power supply limit to the building/house, solar panels power generation and the building/house power consumption. The next equation shows how the available power can be calculated according to the power status:

$$\begin{aligned} &available\ power(kW) \\ &\qquad = grid\ power\ limit(kW) + PV(kW) - power\ w/EVs\ (kW) \end{aligned} \tag{3.5}$$

where $grid\ power\ limit$ is the grid power supply limit to the building/house, PV is the solar panels power generation and $power\ w/EVs$ is the power consumption without considering the EVs.

To prevent a power building/house power blackout a safety power has to be considered and added to the Equation 3.5, where PS is the safety power:

$$\begin{aligned} &available\ power(kW) \\ &\qquad = grid\ power\ limit(kW) + PV(kW) - power\ w/EVs\ (kW) \\ &\qquad - PS\ (kW) \end{aligned} \tag{3.6}$$

There are several ways to obtain the power safety. One way is to analyse a power consumption profile and obtain the variation of power consumption with the maximum and minimum value. The safety power is obtain using the following equation:

$$PS(kW) = \frac{\max\big(f(t)\big) - \min(f(t))}{2} \tag{3.7}$$

where t is time, $f(t)$ is a power consumption profile, $\max(f(t))$ is the maximum value of the power consumption profile that occur in a period of time and $\min(f(t))$ is the minimum value of the power consumption profile that occur in a period of time.

The server has a priority charging system that distributes the available power equally among all the connected EVs that are charging and those how needs charging. To allow a EV to start charging the available power, divided by all the connected EVs, needs to be higher than the minimum charging power permitted by the EV (for three-phase charging is 4.14 kW and for single-phase charging is 1.38 kW). If the divided available power is less than the permitted the EV does not start charging. If the EVs are charging and the available power is less than the permitted the last EV connected stops charging, meaning that the first EV connected is going to continue to charge until there is not enough available power to charge or it is fully charge.

Knowing how the priority system operates, the available power, on Equation 3.6, is divided by the number of the EVs that are charging and those how needs charging:

$$available\ power\ (kW) = \frac{grid\ power\ limit(kW) + PV(kW) - power\ w/EVs\ (kW) - PS\ (kW)}{Number\ of\ EVs} \tag{3.8}$$

where $grid\ power\ limit$ is the grid power supply limit to the building/house, PV is the solar panels power generation, $power\ w/EVs$ is the power consumption without considering the EVs, PS is the safety power and $Number\ of\ EVs$ is the number of EVs that are charging and those how needs charging.

Figure 3.36 shows the priority charging system operation.

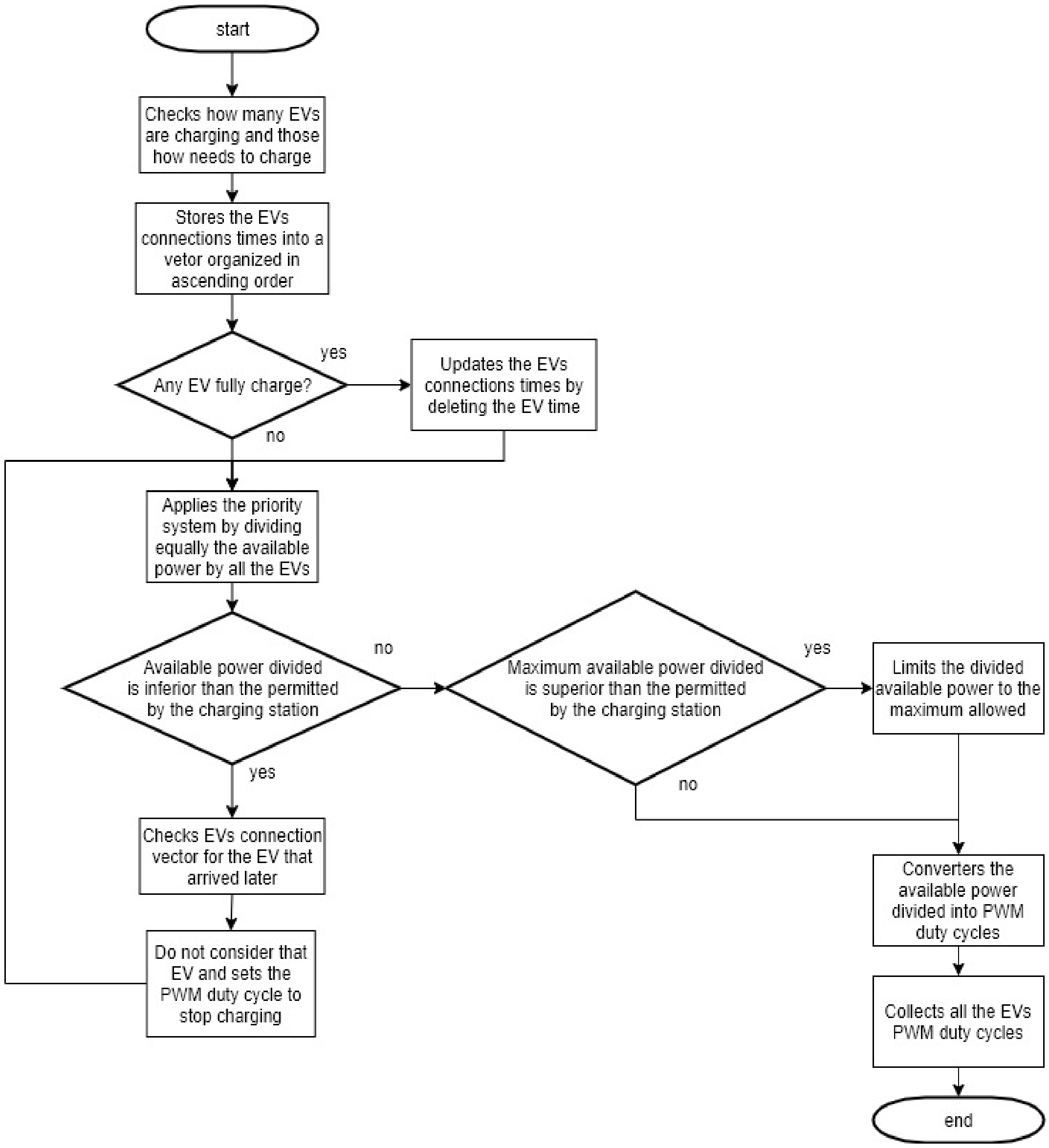

Figure 3.36 - Flowchart of the EV priority system operation.

Chapter 4

Simulation of EV charging scenarios

This chapter presents the simulation of the plug and charge scenario, the smart charging scenario, how the increase of EVs in Madeira Island can affect the electrical grid power supply and a conclusion on which EV charging scenario is better to implement to manage the EV power consumption.

4.1. Plug and charge scenario

This scenario is the most common used to charge an EV and the control of the EV power consumption is accomplish by having a fixed power rate according to the charging station maximum power delivery. To test this scenario the following power consumption (Figure 4.1) from a building with 8 apartments was considered and 3 EVs charging stations of 22 kW.

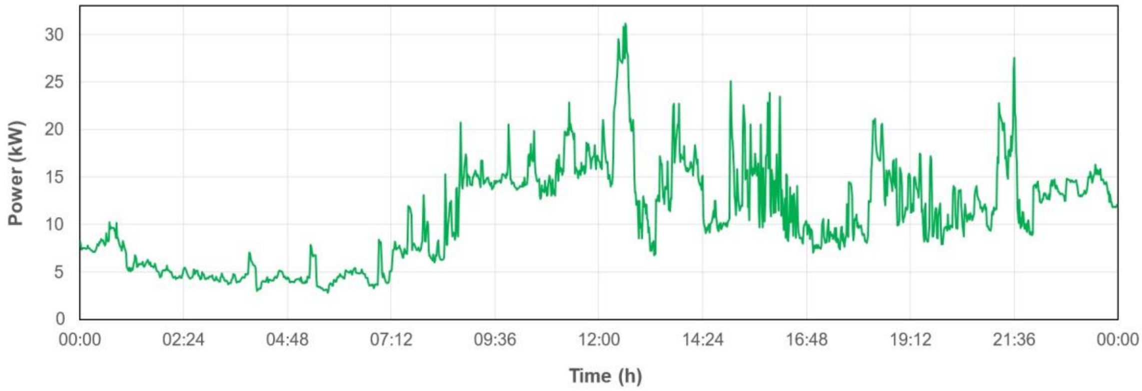

Figure 4.1 – Plug and charge power consumption profile considered.

According to the Portuguese technical rules for low voltage electrical installations [77], the limit of the grid power supply to a building can be obtain by the following equation:

$$power\ limit(kW) = 6.9\ kW \times 0.75 \times N \tag{4.1}$$

Where 6.9 kW represents an estimated value of the grid power consumption for each installation, 0.75 represents a coefficient and N represents the number of apartments that the buildings has. With this, a building with 8 apartments, the grid power supply limit can be seen as the grid power consumption limit and can be calculated:

$$power\ limit(kW) = 6.9\ kW \times 0.75 \times 8 = 41.4\ kW$$

Having a charging station with a power capacity of 22 kW means that the charging power rate is 22 kW and with 3 charging stations the total power consumption is 66 kW, which is much higher that the grid power supply limit allowed by a building with 8 apartments. If 3 EVs are charging at the same time the power supplied to the building is going to be shut down for safety reasons because the grid power limit was exceeded.

Figure 4.2 shows the scenario application where the grid power limit is represented by the red colour, the building power consumption without the EVs is the black, the building power consumption is the green and the others colours represents each EV power consumption when charging. Analysing the building power consumption between $00:45$ h and $07:12$ h it is notice three levels of power consumption, meaning that 3 EVs where charging because there is an increase of the building power consumption by 22 kW in each level comparing with the power consumption without EVs. With this, the grid power limit was exceed at $02:52$ h and at $05:02$ h, and analysing the building power consumption throughout the 24 hours it is notice that only one EV is allow to charge. However, there are three time frames (between $08:28\ \mathrm{h} - 12:57\ \mathrm{h}$, $15:07\ \mathrm{h} - 18:43\ \mathrm{h}$ and $20:52\ \mathrm{h} - 22:19\ \mathrm{h}$) that only one EV is charging and the power limit was exceeded. The best time frame to charge an EV is between the $22:19$ h and $08:38$ h because the power consumption without the EV is below approximately 19.4 kW (difference between the charging station power rate and the power limit) which represents the maximum power consumption allowed to charge an EV to prevent a power blackout.

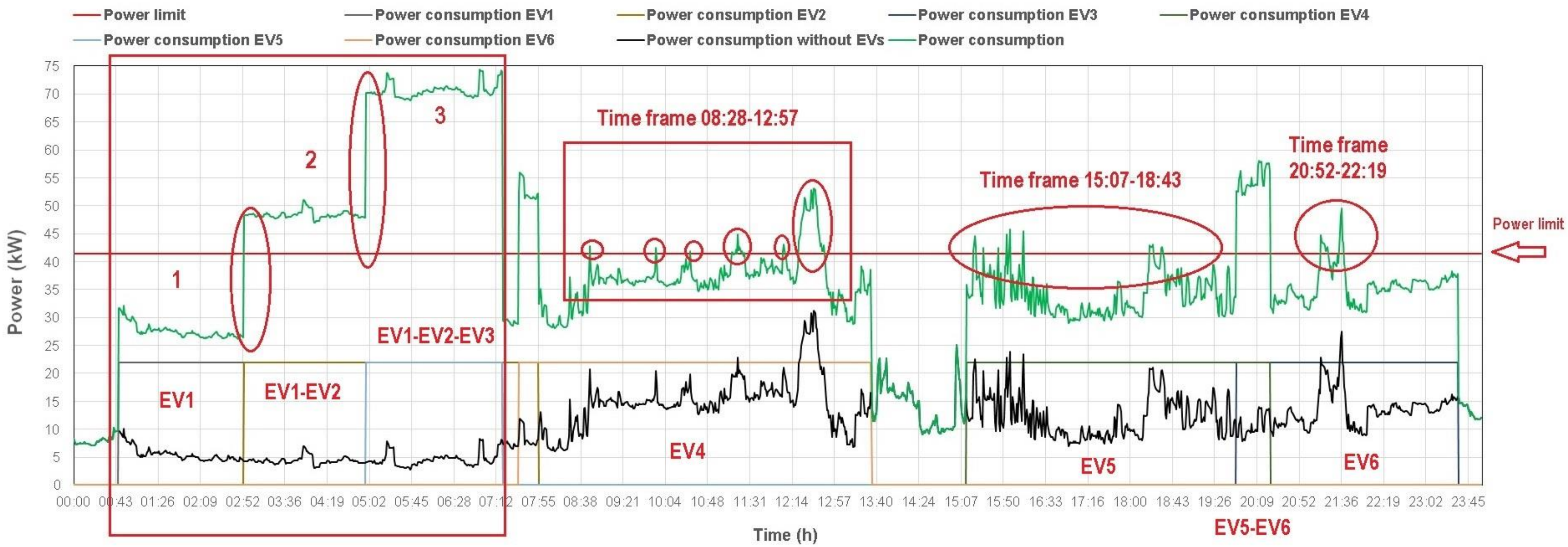

Figure 4.2 - Plug and charge scenario results considering charging stations with 22 kW.

Instead of having the charging stations with a power rate of 22 kW, we can analyse if the power rate is reduce to half, meaning that the each charging station has a power rate of 11 kW. If the building has 3 charging stations, it means that if 3 EVs are charging the total power consumption is 33 kW, which is below the building power limit (41.4 kW).

Figure 4.3 shows the scenario application but with 3 charging stations of 11 kW. Analysing the building power consumption throughout the 24 hours, between 12: 14 h and 12: 57 h, the power limit was exceeded. If it was added one more EV, between 08: 38 h and 12: 14 h, there was more than one time that the power limit can be exceeded. For this scenario to work without any power blackout, the power rate needs to be 3.7 kW because between 12: 14 h and 12: 57 h the power consumption without the EVs was more than 30 kW and if 3 EVs were charging at this time frame only approximately 3.7 kW ($41.4\ kW - 30\ kW \approx 3 \times 3.7\ kW$) was available to charge each EV (Figure 4.4 shows this scenario). A charging station of 3.7 kW is not ideal if the charging time is a limitation because, according with the Table 2.3, to charge an EV with a 24 kWh battery it will take 6.5 hours to charge from empty to full capacity. When the available power is higher than 3.7 kW the EV is going to charge only at this power rate, meaning that the EV charging management using a fixed power rate does not consider the power that is available to charge an EV, since it cannot regulate the power rate according to the available power. However, this scenario can be ideal for charging an EV between 22: 19 h and 07: 55 h, because the power consumption without the EVs is lower and the charging power rate can be approximately 7.4 kW ($41.4\ kW - 17.5\ kW \approx 3 \times 7.4\ kW$), meaning that it will take less time to charge an EV battery from empty to full, approximately 3.2 hours for an EV with a 24 kWh battery, according to the Table 2.3. Even with a power rate of 3.7 kW 6.5 hours can be enough to charge an EV battery to full. If the power limit was much higher that than the power consumption, the EV can charge with a higher power rate, since the power limit is a limitation on how much power the EV can charge using a fixed power rate.

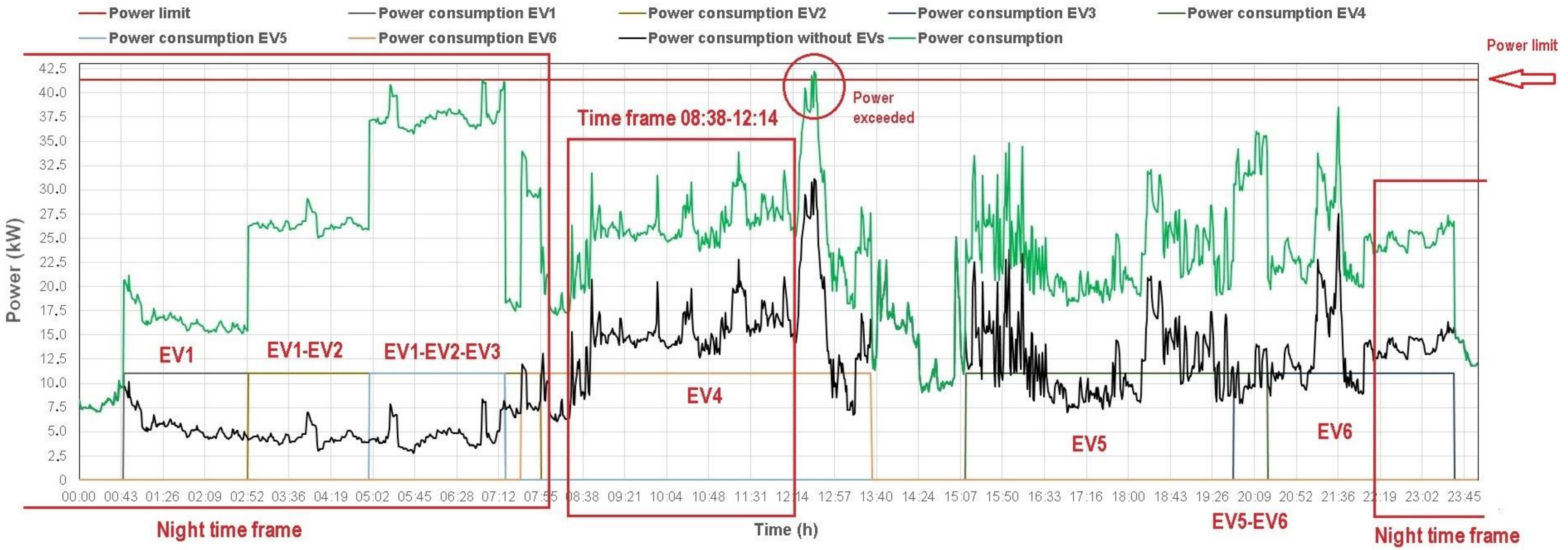

Figure 4.3 - Plug and charge scenario results considering charging stations with 11 kW.

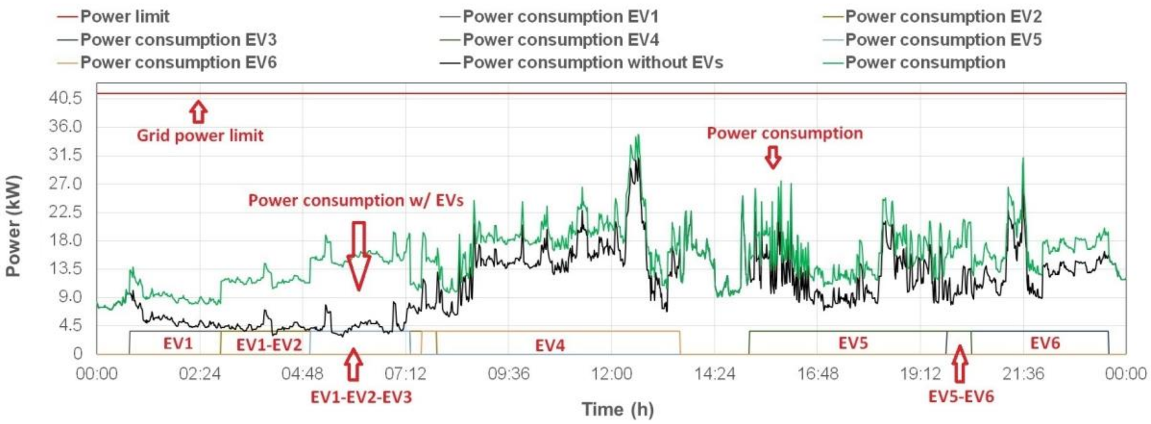

Figure 4.4 - Plug and charge scenario results considering charging stations with 3.7 kW.

The introduction of renewable energy, for example solar panels, does not add value to the plug and charge scenario because of the fixed power rate and only reduces the power consumption profile since the EV power consumption will not change. Figure 4.5 shows the plug and charge scenario results considering charging stations of 11 kW. It is notice that when the solar panels are generating power, the power consumption is reduce and prevents a power blackout. However, the EVs are charging at the same power rate (11 kW) throughout the day and between 04: 48 h and 07: 12 h the consumption reaches the maximum power supplied by the electric grid when there is no power being generated by the solar panels. The unpredictable power generation of the solar panels cannot be considered to obtain the power rate of the charging stations on the plug and charge scenario, because the power rate has to consider the power consumption without solar panels to prevent a power blackout when there is no power being generated by the renewable energy.

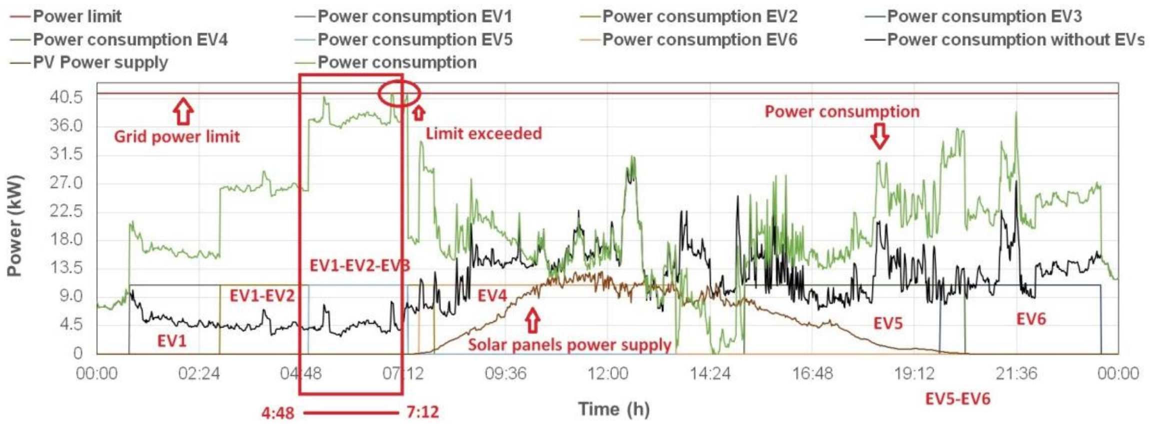

Figure 4.5 - Plug and charge scenario results considering charging stations of 11 kW and solar panels power generation.

The renewable energy only impacts the available power and higher renewable energy means higher is the available power, resulting in a higher charging power rate. The scenario that considers the variations of the available power to charge the EVs is the smart charging.

4.2. Smart charging scenario

The smart charging scenario considers the available power to charge the EV by applying the power rate according to the available power, instead of having a fixed power rate. The introduction of renewable energy, for example solar panels, can add value to this scenario, since the EV power consumption is going to change according the available power. However, the power rate is always limited by the charging station maximum power delivery.

To test this scenario it was considered the same test scenario as the plug and charge scenario (building with 8 apartments, grid power supply limit and 3 charging station of 22 kW each), power consumption without EVs, a safety power, solar panels that can produce a maximum of 13.1 kW of power and a priority system to prioritise the EV that started charging first.

To implement the smart charging scenario, the Equation 3.8 was considered because it has the available power according to the EV priority charging system, meaning that the available power is distributed equally among all the connected EVs that are charging and those how needs charging, allowing every EV to charge at the same power rate. If the power rate is lower than the minimum charging power permitted by the EV (for three-phase charging is 4.14 kW and for single-phase is 1.38 kW), the EV will not charge. If the EV arrives after the others EVs and if the available power divided by all of them is lower than the permitted, the EV that arrived after will not charge to allow the others EVs to continue to charge, meaning that is going to charge only when the available power is higher than the permitted or if an EV stops charging and there is enough power to charge the EV, if not, the EV will wait until there is enough available power.

