POLITECNICO DI TORINO

Master Degree in Electrical Engineering

Master's Degree Thesis

Design and Preliminary Testing of a V2G Smart Charging Lab for Electrical Vehicles in Bidirectional Charging Applications

Supervisor:

Prof. Paolo Guglielmi

Candidate:

Edoardo Olivero

2025

Abstract

The energy transition demands advanced solutions to manage the inherent variability of renewable sources such as solar and wind power. A key strategy to address this challenge is the integration of Electric Vehicles (EVs) as distributed storage units through bidirectional charging (V2X). This technology allows vehicles not only to draw energy from the grid but also to feed it back, enabling critical services such as demand balancing, optimization of self-consumption, and peak load reduction. This thesis focuses on the Design and Preliminary Testing of a V2G (Vehicle-to-Grid) Smart Charging Laboratory for Electric Vehicles dedicated to bidirectional charging applications. The primary objective was to conceive and implement an experimental environment that allows for the simulation and validation of communication according to the latest standard. The laboratory was designed to support electrical charging with a power capacity of up to 12 kW. A crucial technical focus is placed on the implementation of communication protocols between the vehicles and the charging infrastructure, with particular attention to the ISO 15118-20 standard (dynamic power management). The thesis describes in detail the system architecture, component selection, and the control software (Power Electronics Communication Controller) developed using the CAN protocol. The Preliminary Testing phase aims to verify the system's functionality and reliability, validating the correct implementation of power flows and bidirectional communication protocols. Initial tests were conducted to evaluate the system's effectiveness in key scenarios such as Vehicle-to-Grid (V2G), providing the first experimental data and valuable insights for the future development of innovative solutions that enhance the integration between electric mobility and smart power grids.

"Brilliant thinking is rare, but courage is in even shorter supply than genius."

- Peter Thiel

Contents

List of Tables

List of Figures

Acronyms

EV Electric Vehicle Supply Equipment

EVSE Electric Vehicle

BEV Battery Electric Vehicle

HEV Hybrid Electric Vehicle

SOC State of Charge

BMS Battery Management System

V2X Vehicle-to-Everything

V2G Vehicle-to-Grid

V2H Vehicle-to-Home

V2B Vehicle-to-Building

V2L	Vehicle-to-Load
CCS	Combined Charging System
CHAdeMO	Charge de Move
OCPP	Open Charge Point Protocol
PLC	Power Line Communication
CP	Control Pilot
PP	Proximity Pilot

Chapter 1

Introduction

The vehicle-to-grid (V2G) technology represents one of the most promising innovations in the energy and mobility sectors. It enables electric vehicles (EVs) to interact dynamically with the power grid, allowing not only charging but also the return of electricity to the grid when needed. This bidirectional flow can contribute to stabilizing the electrical network, supporting the integration of renewable energy sources, and optimizing energy management systems. The V2G market is projected to grow at a compound annual growth rate (CAGR) of 27.6% between 2024 and 2034, reaching an estimated value of approximately $30.3 billion by 2034[1]. This impressive growth reflects the urgent need to develop flexible and resilient energy infrastructures, particularly in response to the global expansion of renewable energy. Increasing sales pushed the total number of electric cars on the world's roads to 26 million, up 60% relative to 2021, with BEVs accounting for over 70% of total annual growth[2].

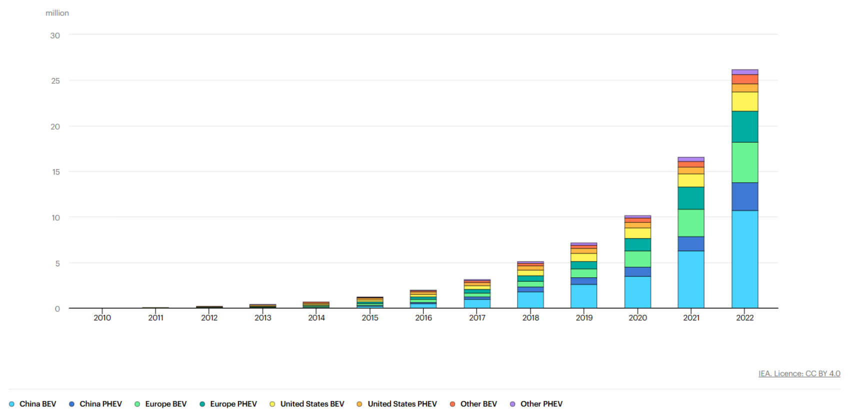


Figure 1.1: *Growth trend of EV around the world 2010-2022.*

The absolute increase in sales from 2021 to 2022 – around 3.5 million vehicles – mirrored the growth seen between 2020 and 2021, although the relative growth rate slowed. This trend reflects a normalization following the exceptional boom in 2021, which was likely driven by post-pandemic recovery. Overall, the 2022 growth rates for both annual EV sales and total stock align closely with the averages observed during 2015–2018, suggesting a robust and steady expansion of the EV market. The aggregation of such a large fleet of mobile storage units presents an unprecedented opportunity for the energy sector. Through V2G technologies, EVs could provide valuable grid services such as frequency regulation, peak shaving, voltage support, and spinning reserves. Pilot programs have demonstrated the practicality of this concept. The Parker Project in Denmark, for instance, successfully integrated EV fleets into the Danish power grid and proved that electric vehicles can consistently deliver grid services without negatively impacting the drivers' mobility needs[3].

Despite these promising developments, several challenges must be overcome before V2G becomes a widespread reality. One of the main technical barriers is the cost of bidirectional chargers, which are currently more expensive than conventional unidirectional chargers. Furthermore, interoperability remains a concern, as standardized communication protocols between vehicles, chargers, and grid operators are still under development. From a regulatory standpoint, grid codes and market participation rules need to be updated to allow vehicles to provide ancillary services legally and economically. Battery degradation is another frequently cited concern. There is a widespread belief that participating in V2G services might accelerate the aging of EV batteries, thereby reducing their lifespan. However, a study conducted by the University of Warwick found that controlled V2G operation can actually reduce battery degradation under certain charging strategies, or at least have minimal impact compared to regular driving and charging patterns[4].

On the policy side, several governments and agencies have started to actively support the deployment of V2G technologies. The European Union has introduced initiatives within the Green Deal framework that promote smart charging and V2G integration as tools for decarbonizing the transport and energy sectors. In the United States, the Department of Energy is funding numerous research programs aimed at developing the next generation of V2G systems and business models. In terms of environmental impact, V2G can significantly enhance the efficiency of renewable energy utilization. By providing distributed energy storage, EVs can mitigate the intermittency of wind and solar power generation, reducing the reliance on fossil-fueled peaker plants. A study by the U.S. National Renewable Energy Laboratory (NREL) estimates that coordinated EV charging and discharging could lower peak demand by up to 15% in future electric grids dominated by renewables[5].

In this context, the ISO 15118-20 standard plays a crucial role by defining communication protocols for bidirectional charging, enabling secure and interoperable energy exchanges between vehicles and the power grid. As implementation of this standard begins to scale, there is a growing need for experimental setups capable of testing these technologies under real or controlled conditions.

The objective of this thesis is to design and implement a laboratory infrastructure—referred to as a Charging Lab—dedicated to testing bidirectional charging operations according to the ISO 15118-20 standard. This facility will serve as a testbed for validating communication interoperability, power exchange control, and overall system behavior in V2G scenarios, contributing to the advancement of electric mobility within smart grid ecosystems.

1.1 Charging Stations and Connectors

Electric vehicle (EV) charging is a pivotal component in the shift toward sustainable mobility. Charging infrastructure, known as Electric Vehicle Supply Equipment (EVSE), acts as the interface between the power grid and the vehicle, ensuring safety, communication, and efficient energy transfer to the battery. EVSE can be primarily categorized into two types:

- **Private or semi-public**: Installed in residential or commercial settings such as homes, shared garages, office buildings, or shopping centers. These stations are typically connected directly to the user's electric meter and require installation by a qualified technician.
- **Public**: Located in accessible public spaces like streets, public parking lots, or service stations. Their installation is generally managed by energy providers or public authorities.

According to the *Global EV Outlook 2024* by the International Energy Agency (IEA), approximately 27 million private chargers were in operation globally in 2023, confirming home charging as the most common method for EV owners. Moreover, projections indicate that this number will increase more than tenfold by 2035, reaching 270 million units in the Stated Policies Scenario (STEPS) and up to 300 million in the Announced Pledges Scenario (APS) [6]. The IEA also reports that there are currently nearly ten times more private chargers than public ones, with the vast majority of EV charging occurring at home [7]. These figures highlight the strategic importance of residential charging infrastructure in facilitating the widespread adoption of electric vehicles. The **IEC 61851-1**[8] standard defines the operational modes for EV charging, establishing technical and safety requirements for various charging types. This regulation is essential to ensure interoperability between vehicles and infrastructure, laying the foundation of the global charging ecosystem. It introduces four charging modes — Mode 1, Mode 2, Mode 3, and Mode 4 — which differ in terms of communication requirements, safety features, and power levels. These modes are discussed in the next section.

1.1.1 Charging Modes According to IEC 61851-1

The IEC 61851-1 standard defines four charging modes —Mode 1 to Mode 4—each representing a different method for the conductive charging of electric vehicles (EVs), with varying degrees of safety, communication, and power capacity.

Mode 1

Mode 1 is the simplest and oldest EV charging mode. The vehicle is connected directly to a standard domestic outlet using a cable with no control or communication functions.

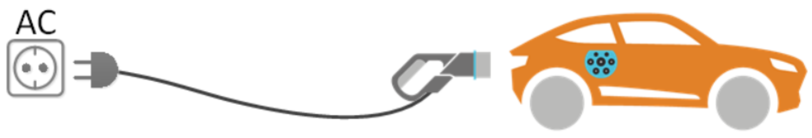


Figure 1.2: *Charging Mode 1.*[9]

- **Power Supply:** Single-phase or three-phase AC; typically up to 16 A; maximum power ≈ 3.7 kW.
- **Protection:** No integrated control; depends entirely on the grid and the vehicle's built-in protection systems.
- **Limitations:** Considered unsafe in many regions; not permitted in most EU countries due to risk of overheating and lack of ground fault protection.
- **Examples:** Only e-bikes or scooters; temporary/emergency use where proper charging equipment is unavailable.

Mode 2

Mode 2 allows charging from a standard socket using a cable that includes an **In-Cable Control and Protection Device (IC-CPD)**, which improves safety.

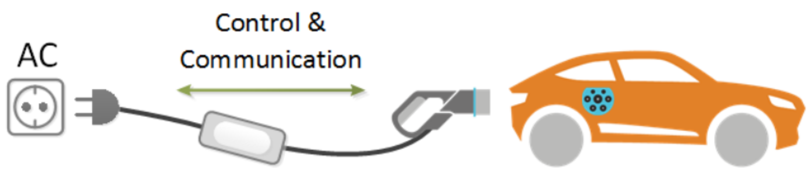


Figure 1.3: *Charging Mode 2.*[9]

- **Power Supply:** Single-phase or three-phase AC; up to 32 A; typical power 3.7–7.4 kW (single-phase), up to 22 kW (three-phase).
- **Protection:** IC-CPD provides overcurrent, ground fault, and temperature protections.
- **Communication:** Basic signaling for authorization.
- **Examples:** Used for private charging without a wallbox; often supplied with the EV.
- **Note:** Requires grounded and certified sockets for safe operation.

Mode 3

Mode 3 uses a dedicated EVSE (Electric Vehicle Supply Equipment) that includes full communication and protection features. It's mandatory for public EVSE and there are different type of connector presented in CEI EN 62196-1[10] and in CEI EN 62196-2[11].

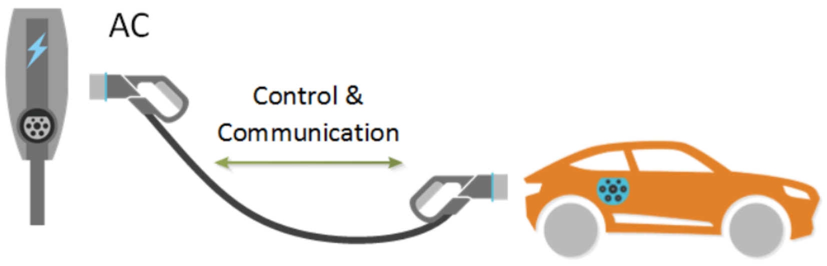


Figure 1.4: *Charging Mode 3.*[9]

- **Power Supply:** Single-phase or three-phase AC; up to 63 A; typical power up to 22 kW (some up to 43 kW).
- **Protection:** Integrated overcurrent, fault detection, and thermal control.
- **Communication:** Pilot control signal enables load management and smart charging.
- **Examples:** Public charging stations, private wallboxes in residential buildings.
- **Connector Types:** Type 1, Type 2, Type 3A, Type 3C. The following table present the standard range values.

Type	**Standard**	**Power**	**Voltage**	**Current**	**Pin**
Type 1	SAE J1772	19 kW	1-phase, 230 V	32 A	5
Type 2	Mennekes VDE-AR-E 2623-2-2	22 kW	1-phase, 250 V 3-phase, 500 V	1-ph: 70 A 3-ph: 63 A	7
Type 3	EV Plug Alliance 3A	4 kW	1-phase, 250 V	16 A	4
Type 3	EV Plug Alliance 3C	22 kW	1-phase, 250 V 3-phase, 500 V	1-ph: 32 A 3-ph: 63 A	7

Table 1.1: Technical comparison of EV connector types Mode 3

Typical outlets used for AC charging are shown in the figure below:

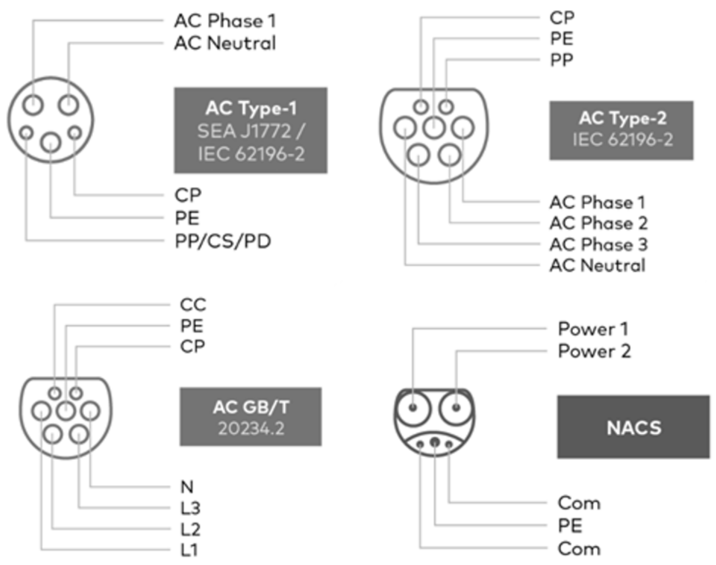


Figure 1.5: *a)AC Type 1 b)AC Type 2 c)AC GB/T d)NACS.*[12]

In the field of alternating current (AC) charging, various connector standards are adopted depending on the geographical region. In North America and Japan, the **Type 1** connector compliant with IEC 62196-2 is widely used, based on the SAE-J1772 standard. It supports single-phase charging with currents from 6 to 32 A, allowing power delivery up to 7.4 kW. In Europe, the prevalent standard is the **Type 2** connector, also defined by IEC 62196-2, which supports three-phase charging up to 63 A, reaching power levels up to 44 kW. Type 2 also allows single-phase charging by omitting the use of L2 and L3 pins, facilitating compatibility and ease of adaptation between Type 1 and Type 2 connectors[13].

In China, the **GB/T** 20234.2 standard is adopted, enabling both single-phase and three-phase charging in Mode 3. The charging cables used in this system employ identical male connectors at both ends, resembling Type 2 in shape, though not electrically compatible [14].

In the United States, the adoption of the **North American Charging Standard (NACS)**, originally developed by Tesla, is rapidly growing beyond the Tesla ecosystem. This connector supports AC charging via the IEC 61851 standard, enhancing interoperability with existing infrastructure. According to SAE International, NACS is currently undergoing standardization under the name SAE J3400, confirming the increasing interest of the U.S. automotive industry [15].

Mode 4

Mode 4 is dedicated to fast DC charging, where the AC/DC conversion is handled externally by the charging station.

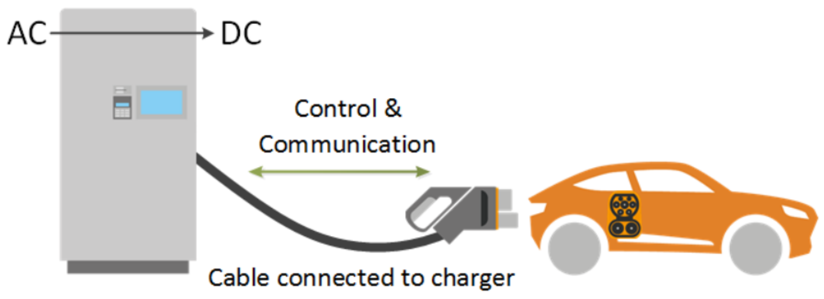


Figure 1.6: *Charging Mode 4.*[9]

- **Power Supply:** Direct current (DC); power range from 25 kW to over 350 kW; voltage from 200 to 1000 V.
- **Communication:** CAN or ISO 15118 protocols; supports Plug&Charge, charge authorization, and load balancing.
- **Protection:** Advanced safety features including interlocks, fault protection, and active cable cooling.
- **Examples:** Highway fast-charging networks (e.g., Tesla Superchargers, Ionity); fleet charging for buses and taxis.
- **Note:** Essential for long-range EVs and commercial applications with large battery packs.

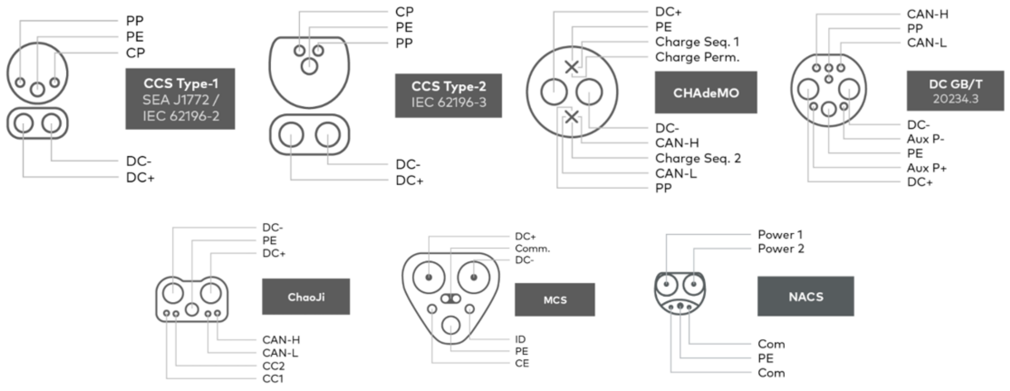


Figure 1.7: *a) CCS Type 1, b) CCS Type 2, c) DC CHAdeMO, d) DC GB/T, e) ChaoJi, f) MCS, g) NACS.* [12]

In the field of high-power DC charging (Mode 4), several standards have been developed to meet the specific requirements of different global markets. **CCS Type 1** (Combo 1) is the predominant standard in North

America: it uses a combined connector for both AC and DC charging, with communication based on the ISO 15118 protocol through Power Line Communication (PLC) [16]. In Europe, the equivalent is **CCS Type 2** (Combo 2), which follows the same design philosophy but is compatible with the widely used Type 2 AC plugs.

CHAdeMO, developed in Japan, was one of the first fast-charging DC standards. It uses CAN bus communication and gained wide adoption across Asia, although its charging power is now limited compared to newer technologies [17]. In China, the official system is **GB/T 27930 DC**, which also uses the CAN bus and features a separate DC connector [18].

To address fragmentation in the Asian market and enhance interoperability, China and Japan collaborated on the development of **ChaoJi**, a next-generation connector capable of supporting over 900 kW, backward compatible with both CHAdeMO and GB/T, and aligned with the ISO 15118 protocol [19].

The **Megawatt Charging System (MCS)** is a new standard designed specifically for heavy-duty and commercial vehicles, supporting charging powers up to 3.75 MW, also based on ISO 15118 [20]. Finally, **NACS** (North American Charging Standard) is Tesla's proprietary connector, compact and versatile, capable of handling both AC and DC high-power charging. NACS is quickly becoming a de facto standard in the U.S., as several other OEMs are adopting it [21].

The following table shows medium charging time using the four different types of charge:

Mode	**Connectors**	**Power**	**Charging Time (20–80%)**
Mode 1	Schuko (EU), Type A (US)	2.3–3.7 kW	10–20 h
Mode 2	Schuko, Type A + Type 1/2	2.3–7.4 kW	6–12 h
Mode 3	Type 1, Type 2, NACS	7.4–43 kW	2–6 h
Mode 4	CCS1, CCS2, CHAdeMO, GB/T, ChaoJi, MCS, NACS	50–1000+ kW	15–60 min

Table 1.2: Summary of EV Charging Modes and Connectors. Charging time is indicative for EVs with 50–80 kWh batteries. Power range includes typical and maximum values.

1.1.2 European situation

In Europe, electric vehicle charging is regulated by specific standards to ensure the interoperability and safety of infrastructure. The Type 2 connector, defined by the IEC 62196-2 standard, is the mandatory standard for public alternating current (AC) charging stations in the European Union. For direct current (DC) charging, the Combined Charging System (CCS) Combo 2, described in the IEC 62196-3 standard, is the adopted standard for high-power charging stations. As of December 31, 2024, there were approximately 882,000 public electric vehicle charging points in Europe. The majority use alternating current (AC), accounting for around 84% of the total, typically offering power up to 22 kW and equipped with the Type 2 connector, which is mandatory in the European Union. Direct current (DC) charging stations, mainly used for fast charging, make up about 16% of the total, with a significant portion — around 10% — consisting of High Power Charging (HPC) units delivering power above 150 kW. HPC stations experienced particularly rapid growth between 2023 and 2024. DC chargers use the CCS Combo 2 connector, the European standard for high-power charging. The Netherlands, Germany, and France alone host nearly half of all public charging points in Europe [22, 23].

1.2 Battery for automotive

Lithium-ion batteries currently represent the predominant solution for the electrification of the automotive sector. Their success is the result of a combination of factors, including high energy density, good conversion efficiency, the possibility of relatively fast charging, and the availability of a wide range of cathode and anode materials that allow tailoring of the cell chemistry to specific applications. Despite the existence of numerous chemistries, the market has progressively focused on three main cathode types: **Nickel-Manganese-Cobalt (NMC)**, **Nickel-Cobalt-Aluminium (NCA)**, and **Lithium-Iron-Phosphate (LFP)**. These three chemistries today constitute the pillars of electric mobility, each adopted according to the required trade-offs in terms of specific energy, cycle life, safety, and cost.

1.2.1 NMC Batteries (Nickel-Manganese-Cobalt)

NMC cells are based on ternary oxides of the type $LiNi_xMn_yCo_{1-x-y}O_2$. The ratio between nickel, manganese, and cobalt is variable and has led to several commercial formulations, identified by stoichiometric ratios such as NMC 111 (1:1:1), NMC 532, NMC 622, and NMC 811. Increasing the nickel fraction enhances the specific capacity, while reducing cobalt addresses cost and sustainability concerns.

Electrochemically, NMC batteries exhibit a nominal voltage of about 3.6–3.7 V, with energy density values typically in the range of 200–300 Wh/kg at the cell level. This makes them suitable for vehicles requiring medium to high range. Charge/discharge efficiency generally exceeds 90%, and the relatively low internal resistance allows for sustained discharge currents. Cycle life depends on composition and operating conditions: on average, 1,000–2,000 cycles can be achieved before the capacity falls below 80% of the initial value. Ageing is mainly driven by the growth of the Solid Electrolyte Interphase (SEI) on the anode and structural degradation of the cathode, with oxygen release and thermal instability being more severe in high-nickel formulations [24].

From a safety standpoint, NMC cells are relatively stable but remain susceptible to thermal runaway in conditions of overcharge, short circuit, or excessive thermal stress. For this reason, automotive battery packs always include active thermal management systems. The main limitation of NMC batteries lies in the cost of raw materials, especially nickel and cobalt, whose extraction also raises environmental and social issues [25]. Current research trends aim at lowering cobalt content while maintaining comparable performance.

1.2.2 NCA Batteries (Nickel-Cobalt-Aluminium)

NCA cells ($LiNi_xCo_yAl_yO_2$) are a variant of NMC in which manganese is replaced by aluminium. This substitution improves structural stability while maintaining a high nickel content, thus ensuring very high energy density. At the cell level, nominal voltage is similar to NMC (3.6–3.7V), but the specific capacity is higher, ranging between 240 and 300mAh/g at the cathode, with energy densities exceeding 300Wh/kg. Coulombic efficiency is also high, generally above 92–94%. The very high energy density makes NCA particularly suited for premium and long-range vehicles, as demonstrated by Tesla's use of this chemistry in its top models. However, NCA cells show certain drawbacks. Cycle life is moderate, typically 1,000–1,500 cycles, with degradation accelerating under fast charging or high-temperature operation. Moreover, NCA exhibits reduced thermal stability, with decomposition starting at around 180–200 °C, lower than LFP [24]. In terms of safety, NCA poses higher instability risks, requiring advanced Battery Management Systems (BMS) and liquid cooling to guarantee reliability. This increases system complexity but has not prevented NCA from being a reference technology for applications prioritizing range and performance.

1.2.3 LFP Batteries (Lithium-Iron-Phosphate)

LFP batteries ($LiFePO_4$) use iron phosphate as the cathode material, characterized by a very stable olivine crystal structure. Unlike NMC and NCA, LFP has a lower nominal voltage, around 3.2–3.3 V, and lower energy density, typically between 90 and 160 Wh/kg.

The key advantage of LFP is its excellent thermal and chemical stability. Thermal decomposition occurs at much higher temperatures compared to other chemistries (above 250 °C), drastically reducing the likelihood of thermal runaway. In addition, the absence of critical raw materials such as nickel and cobalt makes LFP both safer and more cost-effective. Another strength is the extended cycle life. LFP cells can exceed 3,000 charge/discharge cycles without significant capacity loss, and under optimized conditions can even reach 6,000 cycles. This makes them particularly suitable for commercial vehicles, electric buses, and entry-level passenger cars, where durability and reliability outweigh maximum driving range[24].

The main drawbacks are their lower energy density, which limits range, and reduced low-temperature performance, with significant power losses below 0 °C. Moreover, their lower nominal voltage requires more complex pack configurations to reach high system voltages, increasing the number of cells in series. Despite these limitations, the combination of low cost, safety, and longevity has made LFP the dominant chemistry in China and increasingly widespread in Europe and the United States, especially for entry-level EVs and urban fleets.

1.2.4 Comparison of the Three Chemistries

The comparison among NMC, NCA, and LFP highlights their distinct application strategies. NCA offers the highest energy density, making it the preferred choice for premium long-range vehicles. NMC provides a balanced trade-off, making it the most widely adopted in mid-range vehicles. LFP, in contrast, sacrifices energy density in favor of safety, longevity, and cost-effectiveness, making it particularly attractive for city cars and light commercial vehicles.

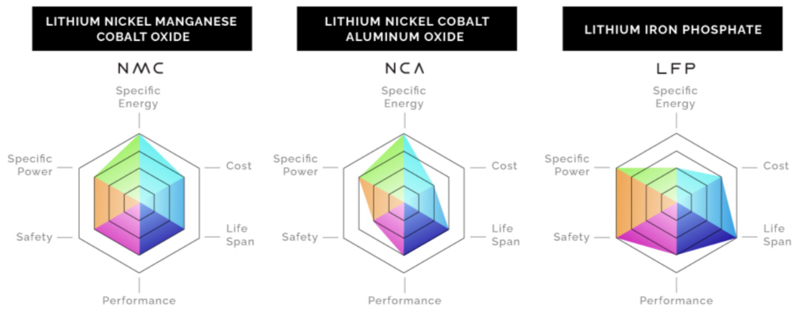


Figure 1.8: *Qualitative comparison of the main lithium-ion battery chemistries (NMC, NCA, LFP).*

1.3 Communication protocols

During the development of the smart charging lab, the integration of various devices, including computers, simulation instruments, and power electronics components, was necessary. To enable proper data exchange and coordination among these units, several communication protocols were adopted. These protocols allow for command management, real-time parameter monitoring, and synchronization of system operations. Choosing the most suitable protocols was essential to ensure the reliability and overall efficiency of the laboratory. The main protocols used will be described in the following.

1.3.1 CAN

The Controller Area Network (CAN) protocol was developed by Bosch in the early 1980s to meet the growing need for reliable communication between electronic control units in vehicles. Unlike point-to-point systems, CAN uses a multi-master architecture and bus communication, making it efficient, scalable, and robust. Its design ensures high noise immunity, effective error handling, and message prioritization based on identifiers. These features have led to its widespread adoption in automotive, industrial, railway, and medical applications. CAN has become the de facto standard for embedded networks and serves as the foundation for more complex protocols such as CANopen and SAE J1939[26].

In the CAN protocol, communication takes place through a broadcasting system: every message sent by a node on the network is received by all other connected nodes without point-to-point addressing. This means that all devices must monitor the bus to recognize messages of interest based on the identifier contained in the frame(ID). This approach simplifies the architecture and enables efficient, deterministic real-time communication, which is essential for critical embedded systems.

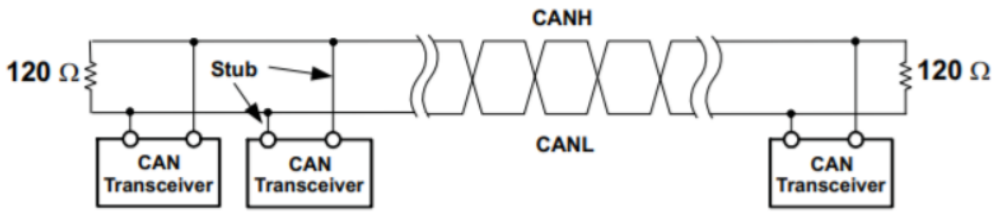


Figure 1.9: *CAN ECU broadcast trasmission* [27]

To prevent signal reflections and ensure communication integrity, the bus is terminated at its ends with two 120-ohm resistors. These termination resistors are essential because they provide proper impedance matching between the bus and the devices, preventing electromagnetic waves from bouncing along the cable and causing interference and signal distortion. Their presence also helps maintain a stable voltage level on the bus, ensuring reliable data transmission even at high communication speeds.

Physically, the CAN network uses a linear bus topology consisting of two twisted wires called CAN_H (High) and CAN_L (Low). These two conductors transmit signals in differential mode, meaning the data is not represented by a voltage with respect to ground but by the potential difference between CAN_H and CAN_L. In the dominant bit state, CAN_H rises to about 3.5 V while CAN_L drops to about 1.5 V, generating a difference of approximately 2 V. In the recessive bit state, both wires stay at an intermediate level around 2.5 V, canceling out the difference. This differential method is chosen to improve signal immunity to electromagnetic noise and interference, as any noise tends to affect both wires symmetrically and is thus canceled out when calculating the difference.

Structure of a standard CAN frame

In the CAN protocol, communication occurs through well-defined frames transmitted sequentially over the bus. Each frame contains specific fields, each with a defined function, useful for data transmission, synchronization, and error management.

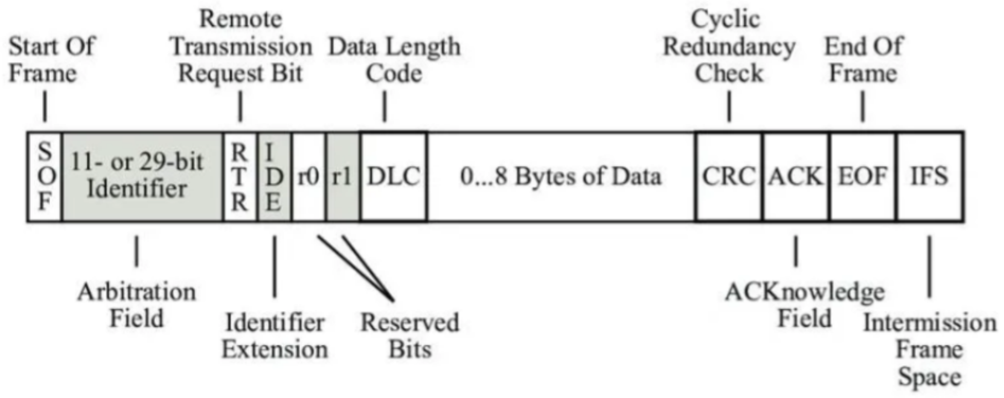


Figure 1.10: *CAN frame.*

Idle State When the bus is not occupied by any frame, it is in the *idle* state. In this condition, both CAN_H and CAN_L lines are at the same logical level (recessive, corresponding to logic 1). This state indicates that the channel is free and ready for a new transmission.

Start of Frame (SOF) Every data frame starts with a dominant bit (logic 0), called the Start of Frame (SOF). This bit informs all nodes on the network that a new transmission is starting, allowing the receiving nodes to synchronize with the transmitting node.

Message Identifier (ID) The next field is the Message Identifier, which is 11 bits long in the standard format. It represents the priority of the message: lower binary values indicate higher priority. In some implementations, certain bits may be masked or reserved; for example, using only 9 effective bits results in 512 possible identifiers (from 0x000 to 0x1FF).

Remote Transmission Request (RTR) The RTR bit distinguishes between:

- Dominant (0): *Data Frame*
- Recessive (1): *Remote Frame* (data request)

Reserved Bits (R1, R0) After the RTR bit, there are two reserved bits, R1 and R0. Originally intended for future protocol extensions, they are currently used as follows:

- R1 is used as the IDE bit to distinguish between standard and extended frames
- R0 is used as the FDF bit to distinguish between CAN and CAN-FD

Data Length Code (DLC) The DLC field is 4 bits long and specifies the number of data bytes in the frame. Its value can range from 0 to 8 in the classical CAN version.

Data Field This field contains the actual data to be transmitted. Its length is defined by the DLC field and can range from 0 to 8 bytes.

Cyclic Redundancy Check (CRC) A 15-bit field used for error detection. The transmitting node calculates and sends the checksum, which the receiving node verifies to validate the frame.

CRC Delimiter A recessive bit that follows the CRC field. It provides time for the receiver to process and validate the checksum.

Acknowledgement (ACK) This field is sent as a recessive bit by the transmitter. All receivers that have successfully received the frame overwrite it with a dominant bit. If the transmitter does not detect a dominant bit, it registers an ACK error.

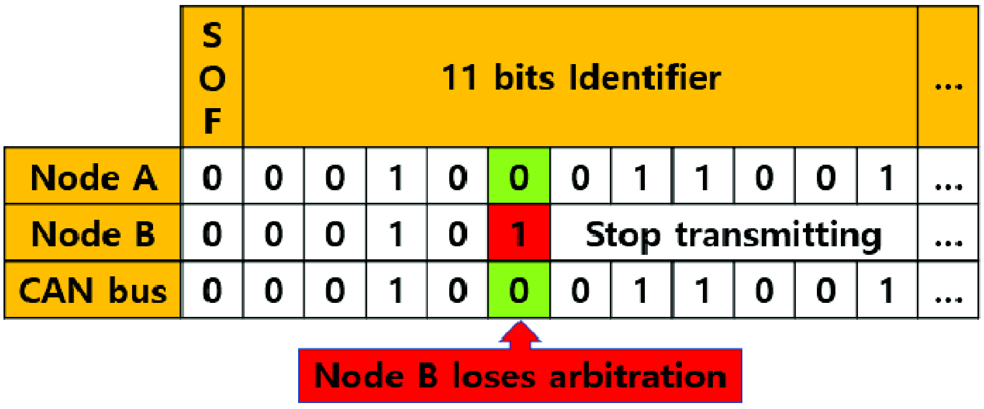
ACK Delimiter A recessive bit that separates the ACK slot from the end of the frame.

End of Frame (EOF) A sequence of 7 recessive bits marking the end of the frame.

Interframe Space (IFS) After the EOF, 3 recessive bits form a pause between consecutive frames, helping to avoid collisions and give nodes time to process the message.

Bus arbitration

One of the most significant features of the CAN protocol is its method of bus arbitration, which ensures that multiple nodes can attempt to transmit simultaneously without data collision or corruption. Arbitration is achieved through a non-destructive bitwise mechanism that takes place during the transmission of the Message ID field. Since the CAN protocol operates with dominant and recessive logic levels—where a dominant bit (logic 0) overrides a recessive bit (logic 1)—nodes can detect if another node is transmitting a message with higher priority.



	SOF	11 bits Identifier											...
Node A	0	0	0	1	0	0	0	1	1	0	0	1	...
Node B	0	0	0	1	0	1	Stop transmitting						...
CAN bus	0	0	0	1	0	0	0	1	1	0	0	1	...

Figure 1.11: *Example of 2 CAN nodes arbitration* [28]

Each message on the CAN bus is identified by an ID, which also represents its priority: the lower the numerical value of the ID, the higher its priority. When two or more nodes start transmitting at the same time, they place their ID bits on the bus one by one. As each bit is transmitted, every node also monitors the bus to check if the actual level matches the one it sent. If a node sends a recessive bit but reads back a dominant level, it immediately stops transmitting, conceding the bus to the node with the higher-priority message. This arbitration mechanism is non-destructive because no data is lost during the contention. The winning node continues transmitting its frame, while the others wait and reattempt after the bus becomes idle again. This strategy is particularly well-suited for real-time systems, where timely and prioritized delivery of critical messages is essential.

Types of CAN Frames

In the Controller Area Network (CAN), communication takes place through logical units called *frames*. Each frame encapsulates a message and follows a well-defined structure. There are four main types of CAN frames, each serving a distinct role in the communication process.

The **Data Frame** is the most common and is used for actual data transmission. It includes several fields such as the identifier (which determines priority), control bits, the data field (up to 8 bytes in standard CAN 2.0), and a CRC for error detection.

The **Remote Frame** is used to request data from another node. It mirrors the structure of a Data Frame but omits the data field. A node receiving this frame responds by sending a Data Frame with the requested identifier.

The **Error Frame** is transmitted by a node that detects a violation in the message structure or a transmission fault. Its purpose is to interrupt the communication and inform all other nodes of the problem, prompting a retransmission.

The **Overload Frame** is used to introduce a short delay in communication when a node needs additional time before it can process or receive further messages. These frame types make the CAN protocol versatile and robust, even in real-time and safety-critical applications.

Bit Stuffing and Synchronization

CAN communication relies on asynchronous transmission without a shared clock line. To preserve synchronization between transmitter and receivers, the protocol uses a mechanism known as **bit stuffing**.

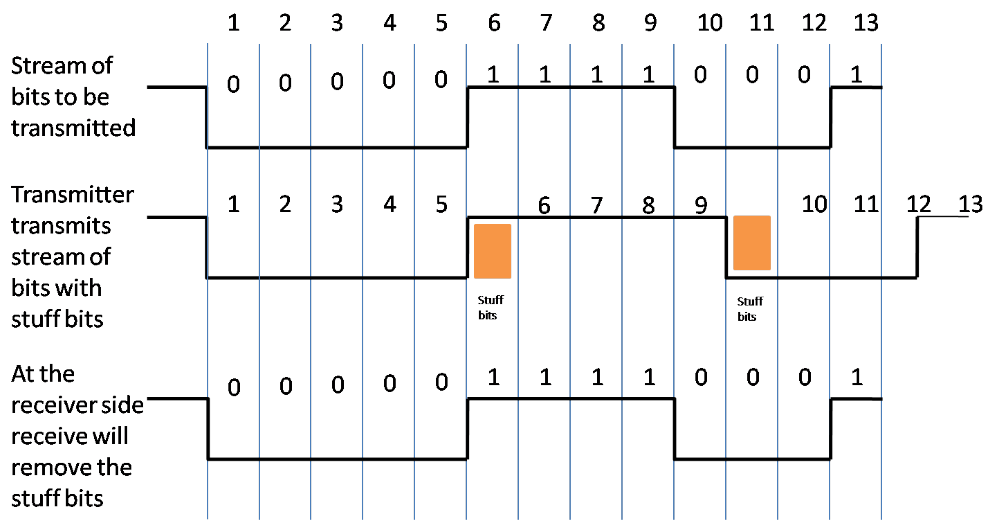

1 2 3 4 5 6 7 8 9 10 11 12 13
Stream of bits to be transmitted
0 0 0 0 0 1 1 1 1 0 0 0 1
Transmitter transmits stream of bits with stuff bits
1 2 3 4 5 6 7 8 9 10 11 12 13
Stuff bits
Stuff bits
At the receiver side receive will remove the stuff bits
0 0 0 0 0 1 1 1 1 0 0 0 1

Figure 1.12: *CAN stuffing and destuffing bits*

When five consecutive bits of the same logical level are transmitted, the sender inserts a sixth bit of opposite polarity. This artificial transition ensures that receivers maintain proper timing synchronization. The receiver, aware of the bit stuffing rule, removes the stuffed bits during decoding to reconstruct the original message. If a violation of this rule occurs (e.g., six identical bits in a row), it triggers a *stuff error*, causing the frame to be discarded and retransmitted. This mechanism is essential to ensure the integrity of data in the absence of a dedicated clock signal [29].

Error Detection and Fault Confinement

One of the main strengths of the Controller Area Network (CAN) protocol is its intrinsic ability to **detect and autonomously handle transmission errors**. This feature is essential to ensure the reliability of distributed embedded systems, especially in safety-critical domains such as automotive applications, where data corruption or message loss can compromise vehicle safety. The protocol implements several control mechanisms to preserve data integrity and ensure robust network operation, even in the presence of disturbances or partial faults. The main types of detectable errors are:

- **Bit Error**: occurs when the bit read on the bus differs from the one transmitted by the node, often due to collisions or noise.

- **Stuff Error**: detected when the bit stuffing rule is violated. This rule requires inserting a complementary bit after five consecutive identical bits to maintain synchronization.
- **CRC Error**: arises when the result of the cyclic redundancy check (CRC) does not match between the sender and receiver, indicating potential data corruption.
- **Form Error**: occurs when one or more frame fields do not respect the expected format (e.g., delimiters or control bits).
- **Acknowledgment Error**: occurs if the transmitting node does not receive any acknowledgment (ACK) from the receivers, indicating that no node confirmed successful reception.

Each CAN node internally maintains two error counters: the *Transmit Error Counter (TEC)* and the *Receive Error Counter (REC)*. These counters are dynamically updated based on the transmission and reception errors detected, and they determine the node's operational state:

- **Error Active**: the node is fully operational and can transmit normally.
- **Error Passive**: the node is still functional but with limited transmission capability; for example, it does not generate active error frames on the bus.
- **Bus Off**: the node is automatically disconnected from the network when the error count exceeds a critical threshold (typically TEC $>$ 255), to prevent it from disrupting overall communication.

Thanks to these self-regulating mechanisms, the CAN protocol provides a high level of **fault tolerance**, making it especially suitable for applications where reliability is a key requirement, such as electric vehicle control units, ABS systems, airbags, and powertrain modules [30].

Transmission Speed and Physical Limitation

The effective performance of a CAN network is influenced by physical layer constraints in addition to protocol logic. In particular, the **maximum transmission speed** is inversely related to the total bus length.

Bus Length	Max Bitrate
$\leq$ 40 m	1 Mbit/s
100 m	500 kbit/s
250 m	250 kbit/s
500 m	125 kbit/s
1000 m	50 kbit/s

Table 1.3: Recommended maximum speed vs. bus length

To minimize signal reflection and degradation, the CAN bus must be terminated with 120 Ω resistors at both ends. The topology should be linear, avoiding long stubs or branches, which could introduce unwanted delay and reflections.

CAN FD: Flexible Data Rate Extension

CAN FD (Flexible Data-rate) is an extension of the classical CAN protocol, developed to meet the increasing demand for higher data throughput and faster communication in embedded systems. It introduces two key improvements over classical CAN (also known as CAN 2.0):

- **Extended data field**: while CAN 2.0 allows a maximum payload of 8 bytes per frame, CAN FD supports up to 64 bytes, significantly increasing the amount of data that can be transmitted in a single message.
- **Dual bit rate**: CAN FD uses the standard CAN bit rate during the arbitration phase, ensuring compatibility with legacy nodes, but allows for a higher bit rate (up to 8 Mbit/s) during the data phase. This is enabled by the Bit Rate Switching (BRS) mechanism.

Despite sharing the same physical layer specifications (ISO 11898-2), classical CAN nodes are not capable of interpreting CAN FD frames.

Therefore, mixed CAN/CAN FD networks must be carefully designed to prevent communication errors, usually by ensuring that legacy nodes do not attempt to process FD messages.

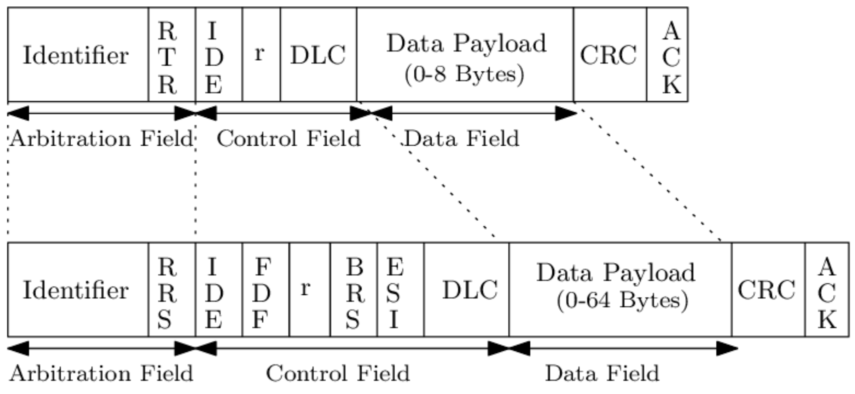


Figure 1.13: Comparison between Classical CAN (CAN 2.0) and CAN FD frame formats.

CAN FD has been widely adopted in modern automotive and industrial applications that require high-speed and high-volume data exchange, such as battery management systems, advanced driver-assistance systems (ADAS), and real-time control units [31].

1.3.2 ETHERNET

Ethernet is the most widely used wired communication technology for local area networks (LANs), standardized as *IEEE 802.3*. Its universal adoption is based on a combination of simplicity, scalability, and backward compatibility. Within the *OSI model*, Ethernet operates mainly at the *Data Link layer*, where it is responsible for physical addressing and reliable delivery of frames.

A key element of the technology is the *Media Access Control (MAC)* structure, which provides unique addresses for each network interface and regulates access to the transmission medium. Furthermore, Ethernet serves as the underlying layer for the *Internet Protocol (IP)*, located at the Network layer, making the interaction between the two fundamental to modern networking infrastructures [32, 33].

Historical Origins and Standardization

Ethernet was developed in 1973 by Robert Metcalfe and David Boggs at Xerox PARC, inspired by the ALOHA radio system. The first prototype used a coaxial bus cable with a speed of 2.94 Mbps [34]. In 1983 the IEEE released the first *IEEE 802.3* standard, which formally defined the rules for medium access and frame structure.

Standardization, combined with the simplicity of implementation and reduced costs, led Ethernet to dominate over competing technologies such as IBM's Token Ring and ANSI's FDDI [35].

Structure of the Ethernet Frame

The Ethernet frame is the fundamental unit of transmission, consisting of several well-defined fields, each serving a specific function to ensure communication between devices:

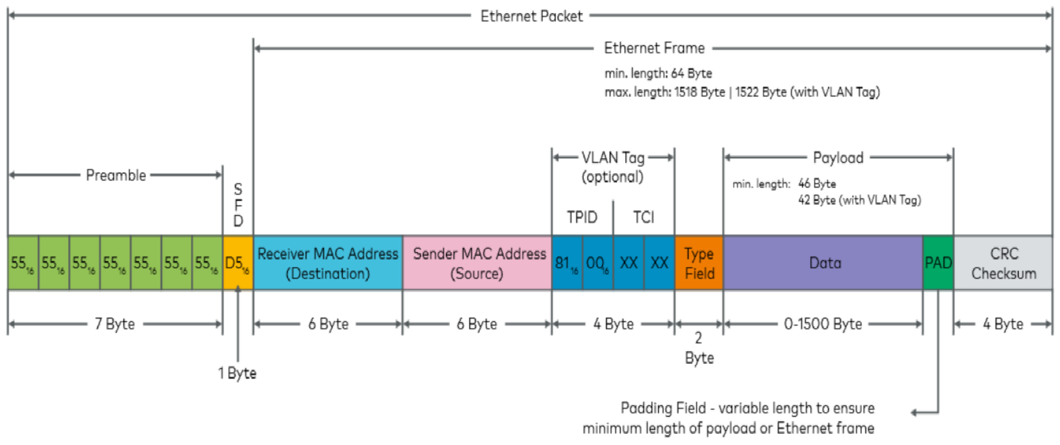


Figure 1.14: *ETHERNET frame.*

- **Preamble (7 bytes) + Start Frame Delimiter (1 byte):** This initial sequence does not carry user data but is crucial for synchronization. The preamble, made of alternating bits, allows the receiving network interface to align with the transmission signal, while the Start Frame Delimiter marks the exact beginning of the frame.

- **Destination MAC address (6 bytes):** This field specifies the intended recipient of the frame. It may correspond to a single host (unicast), a group of hosts (multicast), or all devices on the network (broadcast). This allows Ethernet to perform direct delivery within a LAN without relying on higher-layer addressing such as IP.
- **Source MAC address (6 bytes):** This field indicates the unique identifier of the network interface that transmitted the frame. MAC addresses are globally unique, as they are assigned by hardware vendors using an *Organizationally Unique Identifier (OUI)* prefix. It ensures that the recipient can always identify the origin of the data.
- **Type/Length field (2 bytes):** In Ethernet II frames, now predominant, this field specifies the upper-layer protocol encapsulated, such as IPv4 (0x0800) or IPv6 (0x86DD). In IEEE 802.3 frames, instead, it denotes the length of the payload. In both cases, the function is to enable the receiving device to interpret the payload correctly.
- **Payload or data field (46–1500 bytes):** This is the actual information carried, usually an IP packet. The minimum size of 46 bytes originated from the requirements of CSMA/CD, where transmission time had to be long enough to detect collisions. If the data is smaller, padding bytes are added.
- **Frame Check Sequence (FCS, 4 bytes):** This field contains a 32-bit cyclic redundancy check (CRC-32) that allows the receiver to detect transmission errors. If the calculated checksum does not match the transmitted one, the frame is discarded. Ethernet does not attempt error correction; retransmissions are managed by higher-layer protocols such as TCP.

The stability of the frame format over decades demonstrates the robustness of its design: despite the dramatic evolution in transmission speeds and switching technologies, Ethernet frames remain structurally compatible with the earliest implementations [32, 33].

The Ethernet MAC Layer

Within the IEEE 802.3 standard, the Media Access Control (MAC) sublayer plays a central role in defining how devices access the medium and how they are uniquely identified at the Data Link layer. The MAC layer provides the addressing scheme, regulates the access control mechanism, and ensures the correct encapsulation of higher-layer protocols into Ethernet frames.

Each network interface card (NIC) is assigned a unique MAC address, 48 bits in length, which serves as a hardware identifier. The address is divided into two sections: the first 24 bits form the Organizationally Unique Identifier (OUI), managed by the IEEE and assigned to manufacturers, while the remaining 24 bits are vendor-specific, guaranteeing global uniqueness. This mechanism ensures that no two devices on the same network segment share the same MAC address. Ethernet supports different modes of addressing. In unicast communication, a frame is directed to a single specific host, identified by its MAC address. In broadcast, the destination field is set to the all-ones address (FF:FF:FF:FF:FF:FF), which is recognized by all nodes on the LAN. Finally, multicast addresses allow communication with a defined subset of hosts, a feature widely used by IPv6 and multimedia applications [35].

The original versions of Ethernet relied on the Carrier Sense Multiple Access with Collision Detection (CSMA/CD) algorithm to control medium access. In this scheme, devices listen to the medium before transmitting and detect collisions by monitoring the voltage on the line. When a collision is detected, all transmitting devices stop, wait for a random backoff period, and then retry. This mechanism guaranteed fairness but limited performance as network utilization increased. With the introduction of switches, each port became a separate collision domain, effectively eliminating collisions and rendering CSMA/CD obsolete in modern full-duplex Ethernet. In addition to addressing and medium access, the MAC layer defines the encapsulation of higher-layer data into frames. It ensures that the payload is properly structured, includes the necessary padding when required, and appends the Frame Check Sequence for error detection. In this way, the MAC layer constitutes the foundation upon which higher protocols such as IP can reliably operate. Despite decades of technological evolution, the fundamental logic of the Ethernet MAC has remained stable, proving its robustness and adaptability to new contexts [36].

Integration between Ethernet and IP

The interaction between Ethernet and the Internet Protocol (IP) is fundamental to the operation of modern networks. Ethernet provides the data link layer service, delivering frames within a local segment, while IP operates at the network layer, enabling communication across heterogeneous networks. The integration of the two layers is achieved through encapsulation and through mechanisms that allow IP to map logical addresses onto physical MAC addresses. An Ethernet frame can encapsulate different network-layer protocols, identified by the *Type* field. For instance, the hexadecimal values 0x0800 and 0x86DD indicate that the payload contains an IPv4 or IPv6 packet, respectively. This simple mechanism allows Ethernet to remain independent of the network protocol while ensuring interoperability.

A key aspect of this integration is the resolution of IP addresses into MAC addresses. In IPv4 networks, this is performed by the Address Resolution Protocol (ARP). When a host needs to deliver an IP packet to another device on the same LAN, it broadcasts an ARP request containing the target IP. The host with the corresponding IP responds with its MAC address, which is then cached locally for subsequent communications. In IPv6, ARP is replaced by the Neighbor Discovery Protocol (NDP), which leverages ICMPv6 messages and multicast addressing, improving efficiency and security. Ethernet also supports the transmission of IP multicast traffic by mapping IP multicast addresses into special ranges of Ethernet multicast addresses. This mechanism enables efficient group communication, particularly relevant in applications such as video streaming, online conferencing, or routing protocols like OSPF. However, the mapping is not one-to-one, which may lead to packet delivery to hosts that are not part of the intended IP multicast group, a limitation partially mitigated by switch-level filtering[37].

Finally, Ethernet's role in supporting higher-layer protocols extends beyond IP to include encapsulations such as VLAN-tagged frames (*IEEE 802.1Q*), which separate logical networks within the same physical infrastructure. This capability allows IP subnets to be organized efficiently, providing isolation, security, and traffic management. In this sense, Ethernet does not merely act as a passive transport for IP but provides mechanisms that directly influence the structure and performance of IP-based networks[38].

Ethernet and LAN Switching

In its original form, Ethernet was based on a shared medium, where all devices were connected to the same coaxial cable segment and competed for access using the CSMA/CD mechanism. While this approach was sufficient for small networks, it imposed significant limitations on scalability. Every additional node increased contention, and the entire LAN effectively operated as a single collision domain. As the popularity of Ethernet grew, the need to separate traffic and increase efficiency led to the development of switching technologies.

The first step toward improved scalability came with the introduction of bridges. Operating at the data link layer, bridges separated a LAN into multiple segments while forwarding traffic only when necessary. This segmentation reduced the number of collisions and allowed for more predictable performance. However, bridges were limited in scalability due to their reliance on spanning tree algorithms, which could lead to sub-optimal path selection [39].

Ethernet then transitioned to the use of LAN switches, which essentially function as multiport bridges with higher performance and dedicated hardware support for forwarding. A switch creates a separate collision domain for each port, effectively eliminating the contention problem of shared Ethernet. When operating in full-duplex mode, collisions disappear entirely, enabling simultaneous bidirectional communication. This architectural change marked a decisive shift in Ethernet from a broadcast-based technology to a switched fabric [40]. Switching also introduced important mechanisms for network optimization and management. Features such as Virtual LANs (VLANs) allow logical segmentation of traffic independent of physical topology, reducing broadcast domains and improving security and efficiency. Furthermore, techniques such as link aggregation provide redundancy and load balancing by combining multiple physical links into a single logical channel. These mechanisms have allowed Ethernet to scale from simple local connections to large enterprise backbones.

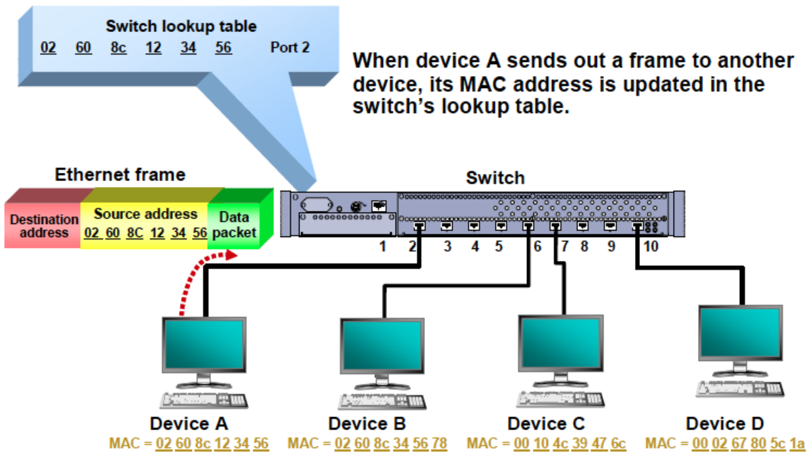


Figure 1.15: *Example of Ethernet's swith device.*[40]

An essential advantage of switched Ethernet is its support for quality of service (QoS) and advanced traffic handling. Modern switches implement prioritization schemes based on IEEE 802.1p, enabling latency-sensitive applications such as voice-over-IP and video streaming to coexist with best-effort traffic. Together, these developments have transformed Ethernet from a basic LAN technology into a flexible, scalable, and high-performance solution suitable for data centers and carrier-grade networks[41].

Evolution of Ethernet Speeds and Standards

Since its introduction in the 1970s, Ethernet has undergone a remarkable evolution in terms of transmission speeds and physical media, while maintaining backward compatibility at the frame format and MAC layer. This continuity has been one of the decisive factors behind Ethernet's dominance in local and wide-area networking. Each new generation has addressed the growing demand for bandwidth, efficiency, and integration with emerging technologies, while preserving interoperability with earlier standards [31].

The original Ethernet standard, ratified as IEEE 802.3 in 1983, operated at 10 Mbps over coaxial cable. This was followed by *Fast Ethernet* (IEEE 802.3u, 1995), which increased throughput to 100 Mbps and introduced twisted-pair cabling as the dominant medium. The next milestone was Gigabit Ethernet (IEEE 802.3z, 1998), achieving 1 Gbps primarily over fiber optic links, with later extensions to copper cabling (*1000BASE-T*). Gigabit Ethernet quickly became the standard in enterprise and data center environments. The need for higher speeds led to the deployment of 10 Gigabit Ethernet(IEEE 802.3ae, 2002), the first Ethernet standard designed without CSMA/CD and exclusively for full-duplex switched environments. This eliminated the historical constraints of shared media and enabled Ethernet to compete directly with technologies like SONET and ATM in backbone networks. Subsequent enhancements expanded Ethernet to 40 Gbps and 100 Gbps (IEEE 802.3ba, 2010), driven largely by the requirements of data centers, cloud computing, and high-performance computing systems[30].

Year	**IEEE Standard**	**Nominal Speed**	**Primary Medium**
1983	IEEE 802.3	10 Mbps	Coaxial (10BASE-5, 10BASE-2)
1995	IEEE 802.3u	100 Mbps (Fast Ethernet)	Twisted Pair (100BASE-TX)
1998	IEEE 802.3z	1 Gbps (Gigabit Ethernet)	Fiber Optics (1000BASE-SX/LX)
1999	IEEE 802.3ab	1 Gbps	Twisted Pair (1000BASE-T)
2002	IEEE 802.3ae	10 Gbps	Fiber Optics (10GBASE-SR/LR)
2010	IEEE 802.3ba	40 Gbps / 100 Gbps	Fiber Optics (40G/100GBASE)
2016	IEEE 802.3bz	2.5 Gbps / 5 Gbps	Twisted Pair (2.5G/5GBASE-T)
2017	IEEE 802.3bs	200 Gbps / 400 Gbps	Fiber Optics (200G/400GBASE)
Ongoing	IEEE 802.3df	800 Gbps / 1 Tbps	Fiber Optics (in development)

Table 1.4: Evolution of Ethernet Speeds and Standards

More recently, the IEEE has standardized 200 Gbps and 400 Gbps Ethernet (IEEE 802.3bs, 2017), with ongoing developments toward 800 Gbps and 1 Tbps. These implementations rely on parallel transmission lanes, advanced modulation schemes, and high-quality optical interfaces. Despite these dramatic increases in bandwidth, Ethernet continues to maintain a consistent frame format and MAC layer operation, ensuring that existing higher-layer protocols such as IP remain unaffected.

The evolution of Ethernet speeds has been accompanied by changes in cabling and physical layer technologies. While twisted-pair copper dominated access networks at 100 Mbps and 1 Gbps, optical fiber has become indispensable at higher speeds due to its superior bandwidth and signal integrity. Additionally, intermediate standards such as 2.5GBASE-T and 5GBASE-T were introduced to extend the life of existing Cat5e/Cat6 infrastructures, bridging the gap between Gigabit and 10 Gigabit Ethernet. These developments illustrate the adaptability of Ethernet to support both cost-sensitive enterprise environments and the bandwidth-intensive demands of hyperscale data centers[41].

Chapter 2

The Charging Lab

The Charging Lab is one of the key infrastructures of the flagship project **ELECTRO**, developed within the framework of the extended partnership **NODES – Nord Ovest Digitale e Sostenibile**, and funded by the **PNRR – National Recovery and Resilience Plan**. The general objective of ELECTRO is to foster applied research and technology transfer in the field of electric mobility, with a specific focus on:

- smart and wireless charging technologies;
- integration of Vehicle-to-Grid (V2G), Vehicle-to-Vehicle (V2V) and Vehicle-to-Everything (V2X) scenarios;
- communication protocols based on the ISO 15118 standard;
- interactions between charging systems and the electric grid, including distributed generation and storage solutions.

The Charging Lab has been designed as an autonomous facility, separate from traditional laboratories, to enable full-scale testing on vehicles ranging from passenger cars to light vans. Its implementation as an equipped container minimizes the required space and simplifies safety management, while ensuring accessibility for different vehicle classes and charging scenarios.