The safety power is obtained considering the day before building power consumption without EVs (Figure 4.1), which is used in the plug and charge scenario, and can be calculated considering the Equation 3.7. The 24 hours of the day before power consumption was divided into 5 time frames (between $22:19\text{ h} - 08:38\text{ h}$, $08:38\text{ h} - 12:14\text{ h}$, $12:14\text{ h} - 14:24\text{ h}$, $14:24\text{ h} - 18:00\text{ h}$ and $18:00\text{ h} - 22:19\text{ h}$) as showed in Figure 4.6. On this time frames, the safety power was calculated with the Equation 3.7 where $f(t)$ is the day before power consumption without EVs profile:

- Time frame 1 (between $22:20\text{ h} - 08:38\text{ h}$):

$$PS(kW) = \frac{16320\ W - 2800\ W}{2} = 6760\ W = 6.76\ kW$$

- Time frame 2 (between$08:39\text{ h} - 12:14\text{ h}$):

$$PS(kW) = \frac{22844\ W - 8360\ W}{2} = 7242\ W = 7.242\ kW$$

- Time frame 3 (between $12:15\text{ h} - 14:24\text{ h}$):

$$PS(kW) = \frac{31178\ W - 6777\ W}{2} = 15250.5\ W \approx 15.25kW$$

- Time frame 4 (between $14:25\text{ h} - 18:00\text{ h}$):

$$PS(kW) = \frac{25096\ W - 7022\ W}{2} = 9037\ W = 9.037\ kW$$

- Time frame 5 (between $18:01\text{ h} - 22:19\text{ h}$):

$$PS(kW) = \frac{27506\ W - 7910\ W}{2} = 9798\ W = 9.798\ kW$$

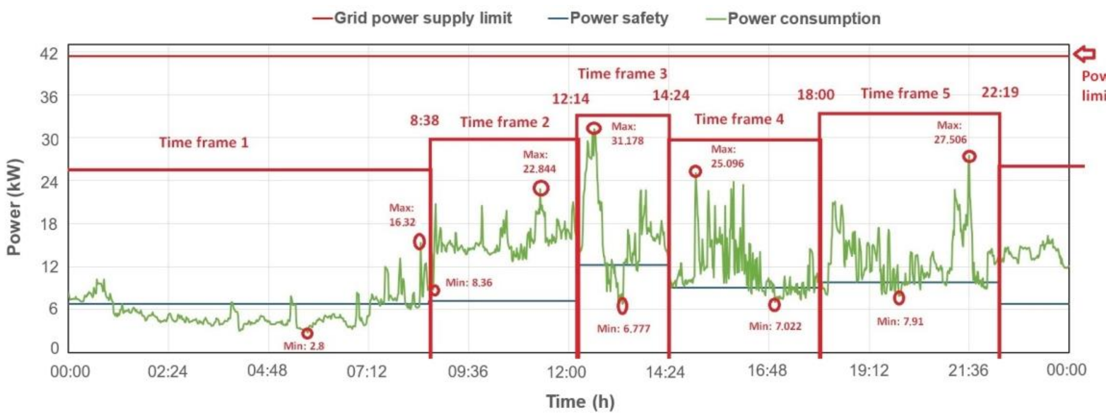

Figure 4.6 - Smart charging scenario safety power obtain from the day before power consumption considering 5 time frames.

Figure 4.7 shows the solar panels power delivery and Figure 4.8 shows the building power consumption without EVs considered.

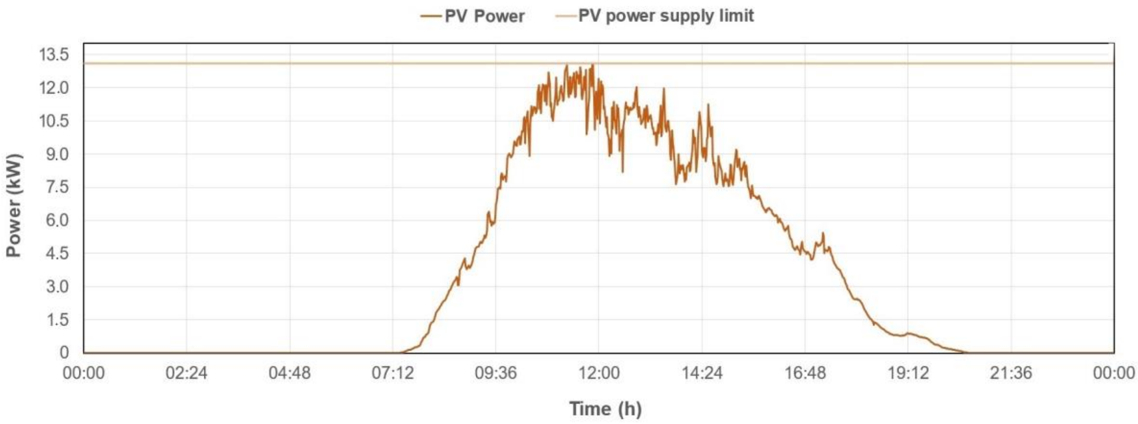

Figure 4.7 - Solar panels power generation considered on the smart charging scenario.

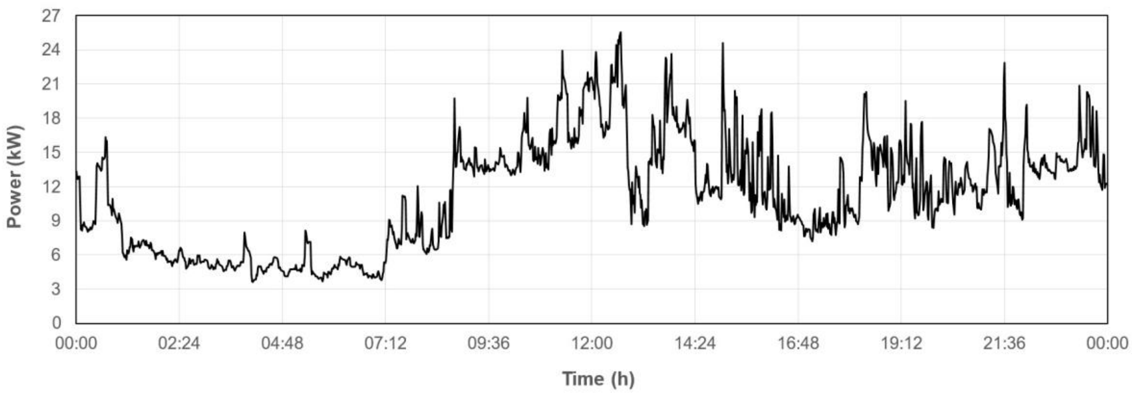

Figure 4.8 - Smart charging building power consumption without EVs.

Figure 4.9 shows the smart charging scenario application where the grid power supply limit is showed by the red colour, the green is the building power consumption, the black is the building power consumption without EVs, the brown is the power generation from the solar panels and the blue colour is the grid power consumption. Analysing the building power consumption between $09{:}36$ h and $13{:}12$ h, the grid power supply limit was exceeded but this does not mean that a power blackout event occur, it means that the power limit was higher because the solar panels where providing power. Between $03{:}36$ h and $07{:}12$ h the building power consumption has a flat profile meaning that EVs were charging in this time frame, as well as between $19{:}12$ h and $22{:}48$ h because the building power consumption without EVs does not have the same flat profile. There is a time frame that none of the charging stations were delivering power to the EVs, meaning that between $13{:}12$ h and $15{:}36$ h there was no EV charging and approximately at $14{:}24$ h the building power consumption is the same as the solar panels generation and the grid power supply was 0 W, meaning that at least 26 kW ($41.4\ kW - 15.2505\ kW \approx 26\ kW$) is available to charge EVs. The priority system can be verified between $03{:}36$ h and $07{:}12$ h meaning that there was more than one EV charging because, between $01{:}12$ h and $02{:}24$ h, the power consumption is the same as the power consumption without EVs with an increase of 22 kW and moments later it was more than 22 kW, so there was more than one EV charging.

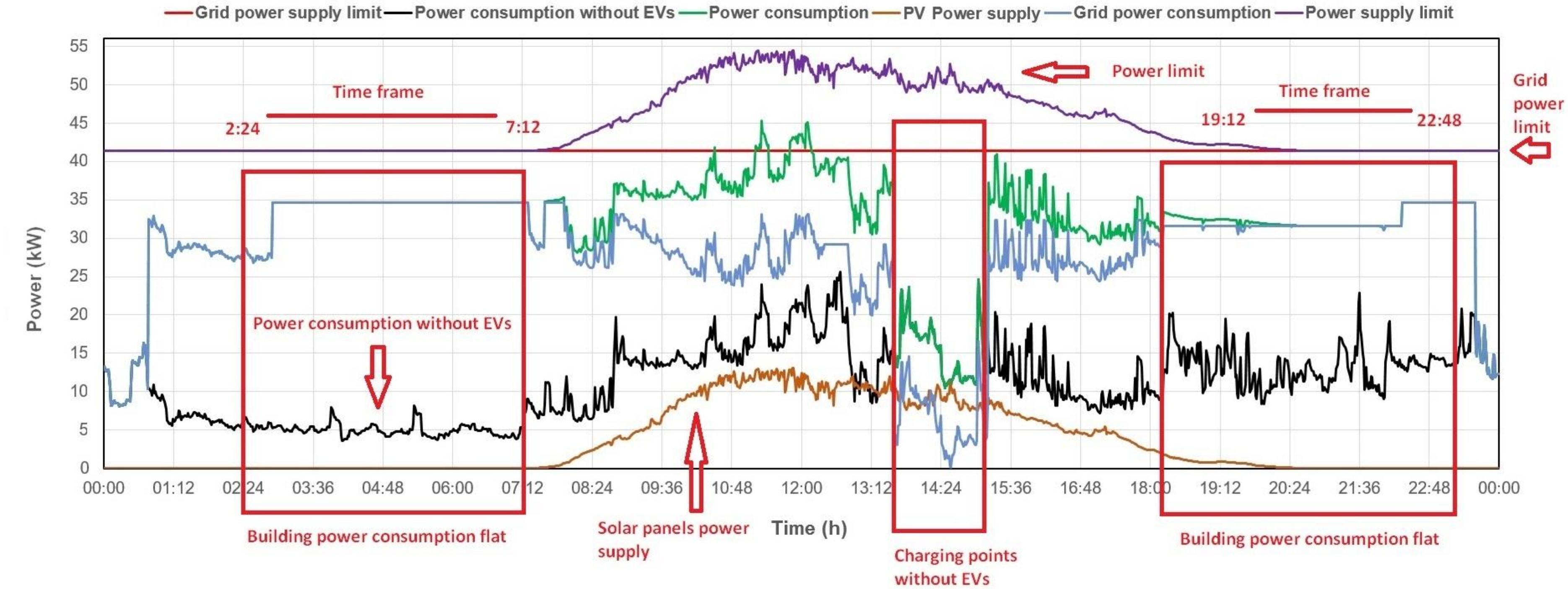

Figure 4.9 - Smart charging scenario results with the safety power updated in 5 time frames.

Figure 4.10 shows with more detail the smart charging scenario application where the green colour is the building power consumption, the black is the power consumption without EVs, the brown is the power generation from the solar panels and the other colours represents the EVs that are charging in the charging stations. Analysing the building power consumption between $03{:}36$ h and $07{:}12$ h, the flat profile appears because there are EVs charging at this time frame and between $04{:}48$ h and $07{:}12$ h there are 3 EVs charging at the same power rate, approximately 10.5 kW, which means that the available power is divided equally among all the EVs. Between $01{:}12$ h and $02{:}24$ h there is an EV charging at the maximum power delivery allowed by the charging station (22 kW) and between $18{:}00$ h and $20{:}24$ h there is an EV charging according to the available power showing that, when possible, the EV charges at 22 kW. Considering the building power consumption throughout the 24 hours, the power limit was never exceeded meaning that the power delivery to the EVs is well manage, considering the building power consumption without EVs, the solar panels power generation, the grid power supply limit and the safety power introduced.

The available power depends as well on the safety power, meaning that more safety power is less available power and less safety power is more available power. Instead of considering 5 time frames from the day before (Figure 4.6), the safety power value is going to be updated every minute considering the time frame of the 30 minutes before to allow a higher available power. The last 30 minutes are going to be calculated firstly with 30 minutes from the day before power consumption without EVs to prevent the EV to wait 30 minutes to start charging. In this case, the first 30 minutes of the safety power are going to have the same value and Figure 4.11 shows an example on how the safety power value is calculated each minute with 30 minutes of time frame. At minute $00{:}30$ h the safety power value is calculated with the 30 minutes from the day before, meaning that at $00{:}31$ h the safety power value is calculated with 1 new value of power consumption without EVs and 29 minutes of the day before and at $00{:}32$ h the safety value is calculated with 2 new minutes and 28 minutes from the day before, and so on. There is a minute that the 30 minutes from the day before is not needed anymore and is considered the 30 minutes before of that day.

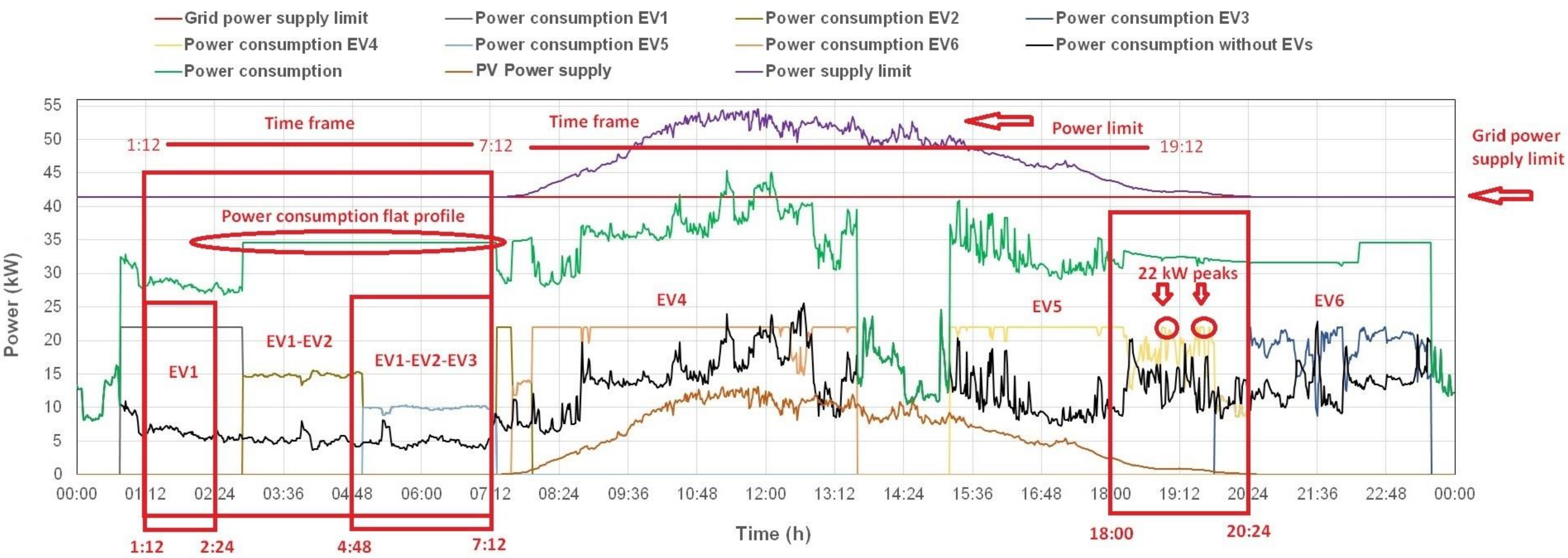

Figure 4.10 - Smart charging scenario considering the safety power is updated in 5 time frames showing each EV power consumption profile.

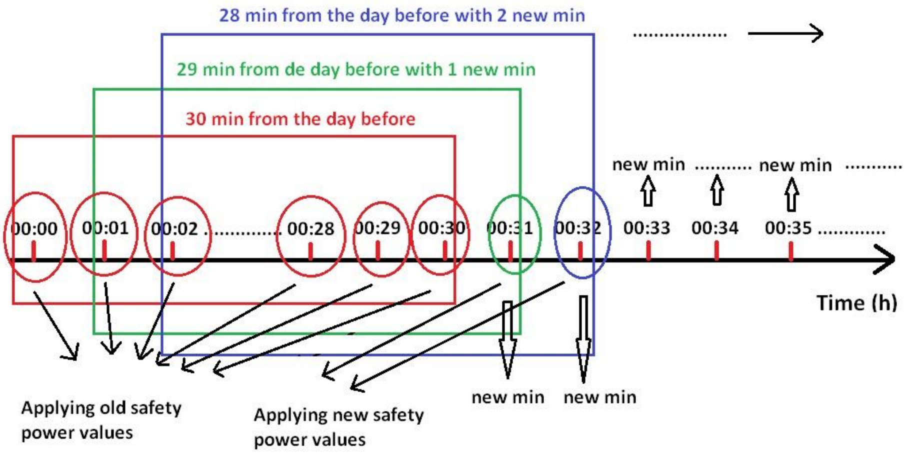

Figure 4.11 - Example of how the safety power can be obtain considering a time frame of the 30 minutes before.

Figure 4.11 shows the safety power obtained every minute considering the 30 minutes time frame and, from left to right, at 00: 00 h there is a flat profile representing the first 30 minutes of the safety values, which are identical, calculated with the power consumption from the day.

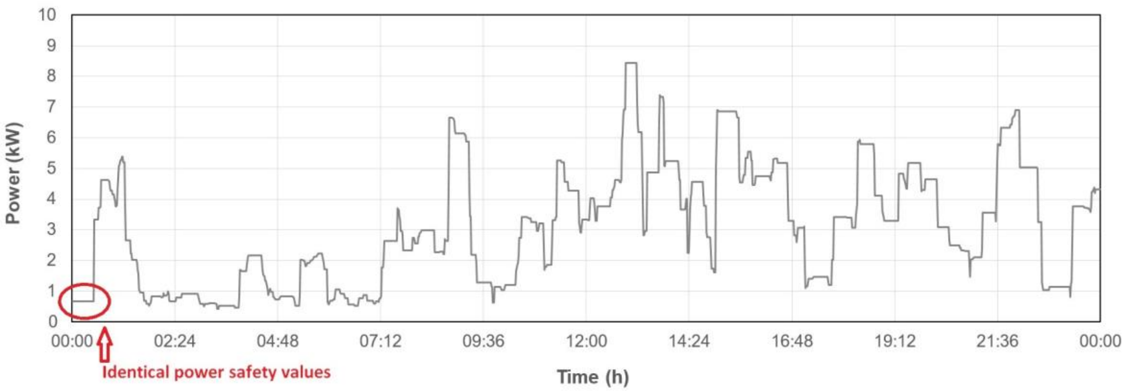

Figure 4.11 – Safety power obtain with the 30 minutes time frame.

Figure 4.12 shows the smart charging scenario with the power safety updated every minute with the 30 minutes time frame. The building power consumption was increased because of the increasing of the available power, meaning that the safety power was reduce. Between 02: 24 h and 07: 12 h the building power consumption is almost reaching the power limit, meaning that the available power was increase allowing more power to charge the EVs. With the safety power being updated every minute with a 30 minutes time frame, the EV power consumption was increased compared with the 5 time frames selected from the day before power consumption, meaning that the available power is better managed.

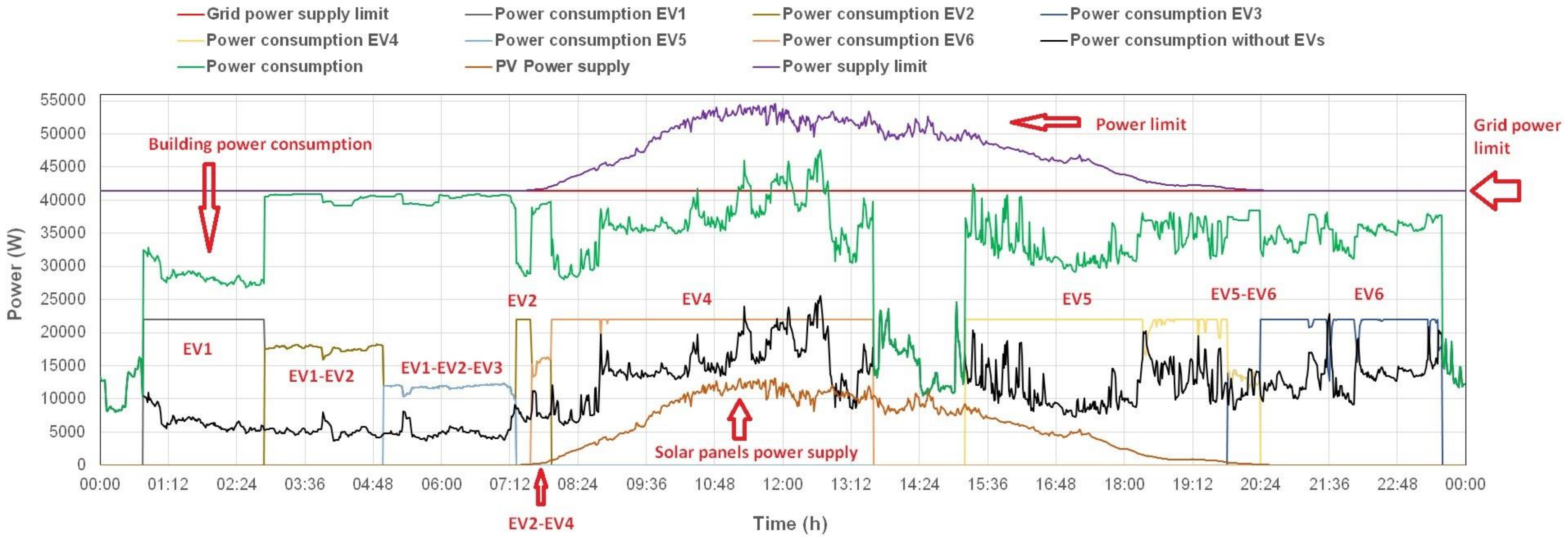

Figure 4.12 - Smart charging scenario results with the safety updated every minute considering the 30 minutes time frame.

To increase more and manage better the available power, the safety power (Figure 4.13) was updated every minute considering a 10 minutes time frame.

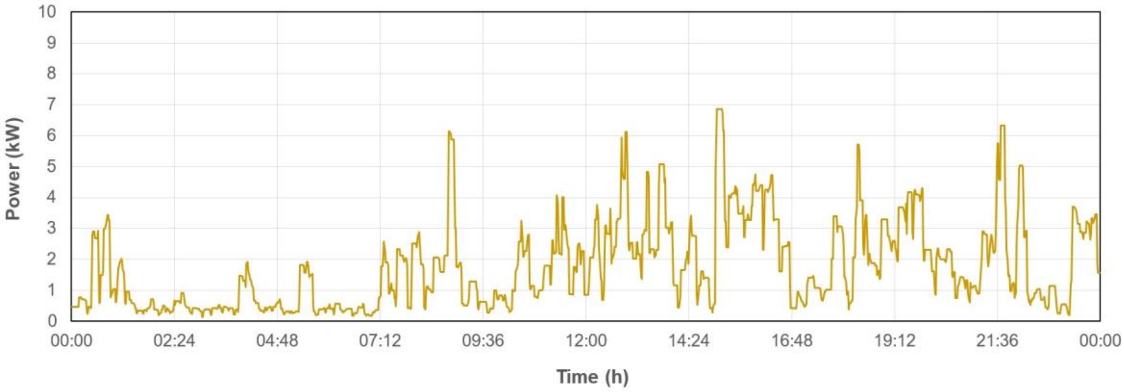

Figure 4.13 - Safety power obtain with the 10 minutes time frame.

Figure 4.14 shows the implemented scenario where the violet colour is the power limit, the black is the EVs power consumption where is considered the calculation of the safety power with 5 time frame from the day before power consumption, the green is the EVs power consumption where is considered that the safety power is updated every minute with a 30 minutes time frame and the yellow colour is the EV power consumption where is considered that the safety power is updated every minute with a 10 minutes time frame. Analysing all the EVs power consumption, between $02:24$ h and $07:12$ h, the safety power with a 10 minutes time frame (yellow colour) is the one how has more available power (Figure 4.15), meaning that the EVs are charging with a superior power rate and the others not (black and green colours). The difference between the EV power consumption (yellow colour) with a safety power updated every minute with a 10 minutes time frame and the other that is updated with a 30 minutes time frame (green colour) is very small. However, analysing throughout the 24 hours the difference can be higher when considering the overall EV power consumption.

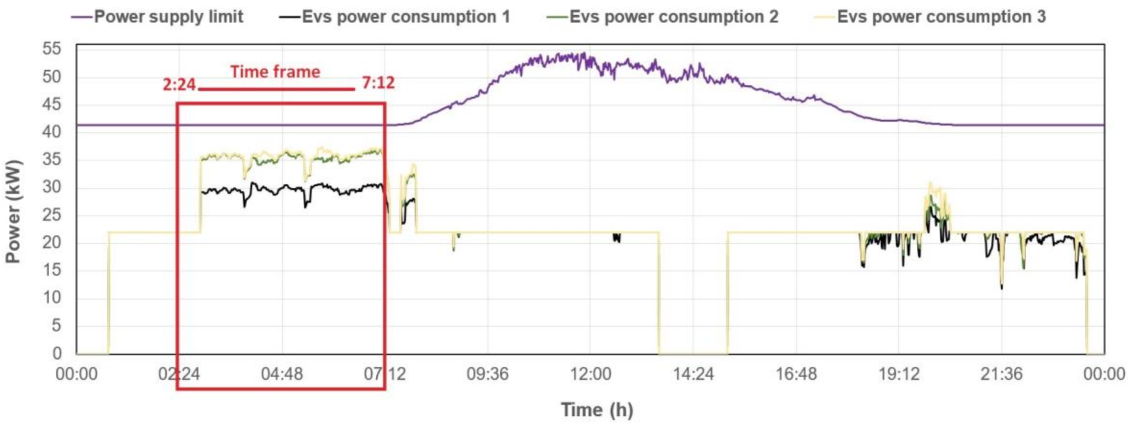

Figure 4.14 - Results of the EVs power consumption considering the safety power obtain with 5 time frames (black colour), with 30 (green colour) and 10 (yellow colour) minutes time frames.

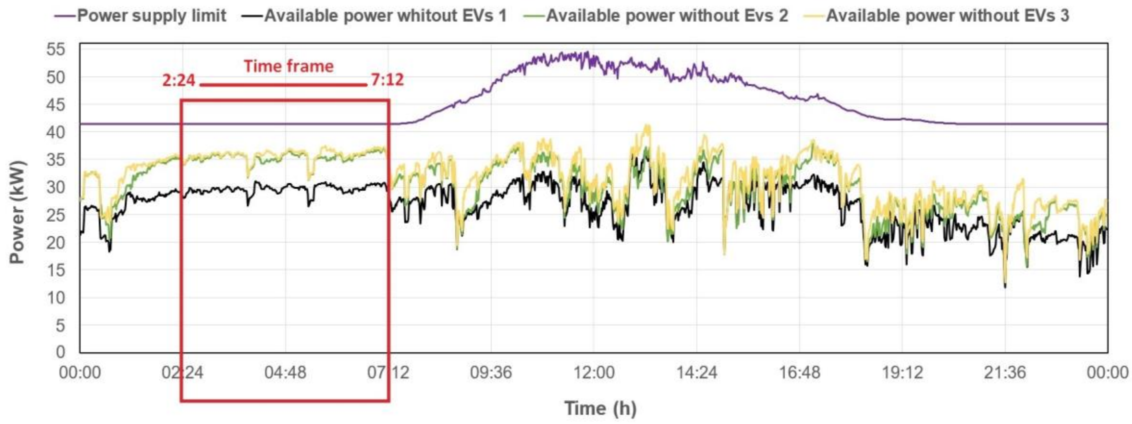

Figure 4.15 - Results of the available power considering the safety power obtain with 5 time frames (black colour), with 30 (green colour) and 10 (yellow colour) minutes time frames.

Figure 4.16 shows with more details the smart charging scenario with the safety power updated every minute with a 10 minutes time frame where the violet colour is the power limit, the green is the building power consumption, the black is the power consumption without EVs, the brown is the solar panels power supply to the building and the other colours are each EV power consumption. Comparing with the building power consumption from Figure 4.9 and Figure 4.12 between $02{:}\,24$ h and $07{:}\,12$ h, the smart charging scenario with the safety power updated every minute with a 10 minutes time frame has more available power, which is deliver to the EVs at a superior power rate, meaning that the EV charging time will be reduce because more available power less is charging time.

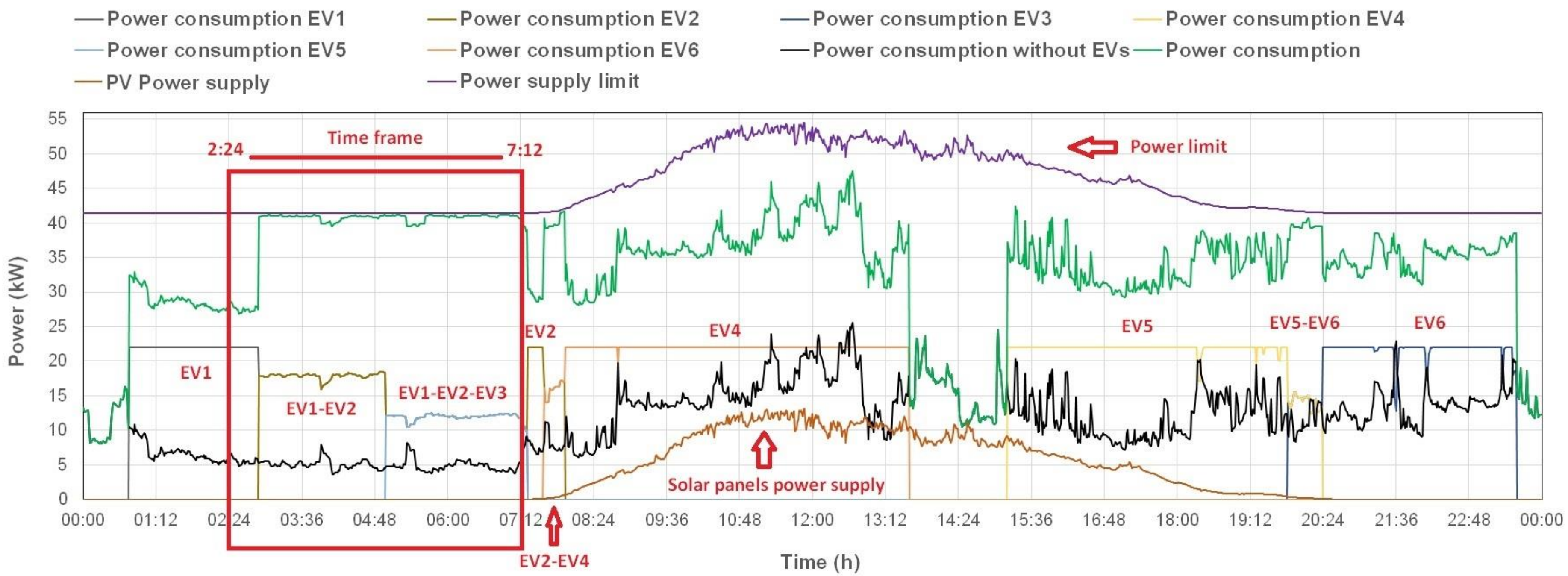

Figure 4.16 - Smart charging results with the safety power update every minute considering a 10 minutes time frame.

The smart charging scenario depends on how much safety power is introduce into the Equation 3.8 to obtain the available power (higher available power lower safety power and lower is the charging time, and lower available power higher safety power and higher is the charging time). The safety power is, as well, a safety feature implemented to prevent a power blackout because this occurs when the building power consumption exceeds the power limit.

Comparing this scenario with the plug and charge scenario, the charging station power rate is higher and can charge EV at superior power rates because the EV power consumption is manage and can be obtained every minute the EV power rate, instead of using a fixed power rate throughout the 24 hours. On the plug and charge scenario the charging power rate allowed is 3.7 kW for each charging station to prevent a power blackout, which is a very low power rate. Despite of the smart charging scenario not applying a fixed power rate, it can calculate every minute the power rate to send to the EV by maximizing the available power considering the building power consumption without EVs and the grid power supply limit. The smart charging scenario can consider the solar panels power generation and increase even more the charging station power rate to charge an EV. The plug and charge scenario does not consider the solar panels power generation because of its unpredicted power supply, meaning that the power rate is the same with or without the solar panels to prevent a power blackout when there is no solar panels power generation.

4.3. Madeira Island electrical grid scenario

With the increase of the number of EVs the charging stations will increase, meaning that the power consumption will rise and can impact the electrical grid by provoking a power blackout or increase the generators power supply to critical levels. Madeira Island does not have a cable connected to the main land to be used when there is not enough power generation, so all the generated power has to be well managed to prevent a grid power blackout. To study the impact of the EVs on the Madeira Island electrical grid it was considered from the annual report of 2015 of the Electricity Company of Madeira [78], which is the company that regulates the power supply and the power grid, the following parameters without considering the distribution and transport losses and limits, as well as, the generator response time to an increase or decrease of power consumption:

- Fuel power supply generator: 203.4 MW (maximum);
- Renewable energies power generation: 122.93 MW (maximum);
- Annual power consumption: $754.35\ \mathrm{GW}h$.

The Madeira Island population in 2015 was 256424 [79]. Assuming that ¼ of the population has an EV, it means that the number of EVs is 64106. With this number of EVs, not all the population has private parking so it was considered the following charging stations:

- 10000 house/building charging stations of 7 kW;
- 20 building/public charging stations of 22 kW;
- 20 public charging points of 50 kW;
- 10 public charging point of 150 kW.

Before the study of this scenario, the annual power consumption was converted to MW and denominated as power consumption without EVs:

$$power\ consumption\ without\ EVs\ (MW) = \frac{754.35\ GWh}{12\ months \times 30\ days \times 24\ hours} \approx 0.087309\ GW = 87.309\ MW$$

The following equation was implemented to calculate the available power:

$$\begin{aligned} available\ power(MW) &= generation\ power\ limit\ (MW) \\ &+ renewable\ power\ generation\ (MW) \\ &- power\ without\ EVs\ (MW) - power\ EVs\ (MW) \end{aligned} \tag{4.1}$$

where $generation\ power\ limit$ is the power supply generator limit, $renewable\ power\ generation$ is the renewable energy power generation, $power\ without\ EVs$ is the power consumption without EVs and $power\ EVs$ is the EVs power consumption.

Adding all the charging stations power delivery, the total EVs power consumption is:

$$10000 \times 7\ kW + 20 \times 22\ kW + 20 \times 50\ kW + 10 \times 150\ kW = 72940\ kW = 72.94\ MW$$

Having all the parameters necessary to study this scenario, Figure 4.17 shows a scenario of a typical day without the renewable energies power supply. Analysing the power consumption without EVs, it means that if the power generator can provide 203.4 MW, the 87.309 MW represents approximately 43.01 % of its power delivery and there is 56.99 % (115.731 MW) that can be used to charge EVs.

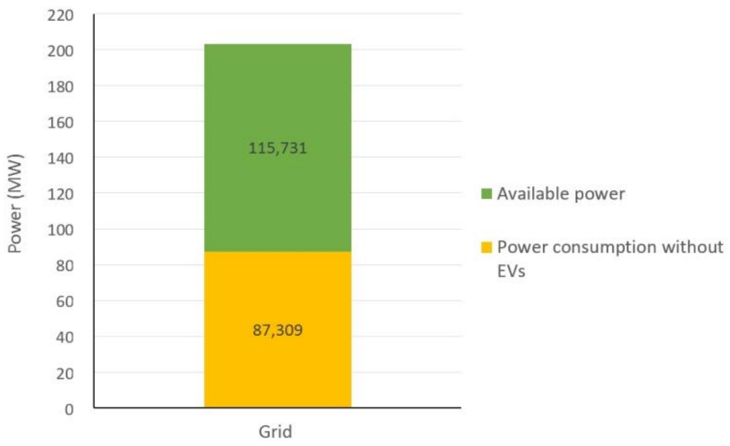

Figure 4.17 - Madeira Island typical day power consumption without renewable energies.

Introducing the EVs total power consumption (72.94 MW), as showed is Figure 4.18, the available power is reduce to 42.791 MW ($115.731\ MW - 72.94\ MW = 42.791\ MW$) representing approximately 21.07 %, considering that 100 % is 203.4 MW. This percentage of available power can be critical to provoke an alarm that the generator is reaching its full power delivery capacity, if the threshold value of the generator is 20 % to provoke an alarm because the generator is at 78.93 %, almost reaching 80 %.

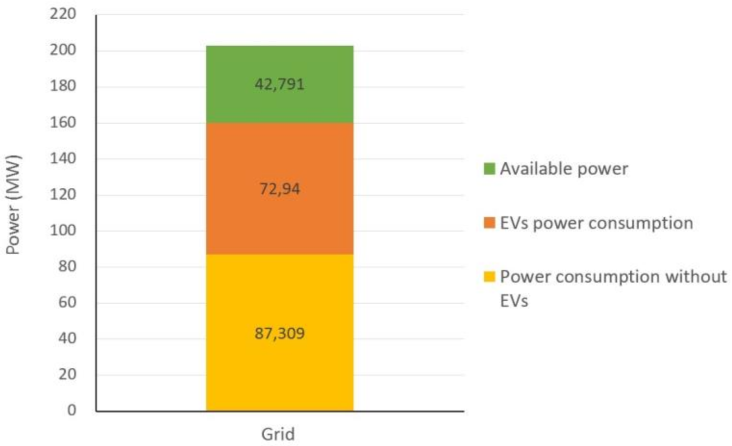

Figure 4.18 - Madeira Island power consumption with the introduction of the EVs power consumption.

Adding even more EVs charging stations (Figure 4.19), for example 5000 charging stations of 7 kW, represents adding more 35 MW to the EVs power consumption. The EVs power consumption will increase and the generator it going to be approximately at 96.16 %, meaning that it is almost at its maximum power delivery capacity (203.4 MW) and can provoke a power blackout. Furthermore, the grid may not support the increase of power do to distribution and transport limits.

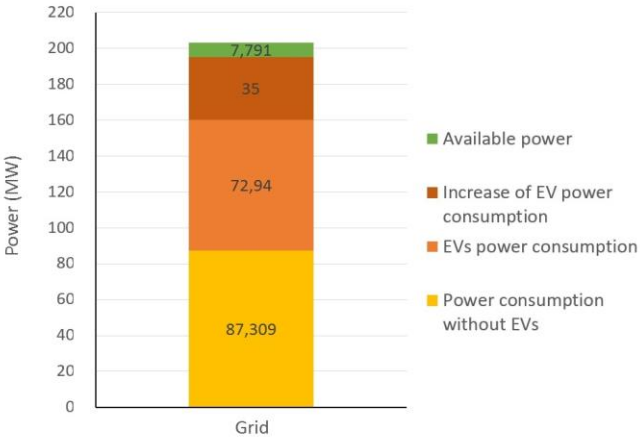

Figure 4.19 - Madeira Island power consumption adding more EVs power consumption.

According to the annual report of 2015, the average renewable energy delivery to the electrical grid was 25.4 %. Assuming that the renewable energy can provide an additional 31.224 MW of power, the new power limit is 234.264 MW. Figure 4.20 shows that with the introduction of the renewable energy the available power will increase from 7.791 MW to 39.015 MW ($7.791\ MW + 31.224\ MW = 39.015\ MW$). This might be enough to prevent a power blackout, but might not be enough to prevent an alarm on the generator if the threshold value is 20 % to provoke an alarm, because the generator is approximately at 84.62 % ($203.04\ MW - 31.224\ MW = 171.816\ MW$).

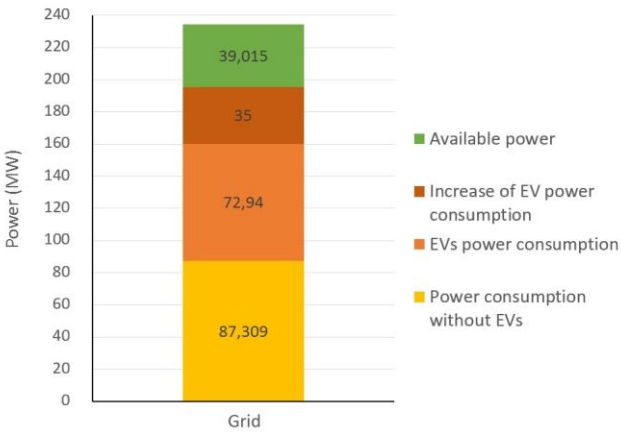

Figure 4.20 - Madeira Island power consumption with the renewable energies and EVs power consumption.

To reduce the probability of a power blackout or an alarm on the generator, the charging stations can have a control unit that is connected to the facility that regulates the generation power delivery to control the charging stations if threshold values are reached. With a smart charging scenario, like the one simulated in the subchapter 4.2, the power delivery to the EV can be reduce to a minimum of 1.38 kW (monophasic charging systems) or 4.14 kW (three phasic charging systems) and this can be enough to prevent an alarm or even a power blackout.

Considering that all the charging stations are smart and the power delivery was reduce to 4.14 kW, the EVs power consumption was reduce from 107.94 MW to 63.307 MW, which represents a power reduction of approximately 41.34 %. With this power reduction, the available power will increase to 84.648 MW ($39.015\ MW + 45.633\ MW =$

84.648 MW), meaning that the renewable energy is at 31.224 MW and the generator is at 118.392 MW allowing for example a good generation operation, considering that is operating at 58.30 % of its maximum power delivery. Figure 4.21 shows the implemented scenario where is shown the available power, EVs power consumption and the power consumption without EVs.

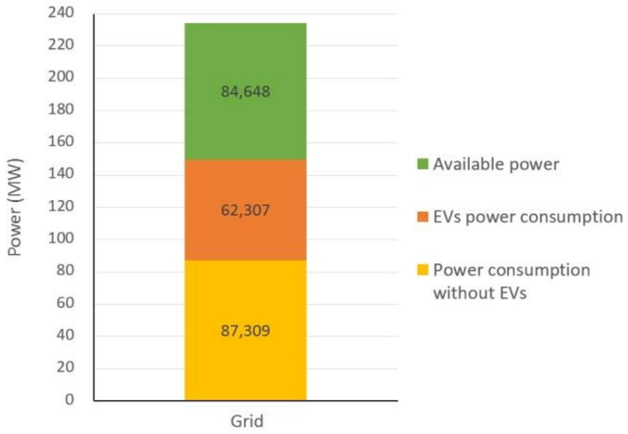

Figure 4.21 - Madeira Island power consumption with the renewable energies and using smart charging stations to manage the EVs power consumption.

Even without renewable energy, the generator will be below the threshold value (considering 20 % to reach its maximum power delivery), meaning that the generator is going to be approximately at 149.616 MW ($118.392\ MW + 31.224\ MW = 149.616\ MW$), representing 73.68 % of its power capacity. Figure 4.22 shows the implemented scenario without the renewable energy.

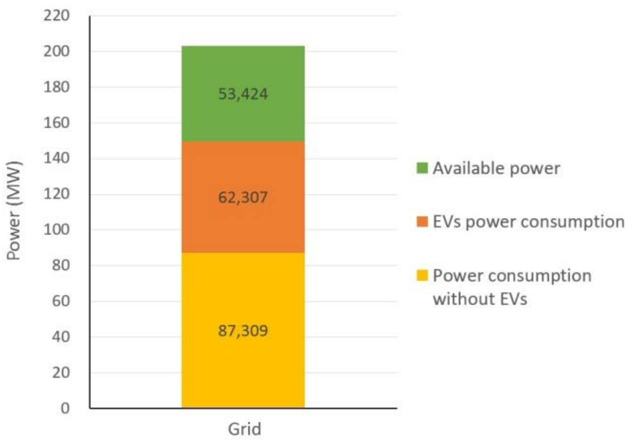

Figure 4.22 - Madeira Island power consumption with EVs power consumption, using smart charging stations, and without renewable energies.

The smart charging station can interrupt the EV charging by shutting down the power supply if there is not enough power to charge the EV at 4.14 kW, or at 1.38 kW, or even if it is needed. With this feature, the power gird blackout can be prevented even with the increase of the number of the charging stations, meaning that the EV charging can be manage, as showed in the smart charging scenario (subchapter 0), but considering the generator as the grid power supply limit, the Madeira Island power consumption as power consumption without EVs, the renewable energy as solar panels power generation and the safety power as a threshold value of the generator.

4.4. Comparison of results

The plug and charge scenario is the most conventional EV charging scheme. The EV is plugged-in and charged as any other load by applying a fixed charging power rate. This type of scenario is not the ideal to apply on a charging station because the power rate is obtain considering the maximum value of the peak consumption that occur and not the variations of the power consumption that changes throughout the day. A low power rate means a longer charging period and can be a limitation if the charging time is a priority. However, the fixed power rate applied on the charging station can be the ideal if the EV charging occurs at night, where the power consumption is lower, and when the charging time in not a limitation.

The smart charging scenario allows the EV to charge according with the available power, which is obtain considering the power consumption and the power limit. When the power consumption is lower, the available power is higher and vice-versa. The power consumption is always changing throughout the day which provokes the variation of the available power throughout the day. This variation can be obtain every 3 hours, 1 hour, 30 minutes or even every minute. The variation is important to control the EV charging because, without this variation, the smart charging can be is seen as the plug and charge scenario if each day is obtain the available power and does not change throughout the day. The smart charging scenario has better control on the EV charging if the variation to obtain the available power is lower, for example every minute or even every second is calculated the available power.

Comparing the smart charging scenario with the plug and charge scenario, the charging station has a higher power rate because it has EV power consumption management allowing to obtain every minute the power rate, instead of using a fixed power rate throughout the day. This can reduce the EV charging time and prevent a power blackout because the plug and charge scenario does not prevent a power blackout if the power consumption exceeds a certain value (exceeds the peak consumption considered). Despite of the plug and charge scenario does not consider solar panels to increase the charging power rate because the unpredicted power generation can provoke a power blackout, the smart charging scenario considers to increase even more the EV charging power rate.

With the increase of the number of EV, the charging stations will increase and the power consumption will increase on the electrical grid. Madeira Island has the unique feature of not having a power cable connected to the main land and a smart charging solution can control the sudden increase of EV power consumption or even a grid power blackout. Implementing smart charging stations that are connected to the facility that regulates the power generation is important to allow a better EV power consumption management provoked by the rising of the EV charging stations.

Overall, a smart charging solution is the best way to charge an EV, comparing with the EV charging as a fixed load. With the increase of the number of the EV, the plug and charge scenario does not consider if the grid is going to have a power blackout with the increase of the power consumption without EVs. The smart charging scenario prevents an electrical grid, building or house power blackout by reducing the EVs power consumption to safety values or even switch off the EVs that are charging if needed.

Chapter 5

Experimental verification of the EV charging system

This chapter presents the EV charging system test, the smart charging scenario implemented on the designed charging system and conclusions.

5.1. Test of the charging system

Knowing that the smart charging scenario is going to be implemented on the EV charging system, it was design a charging station with two charging points of 22 kW each based on the Chapter 3 configurations. The charging station is composed by two EM340 energy meters, two circuit breakers, two ground fault circuit breakers, two contactors, two type 2 sockets with actuators to lock the plug, a 5 VDC and 12 VDC power supply, Raspberry Pi 3 B+ with the RS485 converter and two custom circuit boards with all the hardware design on the subchapter 3.3. Figure 5.1 shows the interior of the charging station and Figure 5.2 shows the custom print circuit board with all the design hardware.