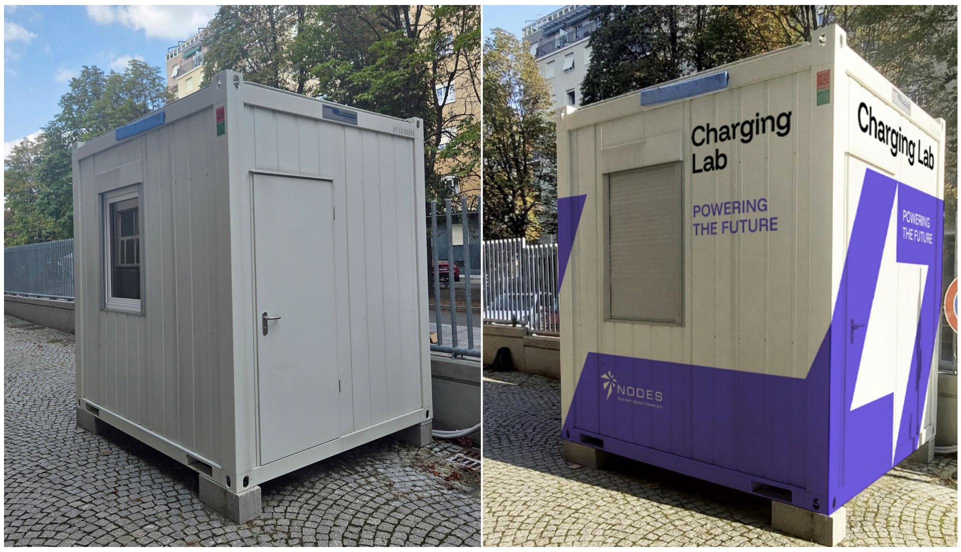


Figure 2.1: *External view of the Charging Lab in container configuration.*

The laboratory is located at the Politecnico di Torino, in **Corso Montevecchio 71**, close to the DENERG laboratories. This positioning enables strong integration between academic research activities and industrial applications, leveraging the expertise and infrastructures available within the Department of Energy.

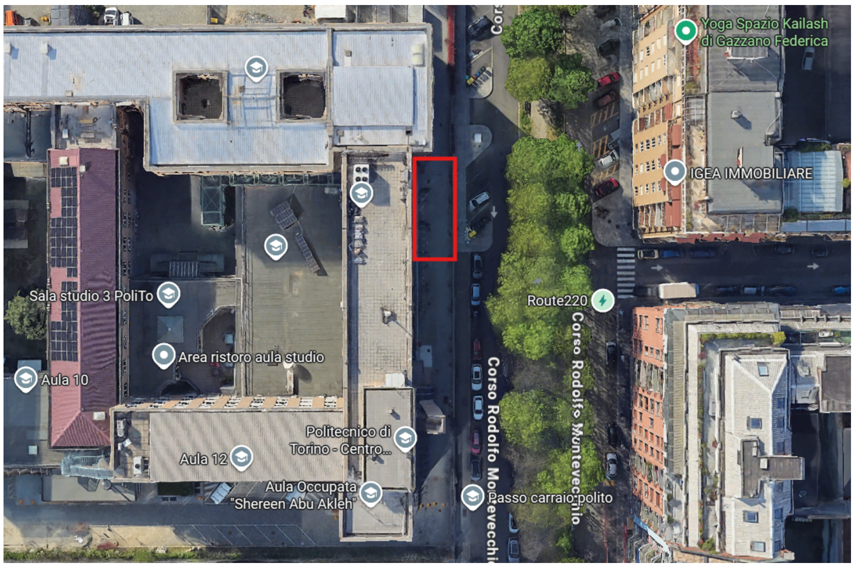

Yoga Spazio Kailash di Gazzano Federica
IGEA IMMOBILIARE
Sala studio 3 PoliTo
Route220
Area ristoro aula studio
Aula 10
Corso Rodolfo Montevecchio
Politecnico di Torino - Centro...
Aula 12
Aula Occupata "Shereen Abu Akleh"
Passo carraio polito

Figure 2.2: *Location of the Charging Lab within the Politecnico di Torino campus, Corso Montevecchio 71*

From a scientific perspective, the project is coordinated by the Politecnico di Torino under the supervision of Prof. Paolo Guglielmi. The main research activities carried out within the Charging Lab include:

- testing of vehicle-to-charger communication protocols;
- interoperability assessment for multi-standard charging infrastructures;
- experimental evaluation of V2G and V2V configurations and their impact on the power grid;
- support for the definition of architectures and services for sustainable electric mobility.

The establishment of the Charging Lab is also embedded within the regional context of Piedmont, a territory that hosts 33.5% of the Italian automotive supply chain (737 active firms), accounting for approximately 35% of the national turnover in the sector (EUR 18.6 billion) and 37% of the total workforce (over 60,000 employees). This strong industrial and technological concentration provides an ideal environment for applied research and for the direct transfer of innovations to the national and regional industrial ecosystem.

The Charging Lab provides two power supplies: a single-phase supply for computers and auxiliary devices, and a three-phase supply required to power the high-power section essential during charging. The diagram below shows the wiring of the three-phase supply:

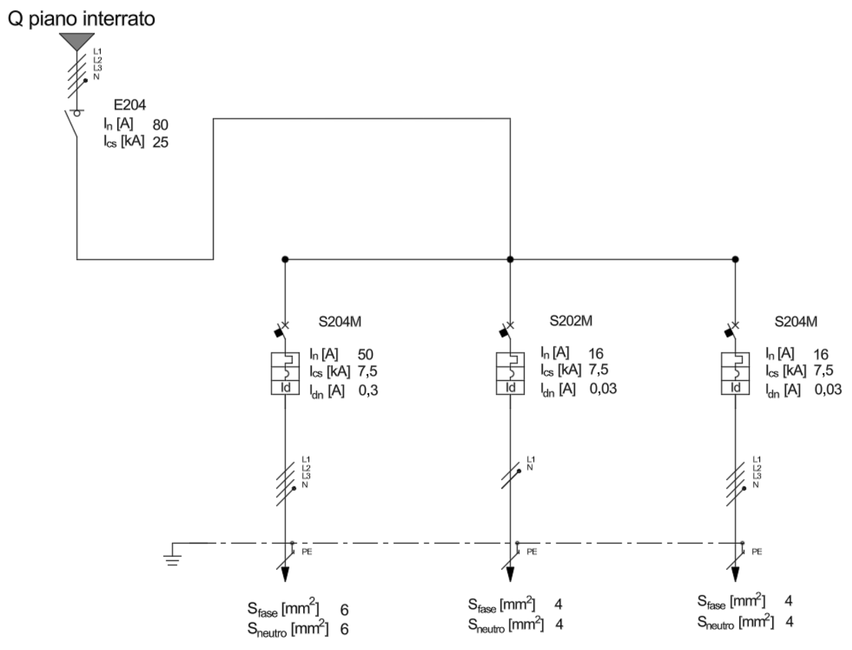


Figure 2.3: *Three phase electrical scheme*

2.1 Laboratory devices

2.1.1 Grid Simulator

Within the Charging Lab, a regenerative grid simulator, model ITECH IT7915P-350-90 from the IT7900P series, is employed. This instrument represents an advanced testing platform, capable of generating and accurately controlling the supply voltage under both single-phase and three-phase conditions. Its main function is to emulate the electrical grid in a laboratory environment, allowing realistic operating scenarios to be reproduced without the need to rely on an external and non-controllable grid.

Figure 2.4: *IETCH Grid simulator*

From a technical perspective, the device can deliver up to 15 kVA, with a nominal voltage of 350 V (line-to-neutral) and a maximum current of 90 A. A distinctive feature of the IT7900P series is its regenerative capability: the simulator not only supplies power but can also absorb energy from the system under test and feed it back into the mains.

Another relevant aspect is the flexibility in waveform programming. The IT7915P can generate not only an ideal sinusoidal waveform at nominal frequency (50/60 Hz), but also arbitrary waveforms, timed sequences, and typical grid disturbances such as sags, swells, and variations in frequency and phase. This makes it an essential tool for verifying the robustness of charging systems under non-ideal grid conditions, which are frequently encountered in practice, especially in scenarios involving large-scale penetration of electric mobility.

Regarding its integration into the laboratory, the simulator is equipped with a local graphical interface via touch screen, as well as numerous remote communication options (USB, LAN, CAN, digital I/O, and standard protocols such as SCPI or Modbus). This makes it easily integrable into automated test benches, where synchronization with other measuring and control instruments is required.

2.1.2 Supply Equipment Communication Controller

The vSECC (Supply Equipment Communication Controller) developed by Vector Informatik GmbH acts as the central communication control unit within an electric vehicle charging station. It represents the logical interface between the electric vehicle (EV), the power electronics of the charging infrastructure (EVSE), and the remote management systems, commonly referred to as the Charging Station Management System (CSMS). Functionally, the vSECC can be regarded as the "brain" of the charging infrastructure, responsible for managing the data–energy interactions that enable a safe, efficient, and flexible charging process.

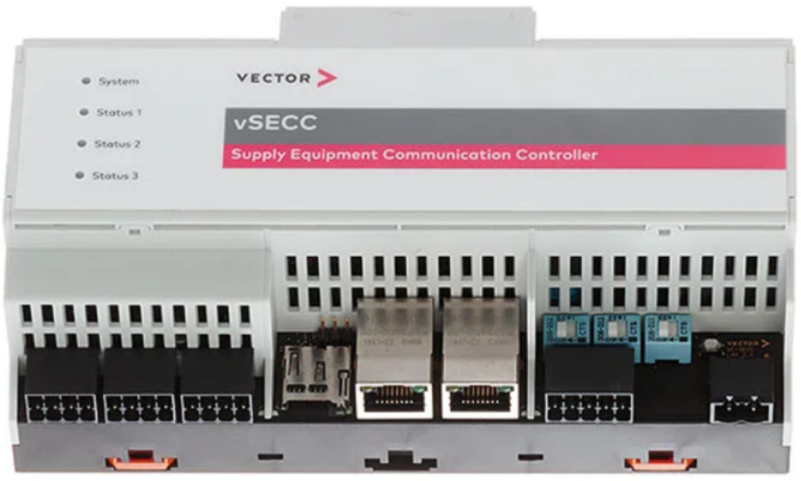


Figure 2.5: *Supply Equipment Communication Controller*

From a communication perspective, the vSECC implements the main standards currently used in the field of e-mobility: ISO 15118 and DIN SPEC 70121 for data exchange between the vehicle and the infrastructure, and OCPP (Open Charge Point Protocol) version 2.0.1 for back-end communication. In this way, the device does not merely establish a dialogue with the vehicle but also integrates into distributed charging networks, enabling centralized management of authentication, authorization, billing, and real-time monitoring.

In terms of hardware architecture, the vSECC is designed for cabinet installation and provides a wide range of interfaces: multiple Ethernet ports for CSMS and subsystem connections, digital and analog inputs/outputs for monitoring safety signals and sensors, as well as dedicated connections for the Control Pilot and Proximity Pin required by the CCS standard. The system is typically powered at 24 V DC and includes intrinsic safety mechanisms, such as interlock and supervision relays, ensuring that power transfer occurs exclusively under safe operating conditions.

A distinctive feature of the vSECC is its scalability. The device is capable of managing two charging points in parallel and can be upgraded to support different connectors and protocols (e.g., CHAdeMO or GB/T), thereby broadening its compatibility with various vehicles and regional markets. Furthermore, the modular firmware, which can also be updated remotely, ensures that the device remains aligned with evolving regulatory frameworks and technological advances in the charging domain.

From an application standpoint, the vSECC is particularly suited for smart charging scenarios, as it integrates dynamic charging profile management. This enables the charging power to be adapted according to external variables such as real-time electricity prices, grid conditions, or renewable energy availability. In addition, advanced diagnostic and logging functionalities are included, providing valuable tools for operational monitoring as well as for experimental research in both academic and industrial contexts.

It should be noted that certain functionalities, particularly those related to AC charging and bidirectional operation (Vehicle-to-Grid), are subject to progressive developments and depend on the software version in use. Nevertheless, thanks to its open architecture and high levels of functional and cybersecurity compliance, the vSECC represents a state-of-the-art solution, well-suited both for deployment in commercial charging infrastructures and for experimental use in research laboratories.

2.1.3 Ethernet/USB/CAN Interface

The VN5610A from Vector Informatik is a compact interface for simultaneous connection to Ethernet and CAN/CAN-FD buses, widely used in the automotive field for testing, monitoring, and simulation purposes. The device, powered and connected to the PC via USB, integrates two Ethernet ports (supporting 10/100/1000BASE and BroadR-Reach) and two CAN/CAN-FD channels, along with a digital I/O channel. One of its main features is the ability to provide accurate timestamps, enabling synchronization of communications across different protocols.

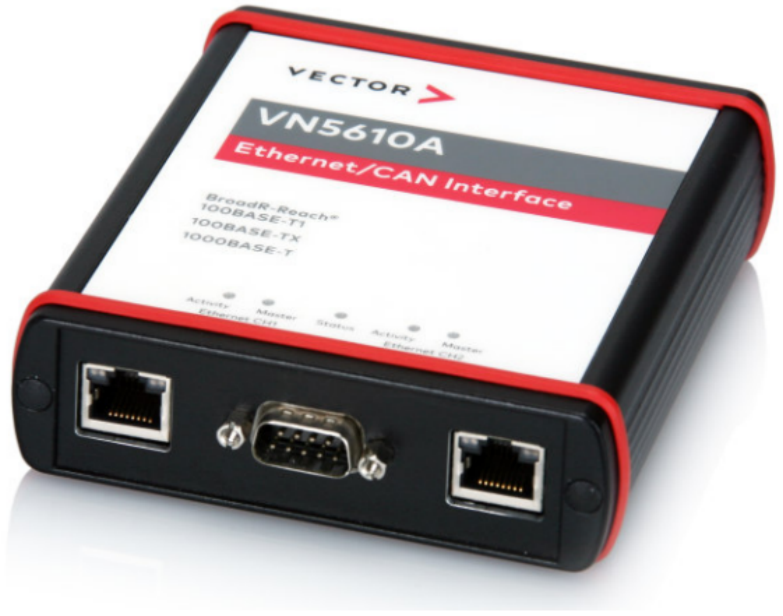


Figure 2.6: *Vector interface*

The main applications include passive network monitoring, remaining bus simulation, and diagnostics over IP (DoIP), typically in combination with Vector software tools such as CANoe and CANalyzer. Thanks to its capability to handle both established and next-generation protocols, the VN5610A represents an essential tool for supporting the transition from CAN-based architectures to automotive Ethernet networks.

2.1.4 Communication Monitoring

The Vector VH5110A is a dedicated tool for passive listening of the communication between electric vehicles and charging infrastructure according to the Combined Charging System (CCS) standard. By connecting to the Control Pilot (CP) line, it allows the acquisition of both low-level signals, such as the PWM modulation defined by IEC 61851, and higher-level communication transmitted via Power Line Communication (PLC) compliant with ISO 15118 and DIN 70121. From a hardware perspective, the device does not interfere with the ongoing transmission, but converts the captured signals into a data stream that can be processed by Vector software, in particular CANoe, for charging protocol analysis. The VH5110A provides an Ethernet interface to the PC and supports a wide input voltage range (9–36 V), which makes it suitable for both laboratory and field applications. It also includes a precise timestamping system, enabling accurate correlation of events and messages, a fundamental requirement for debugging complex charging sessions. An important aspect is related to the connection method: when directly connected to the CP line, it is possible to measure electrical parameters such as voltage, duty cycle, and PWM frequency, while with inductive coupling the measurement is limited to the PLC signal only, without the possibility of acquiring static levels or PWM content.

Figure 2.7: *CCS Listener VH5110A*

To enable this non-intrusive measurement mode, the PREMO MICU300A-S/LF inductive coupler is employed. This device is specifically designed for observing PLC signals present on high-voltage charging cables. Its operating principle relies on the inductive detection of the magnetic field generated by the high-frequency communication signal superimposed on the DC power current. This approach avoids direct contact with the conductors, improving operator safety and preserving cable integrity. The MICU300A-S/LF is rated for nominal currents up to 300 A, with electrical insulation greater than 4.7 kV and an IP65 protection rating, making it suitable for both laboratory use and harsh operating conditions. The coupler exhibits a typical attenuation of about –12 dB in the 100–250 kHz band and –5 dB in the 250–600 kHz band, values that ensure PLC transmission with limited degradation compatible with monitoring activities. The connection to the measuring device is provided via a BNC connector, while the mechanical design allows the accommodation of cables with an outer diameter up to 50 mm.

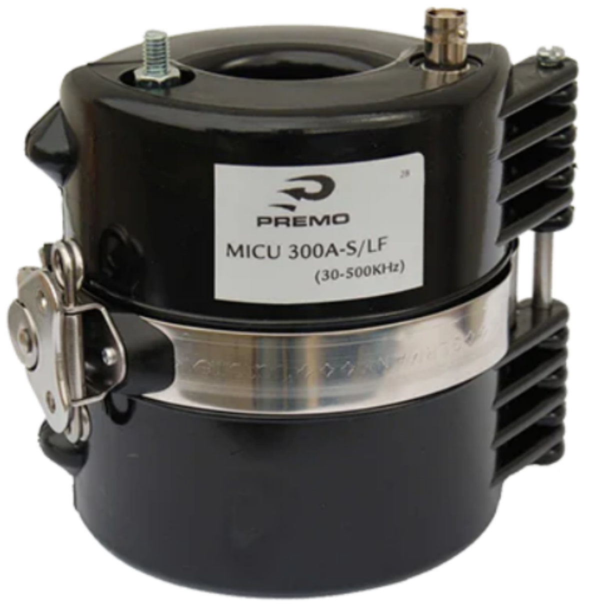


Figure 2.8: *Premo Inductive Coupler*

The combined use of the VH5110A and the MICU300A-S/LF therefore provides a complete system for analyzing CCS charging sessions. The coupler ensures safe inductive access to PLC signals without the need to interrupt or modify the cabling, while the Vector listener decodes and records the data in a synchronized format interpretable by analysis software. This configuration enables interoperability testing, functional validation, and fault diagnosis while maintaining high levels of reliability and safety in laboratory activities.

2.1.5 VT System

The VT System, developed by Vector Informatik, is a modular platform for testing, simulation, and analysis of automotive electronic systems, in particular ECUs (Electronic Control Units) and components related to electric mobility. The system is structured as a modular rack, where each module performs a specific function, while integration between modules and connection to the host PC is achieved through a central module. The modular architecture allows the system to be configured according to laboratory or field testing requirements, combining the measurement of analog, digital, and high-voltage signals with signal simulation and generation.

At the core of the system is the VT6020, which acts as the base module and rack controller. This module is essential to ensure the correct operation of all other connected modules: it manages communication between the PC and the system, distributes power to the installed modules, and provides a central synchronization clock. Thanks to this synchronization, it is possible to correlate acquired and generated signals in real time, a critical aspect in testing activities where the timing of signals can influence the behavior of the ECUs or systems under test. The VT6020 also allows remote management via Ethernet, enabling centralized configuration and monitoring of all modules installed in the rack. Although the VT6020 does not directly generate measurement or output signals, its presence is indispensable for the functional coherence of the entire system.

Figure 2.9: *Real Time module VT6020*

The VT7970 is a digital output module designed to drive logic signals to the ECUs under test or other connected devices. The VT7970's digital channels can be programmed to generate on/off signals, simulate relay activations, or produce complex logic patterns. Integrated channel isolation and configurable voltage levels ensure compatibility with different types of ECUs and automotive systems. The fast switching capability and built-in protection allow repeatable and safe tests, simulating realistic digital control scenarios. This module is particularly useful when emulating input signals to an ECU to evaluate its response under controlled laboratory conditions, allowing researchers to study both functional responses and potential anomalous behaviors or faults.

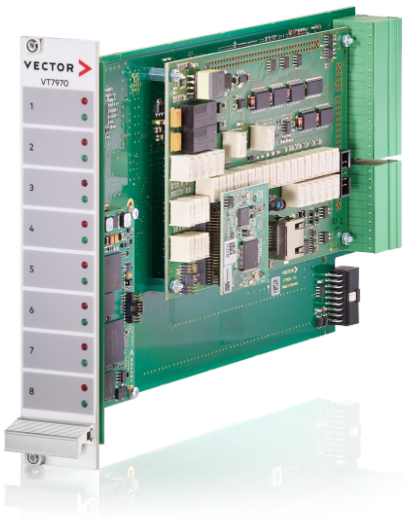


Figure 2.10: *Smart Charging Communication Test Module VT7970*

The VTC8920B is the high-voltage measurement module, capable of measuring signals up to 1000V, making it ideal for applications related to electric mobility and charging systems. This module allows acquisition of both DC and AC voltages, waveform measurements, and monitoring of critical parameters during high-power device testing. The channels are galvanically isolated, ensuring both operator safety and protection of the other rack modules. Integration of the VTC8920B into the VT System allows simultaneous measurement of high-voltage signals and low-power digital and analog signals, synchronized in time via the VT6020. This enables complex experiments and the collection of reliable data for functional analysis, safety verification, or interoperability testing of DC charging systems.

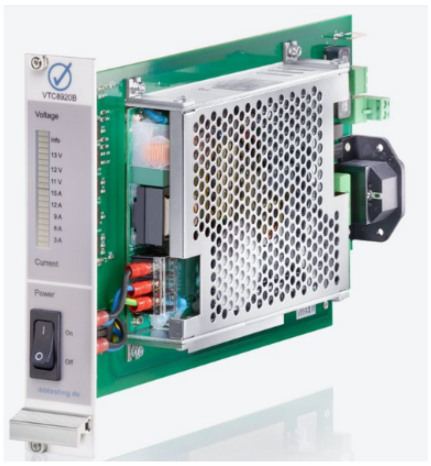


Figure 2.11: *Power Supply Module VTC8920B*

The combined use of these three modules enables a complete and flexible laboratory environment. The VT6020 provides centralized management and signal synchronization; the VT7970 allows realistic digital inputs to stimulate the ECUs; the VTC8920B allows safe acquisition and analysis of high-voltage signals. This configuration is particularly advantageous for validating complex systems, such as bidirectional charging, where multiple signals of different types must be monitored simultaneously to ensure that the ECUs respond correctly to real or simulated scenarios.

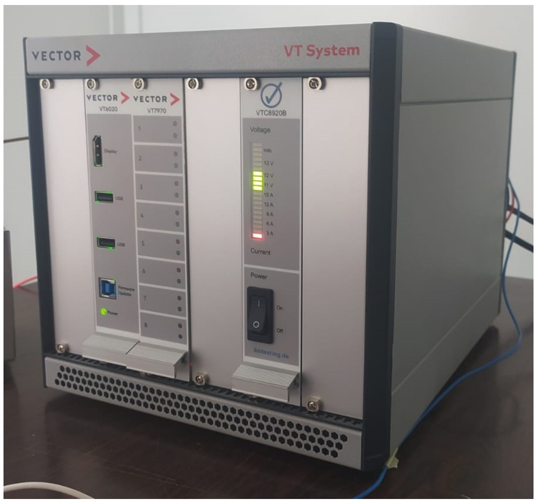


Figure 2.12: *VT System in Smart Charging Lab*

Module integration within the VT System also facilitates test automation. Through connection with software such as CANoe, it is possible to configure simulation scenarios, acquire synchronized data from multiple channels, and analyze complex behaviors in a repeatable manner. Furthermore, the modularity allows the system to be adapted to new testing needs by adding or replacing modules depending on the type of signal to be generated or measured, making the VT System a versatile and indispensable tool in laboratory work on electric vehicles and charging systems.

2.2 AC charging

The first type of charging addressed in this work was alternating current (AC) charging. Although this mode is of lesser interest compared to direct current (DC) charging, a prototype AC charging system was initially developed within the context of this thesis, operating according to the “AC charging via Basic Signalling” standard. The connector used for the prototype is a Mennekes Type 2, defined in the European standard IEC 62196-2, and represents the most widely used interface for AC charging of electric vehicles in Europe. It consists of seven main pins, each with a specific function, as illustrated below:

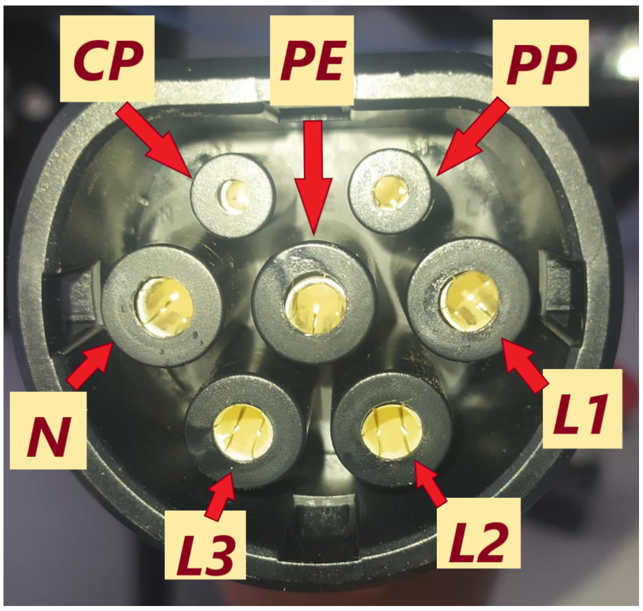


Figure 2.13: *Mennekes plug layout*

- **L1, L2, L3 (Phase Lines):** These pins carry the alternating current from the power source to the vehicle. In a three-phase configuration, the nominal voltage can reach up to 400 V AC, with typical currents up to 32 A for domestic installations and up to 63 A for public charging stations. The three-phase arrangement allows delivering high power without exceeding cable current limits.

- **N (Neutral):** The neutral pin completes the electrical circuit in both single-phase and three-phase configurations. It ensures proper operation of single-phase systems and contributes to overall electrical safety.
- **PE (Protective Earth):** The protective earth pin provides user safety by directing any fault current to the ground. Its presence is mandatory according to IEC 61851-1 and CEI 0-21 standards.
- **CP (Control Pilot):** The CP pin manages communication between the vehicle and the charging station. Through a PWM signal, it allows the detection of vehicle connection, control of the maximum deliverable current, and interruption of charging in case of faults or short circuits. Proper implementation of the CP is essential for safety and dynamic power control during charging.
- **PP (Proximity Pilot):** The PP pin detects the connector's presence and determines the maximum current supported by the charging cable. In combination with the CP, it enables safe adaptation of the charging current to the vehicle and cable capabilities, preventing overheating or overloading.

2.2.1 Cable rating

The Proximity Pilot (PP) is one of the main pins of the Mennekes Type 2 connector and plays a fundamental role in both safety and current management during charging. Its primary function is to detect the presence of the inserted connector and to communicate to the charging station the maximum current that the cable can safely handle. The PP operates through a resistive circuit between the PP pin and the PE pin. According to the IEC 62196-2 standard, the resistance value defines the cable current rating, allowing the Electric Vehicle Supply Equipment (EVSE) to automatically adjust the maximum deliverable current. This prevents overcurrent, overheating, and insulation damage. The typical **test**

voltage applied across the PP–PE circuit is low, usually around 12 V DC, with only a few milliamperes of current, sufficient to measure the resistance without risk. The following table summarizes the typical resistance values used and the corresponding maximum cable current:

PP–PE Resistance	Maximum Cable Current	Test Voltage
2.7 kΩ	13 A	12 V DC
1.3 kΩ	20 A	12 V DC
680 Ω	32 A	12 V DC
150 Ω	63 A	12 V DC

Table 2.1: *PP–PE resistance values, test voltage, and corresponding cable current rating according to IEC 62196-2*

In addition to indicating the cable rating, the PP also ensures that charging cannot start unless the connector is properly inserted. When combined with the **Control Pilot (CP)**, which manages communication and current control, the PP guarantees safe and adjustable charging compatible with different infrastructures.

2.2.2 AC Charging States and Control Pilot (CP) Behavior

In AC charging according to the IEC 61851-1 standard, the **Control Pilot (CP)** line represents the low-level communication interface between the Electric Vehicle Supply Equipment (EVSE) and the Electric Vehicle (EV). Its primary purpose is to manage the connection sequence, detect the presence of the EV, and control the maximum allowable charging current. The CP signal is generated by the EVSE and consists of a **12 V DC** voltage superimposed with a **1 kHz PWM (Pulse Width Modulated)** square wave. This signal is monitored by the vehicle, which in turn modifies the CP line potential through resistive loads to communicate its state. The EVSE applies a 12 V DC potential on the CP pin through a series resistor of 1 kΩ connected to a 12 V source. The EV, on its side, connects various resistors between CP and PE (Protective Earth) to indicate its connection and readiness state. The reference ground for both is the protective earth line (PE).

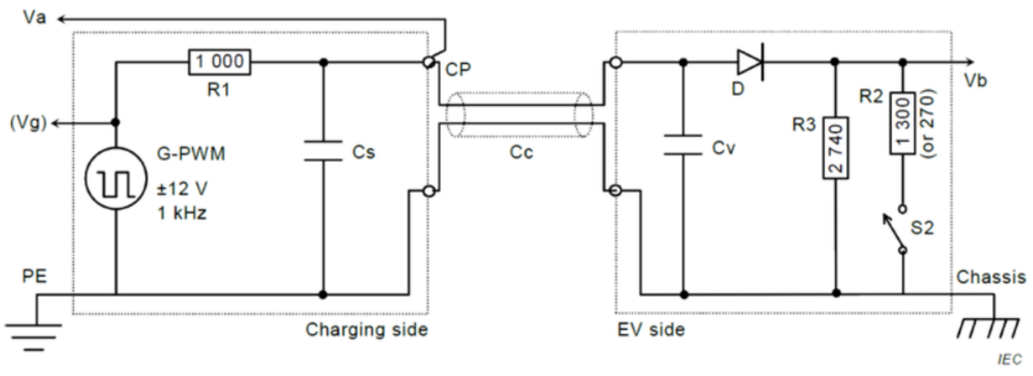


Figure 2.14: *CP circuit for states of charge*

The EVSE measures the resulting voltage at the CP terminal to determine the current state of the connection. The voltage at the CP node (V_{CP}) depends on the voltage divider formed by the EVSE resistor ($R_{EVSE} = 1$ kΩ) and the resistor implemented by the vehicle (R_{EV}). The relation is given by:

$$V_{CP} = 12 \cdot \frac{R_{EV}}{R_{EVSE} + R_{EV}}$$

The different operating states of the charging system are summarized in Table 2.2. Each state corresponds to a specific voltage level on the CP line and determines the next step of the charging process.

State	**CP Voltage**	**Typical Resistance (CP–PE)**
A	12 V	Open circuit
B	9 V	2.74 kΩ
C	6 V	882 Ω
D	3 V	246 Ω + diode
E	0 V	Short circuit

Table 2.2: *AC charging states according to IEC 61851-1.*

State A – No vehicle connected. The CP line remains at 12 V DC, since no load is connected. The EVSE interprets this as an open-circuit condition and does not close the relay that connects the AC mains. The 12 V DC corresponds to the voltage drop across the EVSE internal resistor ($R_{EVSE} = 1$ kΩ) with an open circuit on the EV side.