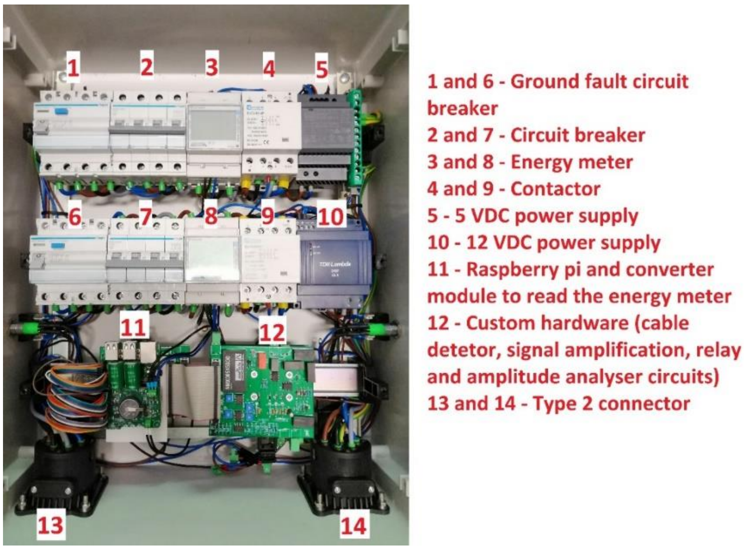

Figure 5.1 – Interior of the charging station with two charging points of 22 kW each.

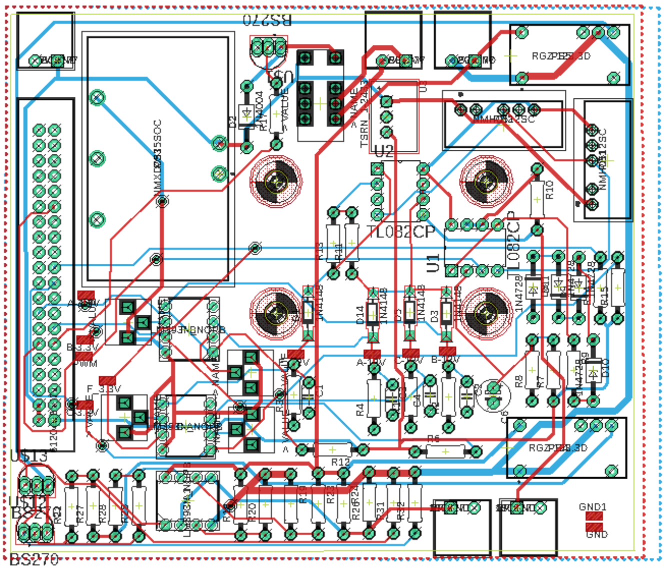

Figure 5.2 – Custom print circuit board.

The coordinator of the EV charging system is shown in Figure 5.3 where it has a Raspberry Pi 3 B+, a cooling fan, a 5 VDC power supply and ethernet connection.

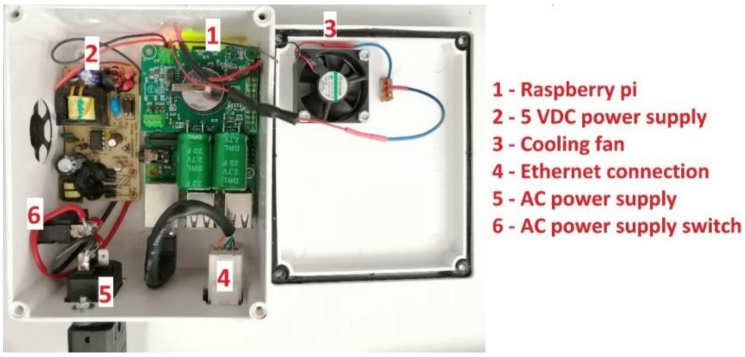

Figure 5.3 – Interior of the coordinator.

To ensure a successful EV charging process, the PWM signal provided by the two charging points of the charging station were checked before any EV connection by setting the duty cycle of the PWM signal of the Raspberry Pi 3 B+ to 50 %. Figure 5.4 shows that the PWM signal meets the requirements of the Table 2.8, such as, positive voltage amplitude is between 11.4 V and 12.6 V, negative voltage amplitude is between −11.4 V and −12.6 V and frequency at 1 kHz.

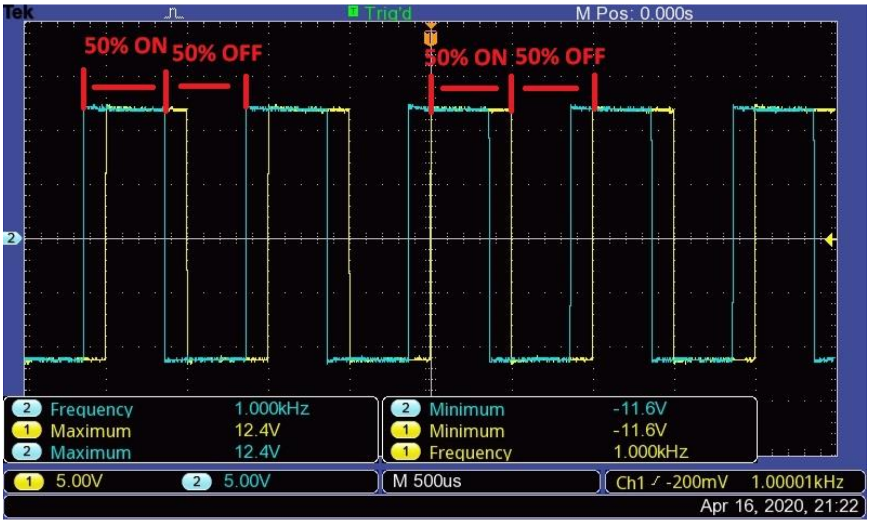

Figure 5.4 - PWM signals generated by the two charging points.

In order to check the operation of the EV charging system, a test was performed considering three charging stations with two charging points each, two EV with three phase charging systems charging at two charging stations and 30.2 kW of power available. This power was equally divided by the two EV on the server, meaning that each EV has to charge at 15.1 kW, and sent to the coordinator to deliver to the charging stations to be applied by the EVs.

Figure 5.5 shows that when the server receives data from the coordinator calculates the available power, in this case is a fixed value of 30.2 kW. With the data received, the server know that EVs are connected and ready to charge, as well as, that the minimum power allowed to charge is 4.14 kW for three phase charging ($230\,V \times 6\,A \times 3 = 4140\,W = 4.14\,kW$), according with the IEC61581 standard. Knowing that two EVs are ready to charge, the available power is equally divided and the PWM duty cycles are calculated using the Equation 3.4:

$$PWM\ duty\ cycle\ (\%) = \frac{15.1\ \times 1000}{138 \times 3} \approx 36\ \%$$

With the duty cycles obtain, the server sends to the coordinator the values to be applied by the EVs.

Figure 5.5 – Server command line interface.

Figure 5.6 shows the response received by the coordinator from the server, meaning that all the data was successfully received when it was sent data to the server by the POST request.


```
----
Received response from Server
[
  {
    "device_id": "0aca11a18257d7eb822c802e08779b8ecbd078b04c2c40c",
    "duty_cycle": 100
  },
  {
    "device_id": "84ee74a8a119b2f23e8b0db179d925141d89930ee10d0ca",
    "duty_cycle": 100
  },
  {
    "device_id": "e2c2a7b81e77712c9d676d2d563ca9c84c4ffde3ccec848",
    "duty_cycle": 36
  },
  {
    "device_id": "2a2b1bc4b9c6a963594c9cbdce12e7ffcf4d418d8e22812",
    "duty_cycle": 100
  },
  {
    "device_id": "42ec8e6d711a7f760522a0ecb966b907bc74bc1d491edf0",
    "duty_cycle": 100
  },
  {
    "device_id": "d2693dc0b4c45b62410107a9c172dc6d19976d6c5d436b1",
    "duty_cycle": 36
  }
]
----
```


Figure 5.6 – Command line data received from the server on the coordinator.

With the duty cycles received, the “device_id” is converter to charging station and charging point numbers to the coordinator identify the stations, according with the Table 5.1.

Table 5.1 - Consertion from “device_id” to charging station and charging point numbers.

device_id	Charging station number	Charging point number
0aca11a18257d7eb822c802e08779b8ecbd078b04c2c40c	1	1
84ee74a8a119b2f23e8b0db179d925141d89930ee10d0ca	1	2
e2c2a7b81e77712c9d676d2d563ca9c84c4ffde3ccec848	2	1
2a2b1bc4b9c6a963594c9cbdce12e7ffcf4d418d8e22812	2	2
42ec8e6d711a7f760522a0ecb966b907bc74bc1d491edf	4	1
d2693dc0b4c45b62410107a9c172dc6d19976d6c5d436b	4	2

After the conversion of the identification numbers, the duty cycle values are stored into the coordinator local database and the coordinator will send them when the charging station communicates. This means that when the coordinator receives data from the charging station, it sends the duty cycles to be applied on the EVs. To better understand the exchange data, Figure 5.7 shows an example of the representation of data receive by the coordinator, which contains the charging station and charging point identification number, EV connectivity, EV state and power information variables.

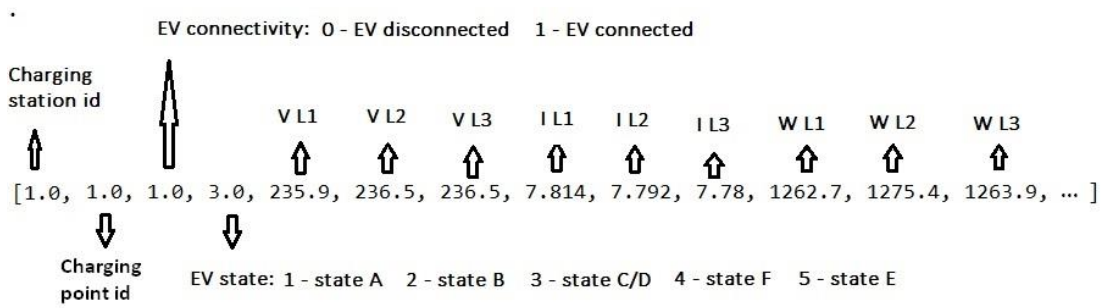

Figure 5.7 – Example of the representaion of the data exchange between the coordinator and charging station.

Figure 5.8 shows the coordinator command interface of the exchange data between the coordinator and the charging station. We can notice that when the charging station sends data, the coordinator sends all the duty cycles to the charging station. In case of the charging station number 2, the charging point number 1 is connected to an EV and when the coordinator receives data from the charging point number 2 it sends the duty cycles values for each charging point, meaning that when the coordinator receives data from the charging point number 1 it has data from the EV that is connected. This scenario occurs, as well, in the charging station number 4 where the charging point number 2 is connected to an EV. Knowing that the available power for each EV is 15.1 kW (36 % of duty cycle), the power consumption of the charging point number 1, from the charging station number 2, is 15.086 kW, which is lower than 15.1 kW. If the EV power consumption is lower than the permitted (15.1 kW), it means that there is less power being consumed by the EV, which in this case represent a safety feature. If the EV was consuming more than the permitted a power blackout can occur. This scenario occurs, as well, in the charging point number 2 of the charging station number 4 were the EV power consumption is 15.092 kW.

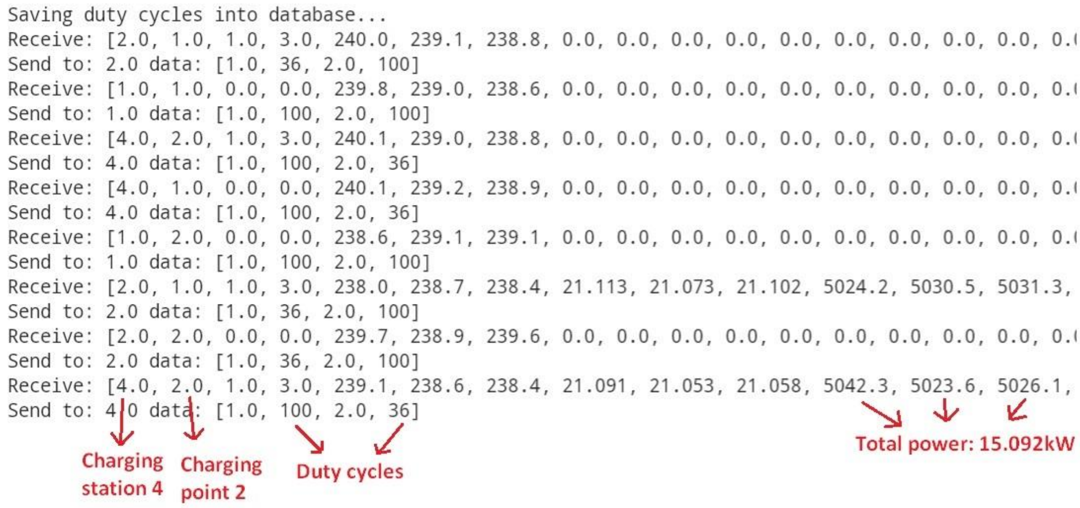

Figure 5.8 – Coordinator command line interface.

To better understand the performed test on the charging station, the EV states are represented as “State A”, “State B”, “State C/D”, “State F” and “State E”, as showed in Figure 5.9.

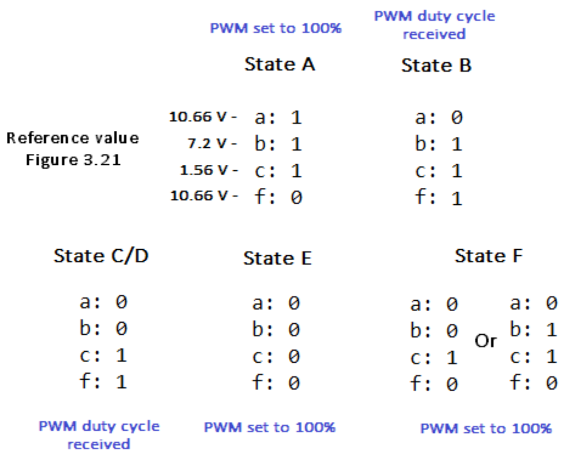

Figure 5.9 - Representation of EV states.

The charging station reads the energy meter EM340 and sends the data to the coordinator, checks if the cable is connected or not, checks the EV state and then applies the duty cycle. Figure 5.10 shows the charging station number 4 command line interface and we can notice that when the EV cable is connected the state A appear, meaning that the cable is not connected yet to the EV, but then the state C appear, meaning that the EV is connected and is ready to charge (state C). When the charging station receives the duty cycle, applies on the PWM signal when the EV is on the state C and the EV power consumption is 15.1 kW, almost 15.1 kW, as shown in Figure 5.11.

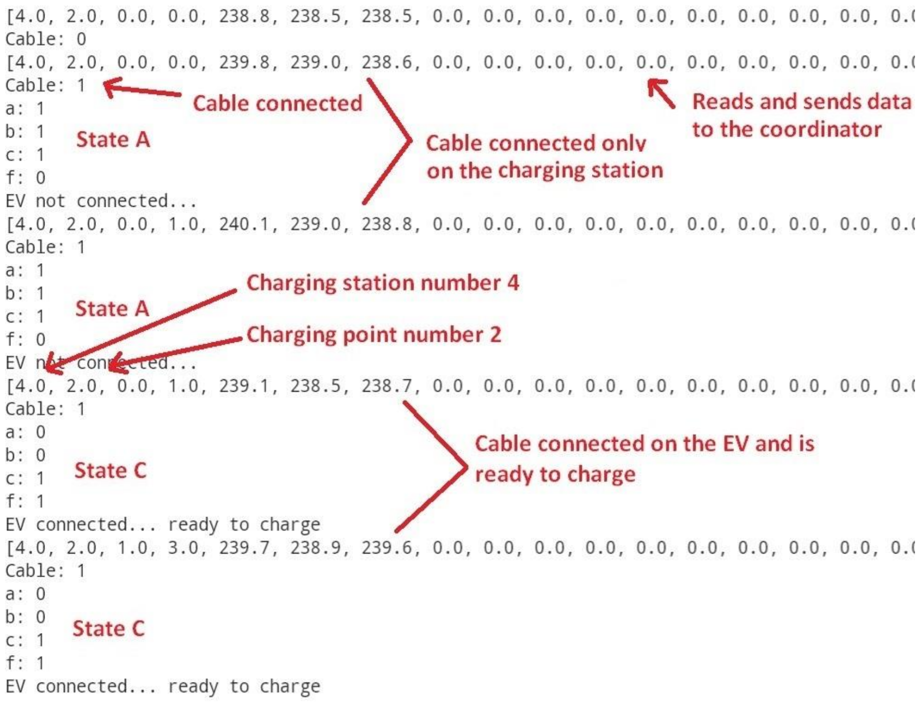

Figure 5.10 - Charging station command line interface showing the EV connection.

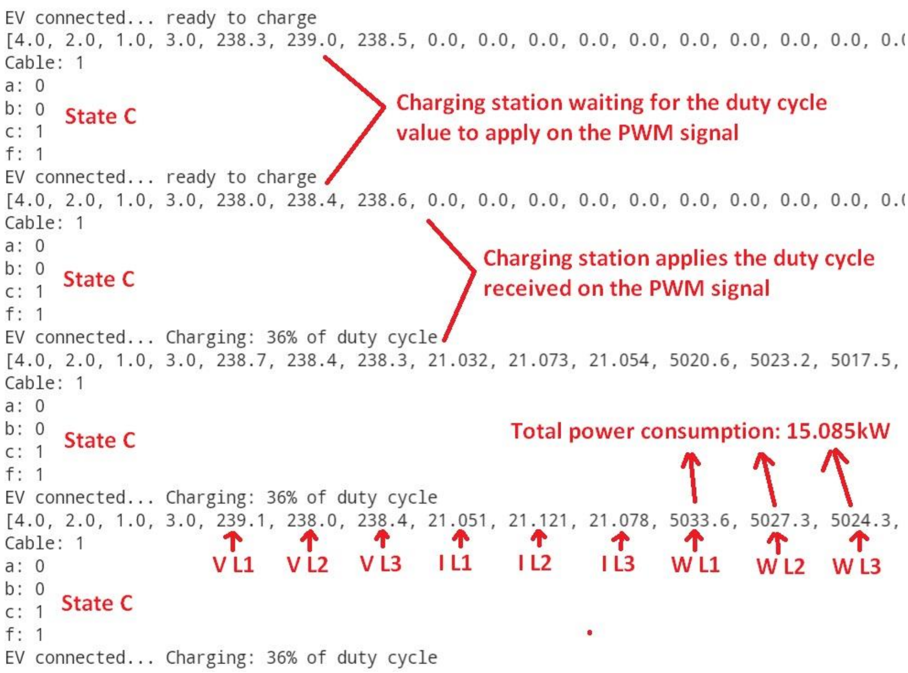

Figure 5.11 – Charging station command line interface showing the EV charging.

Figure 5.12 shows the charging stations number 2 and 4 PWM signals obtain from the control pilot pins using an oscilloscope. The positive amplitude of the signals are 6.2 V and the negative amplitude are −11.8 V showing that the EVs are in state C, according with the Table 2.8. With the EVs in state C, the duty cycles (36 %) are applied on the PWM signals and the frequencies are 1.000 kHz. The rise and fall time does not need to be verified because with a frequency of 1.000 kHz this parameters will be within 2 μs, according to Table 2.8, as shown in Figure 5.13 and Figure 5.14.

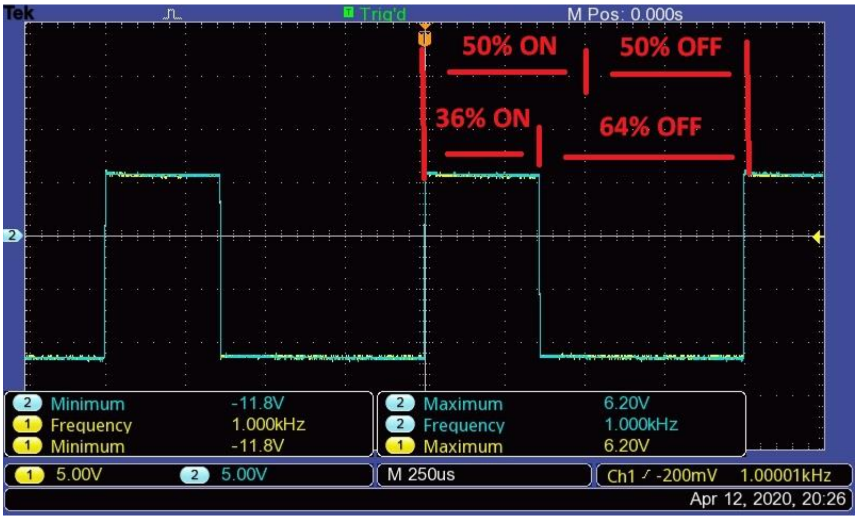

Figure 5.12 – Oscilloscope control pilot signals obtain from charging stations number 2 and 4.

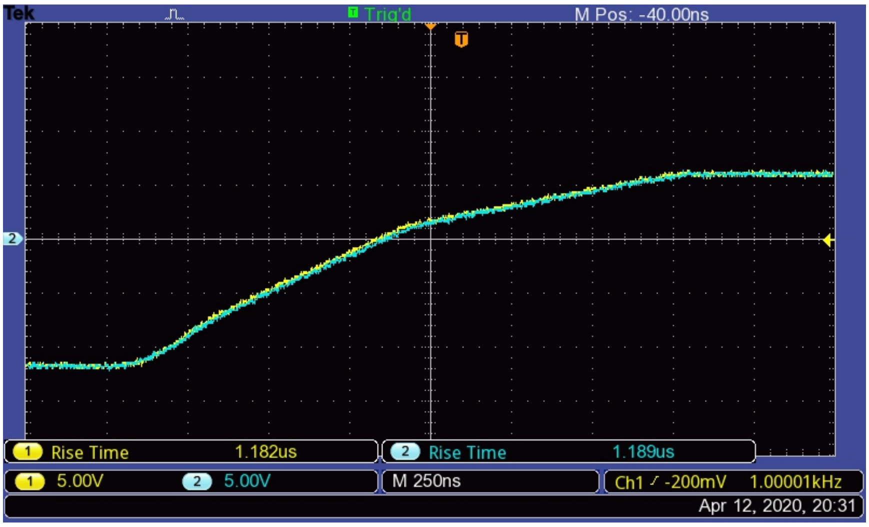

Figure 5.13 - Oscilloscope rise times obtain from the PWM signals of the charging station number 2 and 4.

Figure 5.14 – Oscilloscope fall times obtain from the PWM signals of the charging station number 2 and 4.

With the charging station applying the duty cycle on the PWM signal, the coordinator received the data of the EV power consumption and was deliver to the server. Figure 5.15 and Figure 5.16 shows the web page interface where the EV of the charging station 3 (charging station 2 charging point 1), and 6 (charging station 4 charging point 2) are consuming 15.088 kW and 15.095 kW of power, respectively, meaning that the server is receiving data from the charging stations.

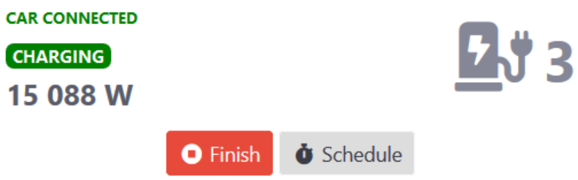

Figure 5.15 – Web page interface of the charging station number 2 charging point number 1.

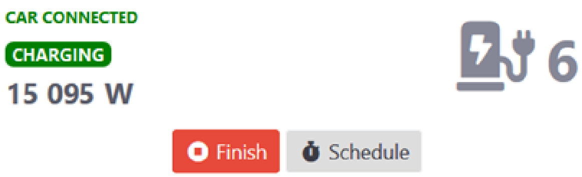

Figure 5.16 - Web page interface of the charging station number 4 charging point number 2.

The test of the EV charging system was successfully accomplish and the server was able to receive and send data to the coordinator. The charging station was able to receive data from the coordinator, generate the PWM signal according to the IEC61581 requirements, detect the EV state and modify the duty cycle according to the available power. After this test, the implementation of the charging scenario is possible to manage all the charging stations and EVs power consumptions, according with the available power.

5.2. Charging scenario

To test the smart charging scenario, three charging stations with two charging points allowing three-phase charging and the coordinator were installed on the Electricity Company of Madeira facilities to provide a safe space to test the EV charging scenario. It was considered the power consumption of 8 houses without solar panels to form a building with 8 apartments, meaning that the power limit is 41.4 kW (Equation 4.1), 6 EVs charging at the charging stations and a safety power obtain every minute considering 10 minutes time frame.

The safety power was obtain using the Equation 3.7 and updated every minute considering 10 minutes time frame. The first 10 minutes of the safety power was obtain considering the 10 minutes of the day before power consumption without EVs. Figure 5.17 shows the day before power consumption without EVs used to calculate the first 10 minutes of the safety power, the safety power and the power consumption without EVs obtain on the test.

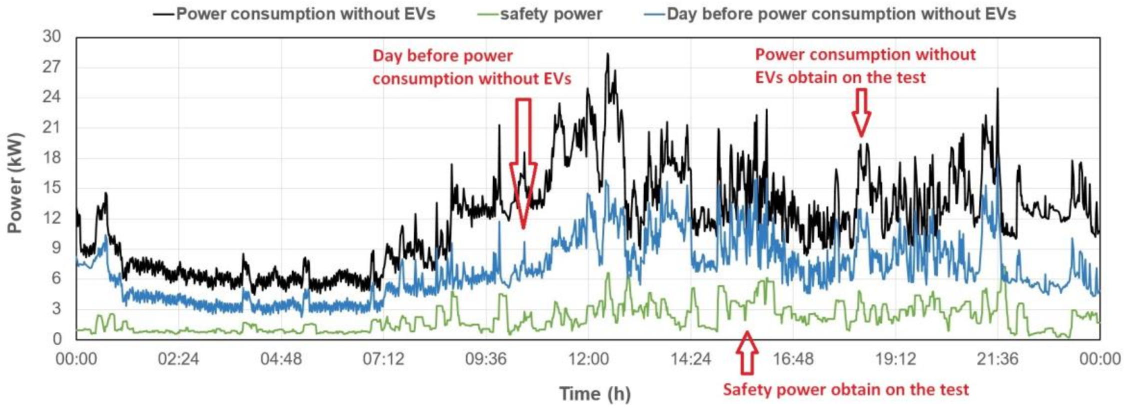

Figure 5.17 – Day before power consumption without EVs, the power consumption without EVs and safety power obtain on the smart charging experimental test.

Considering the charging stations identification numbers on the web page (Figure 3.33), the charging station number 6 (also known as EV6), the EV was charging between $12{:}17\ \mathrm{h} - 17{:}34\ \mathrm{h}$, meaning that the EV was plugged-in at $12{:}17\ \mathrm{h}$ and disconnected at $17{:}34\ \mathrm{h}$, and the schedule method was used to schedule 5 EVs charging sessions:

- On the charging station number 1 (also known as EV1) was set the beginning of the EV charging session at $22{:}03\ \mathrm{h}$ and the end at $03{:}30\ \mathrm{h}$;
- On the charging station number 2 (also known as EV2) was set the end of the EV charging session at $02{:}34\ \mathrm{h}$ and the EV started charging at $14{:}46\ \mathrm{h}$;
- On the charging station number 3 (also known as EV3)) was set the beginning of the EV charging session at $16{:}54\ \mathrm{h}$ and the end at $23{:}44\ \mathrm{h}$;
- On the charging station number 4 (also known as EV4)) was set the beginning of the EV charging session at $14{:}25\ \mathrm{h}$ and the end at $18{:}11\ \mathrm{h}$;
- On the charging station number 5 (also known as EV5) was set the beginning of the EV charging session at $01{:}02\ \mathrm{h}$ and the EV stopped charging at $10{:}02\ \mathrm{h}$.

Knowing all the EVs charging schedules, the smart charging results can be observed in Figure 5.18. The power consumption was increased comparing with the power consumption without EVs, meaning that the EVs where charging at the charging stations. When the power consumption without EVs has higher variations, the safety power has, as well, higher variations, meaning that the available power is lower on those variations. When the variations are lower, the safety power variations are lower and the available power is higher on those variations. This can be seen in two time frame between $21{:}36\ \mathrm{h} - 09{:}36\ \mathrm{h}$ and $09{:}36\ \mathrm{h} - 21{:}36\ \mathrm{h}$, where the time frame $21{:}36\ \mathrm{h} - 09{:}36\ \mathrm{h}$ is seen the lowest power consumption without EVs variations and the time frame $09{:}36\ \mathrm{h} - 21{:}36\ \mathrm{h}$ is seen the highest variations. The power limit was never exceeded meaning that a power blackout never occur throughout the 24 hours.

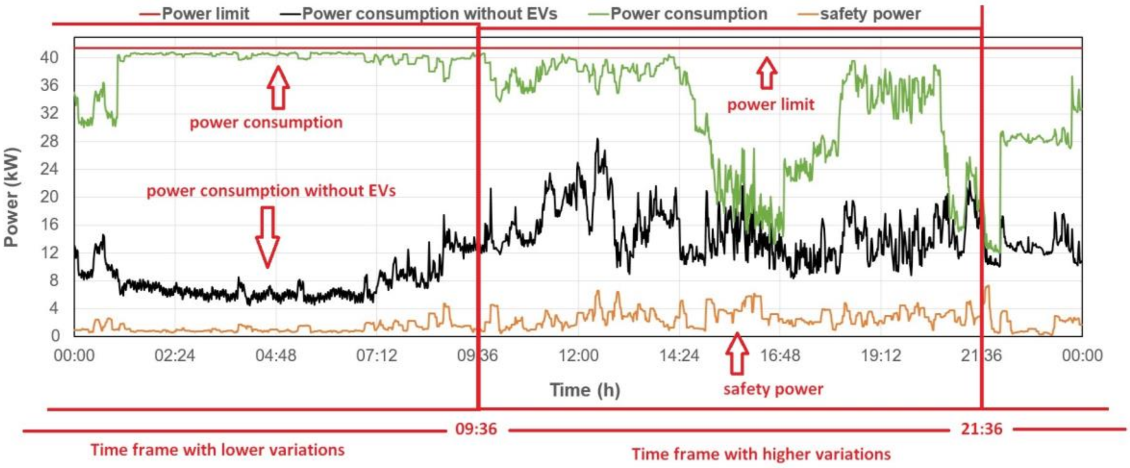

Figure 5.18 – Experimental smart charging results.

Adding the EVs power consumption profile, as showed in Figure 5.19, the EVs are charging, between $16{:}00$ h $-$ $16{:}48$ h there was two EVs charging at the charging stations number 4 and 6, and the power consumption of these two EVs where below the available power, since it is available at least 13.4 kW ($41.4\ kW - 28\ kW = 13.4\ kW$). This means that the EVs where charging at inferior power rates because there batteries where almost at full capacity. At $21{:}36$ h the same scenario happen because the charging stations number 2, 3 and 5 where charging EVs and these ones where charging at inferior power rates meaning that its batteries where almost at full capacity. Throughout the 24 hours it is notice that the EVs power consumption was manage, according to the limitation of the power limit and the safety power. In some cases, the EVs power consumption was limited to 22 kW because only one charging station was charging an EV and the maximum power that a charging station can deliver is 22 kW, even when there is more available power.

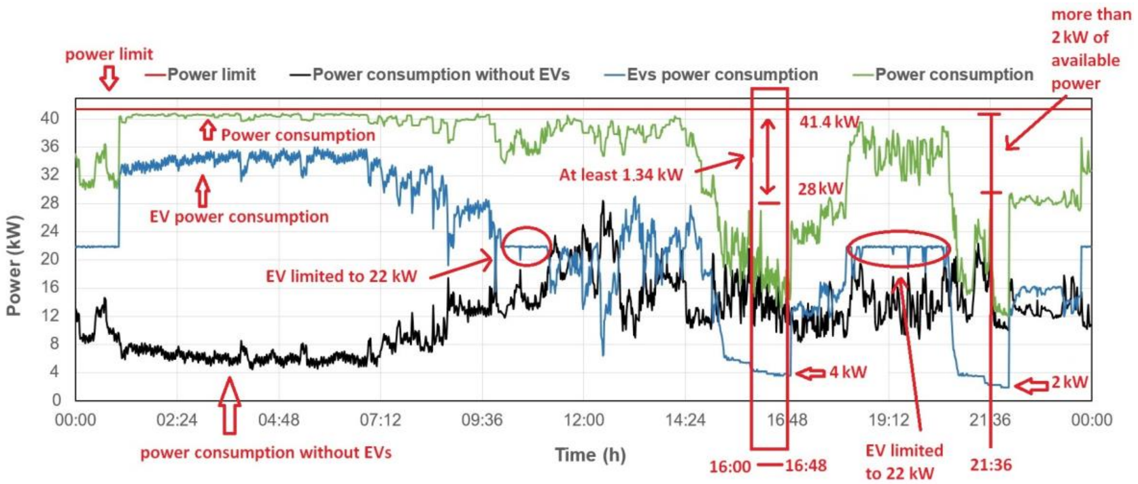

Figure 5.19 - Experimental smart charging results showing the EVs power consumption profile.

Figure 5.20 shows that, between $16{:}00$ h $-$ $16{:}48$ h, two EVs where charging and almost reaching its full battery capacity because before $19{:}12$ h the EVs are disconnected. At $21{:}36$ h the same scenario happen with one EV where it is almost reaching its full battery capacity and before $00{:}00$ h the EV is disconnected.

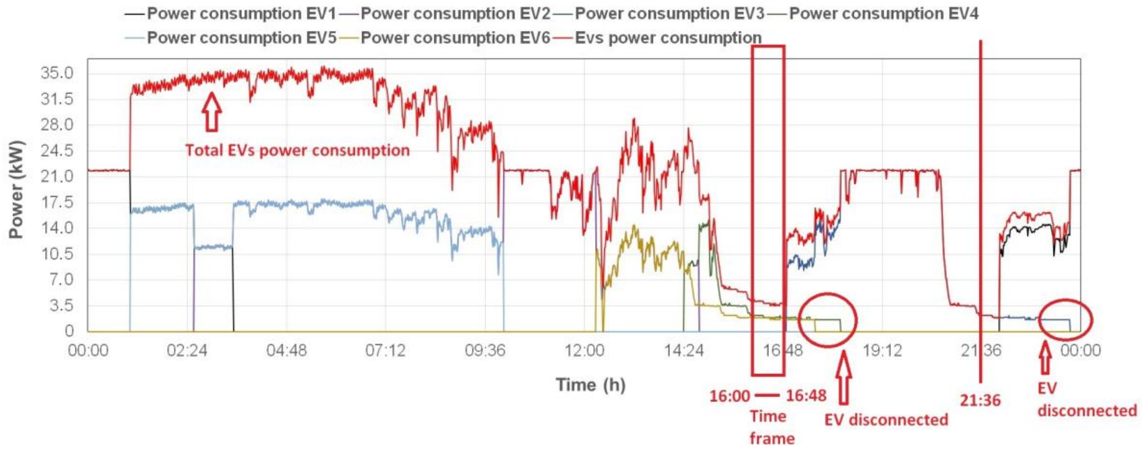

Figure 5.20 - Experimental smart charging scenario showing each EV power consumption and the total EVs power consumption.

Analysing each EV power consumption shown in Figure 5.21, 3 EVs where almost at full capacity, meaning than EV3, EV4 and EV6 where almost fully charge at the end of the charging session. It is also notice that 3 EVs where charging at the maximum charging station power delivery (22 kW), meaning in short periods of time the EV1, EV2 and EV3 where charging at full power delivery. All the EVs where charging according the available power and the priority system. The EV1 power consumption has 3 level of power consumption. On the first level, the EV is charging at the maximum power delivery of the charging station, which is 22 kW. On the second level, the EV power consumption was reduce to almost 17.5 kW because the EV6 started to charge and the power consumption is the same as the EV1. On the third level, the EV1 power consumption was reduce even further, to almost 11.25 kW, because the EV2 started to charge and the power consumption is the same as the EV1 and EV6. With this, the charging priority system was able to equally divide the available power by the EVs. Others scenarios that are visualize the charging priority system are the EV2 and the EV5 power consumption profile and the EV2 and the EV6 power consumption profile. On the EV2 and EV5, it is notice that between $02:24$ h and $09:36$ h the power consumption profile is the same, meaning that the available power was equally divided by two EVs. The same scenario happen on the EV2 and EV6 between almost $12:00$ h and $14:24$ h, where the power consumption has the same profile.

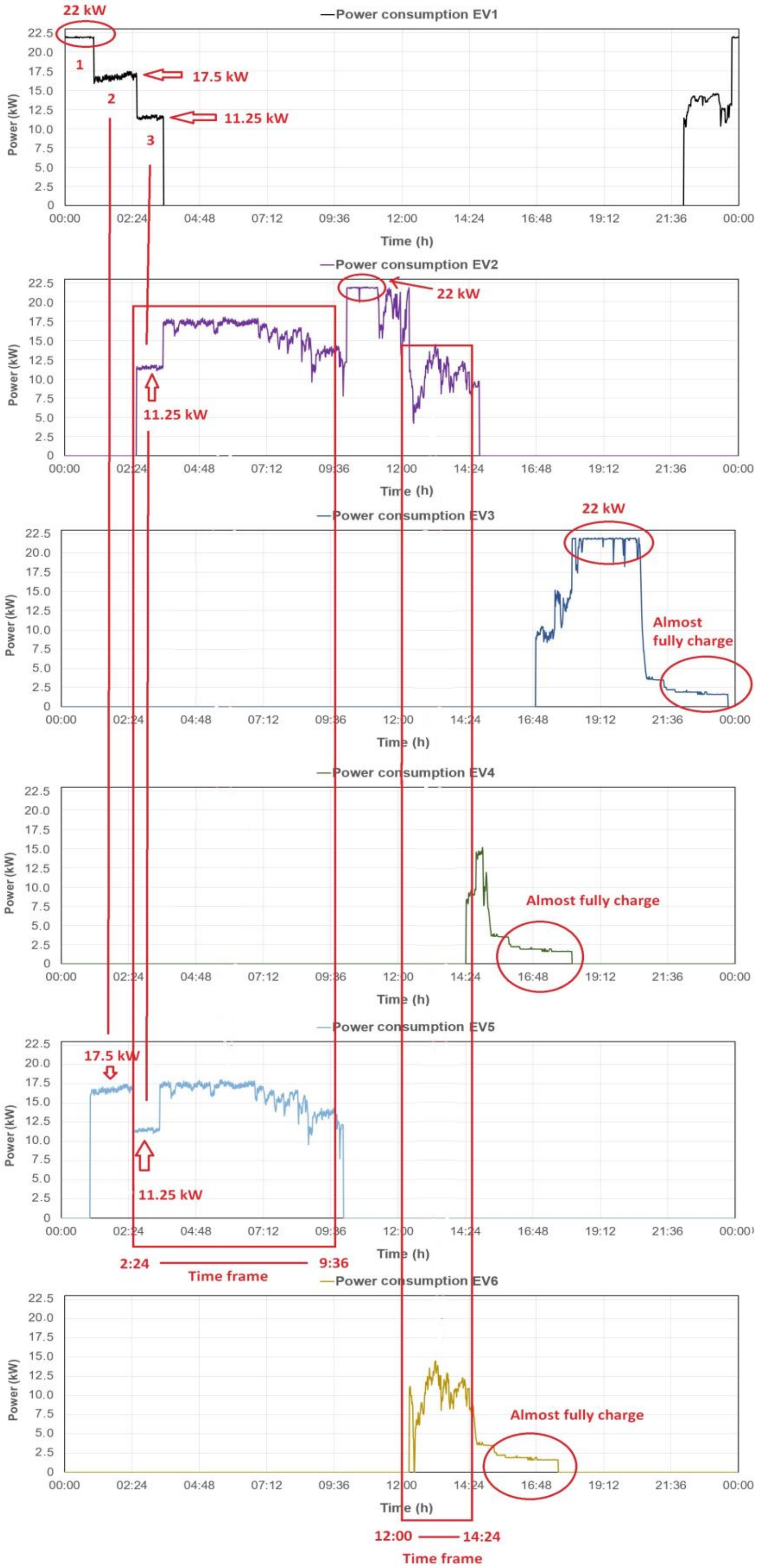

Figure 5.21 - Experimental smart charging scenario showing each EV power consumption separately.

Figure 5.22 shows that only the EV6 charging session was interrupted by the priority system. Approximately at $12:28$ h, the EV6 power consumption is 0 W meaning that there was not available power to charge the EV6 when the EV2 was charging. The available power at $12:28$ h was approximately 7.5 kW and this power divided by two EVs is much lower than the minimum permitted by a three-phase charging system, which is 4.14 kW, and, since the EV2 was charging first, the EV6 charging session was interrupted because the available power was lower than the permitted.

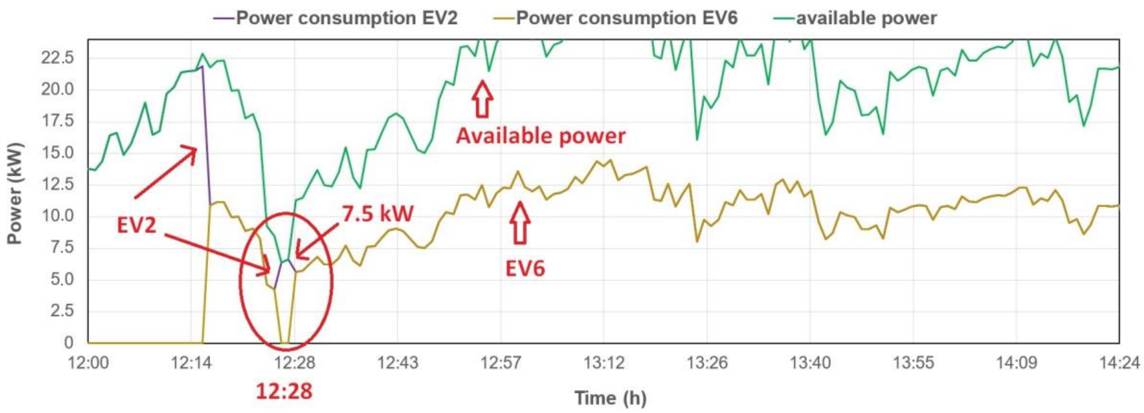

Figure 5.22 - Experimental smart charging scenario showing the priority system disconnecting an EV.

The power consumption profiles are obtained by applying the Equation 3.8. The available power that each EV is allow to charge is set by the PWM duty cycle (Equation 3.4), meaning that the coordinator sends the power information variables, charging station number, charging point number, EV state and EV connectivity to the server and this sends the PWM duty cycle to the coordinator for this to send to the charging station to be applied on the EV.

Figure 5.23 shows the PWM duty cycle applied on the EVs according with the available power (Equation 3.8). When the PWM duty cycle goes to 0 % it means that the available power was lower than the permitted by the three-phase charging, which is 4.14 kW, or the EV stopped charging because it was schedule the end of the EV charging session or the EV was disconnected. Comparing each EV power consumption with each PWM duty cycle, the profile is the same between $00:00$ h and $14:24$ h where 22 kW represents the PWM duty cycle at 53 % ($\frac{22\times1000}{138\times3} \approx 53$ %), 16.5 kW represents 40 % ($\frac{16.5\times1000}{138\times3} \approx 40$ %), 11.2 kW represents 27 % ($\frac{11.2\times1000}{138\times3} \approx 27$ %), 0 W represents 0 % and so on. However, between $14:24$ h and $00:00$ h, the PWM duty cycles and EVs power consumption has different profiles. One of the different profiles is between $14:24$ h and $16:48$ h where the EV4 and EV6 are charging and it is notice that the EVs power consumption are lower than the permitted by the PWM duty cycles. In this time frame there is a power consumption value of the EV4 and EV6 that was approximately 2.5 kW representing a PWM duty cycle of approximately 6 % ($\frac{2.5\times1000}{138\times3} \approx 6$ %). This PWM duty cycle is below the PWM duty cycle permitted, which is 10 % ($\frac{4.14\times1000}{138\times3} = 10$ %), and the PWM duty cycles sent to the charging stations to be applied on the EVs were approximately 30 % meaning that the EVs are almost at full capacity because there was at least 12.42 kW ($\frac{30\ \%\times138\times3}{1000} = 12.42\ kW$) of power available.

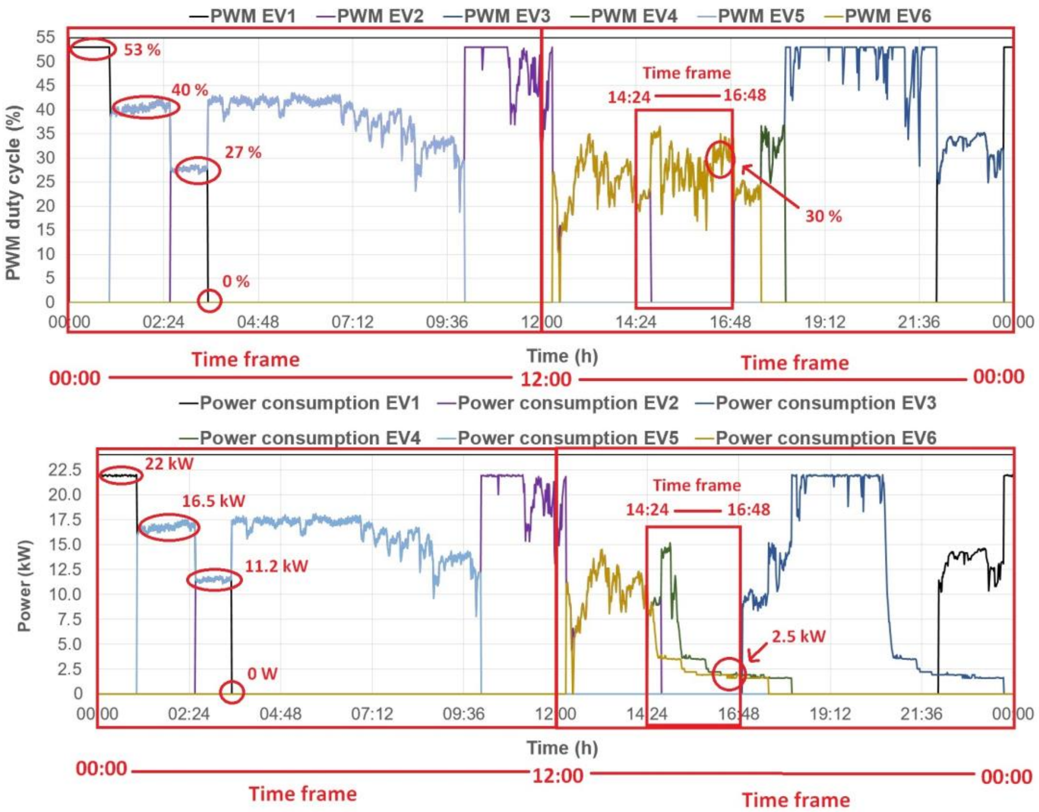

Figure 5.23 - Experimental smart charging scenario showing the PWM duty cycles sent to the charging stations and the each EV power consumption, according to the PWM duty cycle.

With the power consumption without EVs, power limit, safety power, EVs power consumption and the EVs PWM duty cycles, the EV charging system was able to communicate with the server, coordinator and charging stations to implement the smart charging scenario. The best way to charge EVs is when the power consumption without EVs has lower variations, meaning that the safety power is lower and the available power is higher resulting on a higher charging power rate. The lowest variations observed was between $21{:}36$ h and $09{:}36$ h, where the safety power variations are the lowest possible because the power consumption without EVs is the lowest possible, meaning that the available power is the highest possible allowing the EVs to charge at higher charging power rates.