State B – Vehicle connected but not ready. When the vehicle is plugged in, it introduces a 2.74 kΩ resistor between CP and PE. The voltage at the CP terminal drops from 12 V to approximately 9 V DC, according to the voltage divider relation. This state allows the EVSE to detect the presence of the EV but does not yet enable the power contactors. The PWM signal is still present at 1 kHz, but its duty cycle indicates that energy transfer has not yet started.

State C – Vehicle ready for charging. In this configuration, the EV closes an internal relay that adds a 1.3 kΩ resistor in parallel with the 2.74 kΩ detection resistor, resulting in an equivalent resistance of roughly 882 Ω. Consequently, the CP voltage decreases to about 6 V DC. The EVSE interprets this condition as "vehicle ready to charge" and prepares to close the contactors. The PWM duty cycle, typically between 10% and 85%, encodes the **maximum allowable current** (I_{max}) using the following IEC relation:

$$I_{max} = D_{PWM} \times 0.6 \text{ [A]} \quad \text{for } D_{PWM} < 85\%$$

$$I_{max} = (D_{PWM} - 64) \times 2.5 \text{ [A]} \quad \text{for } D_{PWM} > 85\%$$

For example, a duty cycle of 50% corresponds to a maximum current of 30 A.

State D – Charging with active ventilation. If the EV requires ventilation during charging (e.g., when charging lead-acid batteries), it adds a diode and a smaller resistor (about 246 Ω) in series between CP and PE. The diode introduces a voltage drop of approximately 0.7 V, leading to a CP voltage of around 3 V DC. The EVSE detects this voltage and activates the ventilation system before enabling current delivery.

State E – Fault or disconnection. A voltage close to 0 V indicates a short circuit between CP and PE or a severe fault. The EVSE immediately opens its power contactors, stopping the charging process to ensure safety.

The PWM frequency of the CP signal is fixed at **1 kHz** with a duty cycle resolution of 1%. The EV is required to filter the CP waveform using an RC low-pass network to extract the average voltage for state detection, while the EVSE continuously monitors both the high and low levels of the PWM waveform to detect abnormal conditions such as shorts or open circuits.

The transition between states B, C, and D typically occurs within a few hundred milliseconds, ensuring fast but safe coordination between the EV and the EVSE before current delivery. The circuit is designed so that no dangerous potentials are ever present on the connector before proper handshaking is complete.

2.2.3 Laboratory setup

The alternating current (AC) charging of electric vehicles consists in supplying energy to the on-board charger (OBC), a power converter integrated into the vehicle that manages and controls the charging process. The OBC rectifies the AC voltage into DC and regulates the charging current according to the battery limits and the information exchanged with the charging station. The proposed functional diagram is shown below:

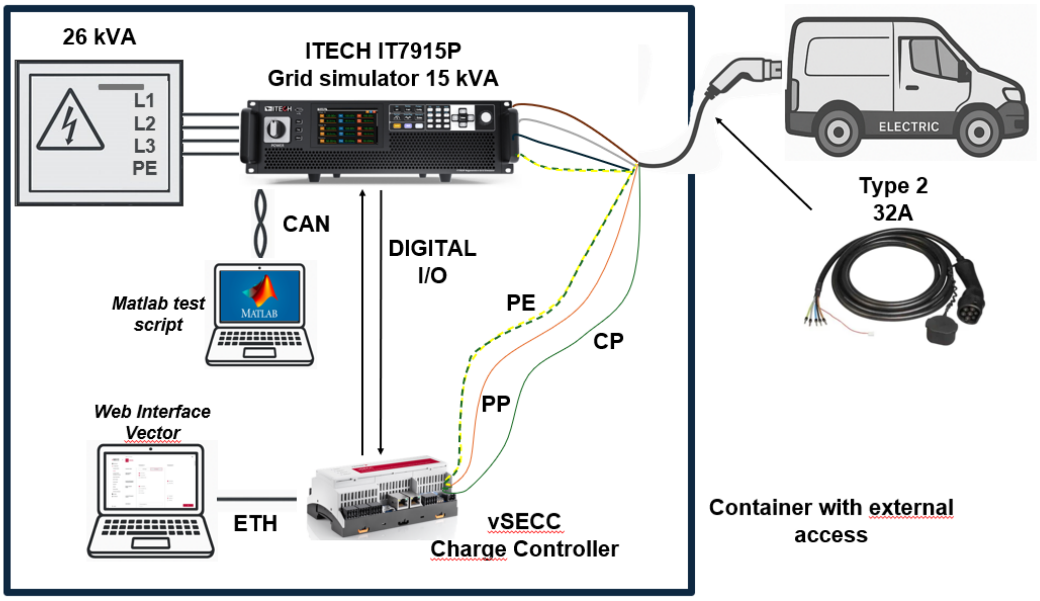


Figure 2.15: *Functional Setup AC charging*

The AC charging of electric vehicles consists in supplying the charging voltage to the on-board charger (OBC), a power converter integrated into the vehicle that manages and controls the charging process, converting the AC power into DC power required to charge the vehicle's battery. The current limitation of the EVSE developed in the laboratory can be configured by accessing the VSECC web interface and setting the desired current value in amperes (A), valid for both single-phase and three-phase operation. The first step involves setting the maximum current deliverable by the charging system. This value is communicated to the EV through the CP state in order to inform it about the maximum current that the EVSE can supply. The configuration requires connecting the VSECC to the web interface developed by **Vector Informatik** via a LAN network. After logging in with the admin credentials and password, the main page is displayed: two connectors are available for control, and for AC charging, a separate license must be purchased. The main configuration parameter for each connector is the communication mode with the Power Electronics, which can be of three types:

- **CAN:** The VSECC sends messages over the CAN network with addresses `0x300 / 0x400` (plus different addresses depending on which of the two connectors is used), providing settings for the

Power Electronics limits and status, as well as information about the current charging state. It also provides the status of inputs/outputs and any temperature sensors.

- **Simulation:** The Power Electronics (PE) is simulated during the various sample configurations in CANoe PRO, so the necessary parameters (such as Voltage Limit, Current Limit, Power Limit) are manually defined by the user.
- **WebSocket:** The most complete interface, where a PECC URL is defined and, through external drivers (e.g., **NI VISA**), messages can be read via JSON format and used to control the Power Electronics through **SCPI** (Standard Control Programmable Instrument) commands.

For the current setup, the choice of communication mode is irrelevant, since the control is managed via digital I/O signals.

The maximum current per phase can be set through the web configuration interface under **Configuration / Vehicle**, or by configuring the variable `ac_charging_setpoint_1/2` using the **Provisioning Tool**. Negative values are not permitted, as the standard does not support discharging. It should be noted that the vehicle is free to draw less power at any time.

During the alternating current (AC) charging tests, it was necessary to properly configure the voltage and current limits of the **ITECH IT7915P** power supply, in order to ensure consistency with the safety parameters defined in the **VSECC (Vehicle Smart Electric Charging Controller)** software and to prevent overcurrent conditions during the charging simulation.

In this case, the VSECC defined a maximum limit of **10 A per phase** in AC mode. This value was established based on the parameters of the vehicle under test and the nominal operating conditions of the charging system. To ensure that the power supply operated within this range, the *Current Limit* settings of the ITECH IT7915P were configured accordingly.

Being a programmable multi-channel power supply, the device allows individual configuration of current limits for each output channel. In the three-phase configuration, the maximum output current for each phase was therefore set to **10 A RMS**, corresponding to the maximum current defined by the VSECC controller. The configuration was carried out either via remote SCPI commands or directly from the front panel through the *"Protection → Current Limit"* menu.

The maximum apparent power deliverable in a balanced three-phase configuration is given by:

$$S_{\text{tot}} = 3\, V_{L\text{–}N}\, I_{\max} = \sqrt{3}\, V_{L\text{–}L}\, I_{\max}$$

where *V(L-N)* is the line-to-neutral voltage (230 V) and *V(L-L)* is the line-to-line voltage (400 V). For the imposed current limit of 10 A per phase, the total apparent power is:

$$S_{\text{tot}} = 3 \cdot 230 \text{ V} \cdot 10 \text{ A} = 6{,}9 \text{ kVA}$$

a value fully compatible with the AC charging profiles considered in the test scenario.

In addition to the current limit, the parameters for overvoltage (**OVP**) and overcurrent (**OCP**) protection were configured with values slightly higher than the nominal operating thresholds, in order to allow some dynamic tolerance during load transitions. Specifically:

- **OVP (Over Voltage Protection)** set to 260 V;
- **OCP (Over Current Protection)** set to 10.5 A per phase;
- **OPP (Over Power Protection)** automatically enabled by the firmware according to the previous limits.

These settings ensure that any transient variations, such as those occurring during contactor activation or the initial handshake phase, do not cause premature interruption of the test procedure.

The control and communication between the VSECC and the power electronics were developed through a connection circuit using digital inputs and outputs. In this case, only basic functionality is required: the vSECC controller manages a set of contactors for each connector via a digital output pin, setting it to HIGH or LOW. A HIGH level corresponds to closing the contactors, while LOW indicates opening them. At this control level, no distinction is made between single-phase or three-phase AC charging.

In addition to the control signal, a feedback pin is monitored, which must replicate the output state within a maximum of 1000 ms. If the feedback does not correctly follow the command, the situation is considered a severe fault. In such a case, the charging process is immediately interrupted, the control pin is set to LOW (opening the contactors), and the CP state is set to F, rendering the entire connector inactive and preventing any further charging attempts. The scheme below resume all the circuits necessary:

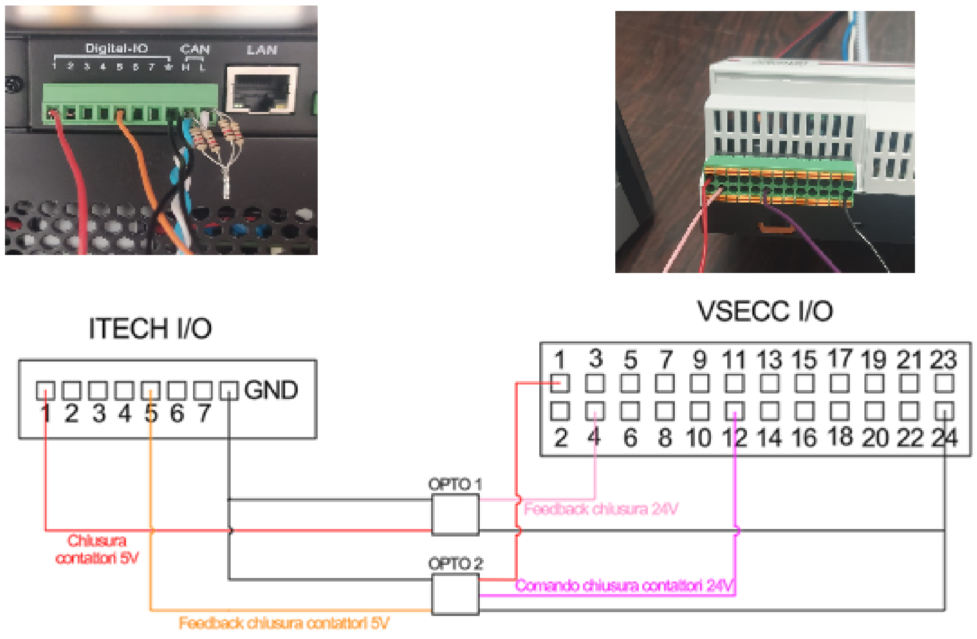


Figure 2.16: *I/O functional circuits scheme*

The connection cannot be made directly and requires an auxiliary circuit because the VSECC provides a digital output with a HIGH level ranging from 18 V to 30 V (typically 24 V), while the digital inputs of the ITECH operate typically at 5 V, with a maximum allowed value of 15 V. Another reason lies in the nature of pin 1 on the ITECH, which is a “sink”-type input — meaning it always provides a 5 V potential but changes its logical state depending on the closure of the circuit between Vcc (5 V) and GND (0 V). Internally, the converter input includes a pull-up resistor, which stabilizes the output when the circuit is open. The last reason concerns the output signal from the ITECH, which provides 5 V for the HIGH state, whereas the VSECC input features a Low-to-High threshold voltage of 6 V (High-to-Low threshold 4.4 V) and typically operates at 24 V. To interface the two devices efficiently and safely, the most suitable solution is to use an optoisolator. This component ensures galvanic isolation, as it consists of an infrared LED and a phototransistor coupled optically. When the LED is forward biased, the emitted light activates the phototransistor on the output side, allowing the transmission of an electrical signal between two electrically isolated circuits, providing protection against overvoltages and noise. For the proposed application, two modules containing four optoisolators each were used — one for the 24V–5V conversion and the other for the 5V–24V conversion:

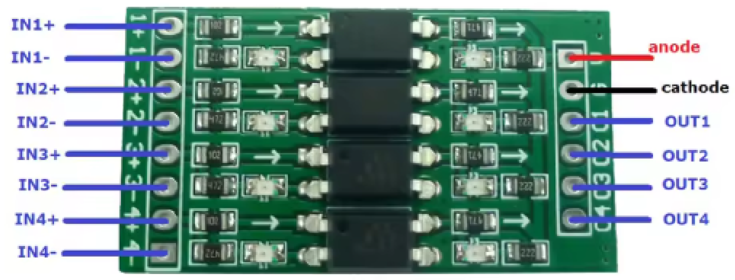


Figure 2.17: *Optocupler 4 channels module*

The individual integrated component is an EL817 module, in the C446 variant for the 5–24 V module and the C352 variant for the 24–5 V module.

Regarding the module responsible for the contactor closing signal on the ITECH side, the voltage is not taken from pin 01; instead, the connection is made directly between pins V (5 V) and G (GND), since the required information is transmitted based on the state of the low-voltage circuit (open circuit / closed circuit).

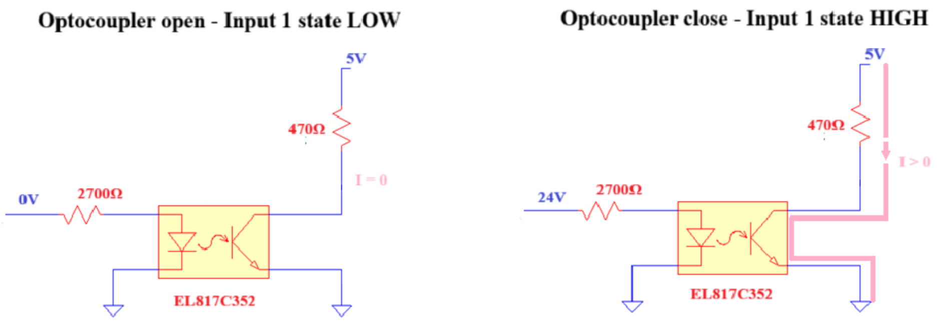
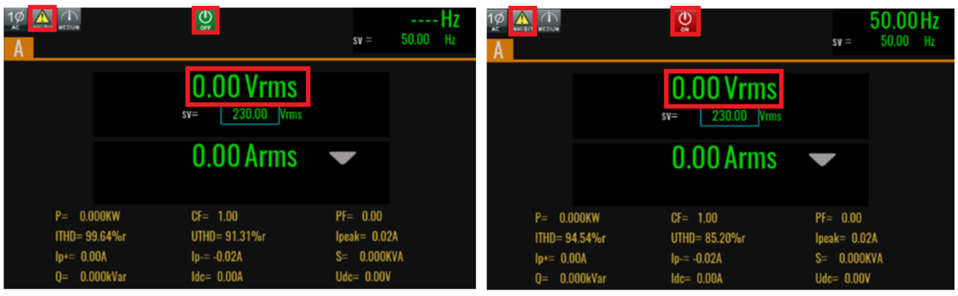

Optocoupler open - Input 1 state LOW
5V
470Ω
I = 0
0V
2700Ω
EL817C352
Optocoupler close - Input 1 state HIGH
5V
470Ω
I > 0
24V
2700Ω
EL817C352

Figure 2.18: *Contactor status sink input level*

On the ITECH firmware, I/O pin 1 is normally assigned to the Inhibit-Latch function in Reverse ON mode: its main purpose is to integrate a normally closed (NC) emergency stop button. If the button is not pressed, the converter remains enabled; if the button is pressed, the circuit opens, and since it operates in Reverse mode, the converter output is disabled.

For safety reasons, if the emergency button is returned to its original state, the circuit closes, but the power output remains locked, and operator intervention is required to restore the system (Shift + Esc keys). For the application under consideration, I/O pin 1 has been assigned to the Inhibit-Living function, meaning that the power output of the power electronics remains disabled until the input circuit closes, allowing power delivery. During this inhibited state, the output may change status (OFF–ON), but it remains deactivated.



```
----Hz
SV =    50.00   Hz
0.00 Vrms
SV=  230.00 Vrms
0.00 Arms
P=    0.000KW        CF=   1.00          PF=   0.00
ITHD= 99.64%r        UTHD= 91.31%r       Ipeak= 0.02A
Ip+=  0.00A          Ip-= -0.02A         S=    0.000KVA
Q=    0.000kVar      Idc=  0.00A         Udc=  0.00V
```

```
50.00Hz
SV =    50.00   Hz
0.00 Vrms
SV=  230.00 Vrms
0.00 Arms
P=    0.000KW        CF=   1.00          PF=   0.00
ITHD= 94.54%r        UTHD= 85.20%r       Ipeak= 0.02A
Ip+=  0.00A          Ip-= -0.02A         S=    0.000KVA
Q=    0.000kVar      Idc=  0.00A         Udc=  0.00V
```

Figure 2.19: *Left side output OFF, right side output blocked ON*

For the 5 V–24 V module, a series connection of two optoisolators was implemented because using a single isolator would result in inverted logic: when the feedback command is HIGH, the phototransistor would conduct, and the voltage at output 01 would drop to 0 V, corresponding to "contactor open" feedback. This issue could be easily solved in the firmware by inverting the logical state using the Reverse ON command.

For safety reasons, however, it was preferred to follow the principle of positive safety: if the power supply were to fail accidentally, the phototransistor would remain open, bringing the optoisolator output to 24 V and allowing the VSECC to keep the CP in state C without interrupting the charging, which could make the charging process unsafe.

To change the logic, it was necessary to act at the hardware level by inserting a second optoisolator in series with the first, using one of the other two available on the same module: if the input of the first branch is LOW, the phototransistor remains open, keeping the output pin at 24 V. This output is then fed into the input of the second optoisolator, which, being HIGH, closes the second optoisolator and brings the output of interest to 0 V (LOW state). When the input switches to HIGH, the states along the circuit change accordingly. The circuit was simulated with LTSpice to determine the resistor to be placed between the first and second optoisolator, since the primary side is not rated to be powered at 24 V.

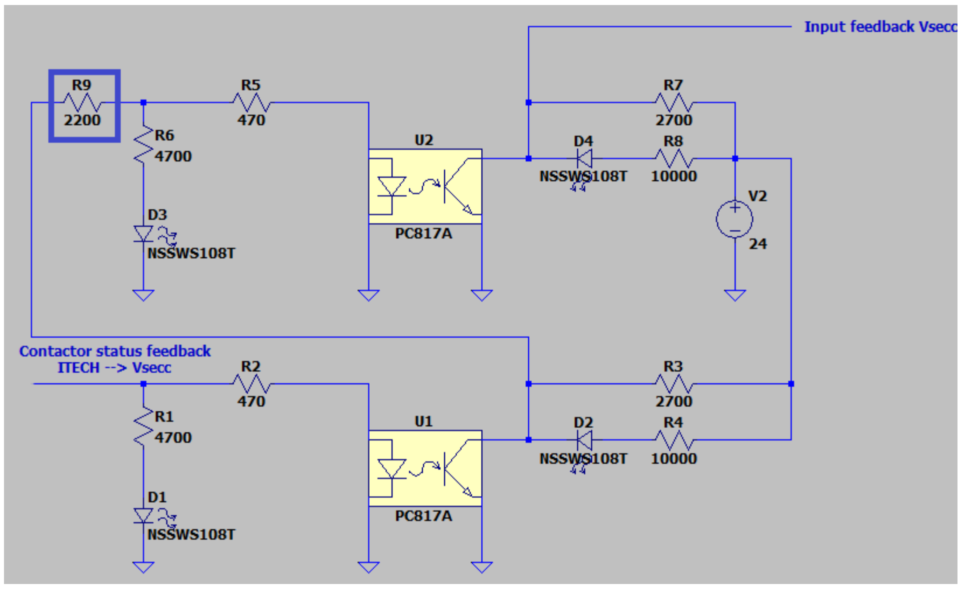


Figure 2.20: *Opto series with limiting resistor*

The simulation without the limiting resistor showed a current of approximately 52 mA between the two devices, which exceeds the maximum forward current (50 mA for the EL817 C446). Through iterative calculations, a commercial resistor of 2200Ω was determined, reducing the current to 8.2 mA, which falls within the typical operating range for the device (5–10 mA). The last issue addressed concerns the maximum current deliverable by an ITECH output, which is less than 1 mA. This value is insufficient to supply the necessary power to drive the optoisolator's LED and allow it to perform its function. Therefore, it is necessary to introduce a circuit that acts as a discrete buffer, preserving the information on the ITECH output state while providing higher current (power), with the nominal voltage remaining limited to 5 V.

A CMOS logic integrated circuit, the MC74HC132N, was used for this purpose; it is a quadruple NAND Schmitt trigger. A Schmitt trigger is a circuit that stabilizes digital signals by introducing two distinct switching thresholds: one for the transition from LOW to HIGH V_{th} and one for the transition from HIGH to LOW V_{tl}. This difference, known as hysteresis, prevents small fluctuations or noise on the input signal from causing undesired changes in the output logic state.

Input A	Input B	Output (Y)
0	0	1
0	1	1
1	0	1
1	1	0

Table 2.3: *Truth table of a 2-input NAND gate*

The gate was powered at 5 V (Vcc), which is also the output level when the gate output is HIGH. Assuming a single NAND gate with one of the two inputs permanently set to HIGH, if the other input is LOW, the output will be HIGH — the inverse of the desired logic. To obtain the correct logical states, a second NAND gate was connected in series. Before constructing the board, a simulation was carried out using LTSpice. The complete circuit is shown below and is presented in greater detail in Annex A of this document:

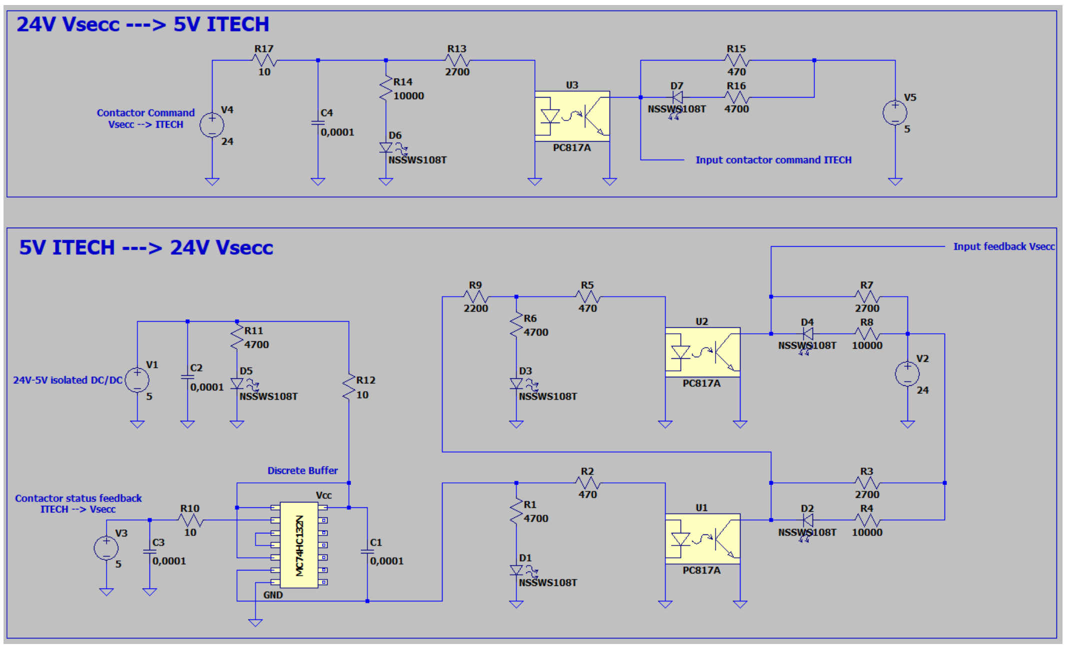


Figure 2.21: *LTspice simulation of the entire board*

During the construction of the electronic board, **100 nF ceramic capacitors** were placed near the power supply and close to the inputs of the main components. These capacitors serve as *bypass/decoupling* elements, stabilizing the supply voltage and reducing high-frequency noise generated by switching transients of the digital devices. Their placement close to the power pins minimizes the effect of parasitic inductance from the PCB traces. Moreover, some connections include **10Ω series resistors**, which act as *current limiters and dampers.* These resistors help attenuate oscillations and current spikes during rapid logic transitions, improving circuit stability and reducing interference or unwanted coupling between traces.

The complete setup was assembled on a perfboard, where the two opto-isolator modules were soldered and mounted using standard-pitch sockets to provide spacing from the board. The MC74HC132N IC was initially soldered directly onto the board, but repeated soldering operations damaged one of the input pins, rendering the IC unusable. To prevent the persistent heat from the soldering iron tip from damaging the next CMOS IC, a 14-pin DIP socket was soldered first, into which the IC was later inserted. The main power supply for the board is provided by pin 1 of connector X306 from the Vsecc, operating at 24V with a maximum current output of 200mA. To power the 5V sides of the two opto-isolator modules and the discrete buffer, the isolated DC/DC converter SCT01F24S15-H2 was used. Isolation is necessary because working with different voltage and current levels requires separating low-voltage and high-voltage circuits, protecting components from overvoltage, short circuits, and interference, thereby ensuring greater safety and system reliability. The following image shows the final result:

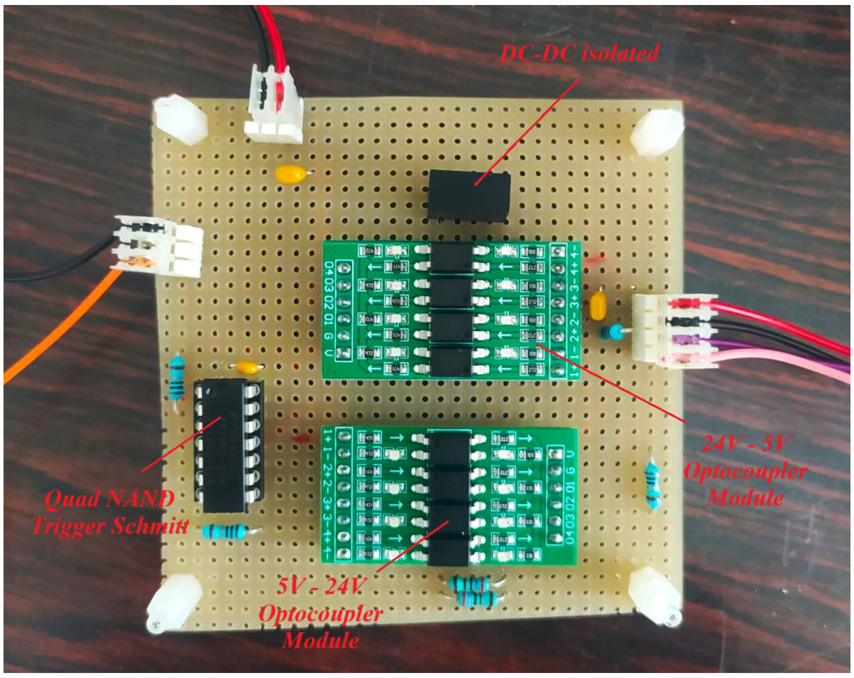

DC-DC isolated
24V - 5V Optocoupler Module
Quad NAND Trigger Schmitt
5V - 24V Optocoupler Module

Figure 2.22: *Final board for I/O pin communication.*

2.3 DC charging

Direct Current (DC) Charging (Mode 4) relies on two fundamental topological architectures. The choice between isolated and non-isolated design directly dictates the Electric Vehicle Supply Equipment's (EVSE) safety profile, efficiency, and compliance with standards like IEC 61851-23.

1. Isolated Architecture (Isolated DC Charging): This topology integrates a **galvanic isolation transformer**, typically within the DC/DC stage (e.g., Dual Active Bridge, DAB). Its primary goal is **safety**, ensuring an electrical separation between the AC grid and the high-voltage DC vehicle circuit. This significantly reduces the risk of electric shock from ground faults on the DC side. While this design involves higher cost, size, and minimal efficiency losses in the magnetic core, it is the **most robust** and common solution for public high-power stations.

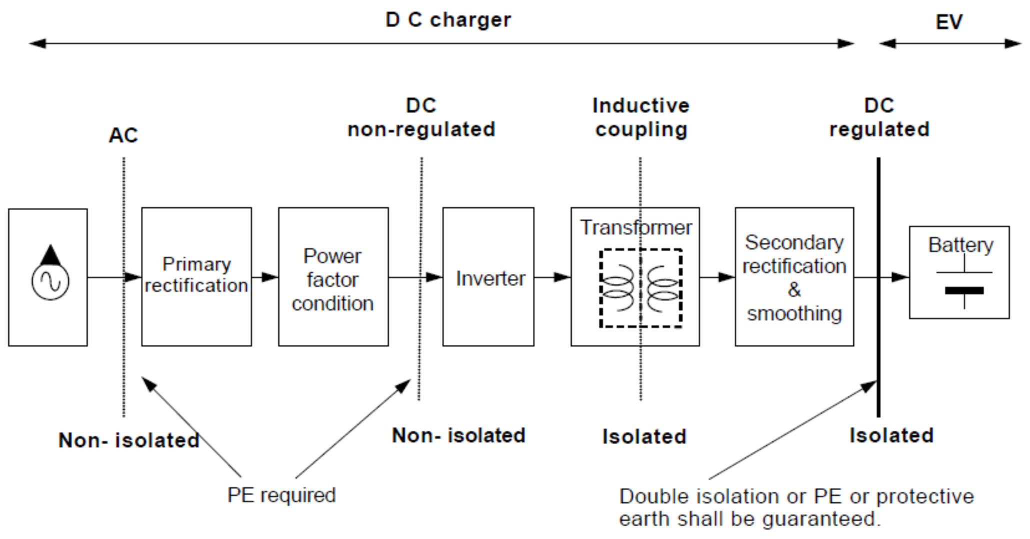


Figure 2.23: *Example of a isolated DC EVSE*

2. Non-Isolated Architecture (Non-Isolated DC Charging): In this setup, the galvanic separation transformer is omitted. The DC bus within the EVSE is **electrically coupled** to the AC input. Key advantages are higher **efficiency** (no transformer losses) and reduced cost/size. However, the lack of intrinsic isolation necessitates an **extremely fast and sophisticated** Ground Fault Detection system to maintain safety standards. This topology is often favoured where efficiency is paramount or for bidirectional systems (V2G) due to the simpler topological design of non-isolated converters.