5.3. Comparison with simulation results

The EV charging system was successfully implemented and the PWM duty cycles calculated by the server were delivered to the coordinator to send to the charging stations. The charging station was able to successfully apply the duty cycle on the PWM signal and the EV was able to accept the PWM signal by being ready to charge. The custom printed circuit board, with all the design circuitry, was able to receive the PWM signal from the Raspberry Pi 3 B+ and amplify to the IEC61581 requirements, as well as, detect the EV states and act according with each state by applying the correct duty cycle value, for example, in state C, or B, applies the duty cycle received from the coordinator and in the others states applies a duty cycle of $100\,\%$ because an error occur or the EV is not connected or is not ready to charge.

On the implemented charging scenario, the power consumption, most of the time, was near the power limit, but was never exceeded, and it was increased comparing with the power consumption without EVs. The available power was well manage and deliver to the EVs throughout the PWM duty cycles. When the available power is the highest possible, the safety power is the lowest possible and the PWM duty cycle is the maximum value possible, according with the maximum charging station power delivery. This situation was verified when the power consumption without EV was the lowest possible.

The priority system was able to divide the available power equally among all the connected EVs, when more than one EV were connected at the same time. In one situation, the available power divided by two EVs was lower than the permitted by the three-phase charging system and the EV that arrived first was able to continue charging, meaning than the available power was higher than the permitted, and the other EV charging session was interrupted.

Each EV power consumption, most of the time, has the same profile as the PWM duty cycle, considering that 53 % is 22 kW. The PWM duty cycle is the most important variable in the charging station and is able to impose a variable power consumption limit on the EV. However, sometimes the power consumption limit is not accepted by the EV when the EV battery is almost at fully capacity. With the battery almost at fully capacity, the EV limits the power consumption to a certain value and applies the value, if the PWM duty cycle allows the EV to charge a higher power rate.

Comparing with the simulation results, the smart charging scenario implemented was able to manage throughout the 24 hours the EV power consumption according with the available power, which depends on the power consumption and power limit and priority system. The simulation results does not show the EV power consumption at the end of its full battery capacity because the EV charges, or not, at the establish power rate, meaning that, depending on the battery capacity, the EV will charge at lower power rates than the permitted. If the EV is charging at lower power rate, there is more available power that can be shift to other EV that is charging and the simulation results does not show this scenario.

The implemented smart charging scenario does not consider solar panels power supply and the simulation scenario has higher charging power rates because of that power injection. However, the control of the EV power consumption was accomplish and in one situation an EV charging session was interrupted because the power rate was lower than the permitted.

Both smart charging results (simulation and experimental) shows that the best time frame to charge an EV is at night because the power consumption without the EVs is lower, meaning that the available power is higher because the safety power is lower. However, an EV can be plugged-in and ready to charge at any time if the available power is higher than the permitted, for example, a three phase charging system the minimum power allowed is 4.14 kW and for a single phase charging system is 1.38 kW.

Overall, the smart charging scenario, the communication between the server, coordinator, charging stations and EVs where successfully implemented and none of the charging stations were shut down during the charging sessions because the power consumption never exceeded the power limit and was according with the available power.

Chapter 6

Conclusions

This chapter presents the overall conclusions about the EV charging system and the future work to improve the implemented charging system.

6.1. Overall conclusions

On this project it was study the EV history, types, advantages and disadvantages regarding the internal combustion engines, batteries types and the EV description, to acquire the EV background.

The main function of the EV is to provide mobility services expected by its user. At higher penetration levels, uncontrolled EV charging can congest the power grid operation and smart charging stations can control the EV charging session by imposing the charging power rates, leading to an improvement of the power grid operations.

Inductive charging is the most common method used because of the charging mode 3. This mode uses the type 2 connector for the EV connection and the charging station can have a maximum power delivery up to 50 kW.

The most conventional charging scenario for charging EVs is the plug and charge. The charging process starts immediately when the user arrives at the charging station and plugs the EV to charge as a fixed load. However, the smart charging scheme is ideal for smart grid application.

The EV charging stations on the market for building and house do not consider the power consumption status to control the EV charging. Instead, the EVs are plugged directly to a power outlet that monitors the energy consumption and can schedule the EV charging. To provide a system that is able to read and communicate the power consumption in real-time is important to implement energy management algorithms. Designing the charging station hardware is a critical element to develop an EV charging system that can apply energy management algorithms.

EVs charging strategies focuses on improving the charging capabilities, reducing power demand peaks and exploiting the overnight demand valley filling. Most of these strategies relies on simulations to assess their proposals and does not meet with the EV charging requirements and specifications.

A layer of communication exists between the charging station and the EV. The EVs are potentially capable of having their power requirements served according to a power rate value controlled by the PWM signal. The IEC61851 standard does not dictate that the EV should necessarily charge at the maximum power rate permitted by the charging station and a lower power rate is also considered a normal charging behaviour.

The EV charging system prototype is compose by a coordinator, server and charging station. With the interaction of this three components, the EV power consumption can be remotely controlled by a server that has energy management algorithms that considers the building or house power consumption status and power limit. This is done by modifying the PWM signal applied by the charging station on the EV, representing a way of introducing controllable charging without the need to modify locally the PWM signal.

In the simulations, the plug and charge scenario is not the ideal to apply on the developed charging system because the charging power rate is fixed, low and limited to the maximum value of the peak consumption. If only night charging is permitted the power rate has a considerable value to charge the EV, comparing with the power rate applied throughout the 24 hours. In this scenario, a power blackout event is more likely to occur if the peak power consumption has an abnormal value and if the unpredicted renewable power generation is considered.

The smart charging scenario, comparing with the plug and charge, is the best charging scenario allowing the EV to charge according the available power, which is obtain considering the power consumption status and power limit. Throughout the tests the available power was well manage and distributed equally to the EVs. This charging scenario can prevent a power blackout because the charging power rate changes according with the available power and can consider renewable energy power generation to increase even more the charging power rate.

Madeira Island has the unique feature of not having a power cable connected to the main land and, with the increase of the charging stations, a smart charging solution can control the sudden increase of EV power consumption or even prevent a blackout on the electric grid.

The smart charging scenario was implemented on the developed charging system. The communication between the server, coordinator, charging stations and EVs where successfully implemented and none of the charging stations was shut down during the charging sessions. The available power was well manage and deliver to the EVs by the charging stations throughout the PWM signals. When the available power is the highest possible, the PWM duty cycle is the maximum value possible, which is 53 %, according with the maximum charging station power delivery (22 kW).

Comparing the experimental and simulation results, the experimental smart charging scenario does not consider solar panels power generation, but the implemented scenario was able to distribute equally the available power by all the connected EVs and control the EV power consumption. Both smart charging results shows that a power blackout never occur and the best time to charge an EV is when the power consumption without EVs is the lowest possible, representing the highest value of available power possible.

6.2. Future work

The developed EV charging system can be implemented on houses with renewable energy to be able to analyse the EV integration, leaving work to explore more energy monitoring by integrating on the coordinator all the systems and aggregate all power information.

The coordinator communication with the server can be reduce to seconds to allow a faster response time, instead of every minute, by responding more quickly to an available power change.

In the charging station a display can be mounted to allow the EV user interaction without the need to access the web page to schedule or stop the EV charging session.

An approach to control the EV power consumption is using forecast algorithms to anticipate the power consumption and obtain the safety power to be implemented on the smart charging system.

Other area that can benefit from this system is the integration of a smartphone app to show real-time EV power consumption and power consumption pricing, as well as, to allow the full access to the EV charging session, such as, schedule, change power rate and to stop the EV charging.

References

[1] IEA, “Global EV Outlook,” 27 May 2019. [Online]. Available: https://www.iea.org/publications/reports/globalevoutlook2019/. [Accessed: 5 August 2019].

[2] V. Monteiro, J. A. Afonso and J. L. Afonso, “Vehicle: Electrification: New Challenges and Opportunities for Smart Grids,” *Energies,* vol.12, p. 118, 2018.

[3] R. Faria, P. Moura, J. Delgado and A. T. Almeida, “Energy Conversion and Management: A sustainability assessment of electric vehicles as a personal mobility system,” *Elsevier,* vol. 61, pp. 19-30, 2012.

[4] K. Gyimesi and R. Viswanathan, “The shift to electric vehicles,” IBM Global Business Services, New York, 2011.

[5] K. T. Chau e C. C. Chan, Modern Electric Vehicle Technology, United Kingdom: Oxford University Press, 2001.

[6] C. C. Chan, “The Past, Present and Future of Electric Vehicle Development,” in *3rd Internationl Conference on Power Electronics and Applications*, Hong Kong, 2009.

[7] C. C. Chan, “The State of the Art of Electric and Hybrid Vehicles,” *Proceedings of the IEEE,* vol. 90, pp. 247-275, 2002.

[8] B. Custers, The Future of Drone Use Opportunities and Threats from Ethical and Legal Perspectives, Netherlands: Springer, 2016.

[9] Edison Tech Center, “Electric Cars,” 2014. [Online]. Available: https://edisontechcenter.org/ElectricCars.html. [Accessed: 05 August 2019].

[10] A. Emadi, K. Rajashekara, S. S. Williamson and M. Lukic, “Topological Overview of Hybrid Electric and Fuel Cell Vehicular Power System Architectures and Configurations,” *IEEE Transactions on Vehicular Technology,* vol. 54, pp. 763-770, 2005.

[11] U. S. Department of Energy, “Electric Vehicles Basics,” [Online]. Available: https://www.energy.gov/eere/electricvehicles/electric-vehicle-basics. [Accessed: 07 August 2019].

[12] C. C. Chan, “The State of the Art of Electric, Hybrid, and Fuel Cell Vehicles,” *Proceedings of the IEEE,* vol. 95, pp. 704-718, April 2007.

[13] M. Rousselle, *Impact of the Electric Vehicle on the Electric System*, KTH Electrical Engineering, 2009.

[14] A. Dinger, R. Martin, X. Mosquet, M. Rabl, D. Rizoulis, M. Russo and G. Sticher, “Batteries for Electric Cars: Challenges, Opportunities, and the Outlook to 2020,” The Boston Consulting Group, Boston, 2010.

[15] C. Iclodean, B. Varga, N. Burnete, D. Cimerdean and B. Jurchis, “Comparison of Different Battery Types for Electric Vehicles,” in *IOP Conference Series: Materials Science and Engineering*, Indonesia, 2017.

[16] C. Liu, K. T. Chau, C. Wu and S. Gao, “Opportunities and Challenges of Vehicle-to-Home, Vehicle-to-Vehicle, and Vehicle-to-Grid Technologies,” *Proceedings of the IEEE*, vol. 101, pp. 2409-2427, 2013.

[17] A. Khaligh and Z. Li, “Battery, Ultracapacitor, Fuel Cell and Hybrid Energy Storage Systems for Electric, Hybrid Electric, Fuel Cell and Plug-In Hybrid Electric Vehicles: State of the Art,” *IEEE Transactions on Vehicular Technology*, vol. 59, pp. 2806-2814, 2010.

[18] Battery University, “BU-107: Comparison Table of Secondary Batteries,” 06 08 2019. [Online]. Available: https://batteryuniversity.com/learn/article/secondary_batteries. [Accessed: 07 August 2019].

[19] Beama, “Guide to Electric Vehicle Infrastructure,” 08 04 2015. [Online]. Available: http://www.beama.org.uk/resourceLibrary/beama-guide-to-electric-vehicle-infrastructure.html. [Accessed: 07 August 2019].

[20] C. Mahmoudi, A. Flah and L. Sbita, “An Overview of Electric Vehicle Concept and Power Management Stratagies,” in *International Conference of Science and Electrical Technologies of Maghreb*, Algeria, 2015.

[21] A. Emadi, Y. J. Lee e K. Rajashekara, “Power Electronics and Motor Drives in Electric, Hybrid Electric, and Plug-In Hybrid Electric Vehicles,” *IEEE Transactions on Industrial Electronics*, vol. 55, pp. 2237-2245, 2008.

[22] Arthur D Little, “A Comparison between EVs and standard Combustion Vehicles,” 15 08 2016. [Online]. Available: https://www.adlittle.com/en/insights/viewpoints/a-comparison-between-evs-and-standard-combustion-vehicles. [Accessed: 07 August 2019].

[23] Arthur D Little, “Battery Electric Vehicles vs. Internal Combustion Engine Vehicles,” 29 11 2016. [Online]. Available: https://www.adlittle.com/en/insights/viewpoints/battery-electric-vehicles-vs-internal-combustion-engine-vehicles. [Accessed: 07 August 2019].

[24] M. B. Nina, *Introduction of Electric Vehicles in Portugal*, Lisboa: Instituto Superior Técnico, 2010.

[25] M. H. Alkhafaji, P. C.-K. Luk e A. F. Bati, “Integration of electric vehicles within microgrid,” in *11th International Conference on Control*, Belfast, 2016.

[26] K. M. Tan, V. K. Ramachandaramurthy and J. Y. Yong, “Integration of electric vehicles in smart grid: A review on vehicle to grid technologies and optimization techniques,” *Renewable and Sustainable Energy Reviews,* vol. 53, pp. 720-732, 2016.

[27] A. S. Hassan, A. Firrincieli, C. Marmaras, L. M. Cipcigan and M. A. Pastorelli, "Integration of Electric Vehicles in a Microgrid with Distributed Generation,*"49th International Universities Power Engineering Conference,* Romania, 2014.

[28] S. S. Hosseini, A. Badri and M. Parvania, “Smart Parking Lot to Minimize Residential Grid Losses Based on Customer Priorities,” em *International Conference on Power, Energy and Control*, Iran, 2013.

[29] K. Dayem, C. Mercier, P. May-Ostendorp e X. Consulting, “Electric Vehicle Charging Control Strategies,” NRECA Business and Technology Strategies department, Arlington, 2019.

[30] A. A. Salam, A. Mohamed and M. A. Hannan, “Technical Challenges on Microgrids,” *Journal of Engineering and Applied Sciences,* vol. 3, pp. 64-69, 2008.

[31] N. Karali and A. R. Gopal, “Vehicle-Grid Integration,” International Energy Studies Group, Berkeley, 2017.

[32] J. A. Lopes, F. J. Soares and P. M. Rocha, “Integration of Electric Vehicles in the Electric Power System,” *Proceedings of the IEEE,* vol. 99, pp. 168-183, 2011.

[33] A. Ul-Haq, C. Cecati e E. A. Al-Ammar, “Modeling of a Photovoltaic-Powered Electric Vehicle Charging Station with Vehicle-to-Grid Implementation,” *Energies,* vol. 10, no. 2, p. 4, 2016.

[34] F. Mwasilu, J. J. Justo, E.-K. Kim, J.-W. Jung and T. D. Do, “Electric vehicles and smart grid interaction: A review on vehicle to grid and renewable energy sources integration,” *Renewable and Sustainable Energy Reviews,* vol. 34, pp. 501-516, 2014.

[35] A. M. Foley, I. J. Winning and B. P. Gallachoir, “State-of-the-Art in Electric Vehicle Charging Infrastructure,” in *IEEE Vehicle Power and Propulsion Conference*, France, 2010.

[36] Automotive Industry Standard Committee, “Electric Vehicle Conductive AC Charging System,” Department of Road Transport & Highways Government of India, New Delhi, 2017.

[37] J. M. Green, B. Hartman and P. F. Glowacki, “A System-based View of the Standards and Certification Landscape for Electric Vehicles,” *World Electric Vehicle Journal,* vol. 8, pp. 564-575, 2016.

[38] A. Dias, “Present and Future of EV Charging in Portugal,” 17 April 2019. [Online]. Available: http://www.imt-ip.pt/sites/IMTT/Portugues/Noticias/Documents/2019/Noticia%20-%20Workshop-Sistemas%20de%20Estradas%20Eletricas%20%E2%80%93%20uma%20solucao%20para%20o%20futuro/20190417%20Andre%20Dias%20CEiiA.pdf. [Accessed: 15 August 2019].

[39] H. Shareef, M. M. Islam and A. Mohamed, “review of the stage-of-the-art charging technologies, placement methodologies, and impacts of electric vehicles,” *Renewable and Sustainable Energy Reviews*, vol. 64, pp. 403-420, 2016.

[40] K. Salah and N. Kama, “Unification Requirements of Electric Vehicle Charging Infrastructure,” *International Journal of Power Electronics and Drive System*, vol. 7, pp. 246-253, 2016.

[41] Zap Map, “Guide to EV charging,” 2019. [Online]. Available: https://www.zap-map.com/charge-points/. [Accessed: 19 August 2019].

[42] PLUGIT, “Types of Charging Plugs for Electric Vehicles,” [Online]. Available: https://plugit.fi/en-gb/article/etusivu/types-of-charging-plugs-for-electric-vehicles/135/. [Accessed: 19 August 2019].

[43] P. Schneider, “Electric Vehicle Charging Type Plugs,” 2018. [Online]. Available: https://dribbble.com/shots/5659804-Electric-Vehicle-Charging-Type-Plugs. [Accessed: 19 August 2019].

[44] Wikimedia Commons, “CHAdeMO Plug,” 2019. [Online]. Available: https://commons.wikimedia.org/wiki/File:CHAdeMO_Plug_VacavilleDavisStDC2.jpg. [Accessed: 20 August 2019].

[45] UFIK, “Golf GTE 3 Pin Charging Cable,” [Online]. Available: https://vw.ufik.co/golf-gte-3-pin-charging-cable/. [Accessed: 20 August 2019].

[46] P. V. Bossche, N. Omar and J. V. Mierlo, “Trends and Development Status of IEC Global Electric Vehicle Standards,” *Journal of Asian Electric Vehicles*, vol. 8, pp. 1409-1414, 2010.

[47] P. V. Bossche, Electric and Hybrid Vehicles, Elsevier: Brussels, 2010.

[48] IRENA, “Electric-Vehicle Smart Charging,” IRENA, Abu Dhabi, 2019.

[49] efimarket, “efimarket,” [Online]. Available: https://www.efimarket.pt/punto-recarga-new-wallbox-32-socket. [Accessed: 23 August 2019].

[50] I9charge, “Charging Your Future,” [Online]. Available: https://i9charge.com/Wallbox-KEBA-B-series-22KwPT?gclid=CjwKCAjw0N3nBRBvEiwAHMwvNo94TpNPAQIQWmqE7lGtNp0EMjQZpyxfUxXD6LyQ7G65G1Lo9BTTYBoC5EUQAvD_BwE. [Accessed: 23 August 2019].

[51] I9charge, “Charging Your Future,” [Online]. Available: https://i9charge.com/index.php?route=product/product&product_id=115. [Accessed: 23 August 2019].

[52] I9charge, “Charging Your Future,” [Online]. Available: https://i9charge.com/index.php?route=product/product&product_id=117. [Accessed: 23 August 2019].

[53] I. Sharma, C. Canizares and K. Bhattacharya, “Smart Charging of PEVs Penetrating Into Residential Distribution Systems,” *IEEE Transactions on Smart Grid,* vol. 5, pp. 1196-1209, 2014.

[54] Y.-M. Wi, J.-U. Lee and S.-K. Joo, “Electric Vehicle Charging Method for Smart Homes/Buildings with a Photovoltaic System,” *IEEE Transactions on Consumer Electronics,* vol. 59, pp. 323-328, 2013.

[55] S. Faddel, A. A. Mohamed and O. A. Mohamed, “Fuzzy logic-based autonomous controller for electric vehicles charging under different conditions in residential distribution systems,” *Electric Power Systems Research,* vol. 148, pp. 48-58, 2017.

[56] V. Tikka, H. Makkonen, J. Lassila e J. Partanen, “Case Study: Smart Charging Plug-In Hybrid Vehicle Test Environment with Vehicle-To-Grid Ability,” *in 16th European Conference on Power Electronics and Applications*, Lappeenranta, 2014.

[57] Z. Ma, D. S. Callaway and I. A. Hiskens, “Decentralized Charging Control of Large Populations of Plug-in Electric Vehicles,” *IEEE Transactions on Control Systems Technology,* vol. 21, pp. 67-78, 2013.

[58] L. Gan, U. Topcu and S. H. Low, “Optimal decentralized protocol for electric vehicle charging,” *IEEE Transactions on Power Systems,* vol. 28, pp. 940-951, 2013.

[59] S. Ruthe, J. Schumutzler, C. Rehtanz and C. Wietfeld, “Study on V2G Protocols against the Background of Demand Side Management,” *Interoperability in Business Information Systems,* 2011.

[60] A. B. Pedersen, S. Martinenas, P. B. Andersen, T. M. Sorensen and H. S. Hoj, “A method for remote control of EV charging by modifying IEC61851 compliant EVSE based PWM signal,” *IEEE Smart Grid Communication,* 2015.

[61] W. Tang, Y. Wu e J. Qin, “Comparative study on electric vehicle charging,” *Advanced Materials Research,* vol. 608, pp. 1553-1559, 2013.

[62] Phoenix Contact, “EV Charge Control Standard-compliant control of the Control Pilot and Proximity Plug interfaces between the electric vehicle and charging station,” Phoenix Contact, Pennsylvania, 2015.

[63] V. Schwarzer and R. Ghorbani, “Current State-of-the-Art of EV Chargers,” U. S. Department of Transportation, Honolulu, 2015.

[64] SAE, “SAE Electric Vehicle Conductive Charge Coupler,” SAE, Pennsylvania, 2017.

[65] The MariaDB Foundation, “About MariaDB,” 2019. [Online]. Available: https://mariadb.org/about/. [Accessed: 15 September 2019].

[66] Oracle, “The Java Tutorials,” 2019. [Online]. Available: https://docs.oracle.com/javase/tutorial/networking/sockets/definition.html. [Accessed: 15 September 2019].

[67] C. H. Chuan, “HTTP (HyperText Transfer Protocol),” 20 October 2009. [Online]. Available: https://www.ntu.edu.sg/home/ehchua/programming/webprogramming/HTTP_Basics.html. [Accessed: 20 September 2019].

[68] RS, “RS,” 2019. [Online]. Available: https://pt.rs-online.com/web/p/contactores/9160350/. [Accessed: 20 September 2019].

[69] ETA, “Commitment to Automation,” 2019. [Online]. Available: https://eta.gr/en/catalog/product/hlektronikos-trifasikos-metrhthsanalyths-energeias-gia-apey8eias-syndesh-fortiwn-ews-65-carlo-gavazzi/. [Accessed: 25 September 2019].

[70] Witte Software, “modbus tools,” 2019. [Online]. Available: https://www.modbustools.com/modbus.html. [Accessed: 25 September 2019].

[71] Carlo Gavazzi, “EM300 Series and ET300 Series,” 2016. [Online]. Available: https://www.enika.eu/data/files/produkty/energy%20m/CP/3-phase%20Modbus%20serial%20protocol%20EM%20and%20ET-rev.2.11.pdf. [Accessed: 25 September 2019].

[72] Juice4Halt, “The 5V model,” 2019. [Online]. Available: https://juice4halt.com/products. [Accessed: 28 September 2019].

[73] Leroy Merlin, 2019. [Online]. Available: https://www.leroymerlin.pt/Produtos/Eletricidade-e-domotica/Quadros-e-disjuntores/WPR_REF_14551201?gclid=Cj0KCQjw0brtBRDOARIsANMDykYWY3AO5sg0MizFGFeAJA1_4LG8BcXzRhcphO9Eqj3M7bRV7FdpkZYaAobnEALw_wcB. [Accessed: 28 September 2019].

[74] Leroy Merlin, 2019. [Online]. Available: https://www.leroymerlin.pt/Produtos/Eletricidade-e-domotica/Quadros-e-disjuntores/WPR_REF_14535675. [Accessed: 28 September 2019].

[75] techopedia, “Server,” 2019. [Online]. Available: https://www.techopedia.com/definition/2282/server. [Accessed: 30 September 2019].

[76] mongoDB, “The database for modern applications,” 2019. [Online]. Available: https://www.mongodb.com/. [Accessed: 30 September 2019].

[77] Direção-Geral de Energia e Geologia, “Regras Técnicas das Instalações Elétricas de Baixa Tensão," Direção-Geral de Energia e Geologia, Lisbon, 2000.

[78] Empresa de Eletricidade da Madeira, “Relatório e Contas 2015,” Empresa de Eletricidade da Madeira, Funchal, 2015.

[79] Direção Regional de Estatísticas da Madeira, “Estatísticas Demográficas da Região Autónoma da Madeira 2015,” Direção Regional de Estatísticas da Madeira, Funchal, 2015.

Attachment A - Coordinator code

In this section are presented the coordinator configuration file, as well as, the implemented code, using the programing language python, for the main loop, socket connection, server communication and database.

A.1 Configuration

{"11_single_phase": false, "11_name": "EV11",

"12_single_phase": false, "12_name": "EV12",

"31_single_phase": false, "31_name": "EV31",

"32_single_phase": false, "32_name": "EV32",

"41_single_phase": false, "41_name": "EV41",

"42_single_phase": false, "42_name": "EV42",

"id11": "0aca11a18257d7eb822c802e08779b8ecbd078b04c2c40c",

"id12": "84ee74a8a119b2f23e8b0db179d925141d89930ee10d0ca",

"id31": "e2c2a7b81e77712c9d676d2d563ca9c84c4ffde3ccec848",

"id32": "2a2b1bc4b9c6a963594c9cbdce12e7ffcf4d418d8e22812",

"id41": "42ec8e6d711a7f760522a0ecb966b907bc74bc1d491edf0",

"id42": "d2693dc0b4c45b62410107a9c172dc6d19976d6c5d436b1",

"multicast_group":"224.10.10.10",

"port": 10000,

"maximum_bytes_read": 162,

"communication_interval": 5.0,

"head_code": "85291643",

"max_error": 60,

"main_interval": 1,

"HTTPS": ****, "chunks_size": 60}

A.2 Main loop

```
import datetime
import random
import json
from subprocess import call
from time import sleep
import math

from oauth2client.file import Storage
import httplib2
from apiclient.discovery import build
from email.mime.text import MIMEText
import base64

from pymodbus.client.sync import ModbusSerialClient as ModbusClient
from pymodbus.constants import Endian
from pymodbus.payload import BinaryPayloadDecoder

import server
import database_duty_cycle
import socket_communication

local_path = "/home/pi/Coordinator/"

class Coordinator_main(object):

    def __init__(self):

        #Initial server settings
        cfg = json.load(open(local_path + "conf.json"))
        self.server_url = cfg['HTTPS']
        self.server_chunks_size = cfg['chunks_size'] - 1 #Max number of minutes we can read from database at
once

        #Server cars id info
        self.id11 = cfg['id11']
        self.id12 = cfg['id12']
```

```
        self.id31 = cfg['id31']
        self.id32 = cfg['id32']
        self.id41 = cfg['id41']
        self.id42 = cfg['id42']

        #Initial Socket Settings
        self.multicast_group = cfg['multicast_group']
        self.socket_port = cfg['port']
        self.maximum_bytes = cfg['maximum_bytes_read']
        self.communication_interval = cfg['communication_interval']
        self.head_code = cfg['head_code']
        self.max_error = cfg['max_error']
        self.main_interval = cfg['main_interval']

        #Init mysql server (needed for reboot)
        call(['sudo','service','mysql','start'])

        #Main function
        self.main()

    #Main
    def main(self):

        #Variables
        self.gateway_data = []
        self.duty_cycle = []
        self.last_check_duty_cycles = datetime.datetime.utcnow()
        single_phase = True

        #Read all cars
        info = database.read_cars()
        if (info['success']):
            cars = info['data']

        while(True):


            self.current_time = datetime.datetime.utcnow()
```

```
        try:

            #Reads from local database
            self.local_database_read()

            #Send duty cycle and receive cars data
            socket_communication.main(self)

            #Records data every minute
            if (len(self.gateway_data) != 0):
                date_to_compare = datetime.datetime.strptime(self.gateway_data[0]['time'], '%Y-%m-%d
%H:%M:%S')
                if (datetime.datetime.utcnow().minute != date_to_compare.minute):
                    if (len(self.gateway_data) > 0):
                        #Change data timestamp
                        self.change_minute_array(cars)
                        #Writes into local database
                        self.local_database_write()
                        self.gateway_data = []

            #Main interval
            sleep(self.main_interval)

        except Exception as error:
            print(str(error) + ".......ERRO Main........" + datetime.datetime.now().strftime('%Y-%m-%d
%H:%M:%S'))


    #Writes data into local database
    def local_database_write(self):
        try:
            #Stores values in local database
            response = database_duty_cycle.bulk_insert_three_phase(self.gateway_data)
            #Check if database storaged did not fail
            if (response['success']):
                sleep(0.2)
                #Send/receive server data
```

```
            server.main(self)
            self.database_gmail = 0
        else:
            print("Database duty cycle failed send/receive")

    except Exception as error:
        print(str(error) + " at " + datetime.datetime.now().strftime('%Y-%m-%d %H:%M:%S'))
        self.gateway_gmail += 1
        if (self.gateway_gmail == self.max_error):
            self.send_email("%s offline after %s attempts" %("RF Data",self.max_error))

#Read from local database
def local_database_read(self):
    try:
        self.duty_cycle = []
        #Reads all data
        data_duty = database_duty_cycle.read_duty_cycle_all()
        #Check if database reading did not fail
        if (data_duty['success'] and len(data_duty['data']) > 0):
            for i in range(0,len(data_duty['data'])):
                self.duty_cycle.append(data_duty['data'][i]['node_id'])
                self.duty_cycle.append(data_duty['data'][i]['car_id'])
                self.duty_cycle.append(data_duty['data'][i]['duty'])
        return

    except Exception as error:
        print(str(error) + "Read database duty cycle offline at " + datetime.datetime.now().strftime('%Y-%m-
%d %H:%M:%S'))

#Change timestamps to not duplicate cars in server side
def change_minute_array(self,cars):
    oldest_timestamp = None
    oldest_minute = None
    new_gateway_data = []
    array = self.gateway_data
    self.gateway_data = []
    #Get all cars
    for car in cars:
```

```
          car_data = []
          #Get cars with same id
          for d in array:
            if (d["data"][0] == car['node_id'] and d["data"][1] == car['car_id']):
              car_data.append(d)
          if (len(car_data) == 0):
            continue
          #Check the first oldest time
          if(oldest_timestamp == None):
              oldest_timestamp = datetime.datetime.strptime(car_data[0]["time"], '%Y-%m-%d %H:%M:%S')
              oldest_minute = oldest_timestamp.minute
          #Divide number of cars by 60 seconds
          seconds_interval = math.floor(60/len(car_data))
          #Set seconds to zero
          seconds_to_set = 0
          #Modify each car time
          for data in car_data:
            data['time'] = datetime.datetime.strptime(data["time"], '%Y-%m-%d %H:%M:%S')
            data['time'] = data['time'].replace(minute=oldest_minute, second=seconds_to_set)
            data['time'] = data['time'].strftime('%Y-%m-%d %H:%M:%S')
            seconds_to_set += seconds_interval
          #Cars time reorganise
          new_gateway_data.extend(car_data)
        self.gateway_data = new_gateway_data
        return

#Executes main class
Coordinator_main()
```

A.3 Socket communication function

```
import socket
import struct
import datetime

def main(self):

    try:
        #Variables
        current_time = datetime.datetime.now()
        past_time = datetime.datetime.now()

        #Create the datagram socket
        sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)

        #Set the time-to-live for messages to 1 so they do not go past the local network segment.
        ttl = struct.pack('b', 1)
        sock.setsockopt(socket.IPPROTO_IP, socket.IP_MULTICAST_TTL, ttl)

        #Convert head code into float
        header_code = float(self.head_code)

        #Set a timeout so the socket does not block indefinitely when trying to receive data.
        sock.settimeout(self.communication_interval)

        #Send data to the multicast group
        sock.sendto(self.head_code.encode(), (self.multicast_group, self.socket_port))

        #Receive/send all responses from all cars
        while True:
            #Check if timeout has passed
            if((past_time - current_time).total_seconds() <= int(self.communication_interval)):
                past_time = datetime.datetime.now()
                try:
                    #Receive data
                    data, server = sock.recvfrom(self.maximum_bytes)
                    #Decode data
```

```
data = data.decode()
#Convert into vetor
recdata = data.split(",")
convdata = list(map(lambda item: float(item), recdata))
#Check header code
if(convdata[0] == float(self.head_code)):
   del convdata[0]
   #Car 1
   if(convdata[0] == 1.0):
      data_duty = []
      msg = ''
      #Select duty cycle data
      data_duty = self.duty_cycle[0:6]
      #Insert header code
      data_duty.insert(0,header_code)
      #Convert into string
      msg = ",".join(map(lambda item: str(item), data_duty))
      #Send data
      sock.sendto(msg.encode(), server)
      #Insert server id car
      if (convdata[1] == 1.0):
         convdata.insert(2,self.id11)
      if (convdata[1] == 2.0):
         convdata.insert(2,self.id12)
      #Records data
      self.gateway_data.append({'time':                    self.current_time.strftime('%Y-%m-%d
%H:%M:%S'),'data': convdata})
      convdata = []
   #Car 3
   if(convdata[0] == 3.0):
      data_duty = []
      msg = ''
      #Select duty cycle data
      data_duty = self.duty_cycle[6:12]
      #Insert header code
      data_duty.insert(0,header_code)
      #Convert into string
      msg = ",".join(map(lambda item: str(item), data_duty))
```

```
                    #send data
                    sock.sendto(msg.encode(), server)
                    #Insert server id car
                    if(convdata[1] == 1.0):
                       convdata.insert(2,self.id31)
                    if(convdata[1] == 2.0):
                       convdata.insert(2,self.id32)
                    #Records data
                    self.gateway_data.append({'time':                    self.current_time.strftime('%Y-%m-%d
%H:%M:%S'),'data': convdata})
                    convdata = []
                 #Car 4
                 if(convdata[0] == 4.0):
                    data_duty = []
                    msg = ''
                    #Select duty cycle data
                    data_duty = self.duty_cycle[12:18]
                    #Insert header code
                    data_duty.insert(0,header_code)
                    #Convert into string
                    msg = ",".join(map(lambda item: str(item), data_duty))
                    #Send data
                    sock.sendto(msg.encode(), server)
                    #Insert server id car
                    if(convdata[1] == 1.0):
                       convdata.insert(2,self.id41)
                    if(convdata[1] == 2.0):
                       convdata.insert(2,self.id42)
                    #Records data
                    self.gateway_data.append({'time':                    self.current_time.strftime('%Y-%m-%d
%H:%M:%S'),'data': convdata})
                    convdata = []

           #Socket timeout exceeded
           except (sock.timeout, sock.error):
              break

        #Timeout exceeded
```

```
            else:
                break
    finally:
        #Closing socket
        sock.close()
        return
```

A.4 Server function

```
import database_server
import datetime
import os
import requests
import json
import argparse

from_date = None
data_array = None
first_upload = False

local_path = "/home/pi/Coordinator/"

server_cfg = json.load(open(local_path + "conf.json"))
server_url = server_cfg['HTTPS']
server_chunks_size = server_cfg['chunks_size'] - 1 #Max number of minutes we can read from database_server at once

def main(self):

    global server_chunks_size
    global first_upload
    global server_cfg
    global from_date
    global data_array

    first_upload = False
    current_date = datetime.datetime.utcnow() - datetime.timedelta(minutes=1) #1 Minute before current time
```

```
    #Read cars info
    info = database_server.read_cars()
    if (info['success']):
        cars = info['data']
        #Check last data sent to server
        if (check_last_data()):
            #Counts number of readings to perform from local database_server
            num_readings = int((current_date - from_date).total_seconds() / (60* server_chunks_size))
            if ((current_date - from_date).total_seconds() % (60* server_chunks_size) > 0):
                num_readings += 1

            #Reads from local database from maximum x minutes
            readings_counter = 0
            while (readings_counter < num_readings):

                #Defines time interval to be read
                if (current_date > (from_date + datetime.timedelta(minutes = server_chunks_size))):
                    to_date = from_date + datetime.timedelta(minutes = server_chunks_size)
                    print("current date maior::::", to_date)
                else:
                    to_date = (current_date + datetime.timedelta(minutes=1)).replace(second=00,microsecond=00)
                    print("To date:::: ",to_date)

                #Loop for iterating each car
                data_array = []
                for i in range(len(cars)):
                    data_L1 = {}
                    data_L2 = {}
                    data_L3 = {}
                    #Get data from local database_server
                    data_L1 =
database_server.readMeasurements("L1",cars[i]['node_id'],cars[i]['car_id'],from_date,to_date)
                    if (not server_cfg[str(cars[i]['node_id']) + str(cars[i]['car_id']) + '_single_phase']):
                        data_L2 = database_server.readMeasurements("L2", cars[i]['node_id'], cars[i]['car_id'],
from_date, to_date)
                        data_L3 = database_server.readMeasurements("L3", cars[i]['node_id'], cars[i]['car_id'],
from_date, to_date)
```

```
            if ((not  data_L1['success']) or (not server_cfg[str( cars[i]['node_id']) + str( cars[i]['car_id']) +
'_single_phase']) and (not  data_L2['success'] or not  data_L3['success'])):
                continue
            #Store in array
            store_array(data_L1,data_L2,data_L3,cars,i)

        #Check if there is data on that interval
        if (len(data_array) != 0):
            sorted_data_array = sorted(data_array, key=lambda k: k['timestamp'])
            recent_time = sorted_data_array[len(sorted_data_array)-1]['timestamp']
            #Check if data was sucessfully received by server
            if (postJsonString()):
                #Update upload_status table from local database_server
                if (first_upload):
                    database_server.createUploadStatus("car",  to_date.strftime('%Y/%m/%d %H:%M'))
                    first_upload = False
                else:
                    database_server.updateUploadStatus("car",  to_date.strftime('%Y/%m/%d %H:%M'))
            else:
                break
        #Updates parameters
        from_date =  to_date# + datetime.timedelta(minutes=1)
        readings_counter += 1

#Check last data sent to server
def check_last_data():

    global first_upload
    global from_date

    #Check upload_status table from local database_server
    upload_status = database_server.readUploadStatus("car")
    if (not upload_status['success']):
        return False

    #If no data was sent before, it sends the data since the first database_server register
    if (not upload_status['data']):
        first_upload = True
```

```
        first_entry = database_server.read_first_entry()
        if(first_entry['success']):
            from_date = first_entry['data'][0]['timestamp']
        else:
            return False
    #Otherwise, it sends data starting from the last datetime
    else:
        from_date = upload_status['data'][0]['last_uploaded_datetime'] #+ datetime.timedelta(minutes=1)

    return True

#Stores cars data in array
def store_array(data_L1,data_L2,data_L3,cars,j):

    global data_array
    global server_cfg

    #Put data on server format
    for i in range(0,len(data_L1['data'])):
        if (server_cfg[str(cars[j]['node_id']) + str(cars[j]['car_id']) + '_single_phase']):
            data_array.append({
                "node_id":  data_L1['data'][i]['ND'],
                "car_id":  data_L1['data'][i]['CD'],
                "device_id":  data_L1['data'][i]['ID'],
                "ev_connect":  data_L1['data'][i]['CC'],
                                    "ev_state":  data_L1['data'][i]['CS'],
                "L1":  {"V":  data_L1['data'][i]['V'],  "I":  data_L1['data'][i]['I'],  "P":  data_L1['data'][i]['P'],  "PF":
data_L1['data'][i]['PF']},
                "P":  data_L1['data'][i]['P'],
                "Q":  data_L1['data'][i]['PR'],
                                    "S":  data_L1['data'][i]['PA'],
                "F": data_L1['data'][i]['F'],
                "timestamp":  data_L1['data'][i]['timestamp'].strftime('%Y-%m-%dT%H:%M:00Z'),
                "measure_cons": data_L1['data'][i]['P']})
        else:
            data_array.append({
                "node_id": data_L1['data'][i]['ND'],
                "car_id": data_L1['data'][i]['CD'],
```

```
            "device_id": data_L1['data'][i]['ID'],
            "ev_connect": data_L1['data'][i]['CC'],
                                    "ev_state": data_L1['data'][i]['CS'],
            "L1": {"V": data_L1['data'][i]['V'], "I": data_L1['data'][i]['I'], "P": data_L1['data'][i]['P'], "PF": data_L1['data'][i]['PF']},
            "L2": {"V": data_L2['data'][i]['V'], "I": data_L2['data'][i]['I'], "P": data_L2['data'][i]['P'], "PF": data_L2['data'][i]['PF']},
            "L3": {"V": data_L3['data'][i]['V'], "I": data_L3['data'][i]['I'], "P": data_L3['data'][i]['P'], "PF": data_L3['data'][i]['PF']},
                                    "P": data_L1['data'][i]['P'],
            "Q": data_L1['data'][i]['PR'],
                                    "S": data_L1['data'][i]['PA'],
            "F": data_L1['data'][i]['F'],
            "timestamp": data_L1['data'][i]['timestamp'].strftime('%Y-%m-%dT%H:%M:00Z'),
            "measure_cons": data_L1['data'][i]['P']})


#Sends POST Request to Server
def postJsonString():

    try:

        global data_array
        global server_url

        headers = {'Content-type' : 'application/json'}
        server_data = requests.post(server_url, data=json.dumps(data_array), headers=headers, timeout=30)
        if (server_data.status_code == requests.codes.ok):
            for i in range(len(server_data.json())):
                duty = server_data.json()[i]["duty_cycle"]
                NID = server_data.json()[i]["node_id"]
                CID = server_data.json()[i]["car_id"]
                database_server.store_dutycycle(NID, CID, duty)
            return True
        else:
            print(str(server_data.status_code) + " at " + datetime.datetime.now().strftime('%Y-%m-%d %H:%M:%S'))
            return False

    except Exception as error:
```

```
    print(str(error) + " at " + datetime.datetime.now().strftime('%Y-%m-%d %H:%M:%S'))
    return False
```

A.5 Database table creation

```
CREATE DATABASE IF NOT EXISTS `carDB`;

USE `carDB`;

# Create car_measurements table

DROP TABLE IF EXISTS `car_duty`;

CREATE TABLE `car_duty` (
  `node_id` INT NOT NULL,
  `car_id` INT NOT NULL,
  `duty` INT DEFAULT 100,
  PRIMARY KEY (`node_id`, `car_id`)
) ENGINE = MYISAM;

#Por os restantes car
INSERT INTO car_duty(node_id, car_id) value (1, 1);
INSERT INTO car_duty(node_id, car_id) value (1, 2);
INSERT INTO car_duty(node_id, car_id) value (3, 1);
INSERT INTO car_duty(node_id, car_id) value (3, 2);
INSERT INTO car_duty(node_id, car_id) value (4, 1);
INSERT INTO car_duty(node_id, car_id) value (4, 2);

# Create car_measurements table

DROP TABLE IF EXISTS `car_measurements`;

CREATE TABLE `car_measurements` (
  `node_id` INT NOT NULL,
  `car_id` INT NOT NULL,
  `device_id` VARCHAR(255) NOT NULL,
  `car_connect` INT NOT NULL,
```

```
  `car_state` INT NOT NULL,
  `L1` blob NOT NULL,
  `L2` blob,
  `L3` blob,
  `potencia_total_ativa` FLOAT NOT NULL,
  `potencia_total_reativa` FLOAT NOT NULL,
  `potencia_total_aparente` FLOAT NOT NULL,
  `frequency` FLOAT NOT NULL,
  `timestamp` DATETIME NOT NULL,
  PRIMARY KEY (`node_id`, `car_id`, `timestamp`),
  foreign key(`node_id`, `car_id`) references car_duty(node_id, car_id)
) ENGINE = MYISAM;

# Create upload_status table

DROP TABLE IF EXISTS `upload_status`;

CREATE TABLE `upload_status` (
  `car` TINYINT NOT NULL AUTO_INCREMENT,
  `table_name` varchar(20) NOT NULL,
  `created` DATETIME NOT NULL,
  `updated` DATETIME NOT NULL ON UPDATE CURRENT_TIMESTAMP,
  `last_uploaded_datetime` DATETIME NOT NULL,
  PRIMARY KEY (`car`)
) ENGINE = MYISAM;
```

A.6 Database duty cycle function

```
from configparser import ConfigParser
from mysql.connector import MySQLConnection, Error
from collections import OrderedDict
import datetime

local_path = "/home/pi/Coordinator/"

#Reads database parameters from .ini file
def read_db_config(filename= local_path + "config.ini", section='mysql'):
```

```
    parser = ConfigParser()
    parser.read(filename)
    db = {}
    if parser.has_section(section):
        items = parser.items(section)
        for item in items:
            db[item[0]] = item[1]
    else:
        raise Exception('{0} not found in the {1} file'.format(section, filename))
    return db

#Returns all rows from a cursor as a list of dicts
def dictFetchAll(cursor):
    desc = cursor.description
    data = []
    for row in cursor.fetchall():
        data.append(OrderedDict(zip([col[0] for col in desc], row)))

    return data
    #return [OrderedDict(zip([col[0] for col in desc], row))
    #       for row in cursor.fetchall()]

#Database write operation
def write_operation(query):
    try:
        db_config = read_db_config()
        conn = MySQLConnection(**db_config)
        cursor = conn.cursor(buffered=True,dictionary=True)
        cursor.execute(query)
        conn.commit()
        data = {"success": True}
    except Error as error:
        print(str(error) + " at " + datetime.datetime.now().strftime('%Y-%m-%d %H:%M:%S'))
        data = {"success": False}
    finally:
        if 'cursor' in locals():
            cursor.close()
```

```
        if 'conn' in locals():
            conn.close()
        return data

#Database read operation
def read_operation(query):
    try:
        db_config = read_db_config()
        conn = MySQLConnection(**db_config)
        cursor = conn.cursor(buffered=True)
        cursor.execute(query)
        data = {"success": True, "data":dictFetchAll(cursor)}
    except Error as error:
        print(str(error) + " at " + datetime.datetime.now().strftime('%Y-%m-%d %H:%M:%S'))
        data = {"success": False}
    finally:
        if 'cursor' in locals():
            cursor.close()
        if 'conn' in locals():
            conn.close()
        return data

def bulk_insert_single_phase(array):
    query = """INSERT INTO car_measurements
(node_id,car_id,device_id,car_connect,car_state,L1,potencia_total_ativa,potencia_total_reativa,potencia_total_aparente,frequency,timestamp) VALUES"""
    for (i, row) in enumerate(array):
        query += """(%s,%s,'%s',%s,%s,
    column_create('voltage',%s,'current',%s,'power',%s,'power_factor',%s),
    %s,%s,%s,%s,'%s')"""
%(row['data'][0],row['data'][1],row['data'][2],row['data'][3],row['data'][4],row['data'][5],row['data'][8],row['data'][11],row['data'][17],row['data'][14],row['data'][15],row['data'][16],row['data'][20],row['time'])
        if i == (len(array) - 1):
            query += ';'
        else:
            query += ','

    return write_operation(query)
```

```
def bulk_insert_three_phase(array):
    query = """INSERT INTO car_measurements (node_id,car_id,device_id,car_connect,car_state,L1,L2,L3,potencia_total_ativa,potencia_total_reativa,potencia_total_aparente,frequency,timestamp) VALUES"""
    for (i, row) in enumerate(array):
        print("row:::",row)
        query += """(%s,%s,'%s',%s,%s,
    column_create('voltage',%s,'current',%s,'power',%s,'power_factor',%s),
    column_create('voltage',%s,'current',%s,'power',%s,'power_factor',%s),
    column_create('voltage',%s,'current',%s,'power',%s,'power_factor',%s),
    %s,%s,%s,%s,'%s')""" %(row['data'][0],row['data'][1],row['data'][2],row['data'][3],row['data'][4],row['data'][5],row['data'][8],row['data'][11],row['data'][17],row['data'][6],row['data'][9],row['data'][12],row['data'][18],row['data'][7],row['data'][10],row['data'][13],row['data'][19],row['data'][14],row['data'][15],row['data'][16],row['data'][20],row['time'])
        if i == (len(array) - 1):
            query += ';'
        else:
            query += ','
    return write_operation(query)


def read_duty_cycle(node_id, car_id):
    query = """SELECT * FROM car_duty WHERE node_id = %s AND car_id = %s"""%(node_id, car_id)
    return read_operation(query)


def read_duty_cycle_all():
    query = "SELECT * FROM car_duty"
    return read_operation(query)


def read_cars():
    query = "SELECT * FROM car_duty"
    return read_operation(query)
```