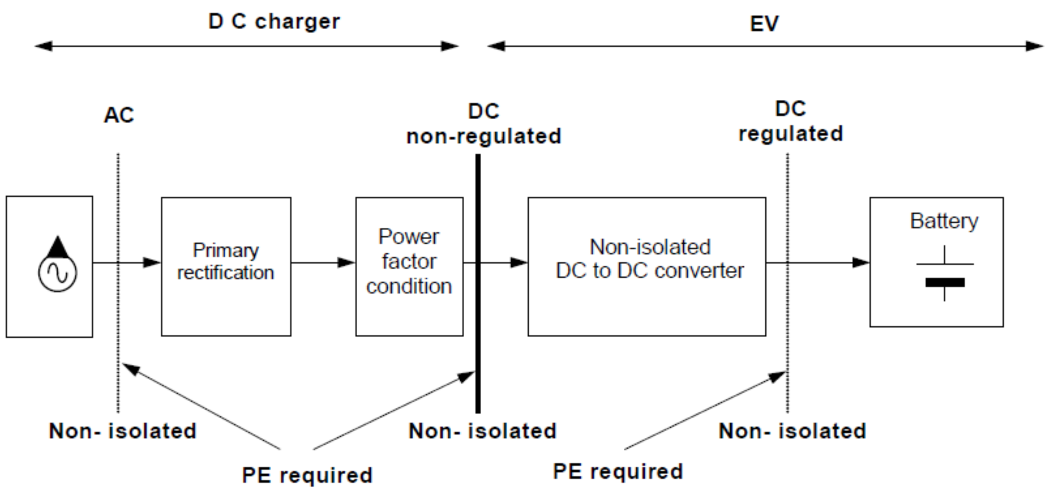


Figure 2.24: *Example of a not isolated DC EVSE*

For the realization of the EVSE, a bidirectional power supply **ITECH IT6642C**, belonging to the IT6600C series, was employed. The device was selected for its architecture with independent and galvanically isolated channels, which allows the system to be configured as a DC source isolated from the mains. This feature is essential to ensure compliance with the safety requirements defined by the standards for DC charging systems (e.g., IEC 61851).

The power supply can operate in both Source and Sink modes, allowing the simulation of both the charging station and the electric vehicle. Each channel can deliver up to 600V and 100A, with a nominal power of 21kW per channel. The channels can be connected either in series or in parallel to achieve higher voltage or current levels. The voltage and current setting resolutions are 1mV and 0,1mA, respectively, allowing precise control of the test parameters. The device also integrates bidirectional energy measurement, overvoltage, overcurrent, and overtemperature protection functions, and supports dynamic programming through arbitrary voltage and current profiles. The IT6642C can be controlled via *LAN*, *USB*, or *GPIB* interfaces and can be fully integrated into automated test environments. These features make it particularly suitable for laboratory testing related to DC charging and CCS2 communication protocol validation, while ensuring operator safety and measurement reliability.

For the DC charging system in the Smart Charging Lab, the CCS Combo connector in its Type 2 version was chosen. The CCS (Combined Charging System) designation indicates that the connector is a combination of an AC charging inlet and a DC charging inlet, allowing for both charging modes through a single port on the vehicle. The Type 2 designation specifically indicates that the AC section of the combined connector is the Mennekes type. This choice was made because the CCS Type 2 connector is the prevalent standard in Europe, while the Type 1 version is predominantly used in the United States and Canada. The following image illustrates the composition of this connector:

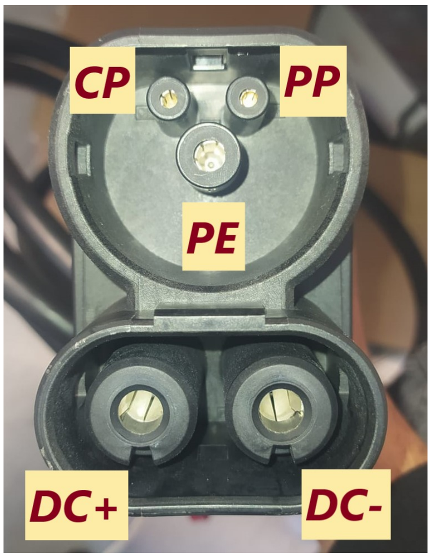


Figure 2.25: *Pins of CCS2 connector*

The CP, PP, and PE pins have the same functionality as the three pins used for AC charging according to the basic signalling scheme. The two lower pins are dedicated to DC power transfer, one for the positive and the other for the negative pole. The connector used is characterized by the absence of the pins (and consequently the wires) for AC charging, since they were unnecessary for the intended application and would have resulted in a cable with a larger cross-section and reduced flexibility.

2.3.1 The ISO/OSI seven-layer model applied to electric vehicle charging communication

Communication between the Electric Vehicle (EV) and the Electric Vehicle Supply Equipment (EVSE) is a key element of the Combined Charging System (CCS). This system integrates not only power transfer but also a complex data exchange between the two devices, necessary for charging parameter negotiation, dynamic energy management, and secure transaction handling. This communication, referred to as High-Level Communication (HLC), is designed according to the Open Systems Interconnection (OSI) model, which forms the conceptual foundation of the ISO 15118-2 and ISO 15118-20 standards. This layered model, composed of seven hierarchical levels, allows the separation of transmission, control, formatting, and data management functions, thus ensuring modularity, interoperability, and scalability of the system.

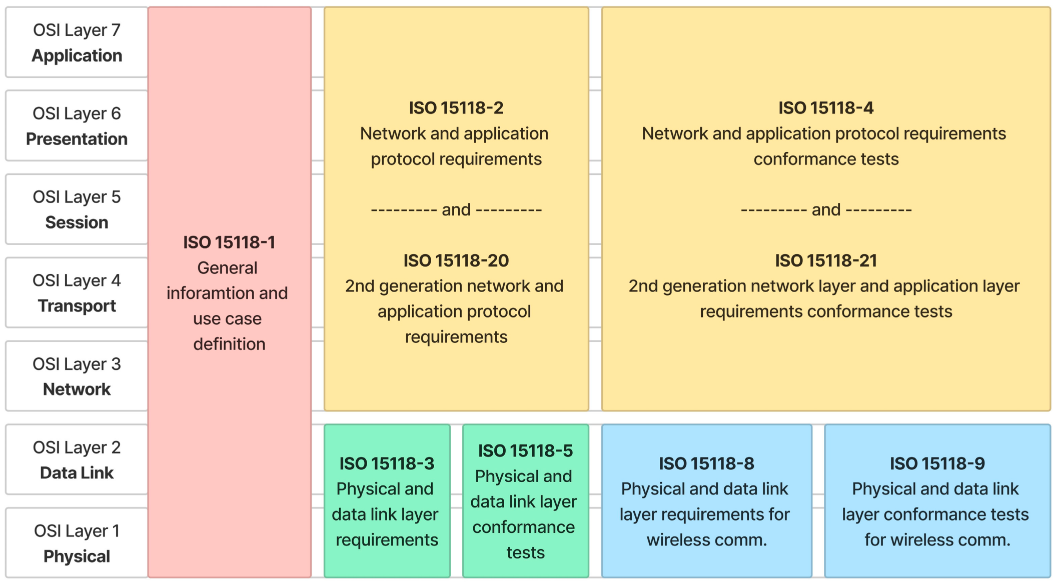

OSI Layer 7 Application
OSI Layer 6 Presentation
OSI Layer 5 Session
OSI Layer 4 Transport
OSI Layer 3 Network
OSI Layer 2 Data Link
OSI Layer 1 Physical
ISO 15118-1 General inforamtion and use case definition
ISO 15118-2 Network and application protocol requirements
--------- and ---------
ISO 15118-20 2nd generation network and application protocol requirements
ISO 15118-4 Network and application protocol requirements conformance tests
--------- and ---------
ISO 15118-21 2nd generation network layer and application layer requirements conformance tests
ISO 15118-3 Physical and data link layer requirements
ISO 15118-5 Physical and data link layer conformance tests
ISO 15118-8 Physical and data link layer requirements for wireless comm.
ISO 15118-9 Physical and data link layer conformance tests for wireless comm.

Figure 2.26: *7 layers ISO/OSI model for EV charging*

Layer 1 – Physical Layer

The physical layer describes the electrical, mechanical, and temporal characteristics of the communication channel. In CCS2, the transmission occurs over the power network using Power Line Communication (PLC) technology in accordance with the HomePlug Green PHY (HPGP) standard.

Communication takes place through the Control Pilot (CP) and Protective Earth (PE) conductors of the CCS connector, where a modulated signal is superimposed in the 2–30 MHz band. The signal is modulated using Orthogonal Frequency Division Multiplexing (OFDM), a multi-carrier technique that divides the frequency band into numerous orthogonal subcarriers, each modulated according to Quadrature Phase Shift Keying (QPSK). This configuration, derived from the HomePlug AV standard, is simplified to reduce hardware complexity and power consumption while maintaining sufficient robustness to interference and impedance variations typical of EV power lines. The transmitted power is limited (on the order of a few dBm) to comply with EMC requirements and to prevent disturbances in adjacent power circuits.

The coupling of the communication signal to the power line is achieved through a PLC coupler, a passive circuit providing galvanic isolation between the power and communication components. The receiver performs OFDM synchronization, channel estimation, and error correction using Forward Error Correction (FEC), thus maintaining link stability even under impulsive noise conditions.

Layer 2 – Data Link Layer

The data link layer defines the frame format and rules for accessing the transmission medium. In ISO 15118, link control is managed by the HPGP MAC layer, which provides error detection and correction, frame integrity (through a 32-bit CRC), and Medium Access Control (CSMA/CA) to prevent collisions.

A crucial function at this level is the Signal Level Attenuation Characterization (SLAC), defined in ISO 15118-3. This procedure, executed before logical channel establishment, allows the EVCC to uniquely associate with the SECC physically connected, avoiding interference with nearby EVSEs. During SLAC, the EVCC transmits a sequence of probe packets of known amplitude; each SECC measures the attenuation level and replies accordingly. The EVCC identifies the valid partner as the one with the minimum attenuation level, indicating a direct physical connection. Therefore, the Data Link Layer ensures reliability, logical isolation, and synchronization between the two devices, forming the foundation of the HLC communication.

Layer 3 – Network Layer

The network layer handles logical addressing and packet routing. In CCS2, the IPv6 protocol is adopted, replacing IPv4 thanks to its extended addressing capabilities and automatic configuration through Stateless Address Autoconfiguration (SLAAC). Each EVCC and SECC possesses a unique IPv6 address, derived from their MAC identifiers and dynamically assigned network prefixes. During connection setup, the Neighbor Discovery Protocol (NDP) is used for node detection and address resolution. This approach simplifies the management of large-scale charging networks, enabling integration with the Internet and backend systems of service providers (CPOs and eMSPs). ISO 15118-20 retains IPv6 but introduces support for multiple simultaneous communications, such as concurrent handling of various services (V2G, diagnostics, firmware update) within a single logical session.

Layer 4 – Transport Layer

The transport layer ensures that data are delivered reliably and in the correct order. ISO 15118 supports TCP, UDP, and TLS depending on the required reliability and security level.

- **TCP (Transmission Control Protocol)** provides a reliable channel with flow control, automatic retransmission, and sequence numbering.
- **UDP (User Datagram Protocol)** is preferred in cases where low latency is critical and occasional packet loss is acceptable (e.g., keep-alive messages).
- **TLS (Transport Layer Security)** introduces end-to-end encryption and authentication using X.509 certificates, which are essential for the Plug & Charge functionality.

In ISO 15118-20, TLS assumes a more central role: charging sessions can be protected through asymmetric key mechanisms, and the vehicle can authenticate automatically without user interaction using certificate chains validated by a Public Key Infrastructure (PKI) dedicated to e-mobility. Therefore, Layer 4 plays a key role in guaranteeing data confidentiality, integrity, and authenticity.

Layer 5 – Session Layer

The session layer is responsible for establishing, managing, and terminating communication sessions. In ISO 15118, this function is handled by the Vehicle-to-Grid Transfer Protocol (V2GTP), which encapsulates the application messages and manages their temporal sequence. V2GTP handles:

- Session initialization after IP connection;
- Assignment of a unique SessionID;
- Synchronization of message flow;
- Timeout and retransmission management;
- Correct session termination in case of error or disconnection.

ISO 15118-20 extends V2GTP by introducing multi-session capability and more granular state management, which is particularly useful for bidirectional charging scenarios (V2G, V2H, V2L) where the power flow may reverse within the same logical session.

Layer 6 – Presentation Layer

The presentation layer defines the data formats and encoding used for transmission. The messages specified by the standard are expressed in XML (eXtensible Markup Language), but for efficiency reasons, they are encoded using EXI (Efficient XML Interchange), a compact binary schema developed by the W3C. EXI encoding reduces message size by up to 90% compared to plain XML, providing higher spectral efficiency and better resilience to noise on power line communication. During transmission, data are further compressed and checked through checksums, ensuring integrity and compatibility across devices.

In ISO 15118-20, the presentation layer gains additional flexibility: different EXI profiles can be applied, and alternative serialization formats (e.g., JSON-EXI or CBOR) are foreseen for future protocol extensions.

Layer 7 – Application Layer

The Application Layer represents the uppermost level of the ISO/OSI architecture and defines the logical framework, data structures, and message exchanges between the Electric Vehicle Communication Controller (EVCC) and the Supply Equipment Communication Controller (SECC). Within the ISO 15118 standard, this layer governs the actual charging logic, organizing it into a series of standardized Request/Response pairs, each corresponding to a particular operational stage of the charging session. All Application Layer messages are encoded using the EXI (Efficient XML Interchange) format and transmitted through the lower layers that were previously established during the physical and data link setup.

The Application Layer communication is conceptually divided into several functional sequences: *communication setup*, *identification and authorization*, *target setting and charge scheduling*, *charging control*, and *session termination*. Each sequence contains a set of specific message pairs defined by ISO 15118-2, with ISO 15118-20 introducing extensions to support bidirectional energy transfer (V2G, V2H, V2L).

Communication setup sequence

SupportedAppProtocolReq/Res: The initial message exchange ensures that both devices operate on compatible versions of the protocol. The EVCC sends a *SupportedAppProtocolReq* message listing all supported ISO 15118 versions along with their priorities. The SECC replies with *SupportedAppProtocolRes*, selecting one of the proposed versions. If no common version exists, the communication session cannot continue, and the ISO 15118 link is terminated.

SessionSetupReq/Res: After confirming protocol compatibility, the EVCC initiates the Vehicle-to-Grid (V2G) communication by sending a *SessionSetupReq*. The SECC responds with a *SessionSetupRes*, which includes a *ResponseCode* to indicate whether the session has been successfully established or resumed from a previous state. A unique *SessionID* is generated and maintained throughout the entire communication.

Identification, authentication and authorization sequence

AuthorizationSetupReq/Res: This marks the start of the authentication phase. The EVCC sends an *AuthorizationSetupReq* message to prompt the SECC to advertise its supported authorization methods. The SECC replies with *AuthorizationSetupRes*, specifying whether Plug & Charge (PnC) or External Identification Means (EIM) are supported, and if certificate installation is available.

AuthorizationReq/Res: The EVCC requests permission to start charging through an *AuthorizationReq* message containing its identification and cryptographic credentials. In Plug & Charge mode, the SECC validates the certificate chain and signature using the e-mobility Public Key Infrastructure (PKI). In EIM mode, authorization depends on an external system (such as RFID or a mobile app). The SECC responds with *AuthorizationRes*, indicating whether authorization succeeded and reporting possible error codes.

ServiceDiscoveryReq/Res: After successful authorization, the EVCC queries the SECC for available services using *ServiceDiscoveryReq*. The SECC answers with *ServiceDiscoveryRes*, listing supported functionalities such as AC charging, DC charging, V2G operation, or certificate updates. Optionally, the EVCC can narrow the query to specific services by providing service identifiers.

ServiceDetailReq/Res: For each selected service, the EVCC can request detailed information through *ServiceDetailReq*. The SECC responds with *ServiceDetailRes*, containing configuration parameters and available options.

ServiceSelectionReq/Res: Once the EVCC has gathered all necessary details, it communicates its choice via *ServiceSelectionReq*. The SECC confirms service availability and readiness with a *ServiceSelectionRes* message.

Target setting and charge scheduling

DC_ChargeParameterDiscoveryReq/Res(3a/3b): At this stage, both controllers negotiate the electrical parameters of the charging process. The EVCC sends a *DC_ChargeParameterDiscoveryReq* message including vehicle-side data such as maximum voltage, current limits, and battery capacity. The SECC replies with *DC_ChargeParameterDiscoveryRes*, detailing the available grid limits and constraints. This negotiation ensures the compatibility of both systems and defines the safe operating range for power transfer.

ScheduleExchangeReq/Res(3.1a/3.1b): Once basic parameters are defined, the energy exchange can be optimized. The *ScheduleExchangeReq* message communicates the EVCC's preferred charging profile, including energy target and departure time. The SECC responds with *ScheduleExchangeRes*, suggesting a schedule that balances user needs and grid management. This process enables advanced energy scheduling and demand-response functionality.

DC_CableCheckReq/Res(4a/4b): Before starting DC power delivery, safety checks are performed. The EVCC sends a *DC_CableCheckReq* to verify cable integrity and locking status. The SECC replies with *DC_CableCheckRes*, confirming whether the system is safe to proceed.

DC_PreChargeReq/Res(5a/5b): Prior to current flow, the pre-charge phase aligns the EVSE output voltage with the EV battery voltage to prevent inrush currents. The EVCC reports its actual voltage via *DC_PreChargeReq*, and the SECC adjusts its output accordingly. When voltages are synchronized, *DC_PreChargeRes* confirms readiness.

PowerDeliveryReq/Res(6a-8a/6b-8b): This message pair transitions the system from setup to active charging. *PowerDeliveryReq* instructs the EVSE to energize its DC output, while *PowerDeliveryRes* acknowledges activation and reports system status or possible faults.

Charging control and re-scheduling

DC_ChargeLoopReq/Res(7a/7b): During active charging, the EVCC continuously monitors and controls the process by sending *DC_ChargeLoopReq* messages specifying desired voltage and current. The SECC responds with *DC_ChargeLoopRes*, providing measured feedback and system diagnostics. This feedback loop allows real-time adjustment of the charging profile and supports intermediate scheduling updates.

End of charging process

DC_WeldingDetectionReq/Res(9a/9b): At the end of the session, a welding detection test is performed as defined in IEC 61851-23 to ensure contactors have not fused. The EVCC initiates this by sending *DC_WeldingDetectionReq*, and the SECC responds with *DC_WeldingDetectionRes*, reporting contactor and open-circuit voltage status.

SessionStopReq/Res(10a/10b): Finally, the EVCC sends a *SessionStopReq* to terminate communication and end power transfer. The SECC confirms with *SessionStopRes*, releasing all logical connections and clearing the session context linked to the *SessionID*. All messages present in the Application Layer are graphed in the following image:

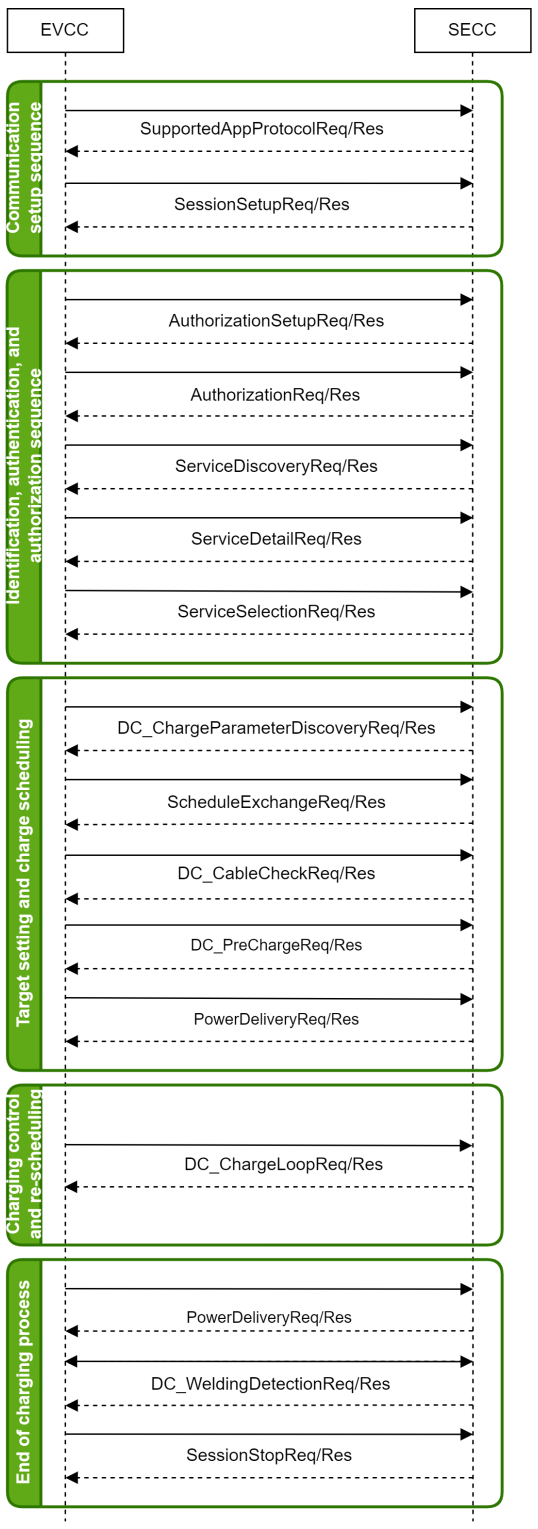


Figure 2.27: *Messages exchanged between EV and EVSE*

Compared to ISO 15118-2, the ISO 15118-20 standard introduces a significant architectural evolution in the Application Layer and in the overall communication model. While ISO 15118-2 was limited to unidirectional power transfer (from the EVSE to the EV) and focused mainly on conductive charging scenarios, ISO 15118-20 extends the communication framework to support **bidirectional power transfer** (BPT), enabling Vehicle-to-Grid (V2G), Vehicle-to-Home (V2H), and Vehicle-to-Load (V2L) functionalities. This enhancement allows the Electric Vehicle (EV) to act not only as an energy consumer but also as a distributed energy resource capable of injecting power back into the grid or supplying external loads. A major conceptual improvement concerns the **Service-based architecture**, which replaces the monolithic message structure of ISO 15118-2 with a modular design. Each service, such as charging, metering, or energy management, is defined as an independent communication unit with its own lifecycle, service identifiers, and parameter sets. This modularity provides scalability and interoperability between different types of energy transfer, including AC, DC, and wireless power transfer (WPT). Moreover, ISO 15118-20 refines the **security framework** by introducing updated cryptographic algorithms, extended certificate handling procedures, and a unified approach to authorization for both Plug & Charge (PnC) and External Identification Means (EIM) modes. The standard also introduces mechanisms for secure firmware and software updates over the same V2G communication link. Another important extension lies in the **data modeling and flexibility** of message exchange. Messages are still encoded using Efficient XML Interchange (EXI), but the schema definitions (XSDs) have been revised to include new data types, optional fields, and extensibility mechanisms that facilitate future compatibility with evolving energy markets and smart grid infrastructures. Finally, ISO 15118-20 introduces advanced **energy management and scheduling features**, allowing negotiation of dynamic charging profiles based on grid conditions, pricing signals, or user-defined constraints. These improvements make the Application Layer not just a vehicle-to-charger communication interface, but a foundational element of the smart charging ecosystem, supporting the integration of electric mobility into the broader context of distributed energy systems.

2.3.2 DC Laboratory setup

Compared to the AC charging setup, the DC charging setup is much simpler since the control electrical circuit between the VSECC and the Power Electronics is not present. A functional diagram of the proposed setup is illustrated below:

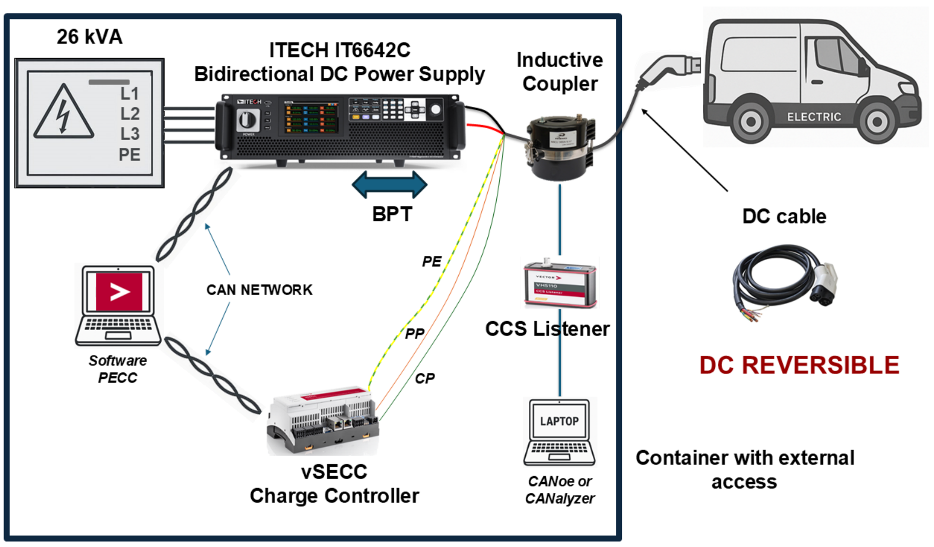


Figure 2.28: *DC laboratory functional setup*

The DC power supply employed in the experimental setup is an ITECH IT6642C model. The positive and negative DC terminals (DC+ and DC–) are connected through a CCS Type 2 charging cable. This cable is routed through an inductive coupler, which enables protocol sniffing, i.e., the non-intrusive monitoring of communication signals exchanged between the Electric Vehicle (EV) and the Electric Vehicle Supply Equipment (EVSE). Such monitoring is achieved by means of a Vector interface, referred to as a *listener*, which captures and reconstructs the transmitted data frames and forwards them to a host PC via USB connection. The data are then visualized and analyzed using Vector's *CANalyzer* tool, specifically designed for network analysis and diagnostic purposes. Among the auxiliary conductors within the CCS2 cable, the Control Pilot (CP), Proximity Pilot (PP), and Protective Earth (PE) lines are connected to the Vector VSECC unit. These conductors implement the low-level signaling defined by IEC 61851 and serve as the physical layer for the Power Line Communication (PLC) channel. The VSECC acts as an interface between the communication link with the vehicle and the power conversion stage of the EVSE. During operation, it exchanges data such as voltage and current limits, charging state, and diagnostic information through a proprietary CAN-based protocol developed by Vector, known as *PEP-CAN*. Every 250 ms, these data are read and processed by a CAPL script, running within the Vector CANoe environment. CAPL (*Communication Access Programming Language*) is a C-like programming language designed for simulation and control tasks in automotive networks. In this setup, the CAPL script functions as the Power Electronics Communication Controller (PECC), transmitting *CANopen* commands to the Power Electronics module. These commands enable or disable the DC output and dynamically adjust the output current and power according to the charging state, thus coordinating the overall charging process between communication and power domains.

2.3.3 Dc Charge sequence

In the latest generation of electric vehicle charging systems, **ISO 15118-20** and **IEC 61851-23** are tightly integrated to enable both digital communication and safe power transfer between the electric vehicle (EV) and the DC charging station. While ISO 15118-20 defines the *high-level communication layer*—managing session setup, energy requests, and vehicle identification—IEC 61851-23 ensures *electrical safety and operational integrity*, governing parameters such as insulation resistance, pre-charge limits, and current response times. The coordination between these two standards guarantees that the charging process evolves from simple plug-in detection to controlled high-power energy transfer in a safe, traceable, and automated manner.

Below is the chronological sequence of events that typically occurs during a DC charging session, after the visual sequence all the parameters and temporal flags are explained:

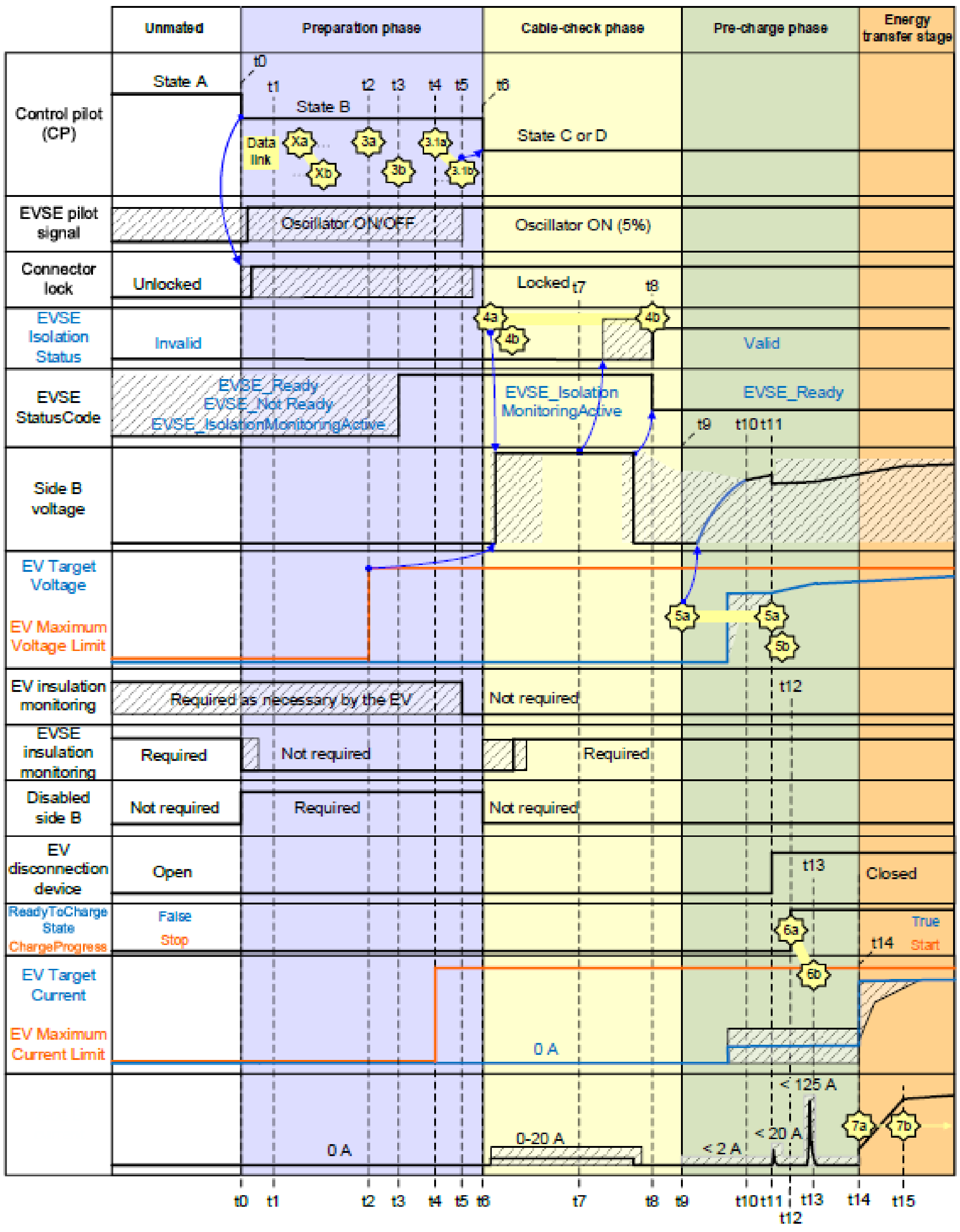


Figure 2.29: *Charging session of a normal startup*

t_0 The present DC voltage at side B is verified to be $\leq 60\,\mathrm{V}$. The EV connector is inserted into the vehicle inlet, changing the Control Pilot (CP) state from A to Bx. Once the CP state reaches P and the Proximity Pilot (PP) is detected, the EV locks the connector. The EVSE disables its IMD on side B and opens the disconnection device, if present. The EV may initiate a data link as defined in

IEC 61851-24.

$\mathbf{t}_0 \rightarrow \mathbf{t}_5$ The EVSE does not trigger an error shutdown due to present voltages at side B, including negative values, which may arise from the vehicle's unmated state behavior.

$\mathbf{t}_0 \rightarrow \mathbf{t}_6$ If the EV fails to lock the connector, it reports an error to the EVSE. Side B of the EVSE retains equivalent electrical behavior (e.g., impedance, discharge rate) as specified.

$\mathbf{t}_0 \rightarrow \mathbf{t}_{13}$ Overvoltage protection at side B is not yet active.

$\mathbf{t}_1$ The EVSE turns on the CP oscillator when ready for energy transfer and maintains a 5% duty cycle unless other conditions require adjustments.

$\mathbf{t}_1 \rightarrow \mathbf{t}_2$ The EV establishes a data link after verifying the CP oscillator is active. Digital communication begins with negotiation of the application layer protocol and exchange of session setup, services, authentication, and value-added services.

$\mathbf{t}_1 \rightarrow \mathbf{t}_{13}$ If the EV or EVSE needs to terminate the session, appropriate error or shutdown sequences are triggered, with messages carrying "ResponseCode = FAILED".

$\mathbf{t}_2$ The EV sends maximum voltage and current limits in the DC_Charge ParameterDiscoveryReq message.

$\mathbf{t}_2 \rightarrow \mathbf{t}_3$ The EVSE performs a compatibility check.

$\mathbf{t}_3$ The EVSE responds with its maximum and minimum voltage and current limits, and the results of the compatibility check. "ResponseCode = OK" indicates success, "FAILED_WrongChargeParameter" triggers EVSE-initiated error shutdown.

$\mathbf{t}_4$ The EV sends parameters according to the selected control mode in the first ScheduleExchangeReq message.

$\mathbf{t}_4 \rightarrow \mathbf{t}_5$ The EVSE verifies that the present voltage at side B is $\leq 60\,\mathrm{V}$ and responds with ScheduleExchangeRes messages indicating ongoing or finished processing. Errors in connector locking or parameter incompatibility are handled with cyclic "Ongoing" messages to allow correction, or trigger error shutdowns if unresolved.

$\mathbf{t}_5$ If the voltage at side B remains $\leq 60\,\mathrm{V}$, the EVSE finalizes schedule exchange, responding “Finished” and “ResponseCode = OK” if successful, otherwise triggering an EVSE-initiated shutdown.

$\mathbf{t}_5 \rightarrow \mathbf{t}_6$ The EV ensures the connector is locked before closing SV3. Failure triggers an EV-initiated shutdown. The EV disables its own IMD if present.

$\mathbf{t}_5 \rightarrow$ The EVSE triggers shutdown if the present voltage at side B is $\leq 60\,\mathrm{V}$ for 400 ms or more.

$\mathbf{t}_6$ After connector locking and IMD disabling, the EV changes CP state from B to C/D by closing SV2. The EVSE enables side B, removing limitations of the previously disabled side. Common-mode and differential-mode disturbances from the EV should be cleared before this transition.

$\mathbf{t}_6 \rightarrow \mathbf{t}_7$ The EV requests the cable check phase using DC_CableCheckReq before pre-charge. The EVSE confirms CE state C and that voltage at side B is below 60 V before responding with DC_CableCheckRes.

$\mathbf{t}_6 \rightarrow \mathbf{t}_8$ The EVSE verifies its IMD and insulation on side B, continuously sending DC_CableCheckRes messages with “EVSEProcessing = Ongoing” and “ResponseCode = OK”. Additional functions such as welding detection may run concurrently.

$\mathbf{t}_7$ The EVSE determines insulation resistance $\geq 100\,\mathrm{k}\Omega$.

$\mathbf{t}_8$ After cable check, the EVSE sends DC_CableCheckRes “Finished” and “ResponseCode = OK” if insulation is valid; otherwise, error shutdown is triggered.

$\mathbf{t}_8 \rightarrow \mathbf{t}_9$ The EVSE may maintain present voltage at side B.

$\mathbf{t}_9$ The EV sends DC_PreChargeReq to start pre-charge, setting “EVTargetVoltage” based on the battery system voltage.

$\mathbf{t}_9 \rightarrow \mathbf{t}_{10}$ EV and EVSE perform pre-charge, with the EVSE controlling voltage to match the target and sending corresponding DC_PreChargeRes messages.

$\mathbf{t}_{10}$ Side B voltage reaches the EV target voltage within defined tolerances.

$\mathbf{t}_{10} \rightarrow \mathbf{t}_{11}$ The EV may adjust target voltage via cyclic DC_PreChargeReq/Res messages to compensate for deviations.

$\mathbf{t}_{11}$ The EV closes its disconnection device only when voltage difference between side B and battery is $\leq 20\,\mathrm{V}$, signaling pre-charge completion with DC_PreChargeReq "Finished".

$\mathbf{t}_{11} \rightarrow \mathbf{t}_{12}$ The EVSE disables pre-charge current limitation and sends DC_Pre ChargeRes acknowledgment.

$\mathbf{t}_{12}$ The EV requests energy transfer by sending PowerDeliveryReq "Start" after closing its disconnection device and receiving pre-charge confirmation.

$\mathbf{t}_{12} \rightarrow \mathbf{t}_{13}$ The EVSE responds with PowerDeliveryRes "OK" and prepares side B for energy transfer, enabling overvoltage protection.

$\mathbf{t}_{13}$ The EVSE confirms readiness for energy transfer and starts overvoltage protection.

$\mathbf{t}_{14}$ The EV sends the first DC_ChargeLoopReq to initiate active energy transfer, specifying parameters per control mode.

$\mathbf{t}_{14} \rightarrow \mathbf{t}_{15}$ The EVSE adjusts side B according to the maximum parameters communicated by the EV and responds with DC_ChargeLoopRes, reporting present current and voltage, min/max values, and status. Insulation monitoring continues throughout.

$\mathbf{t}_{15}$ Current at side B reaches EVMaximumChargeCurrent or EVTargetCurrent within time delay T_d.

$\mathbf{t}_{15} \rightarrow$ The EV continuously adapts current, voltage, and power targets according to its energy transfer strategy via cyclic DC_ChargeLoopReq messages.

The international standard describes also the shutdown and pause procedures of a DC charging session, integrating ISO 15118-20 messages with the operational principles of IEC 61851-23:

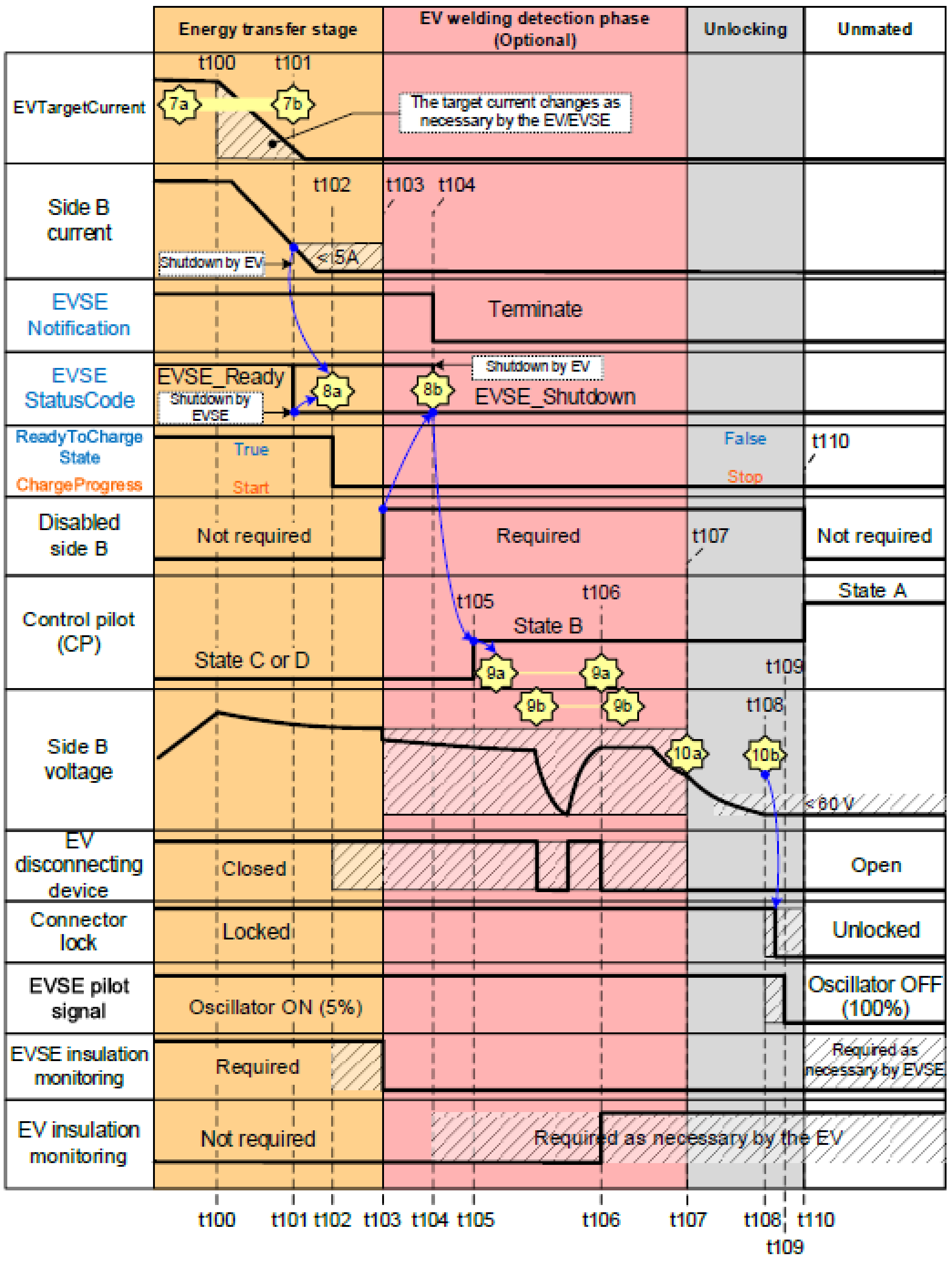


Figure 2.30: *Charging session of a normal shutdown*

t$_{100}$ Cyclic DC_ChargeLoopReq/Res messages occur between the EV (DC_ChargeLoopReq <7a>) and the EVSE (DC_ChargeLoopRes <7b>).

t$_{100}$ If the EV intends to pause or stop the session for a non-critical reason, it requests the present current at side B to be 0 A DC via DC_ChargeLoopReq <7a>messages.

$\mathbf{t}_{100} \rightarrow \mathbf{t}_{101}$ If the EVSE initiates a non-critical stop or pause (for example, user pressed stop), it ramps down side B current to < 5 A DC and updates EVSEMaximumChargeCurrent and EVSEMaximumChargePower in DC_ChargeLoopRes <7b>.

$\mathbf{t}_{101}$ The present current at side B reaches < 5 A DC.

$\mathbf{t}_{101} \rightarrow \mathbf{t}_{102}$ The EVSE sets EVSENotification = 'Terminate' (stop) or 'Pause' (pause, with NotificationMaxDelay = 60 s) in the next message.

$\mathbf{t}_{102}$ The EV requests the EVSE to disable side B via PowerDeliveryReq <8a>after side B current is < 5 A DC.

$\mathbf{t}_{102} \rightarrow \mathbf{t}_{103}$ The EVSE reduces side B current to < 5 A DC within 1 s and stops the overvoltage protection.

$\mathbf{t}_{102} \rightarrow \mathbf{t}_{105}$ The EV may open the EV disconnection device after side B current is < 5 A DC and after sending PowerDeliveryReq <9a>.

$\mathbf{t}_{103}$ The EVSE disables side B and opens its disconnection device (if any). The disabled side B must maintain equivalent electrical behavior (e.g., impedance, discharge rate, etc.).

$\mathbf{t}_{104}$ The EVSE confirms that side B is disabled and current < 5 A DC by sending PowerDeliveryRes <8b>with EVSENotification = 'Terminate' and ResponseCode = 'OK'. The EV may enable its IMD.

$\mathbf{t}_{104} \rightarrow \mathbf{t}_{107}$ If the EVSE has not received a SessionStopReq <10a>within 20 s after sending PowerDeliveryRes <8b>, it triggers an error shutdown.

$\mathbf{t}_{105}$ The EV sets CP state to B after receiving PowerDeliveryRes <8b>.

$\mathbf{t}_{105} \rightarrow \mathbf{t}_{106}$ The EV performs welding detection by sending multiple DC_Welding DetectionReq <9a>messages; the EVSE responds with DC_Welding DetectionRes <9b>messages.

$\mathbf{t}_{106}$ The EV completes welding detection, opens its disconnection device, and allows passive discharge of side B voltage.

$\mathbf{t}_{106} \rightarrow \mathbf{t}_{107}$ The EVSE sends the last DC_WeldingDetectionRes <9b>message with ResponseCode = 'OK'.

$\mathbf{t}_{107}$ The EV sends SessionStopReq <10a>with parameter 'Pause' or 'Terminate' depending on the EVSE notification.

$\mathbf{t}_{107} \rightarrow \mathbf{t}_{108}$ The EVSE reduces and maintains side B voltage $\leq$ 60 V DC within 1 s.

$\mathbf{t}_{108}$ The EVSE sends SessionStopRes <10b>.

$\mathbf{t}_{108} \rightarrow \mathbf{t}_{110}$ The EV unlocks the vehicle connector after receiving SessionStopRes <10b>. If supported, the EV may restart or pause the session, resuming from t_1 in the normal startup sequence.

$\mathbf{t}_{109}$ The EVSE may turn off its CP oscillator 5 s after sending SessionStopRes <10b>.

$\mathbf{t}_{110}$ Un-mating the vehicle connector changes the CP state from B to A. Disabled side B limitations are no longer required[42].

Chapter 3

DC bidirectional charging preliminary test

The initial tests for DC bidirectional charging were performed without an actual electric vehicle, but by utilizing a second Itech IT6642C converter in battery emulator mode. This approach eliminates the risk of charging a real battery during the early tests, which may contain potential code errors, while allowing the overall power system (DC power supply + emulated battery) to operate as two bidirectional converters in a recirculating configuration (or ”power-in-the-loop” setup). Another precaution taken was to power the two converters from the same cabinet, ensuring that during power recirculation, the net positive power drawn from the AC mains is always outgoing towards the load.

3.1 Battery emulator

For the experimental verification and validation of energy management systems and Battery Management Systems (BMS), it is often necessary to have a programmable source capable of realistically reproducing the electrical behavior of an electrochemical cell. To this end, a *Battery Emulator* is used, that is, an electronic device capable of simulating the voltage across a real battery as a function of its State of Charge (SOC) and the current being sourced or sunk.

In this work, the **ITECH IT6600C** emulator was employed, which allows the configuration of a cell model through the *Battery Emulation* function. This model is defined using a **lookup table** that associates, for each discrete SOC value, the Open Circuit Voltage (OCV) and the equivalent Internal Resistance (R_{int}).

3.1.1 Equivalent Cell Model

The electrical behavior of a lithium-ion cell can be approximated using an equivalent circuit model consisting of a voltage source dependent on the state of charge and an internal resistance. The simplest form, known as the **single-resistor model**, is expressed as:

$$V_{\text{term}} = OCV(SOC) - I \cdot R_{\text{int}}(SOC) \tag{3.1}$$

where:

- V_{term} is the terminal voltage of the cell [V];
- $OCV(SOC)$ is the open-circuit voltage [V];
- $R_{\text{int}}(SOC)$ is the equivalent internal resistance [Ω];
- I is the current (positive during discharge) [A].

3.1.2 Relationship Between SOC, Current, and Capacity

The State of Charge (SOC) represents the fraction of remaining capacity with respect to the nominal capacity. It is defined as:

$$SOC(t) = SOC_0 - \frac{1}{C_n} \int_0^t I(\tau)\, d\tau \tag{3.2}$$

where SOC_0 is the initial state of charge and C_n is the nominal capacity of the cell [Ah]. In discrete form, this becomes:

$$SOC_{k+1} = SOC_k - \frac{\Delta t}{C_n} \cdot I_k \tag{3.3}$$

3.1.3 Lookup Table and Interpolation

The **lookup table** represents the core of the model used by the emulator. It contains triplets of values $(SOC_i, OCV_i, R_{\text{int},i})$ for $i = 1, \dots, N$, expressed as:

$$\mathcal{T} = \begin{bmatrix} SOC_1 & OCV_1 & R_{\text{int},1} \\ SOC_2 & OCV_2 & R_{\text{int},2} \\ \vdots & \vdots & \vdots \\ SOC_N & OCV_N & R_{\text{int},N} \end{bmatrix} \tag{3.4}$$

During emulation, for each instantaneous value of $SOC(t)$, the corresponding OCV and R_{int} are obtained by means of **linear interpolation** between the adjacent table points:

$$OCV(SOC) = OCV_i + (OCV_{i+1} - OCV_i) \cdot \frac{SOC - SOC_i}{SOC_{i+1} - SOC_i} \tag{3.5}$$

$$R_{\text{int}}(SOC) = R_{\text{int},i} + (R_{\text{int},i+1} - R_{\text{int},i}) \cdot \frac{SOC - SOC_i}{SOC_{i+1} - SOC_i} \tag{3.6}$$

3.1.4 Typical Behavior of Electrical Characteristics

The functions $OCV(SOC)$ and $R_{\text{int}}(SOC)$ show typical behavior for lithium-ion cells. For the test, a lookup table with 100 points was used, taking into consideration an NMC cell.

- **OCV–SOC:** monotonically increasing curve, with higher slope at low (0–10%) and high (90–100%) charge levels, and an almost flat region in the mid-range:

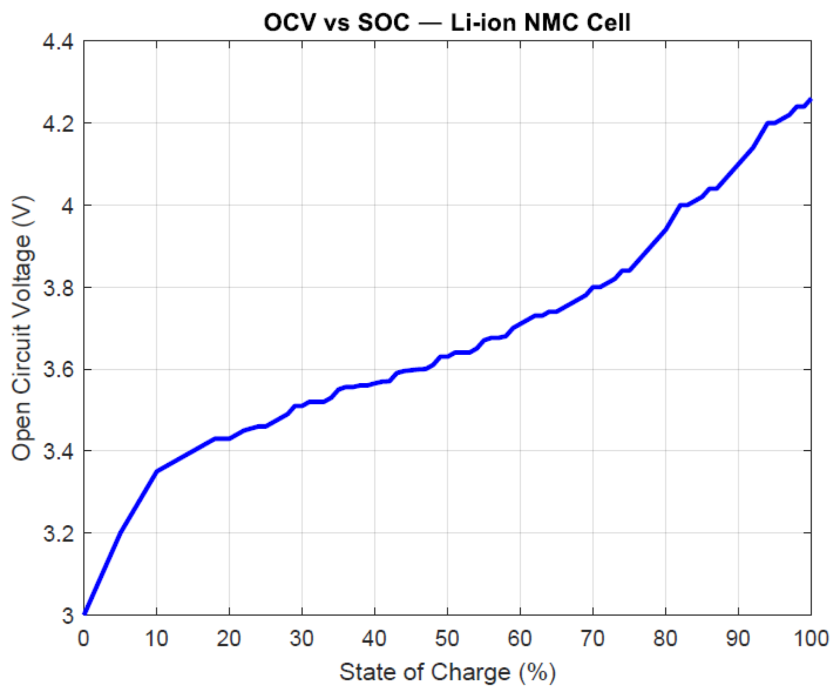


Figure 3.1: *Voc-Soc profile implemented in battery emulator*

- R_{int}**–SOC:** opposite trend, with higher resistance at low and high SOC levels, and a minimum in the intermediate zone.

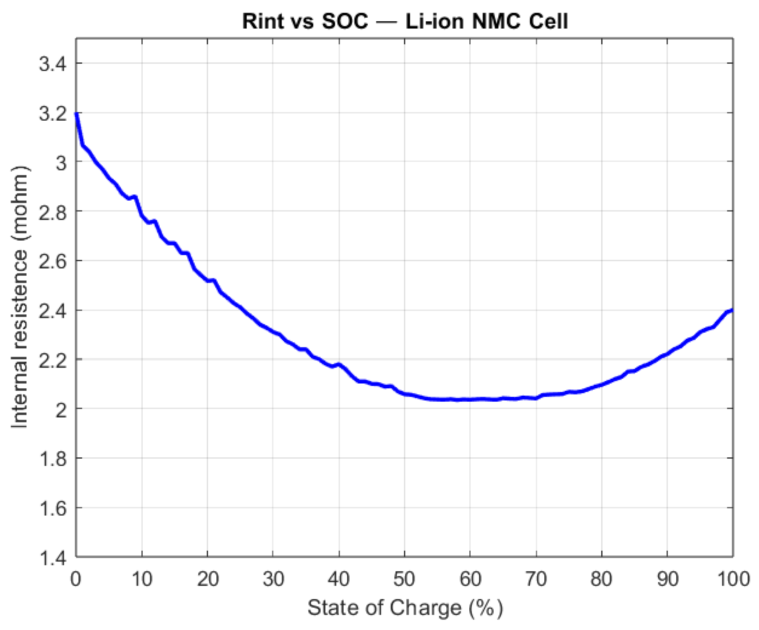


Figure 3.2: *Internal resistence variation for a NMC cell*

In order to obtain the overall electrical parameters of the battery pack starting from a single NMC cell, the series–parallel configuration must be considered. When S cells are connected in series, the total open-circuit voltage and resistance increase linearly, while P parallel branches increase the total capacity and reduce the equivalent internal resistance. The following table reports the resulting pack parameters for a 96S×46P configuration derived from the single-cell data:

SOC	OCV_{cell}	OCV_{pack}	R_{cell}	R_{pack}	Q_{pack}
0%	3.20V	307.20V	3.20mΩ	6.678mΩ	230Ah
55%	3.67V	352.32V	2.035mΩ	4.245mΩ	230Ah
100%	4.26V	408.96V	2.40mΩ	5.009mΩ	230Ah

Table 3.1: *Pack parameters derived from single-cell characteristics (NMC, 96S×46P)*

3.1.5 Operational Equations of the Emulator

During operation, the emulator continuously updates the terminal voltage according to the measured current and the calculated SOC. The main operational equations are as follows:

$$SOC(t+\Delta t) = SOC(t) - \frac{I(t)\,\Delta t}{C_n} \tag{3.7}$$

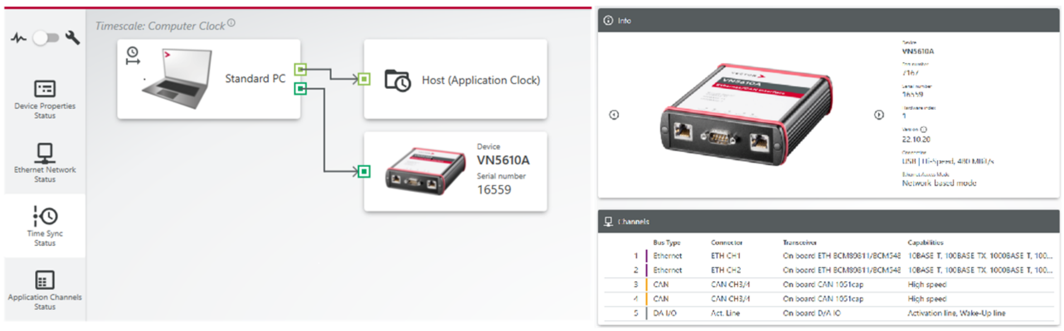
$$(OCV, R_{\text{int}}) = f_{\text{interp}}(SOC) \tag{3.8}$$

$$V_{\text{term}}(t) = OCV(SOC(t)) - I(t)\,R_{\text{int}}(SOC(t)) \tag{3.9}$$

This approach enables the emulator to reproduce the dynamic voltage response of the cell based on empirical or measured data, ensuring realistic emulation of its static electrical behavior.

3.2 Hardware and Network configuration

The connectivity between the vehicular control network (CAN) and the communication channel (Ethernet) was established using the Vector VN5610A interface. This tool was chosen for its capacity to ensure common reference hardware synchronization between the data streams of the two buses, which is crucial for the precise analysis of response times during bidirectional charging. Its correct operation, including the configuration of network parameters, was set up using the Vector Hardware Manager software.



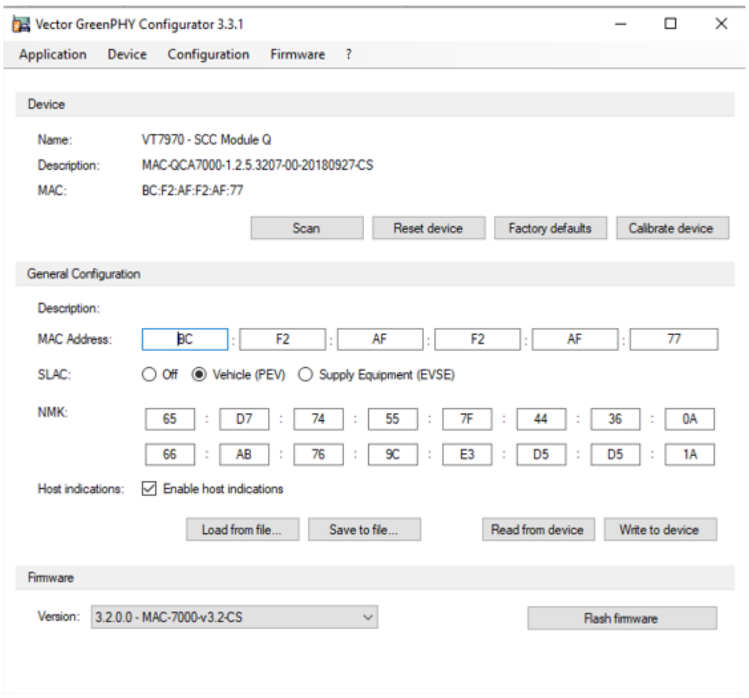
Device: VN5610A
Part number: 7167
Serial number: 16559
Hardware index: 1
Version: 22.10.20
Connection: USB | Hi-Speed, 480 MBit/s
Ethernet Access Mode: Network based mode

	Bus Type	Connector	Transceiver	Capabilities
1	Ethernet	ETH CH1	On board ETH BCM89811/BCM548	10BASE T, 100BASE TX, 1000BASE T, 100...
2	Ethernet	ETH CH2	On board ETH BCM89811/BCM548	10BASE T, 100BASE TX, 1000BASE T, 100...
3	CAN	CAN CH3/4	On board CAN 1051cap	High speed
4	CAN	CAN CH3/4	On board CAN 1051cap	High speed
5	D/A I/O	Act. Line	On board D/A IO	Activation line, Wake-Up line

Figure 3.3: *Hardware Vector manager setup*

To manage the high-level communication according to ISO 15118-20 between the vSECC and the VTsystem, the CANoe Pro software was used, employing a specific configuration for Smart Charging that allows for the visualization of communication with the VTsystem, which emulates the vehicle (not the power section). The networks used in the setup were first defined, and then, given that the bus was real and not simulated, the computer's input ports for interfacing with the aforementioned networks were defined.

The predominant part of the configuration concerns the VTsystem, where it is necessary to configure the interface of the VT7970 board (Smart Charging), which performs measurements on the basic signalling communication, including PP voltage, CP, PWM, and Duty Cycle. The Power Line Communication (PLC) interface of the VTsystem, managed by the VT7970 module and configured using the GreenPhy Configurator, was set up to emulate the Electric Vehicle (EV) role. To this end, the SLAC (Signal Level Attenuation Characterization) parameter was defined as EV. To ensure correct identification within the PLC network and Level 2 addressing, the specific MAC Address BC:F2:AF:F2:AF:77 was assigned to the interface. Furthermore, communication security and the definition of the logical network were guaranteed by utilizing the NMK (Network Membership Key): 65:D7:74:55:7F:44:36:0A:66:AB:76:9C:E3:D5:D5:1A. Finally, the "Enable Host Indication" option was activated to notify the host system of the successful completion of the SLAC process and the subsequent readiness of the interface for the exchange of ISO 15118-20 protocol messages.