A.7 Database server function

```
from configparser import ConfigParser
from mysql.connector import MySQLConnection, Error
from collections import OrderedDict
import datetime

local_path = "/home/pi/Coordinator/"

#Reads database parameters from .ini file
def read_db_config(filename = local_path + "config.ini", section='mysql'):
    parser = ConfigParser()
    parser.read(filename)
    db = {}
    if parser.has_section(section):
        items = parser.items(section)
        for item in items:
            db[item[0]] = item[1]
    else:
        raise Exception('{0} not found in the {1} file'.format(section, filename))
    return db

#Returns all rows from a cursor as a list of dicts
def dictFetchAll(cursor):
    desc = cursor.description
    return [OrderedDict(zip([col[0] for col in desc], row))
            for row in cursor.fetchall()]

#Database write operation
def write_operation(query):
    try:
        db_config = read_db_config()
        conn = MySQLConnection(**db_config)
        cursor = conn.cursor()
        cursor.execute(query)
        conn.commit()
    except Error as error:
        print(str(error) + " at " + datetime.datetime.now().strftime('%Y-%m-%d %H:%M:%S'))
```

```
    finally:
        if 'cursor' in locals():
            cursor.close()
        if 'conn' in locals():
            conn.close()

def createUploadStatus(a_table, a_datetime):
    query = """INSERT INTO upload_status (table_name, created, updated, last_uploaded_datetime)
            VALUES ('%s', NOW(), NOW(), '%s')""" %(a_table, a_datetime)
    write_operation(query)

def updateUploadStatus(a_table, a_datetime):
    query = """UPDATE upload_status SET last_uploaded_datetime = '%s' WHERE table_name ='%s'"""
%(a_datetime, a_table)
    write_operation(query)

#Database read operation
def read_operation(query):
    try:
        db_config = read_db_config()
        conn = MySQLConnection(**db_config)
        cursor = conn.cursor()
        cursor.execute(query)
        data = {"success": True, "data":dictFetchAll(cursor)}
    except Error as error:
        print(str(error) + " at " + datetime.datetime.now().strftime('%Y-%m-%d %H:%M:%S'))
        data = {"success": False}
    finally:
        if 'cursor' in locals():
            cursor.close()
        if 'conn' in locals():
            conn.close()
        return data

def readUploadStatus(a_table):
    query = """SELECT * FROM upload_status WHERE table_name = '%s'""" %(a_table)
    return read_operation(query)
```

```
def readMeasurements(fase, node_id, car_id, a_from_date, a_to_date):
    query = """SELECT node_id AS ND, car_id AS CD, device_id AS ID, car_connect AS CC, car_state AS CS,
      ROUND(AVG(column_get({0},'voltage' as double)),2) AS V,
      ROUND(AVG(column_get({0},'current' as double)),2) AS I,
      ROUND(AVG(column_get({0},'power' as double)),2) AS P,
      ROUND(AVG(column_get({0},'power_factor' as double)),2) AS PF,
      ROUND(AVG(potencia_total_ativa),2) AS P,
      ROUND(AVG(potencia_total_reativa),2) AS PR,
      ROUND(AVG(potencia_total_aparente),2) AS PA,
      ROUND(AVG(frequency),2) AS F, timestamp FROM car_measurements
      WHERE node_id = {1} AND car_id = {2} and
       timestamp BETWEEN '{3}' AND '{4}' GROUP BY date(timestamp), hour(timestamp), minute(timestamp)""".format(fase, node_id, car_id, a_from_date, a_to_date)
    return read_operation(query)


def read_first_entry():
    query = "SELECT * FROM car_measurements ORDER BY node_id AND car_id limit 1"
    return read_operation(query)


def store_dutycycle(node_id, car_id, duty):
        query = "UPDATE car_duty SET duty = {0} WHERE node_id = {1} AND car_id = {2}".format(duty, node_id, car_id)
        write_operation(query)


def read_cars():
    query = "SELECT * FROM car_duty"
    return read_operation(query)
```

Attachment B - Charging station code

In this section are presented the charging station configuration file, as well as, the implemented code, using the programing language python, for the main loop, EV state detection, energy meter reading and the socket connection.

B.1 Configuration


```
{"serial_modbus": "/dev/ttyS0",
"modbus_method": "rtu",
"baudrate_modbus": 9600,
"main_interval": 2,
"pwm_frequency": 1000,
"node_id": 4,
"multicast_group": "224.10.10.10",
"host": "",
"port": 10000,
"maximum_bytes_read": 26,
"communication_interval": 2.0,
"head_code": 85291643}
```


B.2 Main loop

```
import json
import datetime
import socket
import struct
import sys
import pigpio
import RPi.GPIO as GPIO
from time import sleep
from pymodbus.client.sync import ModbusSerialClient as ModbusClient
from EV1_STATE import EV1_state
from EV2_STATE import EV2_state
from Smart_meter import smart_meter
from Socket_communication import socket_communication

local_path = "/home/pi/Documents/Node/"
```

```
class main(object):

    def __init__(self):

        #Initial Main settings
        Node_cfg = json.load(open(local_path + "Conf.json"))
        self.main_interval = Node_cfg['main_interval']
        self.node_id = Node_cfg['node_id']

        #Initial meter settings
        self.reg = [0, 2, 4, 12, 14, 16, 18, 20, 22, 40, 42, 44, 46, 47, 48]#Registers
        meter_port = Node_cfg['serial_modbus']
        meter_baudrate = Node_cfg['baudrate_modbus']
        modbus_method = Node_cfg['modbus_method']
        self.client = ModbusClient(method = modbus_method, port = meter_port, timeout = 1, baudrate =
meter_baudrate)

        #Initial PWM settings
        self.pwm = pigpio.pi()
        self.pwm_frequency = Node_cfg['pwm_frequency']
        self.pwm.hardware_PWM(13, self.pwm_frequency, 1000000)#GPIO 13 - Pin 33
        self.pwm.hardware_PWM(18, self.pwm_frequency, 1000000)#GPIO 18 - Pin 12

        #Initial EV settings
        GPIO.setwarnings(False)
        GPIO.setmode(GPIO.BOARD)#Select board pin number
        self.gpio_out = [11, 35]#Pins
        self.gpio_in = [3, 5, 13, 16, 26, 31, 36, 37]#Pins
        GPIO.setup(self.gpio_out, GPIO.OUT, initial = GPIO.LOW)
        GPIO.setup(self.gpio_in, GPIO.IN, pull_up_down = GPIO.PUD_DOWN)

        #Initial Socket settings
        self.header_code = Node_cfg['head_code']
        self.communication_interval = Node_cfg['communication_interval']
        socket_port = Node_cfg['port']
        self.maximum_bytes = Node_cfg['maximum_bytes_read']
        socket_host = Node_cfg['host']
```

```
        multicast_group = Node_cfg['multicast_group']
        self.sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
        self.sock.bind((socket_host, socket_port))
        self.sock.settimeout(self.communication_interval)
        group = socket.inet_aton(multicast_group)
        mreq = struct.pack('4sL', group, socket.INADDR_ANY)
        self.sock.setsockopt(socket.IPPROTO_IP, socket.IP_ADD_MEMBERSHIP, mreq)

        #Starts main routine
        self.main_routine()

    def main_routine(self):

        #Variables
        self.EV1_state = 0
        self.EV1_connect = 0
        self.EV1_flag = False
        self.EV2_state = 0
        self.EV2_connect = 0
        self.EV2_flag = False
        self.duty_cycle1 = 100
        self.duty_cycle2 = 100
        self.meter_id = 1

        #Main loop
        while(True):

            #Acquire Meter data
            smart_meter(self)
            print("Meter data:::::: ",self.meter_data)

            if(self.meter_id == 1):
                self.meter_data.insert(0, float(self.node_id))
                self.meter_data.insert(1, float(self.meter_id))
                self.meter_data.insert(2, float(self.EV1_connect))
                self.meter_data.insert(3, float(self.EV1_state))
                self.meter_id = 2
```

```
else:
    self.meter_data.insert(0, float(self.node_id))
    self.meter_data.insert(1, float(self.meter_id))
    self.meter_data.insert(2, float(self.EV2_connect))
    self.meter_data.insert(3, float(self.EV2_state))
    self.meter_id = 1

#Check Meter reading failure
if(self.meter_exception == False):
    #Send/receive data
    socket_communication(self)

#Check EV1 cable
cable_state1 = GPIO.input(self.gpio_in[8])

if(cable_state1 == True):
    print("Entrou.....::::: ")
    #Check EV1 state
    EV1_state(self)
else:
    self.EV1_connect = 0
    self.EV1_state = 0

sleep(0.2)

#Check EV2 cable
cable_state2 = GPIO.input(self.gpio_in[9])

if(cable_state2 == True):
    #Check EV2 state
    EV2_state(self)
else:
    self.EV2_connect = 0
    self.EV2_state = 0

sleep(self.main_interval)
```

```
#Executes main class
main()
```

B.3 EV state function

```
import RPi.GPIO as GPIO
import datetime
from time import sleep

def EV1_state(self):

    try:

        a = None
        b = None
        c = None

        c = GPIO.input(self.gpio_in[0])
        b = GPIO.input(self.gpio_in[2])
        a = GPIO.input(self.gpio_in[3])

        print("a: " + str(a))
        print("b: " + str(b))
        print("c/d: " + str(c))
        print("e: " + str(c))

        #Charging station error
        #State E
        if(c == False):
            print("Charging station error....")
            #Deactivate energy relay
            GPIO.output(self.gpio_out[0], GPIO.LOW)
            #Set EV communication
            self.pwm.hardware_PWM(18, self.pwm_frequency, 1000000)
            self.EV1_state = 4
            self.EV1_connect = 0
            return
```

```
#Second case of EV connection / With the duty cycle defined (EV malfunction verified)
if(self.duty_cycle1 <= 50 and self.duty_cycle1 >= 10):
    #State A
    if(a == True):
        print("EV1 not connected....")
        #Deactivate energy relay
        GPIO.output(self.gpio_out[0], GPIO.LOW)
        #Set EV communication
        self.pwm.hardware_PWM(18, self.pwm_frequency, 1000000)
        self.EV1_state = 1
        self.EV1_connect = 0
        return
    #State B
    if(b == True):
        print("EV1 connected / Charging set: ", self.duty_cycle1)
        #Set EV communication
        self.pwm.hardware_PWM(18, self.pwm_frequency, self.duty_cycle1*10000)
        self.EV1_state = 2
        self.EV1_connect = 1
        sleep(0.2)
        f = None
        f = GPIO.input(self.gpio_in[1])
        print("f1: " + str(f))
        #State F
        if(f == False):
            print("EV1 malfunction....")
            #Deactivate energy relay
            GPIO.output(self.gpio_out[0], GPIO.LOW)
            #Set EV communication
            self.pwm.hardware_PWM(18, self.pwm_frequency, 1000000)
            self.EV1_state = 5
            self.EV1_connect = 0
        else:
            #Deactivate energy relay
            GPIO.output(self.gpio_out[0], GPIO.LOW)
        return
```

```
        #State C/D
        if(c == True):
            print("EV1 connected / Charging rate: ", self.duty_cycle1)
            #Set EV communication
            self.pwm.hardware_PWM(18, self.pwm_frequency, self.duty_cycle1*10000)
            self.EV1_state = 3
            self.EV1_connect = 1
            sleep(0.2)
            f = None
            f = GPIO.input(self.gpio_in[1])
            print("f1: " + str(f))
            #State F
            if(f == False):
                print("EV1 malfunction....")
                #Deactivate energy relay
                GPIO.output(self.gpio_out[0], GPIO.LOW)
                #Set EV communication
                self.pwm.hardware_PWM(18, self.pwm_frequency, 1000000)
                self.EV1_state = 5
                self.EV1_connect = 0
            else:
                #Activate energy relay
                GPIO.output(self.gpio_out[0], GPIO.HIGH)
            return
    #First case of EV connection / Without de charging rate defined (EV malfunction not verified)
    else:
        #State A
        if(a == True):
            print("EV1 not connected....")
            #Deactivate energy relay
            GPIO.output(self.gpio_out[0], GPIO.LOW)
            #Set EV communication
            self.pwm.hardware_PWM(18, self.pwm_frequency, 1000000)
            self.EV1_state = 1
            self.EV1_connect = 0
            return
        #State B
```

```
        if(b == True):
            print("EV1 connected....")
            #Deactivate energy relay
            GPIO.output(self.gpio_out[0], GPIO.LOW)
            #Set EV communication
            self.pwm.hardware_PWM(18, self.pwm_frequency, 1000000)
            self.EV1_state = 2
            self.EV1_connect = 1
            return
        #State C/D
        if(c == True):
            print("EV1 connected / Charging rate not define....")
            #Deactivate energy relay
            GPIO.output(self.gpio_out[0], GPIO.LOW)
            #Set EV communication
            self.pwm.hardware_PWM(18, self.pwm_frequency, 1000000)
            self.EV1_state = 3
            self.EV1_connect = 1
            return

    except Exception as error:
        print(str(error) + "....EV1 function... at " + datetime.datetime.now().strftime('%Y-%m-%d %H:%M:%S'))
        #Deactivate energy relay
        GPIO.output(self.gpio_out[0], GPIO.LOW)
        #Set EV communication
        self.pwm.hardware_PWM(18, self.pwm_frequency, 1000000)
        self.EV1_connect = 0
        self.EV1_state = 4
```

B.4 Energy meter function

```
import datetime
from pymodbus.constants import Endian
from pymodbus.payload import BinaryPayloadDecoder
import RPi.GPIO as GPIO
from pymodbus import exceptions
```

```
#Reads values from Carlo meter
def smart_meter(self):

    try:
        #Clean meter data variable
        self.meter_data = []
        self.meter_exception = False

        #Sending requests to meter
        value_array = BinaryPayloadDecoder.fromRegisters(self.client.read_holding_registers(self.reg[0], self.reg[len(self.reg)-1] + 1,unit = self.meter_id).registers, byteorder = Endian.Big, wordorder = Endian.Little)

        #Process Carlo readings
        for i in range(0,len(self.reg)):

            #Decode Carlo readings
            flag = True
            if(i == len(self.reg) - 1 or self.reg[i+1] == self.reg[i] + 1):
                value = value_array.decode_16bit_int()
                flag = False
            else:
                value = value_array.decode_32bit_int()

            #Current and power factor have 3 decimal places, the rest just one
            if((i>2 and i<6) or (i>11 and i<15)):
                div = 1000.0
            else:
                div = 10.0

            #Update array
            self.meter_data.append(value/div)

            #Decode not stored data
            if(flag):
                for j in range(0, int((self.reg[i+1]-(self.reg[i]+2))/2)):
                    value_array.decode_32bit_int()
```

```
        #Just for the frequency

self.meter_data.append(BinaryPayloadDecoder.fromRegisters(self.client.read_holding_registers(51,1,unit=
self.meter_id).registers, byteorder = Endian.Big, wordorder = Endian.Little).decode_16bit_int()/10.0)

        return

    except Exception as exceptions:
        print(str(exceptions) + "...Smart Meter Reading... at " + datetime.datetime.now().strftime('%Y-%m-%d
%H:%M:%S'))
        self.meter_data = []
        self.meter_exception = True
        #Meter not working and EV disconnect
        if(self.meter_id == 2):
            self.pwm.hardware_PWM(13, self.pwm_frequency, 1000000)
            GPIO.output(self.gpio_out[1], GPIO.LOW)
            self.duty_cycle2 = 100

        if(self.meter_id == 1):
            self.pwm.hardware_PWM(18, self.pwm_frequency, 1000000)
            GPIO.output(self.gpio_out[0], GPIO.LOW)
            self.duty_cycle1 = 100
```

B.5 Socket communication function

```
import sys
import datetime
import socket

def socket_communication(self):

    #Variables
    current_time = datetime.datetime.now()
    past_time = datetime.datetime.now()

    #Convert data into a string
    self.meter_data.insert(0, self.header_code)
```

```
msg = ",".join(map(lambda item: str(item), self.meter_data))
print("Msg sent:::: ",msg)

#Receive/send data
while True:
    #Check if timeout has passed
    if((past_time - current_time).total_seconds() <= self.communication_interval):
        past_time = datetime.datetime.now()
        try:
            print('\nwaiting to receive message')
            #Receive data
            data, address = self.sock.recvfrom(self.maximum_bytes)
            print('received {} bytes from {}'.format(len(data), address))
            print('sending acknowledgement to', address)
            #Send data
            self.sock.sendto(msg.encode(), address)
            #Decode receive data
            data = data.decode()
            #Convert data into vetor
            recdata = data.split(",")
            convdata = list(map(lambda item: float(item),recdata))
            print("converted data:::: ",convdata)
            print("Header code :::::: ",self.header_code)
            #Check lenght data higher than header code
            if(len(convdata) > 1):
                #Check header code
                if(convdata[0] == float(self.header_code)):
                    del convdata[0]
                    #Check node id
                    if(convdata[0] == float(self.node_id)):
                        #Check EV1 connect
                        if(self.EV1_connect == 0):
                            #Server did not recognise the disconnection
                            if(convdata[2] <= 50 and convdata[2] >= 10):
                                self.duty_cycle1 = 100
                                self.EV1_flag = False
                            #Server recognise the disconnection
```

```
                        else:
                            self.duty_cycle1 = convdata[2]
                            self.EV1_flag = True
                    #EV1 controlled by server
                    if(self.EV1_connect == 1 and self.EV1_flag == True):
                        self.duty_cycle1 = convdata[2]
                    #Check EV2 connect
                    if(self.EV2_connect == 0):
                        #Server did not recognise the disconnection
                        if(convdata[5] <= 50 and convdata[5] >= 10):
                            self.duty_cycle2 = 100
                            self.EV2_flag = False
                        #Server recognise the disconnection
                        else:
                            self.duty_cycle2 = convdata[5]
                            self.EV2_flag = True
                    #EV2 controlled by server
                    if(self.EV2_connect == 1 and self.EV2_flag == True):
                        self.duty_cycle2 = convdata[5]
                    print("Duty Cycle::::: ", convdata)
                    convdata = []
                    break
        #Socket timeout exceeded
        except (socket.timeout, socket.error):
            print("socket timeout:::::::::::")
            break
    #Timeout exceeded
    else:
        print("Start timeout.................")
        break
return
```

Attachment C – Server code

In this section are presented the server implemented code, using the programing language javascript, to receive from the coordinator the data and calculate the duty cycle values.

C.1 Coordinator data function

```
// Function to receive and process data from coordinator
exports.postSamplesEV = (req, res) => {
  Source.findOne({'config.slug': mongoSanitize(req.params.slug) }, (error, source) => {
    if (error) return res.status(500).json({ error: 'DB error' })
    if (!source) return res.status(404).json({ error: 'Source not found' })

    let cars = []

    // Authentication
    passport.authenticate('producer', (error, producer) => {
        if (error) return res.status(500).json({ error: 'Authentication error' })
        if (!producer) return res.status(401).json({ error: 'Unauthorized' })
        cars = parseData(producer)
        const cars_calculated_data = await processEVs(cars, producer.id)
        return res.json(cars_calculated_data)
    })(req, res)

    // Validating and Parsing Data
    async function parseData(producer) {
      const producerID = (producer && producer.id) || null

      // Transform to Array if not Array
      let data = []
      if (Array.isArray(req.body)) data = [...req.body]
      else data.push(req.body)

      // Convert
      const converter = new sampleConverter(source.config.converter)
      const convertedData = data.map(d => converter.convert(d))

      // Store and calculate duty cycle
```

```
        let devices_data = {}
        convertedData.forEach(d => {
            if (!devices_data[d.device_id])
                devices_data[d.device_id] = []
            devices_data[d.device_id].push(d)
        })

        let cars = []
        for (device_id in devices_data){
            devices_data[device_id].sort((a,b) => { return a.timestamp-b.timestamp })
            let timestamps = devices_data[device_id].map(d => d.timestamp.toDate())
            storage.writeSamples(source.config.name, producerID, device_id, timestamps,
devices_data[device_id], (err, result) => {
                if(err) console.log(err)
                if(!result) return false
                return true
            })
            if (devices_data[device_id].length > 0){
                cars.push(devices_data[device_id][devices_data[device_id].length-1])
            }
        }
        return cars
    }
  })
}
```

C.2 Duty cycle calculation function

```
// Function to process EV data and calculate duty cycle
async function processEVs(cars, producerId) {
  // Get house available power
  let available_power = await get_available_power(producerId)
  const cars_total_power = cars.reduce((sum, car) => sum+(car.data.active_power || 0), 0)
  available_power += (cars_total_power || 0)

  // Get house stations
  let stations = await get_producer_stations(producerId)
```

```
    // Prepare cars
    for (car of cars) {
        car.station = stations.find(station => station.deviceId == car.device_id)
        car.event = await ev_device_helper.lastEvent(car.device_id)
        car.power_assigned = 0
        car.power_required = await calculate_power_required(car)
        car.typology = check_typology(car)
        car.max_power = car.typology == 3 ? car.station.ev.max_power_tri : car.typology == 1 ? car.station.ev.max_power_single : 1380
    }

    // Sort cars by end_timestamp, start_timestamp, and typology
    cars.sort((a,b) => (a.end === null)-(b.end === null) || (a.start === null)-(b.start === null) || (b.event.charging_rate - a.event.charging_rate) || (a.end-b.end) || (a.start-b.start) || (b.typology-a.typology))

    // Get count of cars that will use power
    const connected_cars_count = cars.filter(car => car.power_required > 0).length

    // Iterate through cars
    while (!cars.every(car => car.hasOwnProperty('power_required') && car.power_required == null)) {
        const divided_power = (available_power/connected_cars_count)
        cars.forEach(car => {
            // Assign power to car
            let station_max_power = car.max_power
            let power_to_assign = 0

            if (car.power_required > 0) {
                if (car.power_required <= available_power && car.power_required <= station_max_power) {
                    power_to_assign = car.power_required
                    car.power_assigned += power_to_assign
                    if (car.event.charging_rate > 0) car.power_required = null
                    else car.power_required = 0
                }
                else car.power_required = null
            }
            else if (car.power_required == 0) {
                if (available_power > 0) {
```

```
                if ((car.power_assigned + divided_power) > station_max_power) {
                    power_to_assign = station_max_power - car.power_assigned
                    car.power_assigned += power_to_assign
                    car.power_required = null
                }
                else {
                    power_to_assign = divided_power
                    car.power_assigned += power_to_assign
                }
            }
            else car.power_required = null
        }
        available_power = available_power - power_to_assign
    })
  }

  // Calculate duty cycles fro each car and return data
  return await new Promise((resolve, reject) => {
    let promises = []
    cars.forEach(car => {
        promises.push(new Promise (async (resolve, reject) => {
            let duty_cycle = await calculate_duty_cycle(car, producerId)
            if (car.power_assigned == 0) duty_cycle = 100
            let typology = await update_typology(car, duty_cycle)
            await update_duty_cycle(car, duty_cycle)
            resolve({ device_id: car.device_id, duty_cycle, typology, data: car })
        }))
    })
    Promise.all(promises).then(calculated_cars => {
        calculated_cars.sort((a,b) => a.data.station.ev.number - b.data.station.ev.number)
        return resolve(calculated_cars)
    })
  })
}
```