Device

Name: VT7970 - SCC Module Q

Description: MAC-QCA7000-1.2.5.3207-00-20180927-CS

MAC: BC:F2:AF:F2:AF:77

General Configuration

Description:

MAC Address: BC : F2 : AF : F2 : AF : 77

SLAC: Off [✓ Vehicle (PEV)] Supply Equipment (EVSE)

NMK: 65 : D7 : 74 : 55 : 7F : 44 : 36 : 0A
66 : AB : 76 : 9C : E3 : D5 : D5 : 1A

Host indications: [✓ Enable host indications]

Firmware

Version: 3.2.0.0 - MAC-7000-v3.2-CS

Figure 3.4: *GreenPhy Configurator for EV emulator*

To start a charging session, the network defined in the setup must be specified. By default, the Ethernet network is already configured and includes the VSECC, the VT7970 board, and a monitoring node. The CAN network was defined by introducing the CAN communication interface, a generation node for defining CAN messages for possible testing, and a node called PECC. The latter, the Power Electronics Communication Controller, contains the CAPL file required for reading messages sent by the VSECC and all CAN commands used to control the power converter(see Annex D for more detail). The CAN commands used are interpreted by two different CAN databases: for the VSECC, a database based on PEP-CAN 1.4 messages was used a proprietary protocol developed by Vector while for the Itech converter, despite a database being available, a specific database was created due to compatibility issues, defining the relevant messages and signals. The setup and the databases used are shown in the following figure:

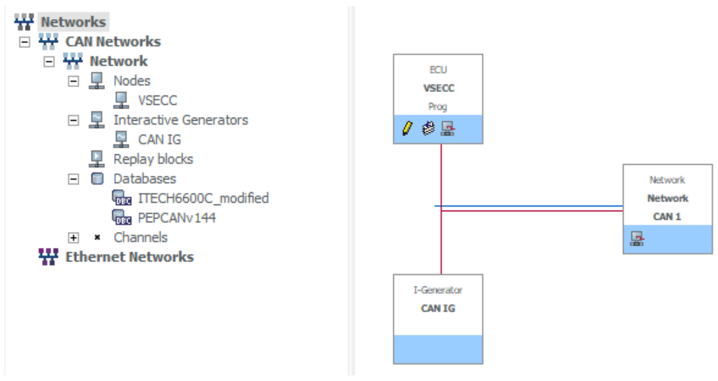


Figure 3.5: *Setup for the charging test*

3.3 Charging session

During the charging session, it is essential to define the limits of both the vehicle and the charging station. For the vehicle, these limits can be set directly in the sample configuration, while the limits of the charging station are defined within the PECC, written in CAPL, in the corresponding node. The interface appears as follows:

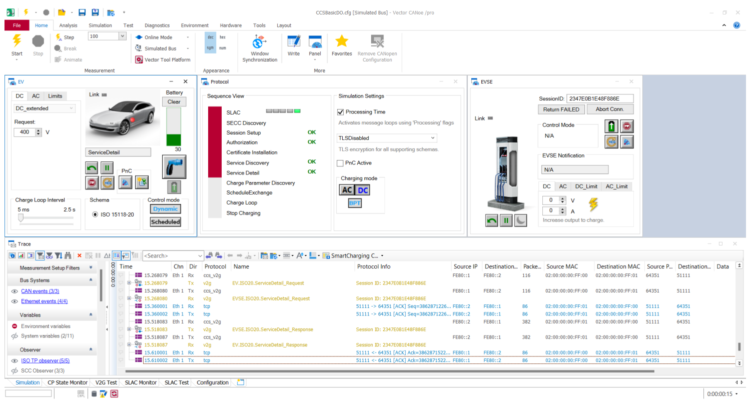


Figure 3.6: *Principal page of a CANoe charging session*

For the test, dynamic mode and BPT (Bidirectional Power Transfer) mode were used. For ISO 15118-20 to function correctly, the TLS (Transport Layer Security) must be active and updated to version 1.3. The TLS 1.3 is a cryptographic protocol used to secure the communication between the charging station (EVSE) and management systems (CSMS) in smart charging (for example, with OCPP 2.0.1). It ensures data confidentiality and integrity (such as charging session details and user data) by preventing eavesdropping and tampering. One of the fundamental innovations introduced by the standard is Plug and Charge. This functionality allows the electric vehicle to automatically authenticate and start charging and payment simply by plugging in the cable, without using apps or cards. For the purpose of this setup, this option was deactivated, given that the ultimate goal is to verify the phases of the protocol that involve power exchange. The proposed setup is depicted in the following figure:

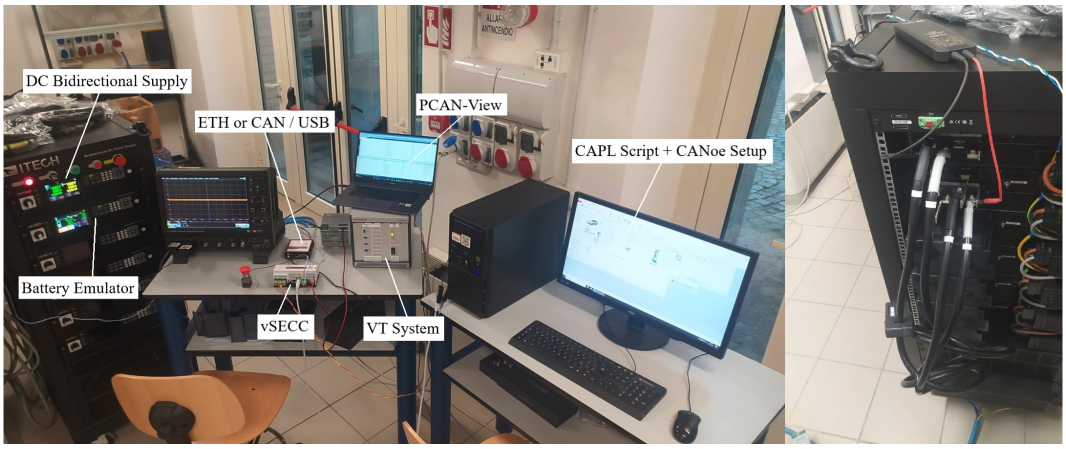
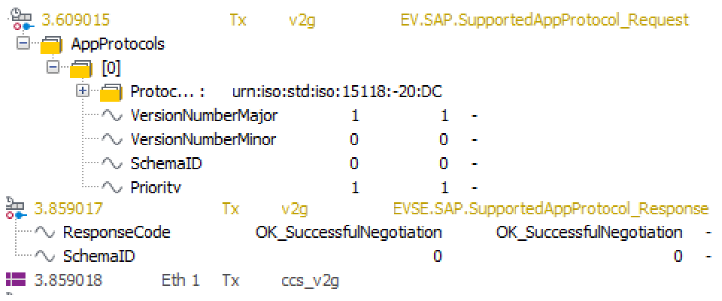


Figure 3.7: *Proposed setup for the preliminary test*

The charging process is initiated through a software interface, where pressing the start button initializes the charging session. After the completion of the SLAC procedure, according to ISO 15118-3, the state machine defined in ISO 15118-20 begins. The first step of the communication concerns the *SupportedAppProtocolReq/Res* exchange, where the vehicle queries the charging station about the application protocols supported by the EVSE. The main available protocols are:

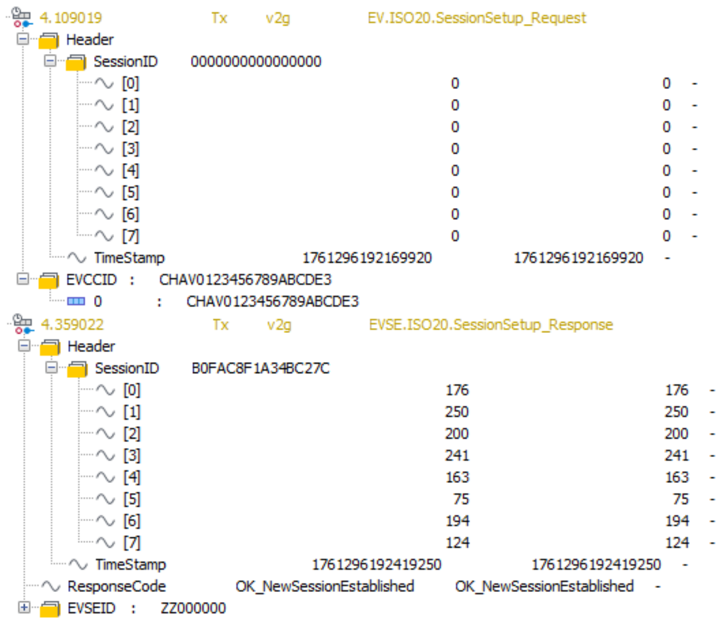
- ISO 15118-2 AC
- ISO 15118-2 DC
- ISO 15118-20 AC
- ISO 15118-20 DC
- ISO 15118-20 WPT (Note: this option has not been licensed in the SECC currently in use)



```
3.609015            Tx      v2g          EV.SAP.SupportedAppProtocol_Request
  AppProtocols
    [0]
      Protoc... :   urn:iso:std:iso:15118:-20:DC
      VersionNumberMajor         1          1   -
      VersionNumberMinor         0          0   -
      SchemaID                   0          0   -
      Priority                   1          1   -
3.859017            Tx      v2g          EVSE.SAP.SupportedAppProtocol_Response
  ResponseCode          OK_SuccessfulNegotiation       OK_SuccessfulNegotiation   -
  SchemaID                                     0                              0   -
3.859018     Eth 1  Tx      ccs_v2g
```

Figure 3.8: *SupportedAppProtocol, the setup works with ISO15118-20 DC*

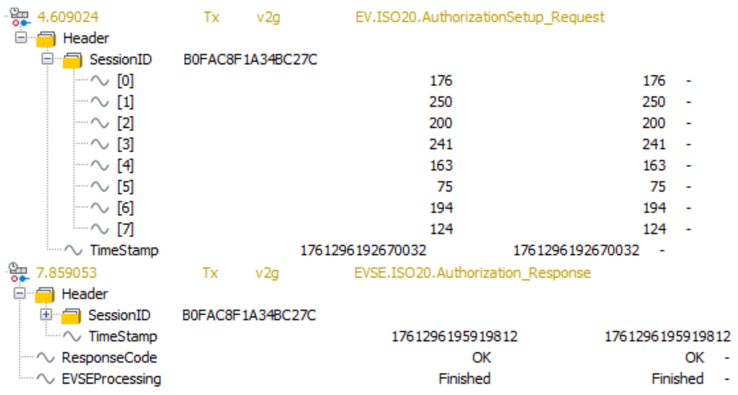
If the EVSE determines that the required protocol is compatible, it sends feedback with the message ”OKSuccessfullNegotiation” and the communication proceeds to the next step. During preliminary tests to verify the reliability of the protocol, only the ISO15118-2 options were enabled, while only ISO15118-20 was maintained on the EV (VT-System). When attempting to start the session, the protocol resulted in an Error due to incompatibility. In the current setup, during the Session Setup phase, the vehicle with identifier EVCCID CHAV0123456789ABCDE3 sends a request to the charging station to initiate a new charging session. The station (EVSEID ZZ000000) responds by confirming the connection with the status “NewConnectionEstablished” and assigns the session identifier B0FAC8F1A34BC27C. From this moment on, communication is formally established, and all subsequent messages in the ISO 15118-20 protocol will reference this Session ID to maintain the continuity of the charging session.



```
4.109019          Tx    v2g        EV.ISO20.SessionSetup_Request
  Header
    SessionID     0000000000000000
      [0]                               0                       0  -
      [1]                               0                       0  -
      [2]                               0                       0  -
      [3]                               0                       0  -
      [4]                               0                       0  -
      [5]                               0                       0  -
      [6]                               0                       0  -
      [7]                               0                       0  -
    TimeStamp              1761296192169920        1761296192169920  -
  EVCCID   :    CHAV0123456789ABCDE3
    0        :    CHAV0123456789ABCDE3
4.359022          Tx    v2g        EVSE.ISO20.SessionSetup_Response
  Header
    SessionID     B0FAC8F1A34BC27C
      [0]                             176                     176  -
      [1]                             250                     250  -
      [2]                             200                     200  -
      [3]                             241                     241  -
      [4]                             163                     163  -
      [5]                              75                      75  -
      [6]                             194                     194  -
      [7]                             124                     124  -
    TimeStamp              1761296192419250        1761296192419250  -
  ResponseCode       OK_NewSessionEstablished    OK_NewSessionEstablished  -
  EVSEID   :    ZZ000000
```

Figure 3.9: *In Session Setup, EVSE assigns an identifier that persists throughout the session*

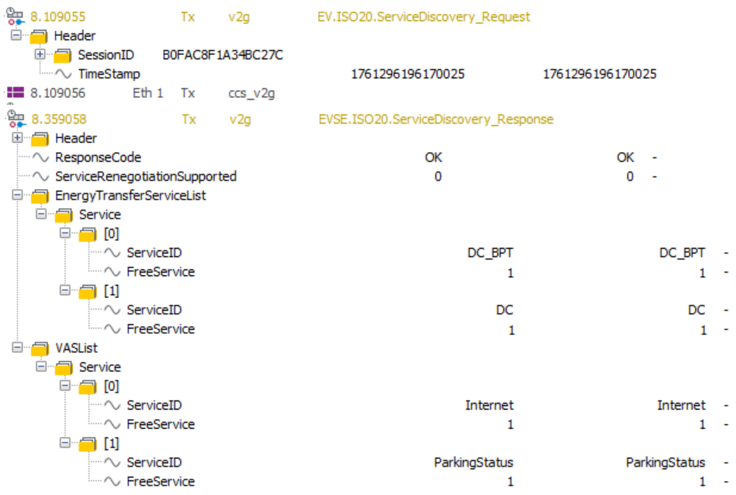
After negotiating the authorization method, the vehicle sends a request to the charging station to authorize the charging session using the supported standard method. The station (EVSEID ZZ000000) verifies the vehicle's identity and responds with a positive outcome, confirming that the charging session can proceed. From this point, the authorization phase is complete, and the process can move on to the discovery of charging services.



```
4.609024          Tx    v2g        EV.ISO20.AuthorizationSetup_Request
  Header
    SessionID     B0FAC8F1A34BC27C
      [0]                             176                     176  -
      [1]                             250                     250  -
      [2]                             200                     200  -
      [3]                             241                     241  -
      [4]                             163                     163  -
      [5]                              75                      75  -
      [6]                             194                     194  -
      [7]                             124                     124  -
    TimeStamp              1761296192670032        1761296192670032  -
7.859053          Tx    v2g        EVSE.ISO20.Authorization_Response
  Header
    SessionID     B0FAC8F1A34BC27C
    TimeStamp                       1761296195919812        1761296195919812
  ResponseCode                                    OK                      OK  -
  EVSEProcessing                            Finished                Finished  -
```

Figure 3.10: *In AuthorizationSetup/Authorization, EVSE define payment type(for the test purpose PnC wasn't used)*

The vehicle then proceeds to the next phase, querying the EVSE (Electric Vehicle Supply Equipment) about the available services, including the possibility of renegotiating the AC or DC services during a charging session. In this specific case, the value 0 indicates that the option is not accepted. The services in question are BPT_DC and DC, and for simplicity, they have been set as free (FreeService = 1).

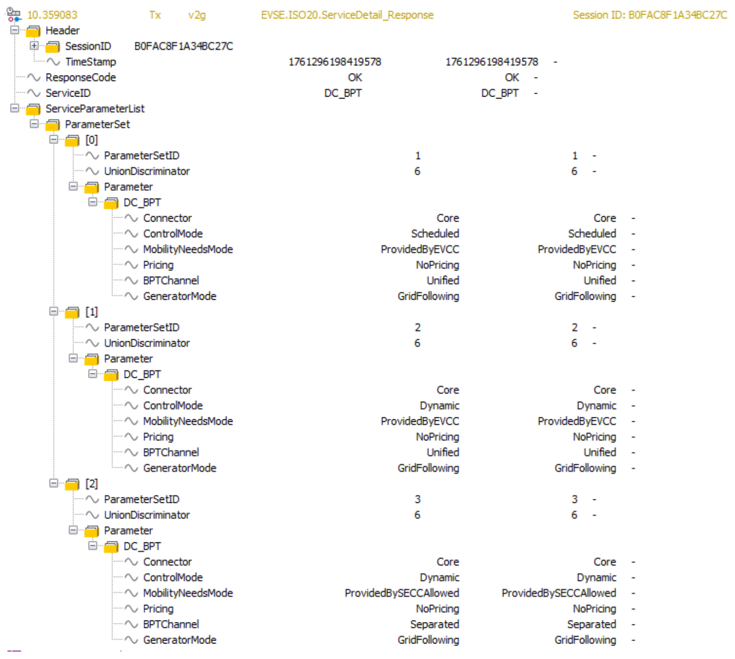


```
8.109055          Tx    v2g        EV.ISO20.ServiceDiscovery_Request
  Header
    SessionID   B0FAC8F1A34BC27C
    TimeStamp                      1761296196170025          1761296196170025
8.109056   Eth 1  Tx    ccs_v2g
8.359058          Tx    v2g        EVSE.ISO20.ServiceDiscovery_Response
  Header
  ResponseCode                                 OK                   OK  -
  ServiceRenegotiationSupported                0                    0   -
  EnergyTransferServiceList
    Service
      [0]
        ServiceID                          DC_BPT               DC_BPT  -
        FreeService                             1                    1  -
      [1]
        ServiceID                              DC                   DC  -
        FreeService                             1                    1  -
  VASList
    Service
      [0]
        ServiceID                        Internet             Internet  -
        FreeService                             1                    1  -
      [1]
        ServiceID                   ParkingStatus        ParkingStatus  -
        FreeService                             1                    1  -
```

Figure 3.11: *In Service Discovery, EVSE define the possible renegotiation of services and VAS)*

The VASList (Value Added Services) are also present, which add services such as parking payment and data traffic offered by the charging station, useful for downloading system updates. In the setup, these services are available and free of charge, despite there being no actual usage.

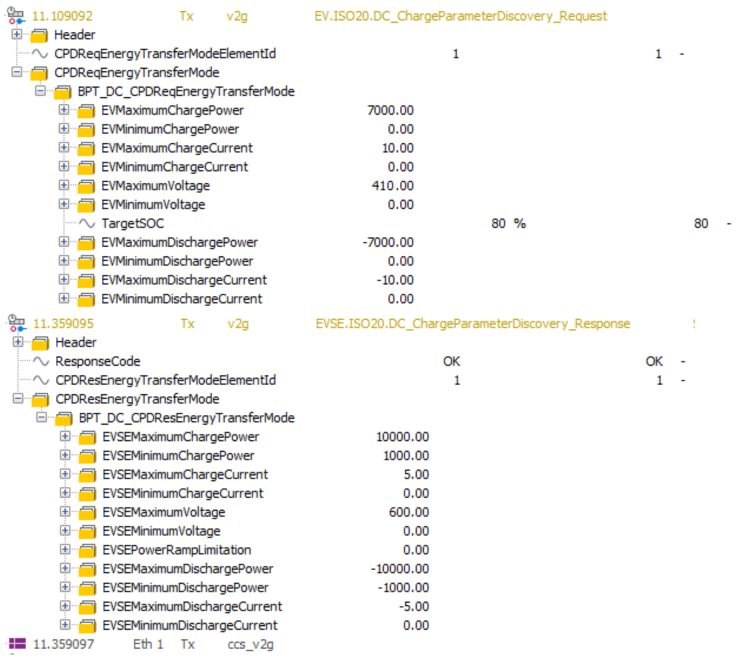
In the current setup, during the Service Detail phase, the charging station provides the vehicle with specific information about the selected energy transfer service. The received parameters indicate that the main connector is used (Connector: Core) with dynamic power control (ControlMode: Dynamic), and no pricing model is applied (Pricing: No Pricing). The vehicle is responsible for defining its own mobility requirements (MobilityNeedMode: ProvidedByEVCC), and communication takes place over a unified channel (BTPChannel: Unified). The system operates in GridFollowing mode, meaning that during energy transfer from the vehicle to the grid, the EVSE will follow the grid voltage and frequency. In the subsequent ServiceSelection message, the particularity is that the EV does not send a new request to the EVSE, but confirms one of the *ServiceDetails* from the previous state, and then receives approval in the *ServiceSelectionResponse*.



```
10.359083          Tx    v2g       EVSE.ISO20.ServiceDetail_Response                        Session ID: B0FAC8F1A34BC27C
Header
  SessionID      B0FAC8F1A34BC27C
  TimeStamp                              1761296198419578          1761296198419578   -
ResponseCode                                           OK                        OK   -
ServiceID                                          DC_BPT                    DC_BPT   -
ServiceParameterList
  ParameterSet
    [0]
      ParameterSetID                                        1                         1   -
      UnionDiscriminator                                    6                         6   -
      Parameter
        DC_BPT
          Connector                                      Core                      Core   -
          ControlMode                               Scheduled                 Scheduled   -
          MobilityNeedsMode                    ProvidedByEVCC            ProvidedByEVCC   -
          Pricing                                   NoPricing                 NoPricing   -
          BPTChannel                                  Unified                   Unified   -
          GeneratorMode                         GridFollowing             GridFollowing   -
    [1]
      ParameterSetID                                        2                         2   -
      UnionDiscriminator                                    6                         6   -
      Parameter
        DC_BPT
          Connector                                      Core                      Core   -
          ControlMode                                 Dynamic                   Dynamic   -
          MobilityNeedsMode                    ProvidedByEVCC            ProvidedByEVCC   -
          Pricing                                   NoPricing                 NoPricing   -
          BPTChannel                                  Unified                   Unified   -
          GeneratorMode                         GridFollowing             GridFollowing   -
    [2]
      ParameterSetID                                        3                         3   -
      UnionDiscriminator                                    6                         6   -
      Parameter
        DC_BPT
          Connector                                      Core                      Core   -
          ControlMode                                 Dynamic                   Dynamic   -
          MobilityNeedsMode             ProvidedBySECCAllowed     ProvidedBySECCAllowed   -
          Pricing                                   NoPricing                 NoPricing   -
          BPTChannel                                Separated                 Separated   -
          GeneratorMode                         GridFollowing             GridFollowing   -
```

Figure 3.12: *In Service Details, different possibilities are available*

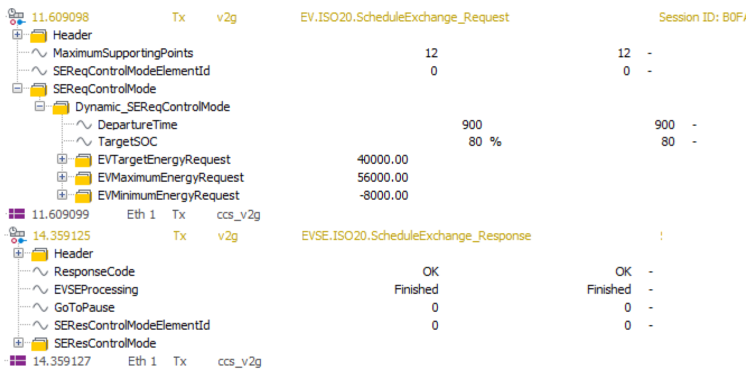
During the Charge Parameter Discovery phase, the vehicle (emulated by the VT System) communicates its operational limits and energy requirements for the charging session to the station. Specifically, the EV declares a maximum charging power of 7 kW, with a maximum current of 10 A and a maximum voltage of 500 V, while also specifying a Target State of Charge (SOC) of 80%. The EV further indicates its ability to support discharging up to –7 kW with a maximum discharge current of –10 A. In response, the charging station (EVSE) reports its actual operational limits: a maximum charging power of 10 kW, but a maximum current capability of only 5 A, and a nominal voltage up to 600 V. The EVSE's discharge range is defined between –1 kW and –10 kW, with a maximum discharge current of –5 A. The comparison shows that the station enforces more conservative constraints, particularly on the current limit, which will define the effective parameters applied during the subsequent Power Delivery phase.



```
11.109092          Tx     v2g          EV.ISO20.DC_ChargeParameterDiscovery_Request
  Header
  CPDReqEnergyTransferModeElementId                     1                          1  -
  CPDReqEnergyTransferMode
    BPT_DC_CPDReqEnergyTransferMode
      EVMaximumChargePower              7000.00
      EVMinimumChargePower                 0.00
      EVMaximumChargeCurrent              10.00
      EVMinimumChargeCurrent               0.00
      EVMaximumVoltage                   410.00
      EVMinimumVoltage                     0.00
      TargetSOC                                      80  %                        80  -
      EVMaximumDischargePower          -7000.00
      EVMinimumDischargePower              0.00
      EVMaximumDischargeCurrent          -10.00
      EVMinimumDischargeCurrent            0.00
11.359095          Tx     v2g          EVSE.ISO20.DC_ChargeParameterDiscovery_Response         !
  Header
  ResponseCode                                          OK                         OK  -
  CPDResEnergyTransferModeElementId                     1                          1  -
  CPDResEnergyTransferMode
    BPT_DC_CPDResEnergyTransferMode
      EVSEMaximumChargePower            10000.00
      EVSEMinimumChargePower             1000.00
      EVSEMaximumChargeCurrent              5.00
      EVSEMinimumChargeCurrent              0.00
      EVSEMaximumVoltage                  600.00
      EVSEMinimumVoltage                    0.00
      EVSEPowerRampLimitation               0.00
      EVSEMaximumDischargePower        -10000.00
      EVSEMinimumDischargePower         -1000.00
      EVSEMaximumDischargeCurrent          -5.00
      EVSEMinimumDischargeCurrent           0.00
11.359097    Eth 1  Tx     ccs_v2g
```

Figure 3.13: *In Charge Parametery Discovery, EV and EVSE show their respective Current, Voltage and Power limits*

During the Schedule Exchange phase, the vehicle communicates its energy goals and limits to the charging station. In the current setup, the EV requests a target energy of 40,000 Wh, with a maximum limit of 56,000 Wh and a minimum of –8,000 Wh, the latter indicating the vehicle's capability to deliver energy back to the grid (Vehicle to Grid). The departure time is set to 900 s, corresponding to approximately 15 minutes, within which the charging session is ideally expected to finish. The parameter SERerControlModeElementId = 0 specifies that the default control profile (in this case Dynamic) is used, with no alternative or multiple scheduling modes selected. Finally, the charging station replies with EVSEProcessing = Finished, meaning that the request has been processed successfully and the charging schedule is ready, allowing the protocol to proceed to the next communication phase.



```
11.609098        Tx    v2g    EV.ISO20.ScheduleExchange_Request                Session ID: B0F
  Header
  MaximumSupportingPoints                          12              12  -
  SEReqControlModeElementId                         0               0  -
  SEReqControlMode
    Dynamic_SEReqControlMode
      DepartureTime                                   900             900  -
      TargetSOC                                        80 %            80  -
      EVTargetEnergyRequest               40000.00
      EVMaximumEnergyRequest              56000.00
      EVMinimumEnergyRequest              -8000.00
11.609099   Eth 1   Tx    ccs_v2g
14.359125        Tx    v2g    EVSE.ISO20.ScheduleExchange_Response             !
  Header
  ResponseCode                                     OK              OK  -
  EVSEProcessing                             Finished        Finished  -
  GoToPause                                         0               0  -
  SEResControlModeElementId                         0               0  -
  SEResControlMode
14.359127   Eth 1   Tx    ccs_v2g
```

Figure 3.14: *In Schedule Exchange, EV notified maximum energy transfer to the battery and the grid*

In the current setup, during the Cable Check phase, the charging station (EVSE) performs an electrical insulation verification of the DC power circuit before proceeding to the pre-charging phase. At this stage, both the EVSE and vehicle main contactors remain open, meaning that no galvanic connection is yet established between the DC bus of the charger and the vehicle battery. The verification is carried out by the Insulation Monitoring Device (IMD), which temporarily closes an internal measurement circuit by connecting the DC+ and DC– poles to the protective earth (PE) through a sensing resistor. In this way, the EVSE applies a small test voltage and measures the leakage current to determine the system's insulation resistance. If the measured value is above the minimum threshold defined by the standard (for example, 100 Ω/V, equivalent to 50 kΩ/V for a nominal voltage of 500 V), the insulation state is declared "Valid," and the charging station can proceed to the subsequent Pre-Charge phase.

In the current setup, an IMD was not yet implemented; however, for the final version of the charging lab, it is required according to IEC 61851-23. The selected device is an ABB ISL-A, where the insulation monitoring threshold is configured at 100 kΩ/V. In addition to the acoustic alarm, communication with the VSECC during the charging process is essential. In the CAPL code, when the charging state transitions to CableCheck, a PEP-CAN command activates a digital output on the VSECC, which—through an optocoupler—closes the contact between pins 23 and 25 of the IMD, initiating the measurement.

If an insulation fault is detected, an internal relay of the IMD is triggered (short-circuiting pins 13 and 14 to enable the Fail Safe function). The corresponding logical state is then read by a digital input pin according to the PEP-CAN DigitalIns message. By monitoring the CAN feedback, the system can force the insulation state to Fault and terminate the charging session in case of an insulation failure.

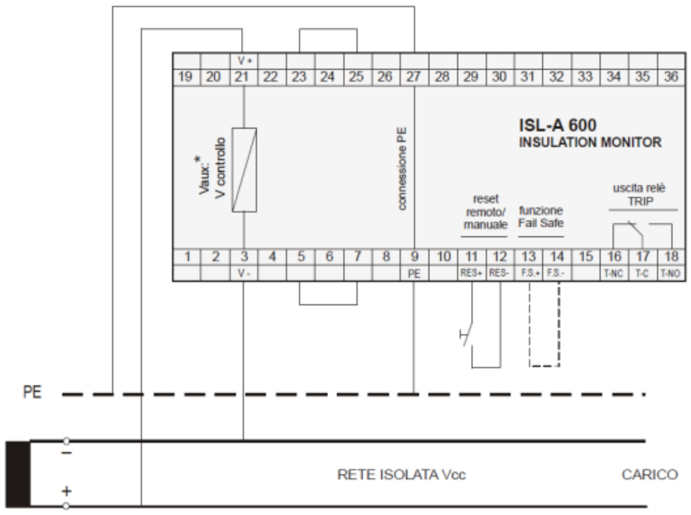
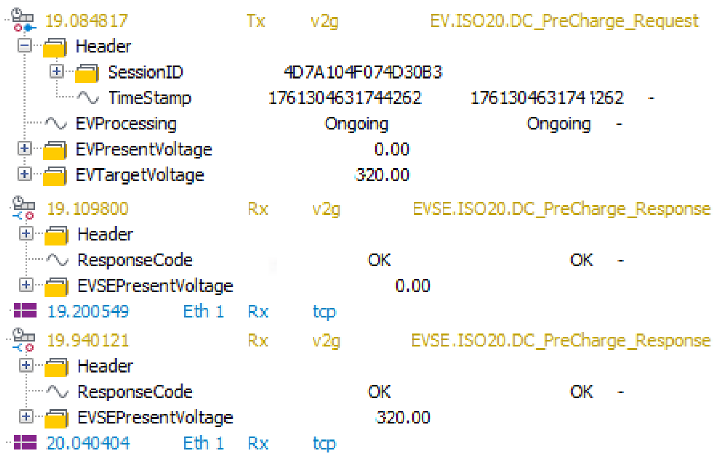


Figure 3.15: *ABB ISL-A IMD connection scheme*

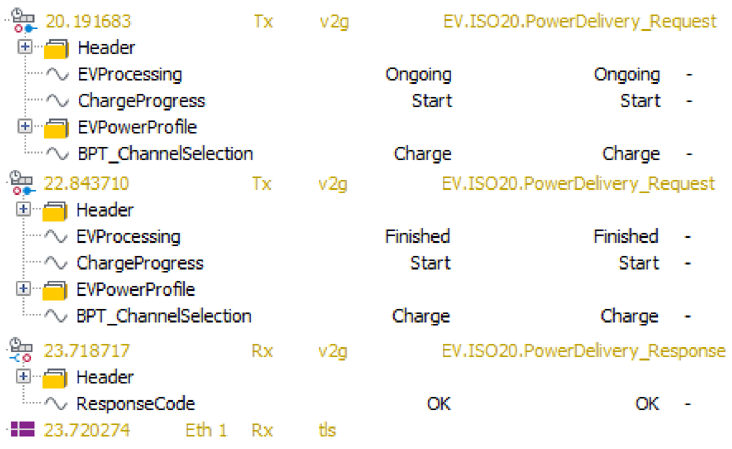
During the Pre-Charge phase, the charging station (EVSE) gradually increases the DC output voltage to match the vehicle's battery voltage before closing the main contactors. In the current setup, the initial PresentVoltage value is 0 V, as the EVSE output is still disconnected from the vehicle bus. The actual EV voltage is monitored through a specific CAN command sent to the ITECH power supply, which measures the real-time output voltage. The EVSE then raises its DC voltage until the measured value (MeasuredVoltage) approaches the requested target of 320 V. Once the voltage difference between the EVSE output and the EV's internal DC bus becomes lower than 20 V, the pre-charge phase is considered successfully completed, and the system transitions to the next stage of the charging process, allowing the main contactors to close and the full current transfer to begin.



```
19.084817          Tx     v2g          EV.ISO20.DC_PreCharge_Request
  Header
    SessionID              4D7A104F074D30B3
    TimeStamp            1761304631744262       1761304631741262   -
  EVProcessing                Ongoing                 Ongoing   -
  EVPresentVoltage               0.00
  EVTargetVoltage              320.00
19.109800          Rx     v2g          EVSE.ISO20.DC_PreCharge_Response
  Header
  ResponseCode                    OK                       OK   -
  EVSEPresentVoltage               0.00
19.200549     Eth 1   Rx     tcp
19.940121          Rx     v2g          EVSE.ISO20.DC_PreCharge_Response
  Header
  ResponseCode                    OK                       OK   -
  EVSEPresentVoltage             320.00
20.040404     Eth 1   Rx     tcp
```

Figure 3.16: *During PreCharge the cable voltage reach the same voltage of the battery.*

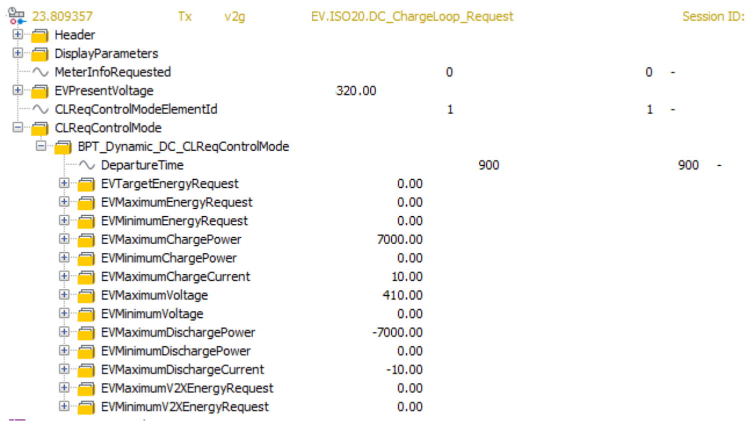
During the Power Delivery phase, the charging session enters the active stage of energy transfer. After completing the Pre-Charge process and closing the main contactors, the charging station (EVSE) enables the delivery of direct current to the vehicle. In this phase, the PowerDeliveryReq message is sent by the EV to confirm its readiness to receive energy and to set the operating state (ChargeProgress = Start). The EVSE responds with PowerDeliveryRes, authorizing the actual power transfer. From this point onward, current control is performed in a regulated mode, according to the voltage and current limits negotiated during the previous phases (ChargeParameterDiscovery and Schedule-Exchange).



```
20.191683        Tx     v2g          EV.ISO20.PowerDelivery_Request
  Header
  EVProcessing                 Ongoing           Ongoing  -
  ChargeProgress                 Start             Start  -
  EVPowerProfile
  BPT_ChannelSelection          Charge            Charge  -
22.843710        Tx     v2g          EV.ISO20.PowerDelivery_Request
  Header
  EVProcessing                Finished          Finished  -
  ChargeProgress                 Start             Start  -
  EVPowerProfile
  BPT_ChannelSelection          Charge            Charge  -
23.718717        Rx     v2g          EV.ISO20.PowerDelivery_Response
  Header
  ResponseCode                      OK                OK  -
23.720274  Eth 1 Rx     tls
```

Figure 3.17: *In Power Delivery EV close the contactor and start power transfer.*

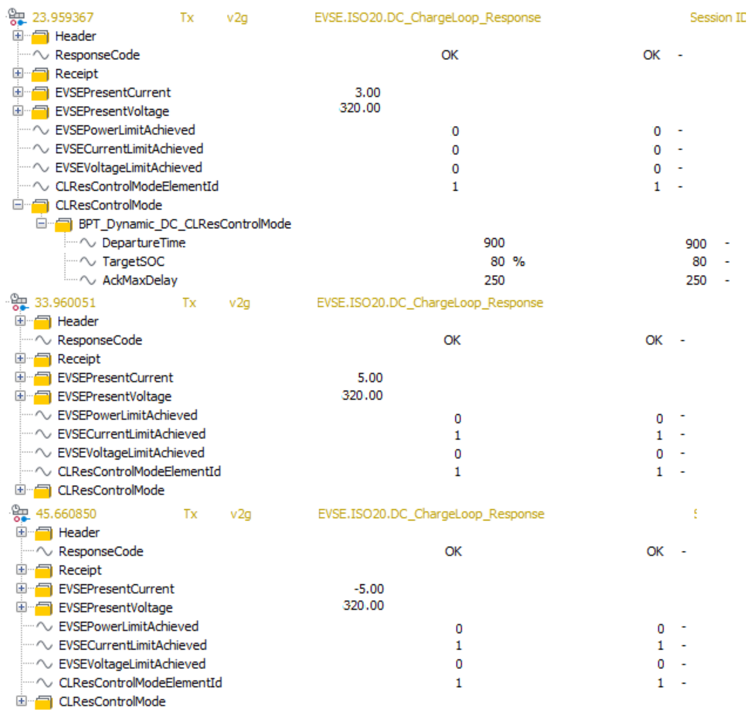
In the Charging Loop phase, the EV sends the PresentVoltage value of 320 V and retransmits its operating limits. Among the various parameters, it also specifies the DepartureTime, informing the EVSE of its planned connection duration of 900 seconds.



```
23.809357        Tx     v2g          EV.ISO20.DC_ChargeLoop_Request              Session ID:
  Header
  DisplayParameters
  MeterInfoRequested                          0                  0  -
  EVPresentVoltage               320.00
  CLReqControlModeElementId                   1                  1  -
  CLReqControlMode
    BPT_Dynamic_DC_CLReqControlMode
      DepartureTime                             900                900  -
      EVTargetEnergyRequest          0.00
      EVMaximumEnergyRequest         0.00
      EVMinimumEnergyRequest         0.00
      EVMaximumChargePower        7000.00
      EVMinimumChargePower           0.00
      EVMaximumChargeCurrent        10.00
      EVMaximumVoltage             410.00
      EVMinimumVoltage               0.00
      EVMaximumDischargePower    -7000.00
      EVMinimumDischargePower        0.00
      EVMaximumDischargeCurrent    -10.00
      EVMaximumV2XEnergyRequest      0.00
      EVMinimumV2XEnergyRequest      0.00
```

Figure 3.18: *In ChargingLoop periodically EV and EVSE send TargetCurrent and TargetVoltage.*

During the test, several current values were tested, starting with 3A set by the EVSE. Subsequently, the second current value was 5A, which represents the maximum deliverable current for the charging session, as it matches the station limit. To test the reverse operation, the configuration includes a button that inverts the charging direction; once pressed, it reverses the current polarity, setting the reference value to −5 A. This value is not arbitrary but corresponds to the static limit defined by the EVSE.



```
23.959367        Tx    v2g    EVSE.ISO20.DC_ChargeLoop_Response             Session IC
  Header
  ResponseCode                                     OK                 OK  -
  Receipt
  EVSEPresentCurrent                       3.00
  EVSEPresentVoltage                     320.00
  EVSEPowerLimitAchieved                           0                   0  -
  EVSECurrentLimitAchieved                         0                   0  -
  EVSEVoltageLimitAchieved                         0                   0  -
  CLResControlModeElementId                        1                   1  -
  CLResControlMode
    BPT_Dynamic_DC_CLResControlMode
      DepartureTime                                    900                 900  -
      TargetSOC                                         80 %                80  -
      AckMaxDelay                                      250                 250  -
33.960051        Tx    v2g    EVSE.ISO20.DC_ChargeLoop_Response
  Header
  ResponseCode                                     OK                 OK  -
  Receipt
  EVSEPresentCurrent                       5.00
  EVSEPresentVoltage                     320.00
  EVSEPowerLimitAchieved                           0                   0  -
  EVSECurrentLimitAchieved                         1                   1  -
  EVSEVoltageLimitAchieved                         0                   0  -
  CLResControlModeElementId                        1                   1  -
  CLResControlMode
45.660850        Tx    v2g    EVSE.ISO20.DC_ChargeLoop_Response             !
  Header
  ResponseCode                                     OK                 OK  -
  Receipt
  EVSEPresentCurrent                      -5.00
  EVSEPresentVoltage                     320.00
  EVSEPowerLimitAchieved                           0                   0  -
  EVSECurrentLimitAchieved                         1                   1  -
  EVSEVoltageLimitAchieved                         0                   0  -
  CLResControlModeElementId                        1                   1  -
  CLResControlMode
```

Figure 3.19: *In ChargingLoop Response EVSE send TargetCurrent and if EVSE Limit are achieved.*

3.3.1 Test measurements

The CANoe setup configuration allows reading the basic signalling, defined by IEC 61851-1. Compared to charging with only low-level communication, the duty cycle remains at 5%, whereas previously it varied to communicate the current limit of the EVSE to the EV.

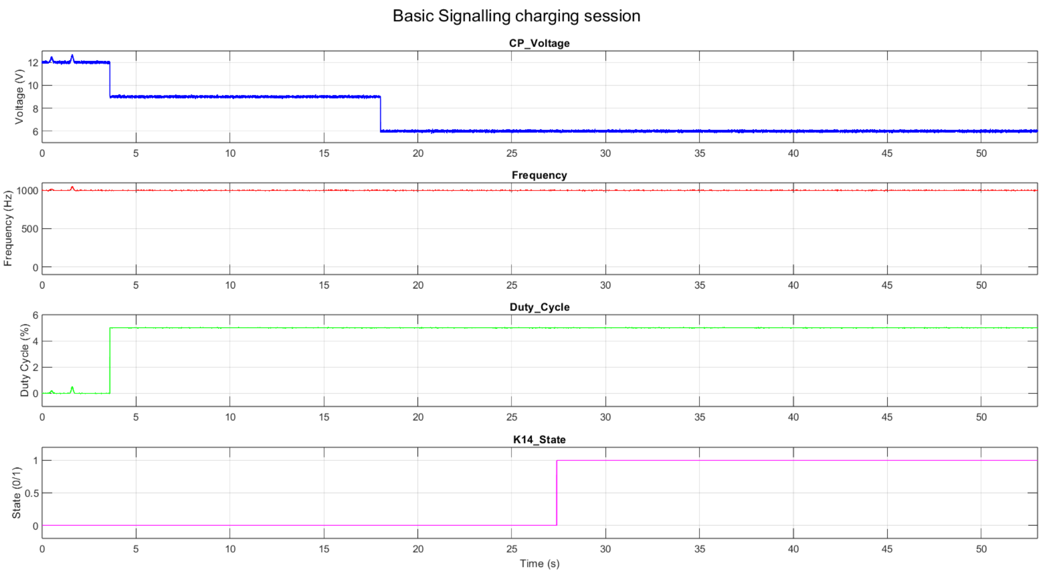


Figure 3.20: *Basic signalling measurement for the charging session.*

The Control Pilot voltage is initially at 12V with a frequency of 1kHz. The cable connection is fictitious and is performed via CANoe using a dedicated button. This action leads to the equivalent connection of an EV-side resistor and the divider voltage drops to 9V. The duty cycle changes to 5% and remains constant as long as the cable is connected. When the system is ready for charging, a contact closes which modifies the resistance value on the EV side, and the divider voltage becomes 6V. The peculiarity is that this state is not activated when the power transfer starts, but only to indicate that the contactors on the EVSE side and the EV side can be enabled. In the specific case, starting from 18s, the cable check is performed (EVSE contactor closed). The actual power transfer is observed at 27.39s, where the closing of the K14 contactor (the name of the contactor in the VT7970 module equivalent to the closing of the emulated EV contactor) allows power transfer.

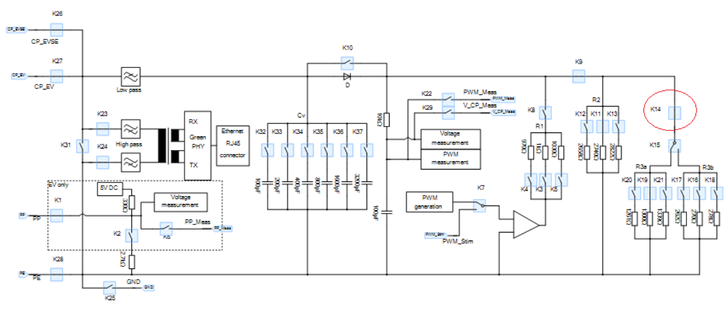


Figure 3.21: *VT7870 with interactive switch.*

The diagram above represents how the VT7970 board is structured: circled in red is the K14 switch, which is responsible for initiating the power transfer. The board provides for the measurement of voltage and PWM (Pulse-Width Modulation) which allows the graphs shown above to be displayed. Measurements for the power transfer between the two converters were taken using an isolated voltage probe (Max 2000V peak) and a current clamp (max 150A). At this setup level, the insulation monitoring system (which will be present later in the charging lab) and the DC-link (which will be present in a real vehicle) are both absent. The following image shows the result of the measurements:

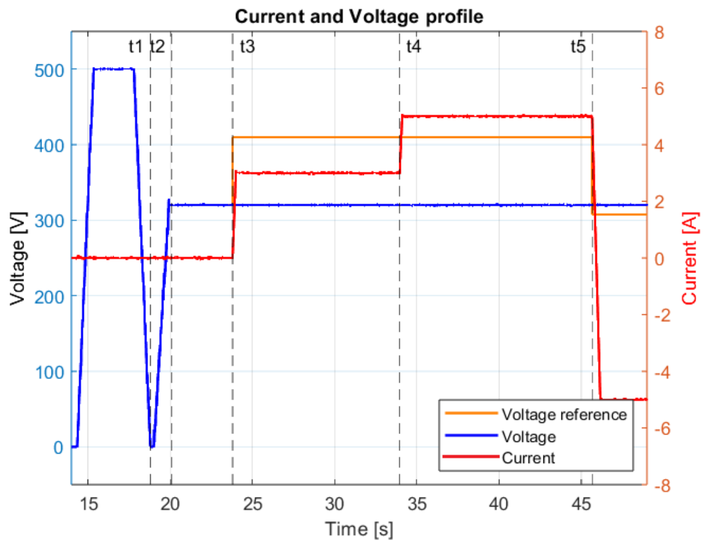


Figure 3.22: *Voltage and Current profile of a charging session.*

The first 10 seconds of the charging session are not shown because they are not very interesting from the perspective of the electrical parameters. The first point of interest is at second 14.4, where the Cable Check is performed and the voltage is brought to 500 V. Due to the lack of an insulation measurement system in the current setup, after a 3500 ms timer the CableCheck status is switched to 1, ending the measurement with a valid state. Next, the system moves to the Precharge phase, where the battery voltage is read (emulator with Itech) and the converter is set to the EVTargetVoltage. Once the Precharge is completed in t2, before transitioning to t3, the PowerDelivery phase occurs, which concludes with the closure of the vehicle's contactor. At second 23.8, the system begins power transfer and the first defined current value is 3 A. At second 33.5, a new setpoint is sent and the system moves to 5 A. When the value is reversed, it goes to the lower dynamic limit and starts discharging at –5 A. It can be observed that the battery voltage remains almost constant: this is reasonable given that the configured capacity (230 Ah) is large compared to the charging current, and therefore over a measurement time of 50 s the voltage does not undergo a significant change. It is also possible to observe the reference voltage, which after the contactor closes at 23.8 s increases to a higher value, equal to the final battery voltage (approximately 410 V). When the current sign reverses, the voltage reference becomes the lower battery voltage limit.

Chapter 4

Work in progress: power quality test for AC Charging Systems

To evaluate the robustness and compliance of the On-Board Charger (OBC) and the Electric Vehicle Supply Equipment (EVSE) with respect to real grid disturbances, specific Power Quality (PQ) tests are commonly performed. These tests aim to reproduce non-ideal grid conditions and to assess the capability of the charging system to maintain stable operation under voltage variations, phase unbalance, and other transient phenomena.

Two of the most representative PQ tests that are typically carried out in AC mode are the *Voltage Level Disturbance Test* and the *Grid Unbalance Test*. Both are described in the framework of international standards, such as IEC 61000-4-11, IEC 61000-4-27, and IEC 61851-21-2, which define the immunity and emission requirements for conductive charging systems. The proposal setup is depicted in the following figure:

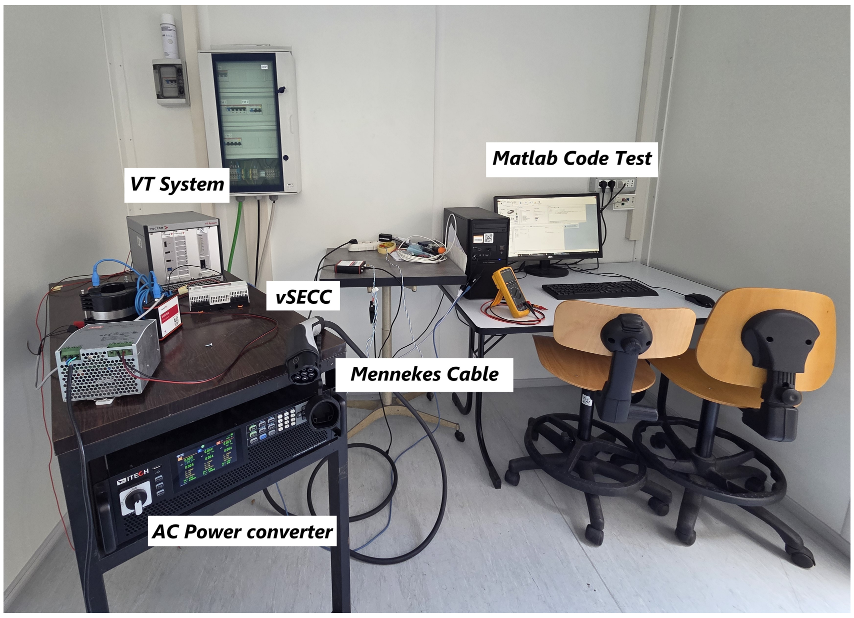


Figure 4.1: *AC est laboratory setup*

The implementation of the power quality tests within the charging lab consists of an initial phase where low-level communication is established between the electric vehicle (EV) and the VSecc, and the system begins operating in the traditional manner. Subsequently, to perform any test where the voltage, frequency, and slew rate of the three phases must change (as defined by the user), a MATLAB script is executed on the lab PC, which is connected to the ITECH power supply via CAN. The two developed codes for the two tests are included in the appendix (Annex C, Annex D). A debugging and communication program called PCANView was used for code development; this program allows sending commands separately and analyzing the response messages.

4.1 Voltage Level Disturbance Test

The voltage disturbance test is designed to verify the OBC's behavior when the supply voltage deviates from its nominal value. In this test, the input voltage is varied within a range of ±30% of the nominal phase-to-phase value, in discrete increments of approximately 15 V per step. For a standard 400 V three-phase system, this corresponds to a voltage sweep between 280 V and 520 V. At each step, the system's response is monitored in terms of:

- charging current and power stability,
- power factor and harmonic distortion,
- proper continuity of the ISO 15118 communication session,
- protection triggering thresholds and fault signaling.

This test allows to determine the limits within which the OBC can safely and efficiently operate, as well as to evaluate whether the EVSE correctly enforces the power derating logic defined in IEC 61851-1. The chosen range of ±30% stems from worst-case scenarios defined in grid immunity standards, such as IEC 61000-4-11, and ensures coverage of both under-voltage (brownout) and over-voltage (temporary surge) events that may occur in weak or unbalanced distribution networks.

The control and automation of the **ITECH IT7915P** programmable power supply were implemented using a **MATLAB** script, leveraging the CAN toolbox for direct communication management. This software is the core element for the automatic and repeatable execution of *voltage disturbance* tests on the Device Under Test (DUT).

Communication Architecture and Initialization

The interaction between the computing environment and the power supply is established via a **CAN 2.0B** bus, configured at a *bitrate* of **250 kbit/s** and managed by a **PEAK-System** hardware interface. The initialization sequence ensures the correct operational setup of the instrument:

- **Channel Setup:** The CAN channel is initialized and started, establishing physical connectivity.
- **Operational Configuration:** The power supply is set to the **three-phase mode** and the **phase balance** feature is enabled via command messages with *ID* `0x601`.

2. Data Encoding Details and Slew Rate Control

The control precision is contingent upon the correct encoding of numerical parameters (voltage and *slew rate*) within the 8-byte CAN message *payload.* The transmission mechanism emulates a simplified **SDO Write** approach and is critical for managing non-discrete values:

- **32-bit Float Transmission:** Physical values are treated as **single-precision floating-point** numbers.
- **Endianness Management:** The combined use of the MATLAB functions `typecast` and `swapbytes` is essential. `typecast` converts the *float* value into a raw 4-byte sequence. Subsequently, `swapbytes` reverses the byte order to comply with the required *Endianness* of the ITECH's electronics, as mandated for write commands (ID `0x601`, e.g., `0x23 0x03 0x30 0x02` for RMS voltage).

The **Voltage Slew Rate**, a critical parameter for dynamic control, is acquired from user input and set via a CAN command, defining the **rate of transition** between different voltage levels during the test.

3. Test Execution and Voltage Disturbance Cycle

The main test is implemented through an iterative `while` loop that automates the application of progressive *voltage sags* and *swells*:

- **Voltage Ramp:** The phase *RMS* voltage, initially set at $\approx 0.7 \times V_n$, is incrementally increased in fixed steps up to $\approx 1.3 \times V_n$ relative to the nominal voltage $V_n = 230\ V$.
- **Command and Validation:** At each *step*, a new write message transmits the new voltage value. The system then awaits a confirmation response (ID `0x581` / 1409 decimal) to validate the successful parameter setting.
- **Dwell Time:** A **5-second** pause is introduced after each variation to ensure the DUT is subjected to the disturbance state for a sufficient time interval for data acquisition and stabilization analysis.

4. Controlled Termination

The conclusion of the test involves a rigorous *cleanup* sequence to return the hardware to a safe state:

- The power output is **disabled**.
- The power supply is reverted to the **single-phase mode** and phase voltages are explicitly set to **0 V** as a safety measure.
- Finally, the CAN communication is terminated (`stop(canch)`), releasing all system resources.

This control methodology ensures the required **parametric accuracy** and **procedural reliability** for academic experimental trials.

4.2 Grid Unbalance Test

The grid unbalance test is intended to evaluate the system performance when the three-phase network loses symmetry in either magnitude or phase angle. During this test, the amplitude of one or more phases is deliberately reduced, while the others remain at their nominal level. The unbalance is gradually increased from 5% up to 50% in incremental steps, in accordance with IEC 61000-4-27 recommendations. Mathematically, voltage unbalance can be expressed through the negative-sequence voltage component V_2 as:

$$U_{\mathrm{unb}} = \frac{V_2}{V_1} \times 100 \ [\%]$$

where V_1 is the positive-sequence voltage and V_2 the negative-sequence component obtained from symmetrical component analysis. High values of U_{unb} can lead to current asymmetry, increased harmonic distortion, and thermal stress in the OBC input stage. The main objectives of this test are:

- to verify whether the OBC maintains correct charging operation or transitions to a controlled stop when unbalance exceeds a defined threshold,

- to measure phase current symmetry and monitor power factor degradation,
- to confirm that no unsafe thermal or electrical conditions occur within the EVSE.

4.2.1 Communication Architecture and Initialization

The interface between the computing environment and the ITECH IT7900P power converter is established via a CAN 2.0B bus, configured at a bitrate of 250 kbit/s and managed by a PEAK-System hardware interface. The initialization sequence ensures that the converter is properly configured for three-phase operation:

- **Channel Setup:** The CAN channel is initialized (`canChannel`) and started (`start(canch)`), establishing physical connectivity. A confirmation message from the DUT (Device Under Test) is awaited to ensure proper communication.
- **Operational Configuration:** The converter is set to three-phase mode by sending specific command messages (ID 0x601), and the phase balancing feature is disabled to allow independent control of each phase voltage. The initial RMS voltage of all three phases (Va, Vb, Vc) is set to the nominal value $V_n = 230\,\text{V}$.

4.2.2 Data Encoding and Transmission Protocol

Precise control over the voltage requires correct encoding of floating-point numerical values within the 8-byte CAN message payload. This ensures that the converter interprets the intended physical values accurately:

- **32-bit Float Transmission:** Phase voltages are represented as single-precision floating-point numbers (`single`).

- **Endianness Management:** MATLAB functions `typecast` and `swapbytes` are used to convert the float into a 4-byte sequence and reorder the bytes to match the Endianness required by the ITECH electronics. Commands such as `0x23 0x03 0x30 0x02` are used to write the RMS voltage to a specific phase.

4.2.3 Test Execution – Voltage Unbalance Cycle

The core test involves a controlled unbalancing of phase Vc while monitoring the corresponding voltages on all three phases. This is implemented using iterative `while` loops:

- **Positive Unbalance Cycle:** Starting from V_n, Vc is incrementally increased in steps of 5% of the nominal voltage until reaching 150% of V_n. At each step, the new voltage is transmitted, and the system awaits a confirmation response (ID 0x581 / 1409 decimal) to ensure successful parameter setting. The RMS voltages of Va, Vb, and Vc are read back and stored for later analysis.
- **Negative Unbalance Cycle:** After resetting Vc to the nominal voltage, the voltage is decreased stepwise to 50% of V_n. Measurements of all three phases are again acquired and recorded at each step.
- **Dwell Time:** A short pause ($\approx$2s) is introduced after each voltage adjustment to allow the DUT to stabilize and ensure reliable voltage readings.

4.2.4 Controlled Termination

The test concludes with a sequence to safely return the hardware to a secure state:

- **Output Disable:** The converter output is deactivated by sending the corresponding CAN command and verifying the acknowledgment message.

- **Single-Phase Reversion:** The device is switched back to single-phase mode, automatically setting all phase voltages to 0 V for safety.
- **Communication Shutdown:** CAN communication is terminated (`stop(canch)`), releasing system resources and confirming the channel is no longer active.

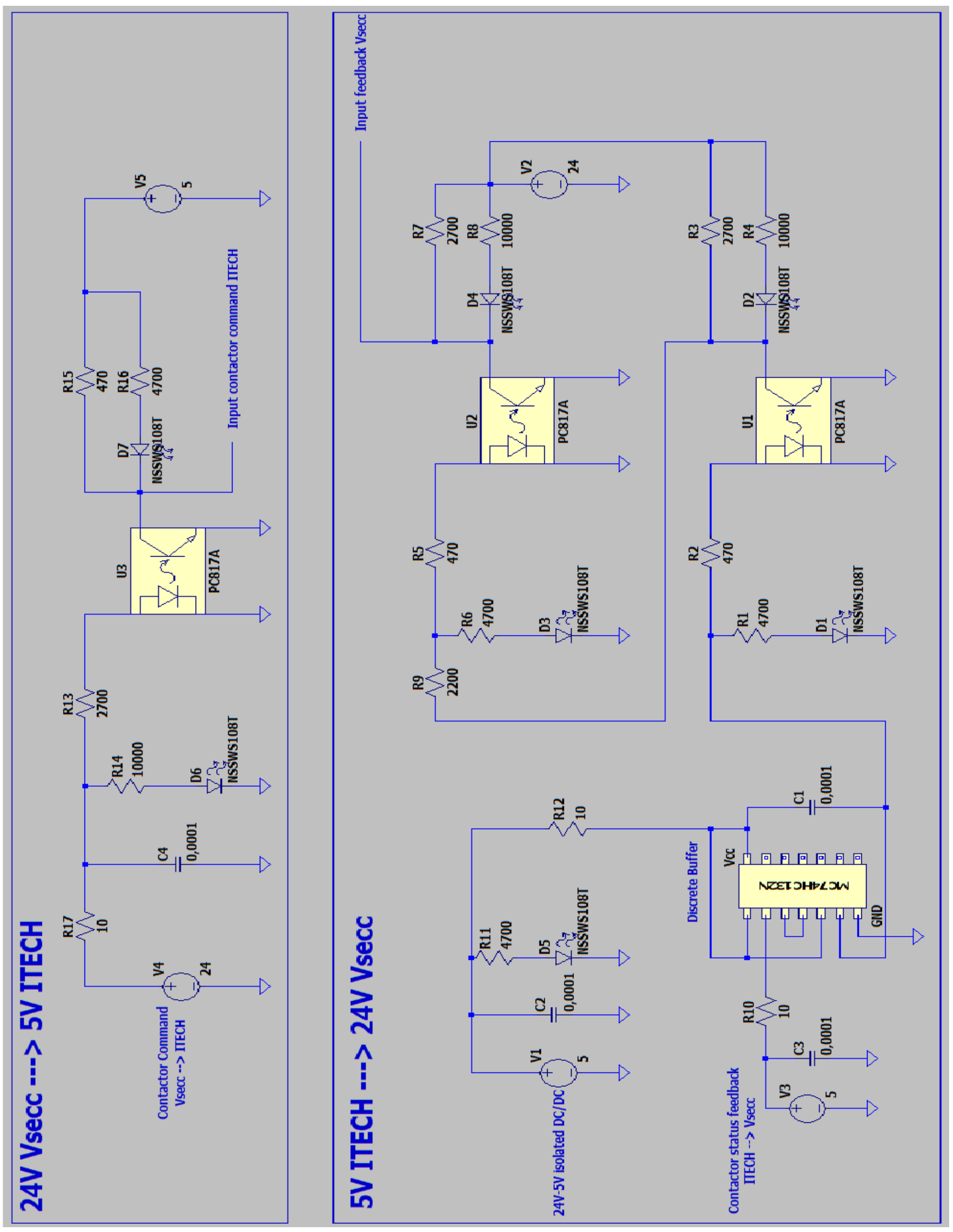
The behavior with a 50% variation is very rare, but necessary to perform the tests in their entirety. The ability to independently control each phase ensures a more accurate emulation of the grid and allows observation of how the onboard charger or any wallbox under test responds to a grid disturbance.

Chapter 5

Conclusion and future work

The objective of this thesis was to develop an experimental setup for bidirectional charging using the high-level communication protocol specified by ISO 15118-20. The measurements performed on both the power profiles and the basic signalling demonstrate that it is possible to integrate the Vector vSECC system with a commercial ITECH converter, instead of the custom converters typically used in real DC charging stations. The integration of the setup through CANoe Pro Network Configuration, together with the development of the PECC using the CAPL language, proved to be essential both for the development of the system itself and for vehicle testing activities. The ability to visualize the protocol and to rely on a software-based PECC makes the system extremely flexible with respect to the vehicle and the specific test procedures that a customer or industrial partner intends to carry out. The main focus of the present work concerns the interaction between the vehicle and the charging station; an interesting direction for future development is the implementation of a software module acting as a CSMS and capable of sending setpoints to the vSECC via OCPP 2.0.1. This additional step would make the system fully integrated with the grid, not only from the power-flow perspective but also with regard to protocol-level communication. The work in progress also presents promising developments: grid emulation remains a constant necessity for manufacturers of electric vehicles and wallboxes, because despite the continuous improvement in energy-quality standards, disturbances and unbalances may still occur in real networks. In the near future, through an industrial collaboration, we will have the opportunity to test both setups with an industrial partner and support companies in an area where the protocol has not yet officially entered into force.

Annex A – LTSpice AC control board



Annex B – CAPL Code for vSECC and IT6642C integration

```
/*@!Encoding:65001*/
variables {
  // ITECH Messages
  message Output_ON itechStart_gen;
  message Output_ON itechStart_load;
  // ITECH_GEN Messages
  message SDO_Client_Generator itech_gen_CCMode;
  message SDO_Client_Generator itech_gen_CVMode;
  message SDO_Client_Generator itech_gen_Vhigh_lim;
  message SDO_Client_Generator itech_gen_Vlow_lim;
  message SDO_Client_Generator itech_gen_Phigh_lim;
  message SDO_Client_Generator itech_gen_Plow_lim;
  message SDO_Client_Generator itech_gen_SetIDC;
  message SDO_Client_Generator itech_gen_MeasureIDC;
  message SDO_Client_Generator itech_gen_SetVDC;
  message SDO_Client_Generator itech_gen_MeasureVDC;
  message SDO_Client_Generator itech_gen_SetV_RiseSlope;
  message SDO_Client_Generator itech_gen_SetV_FallSlope;
  message SDO_Client_Generator itech_gen_SetOutput;
  message SDO_Client_Generator itech_gen_GetOutput;
  message SDO_Client_Generator itech_gen_SetOutputTimer;
  message SDO_Client_Generator itech_gen_SetOutputTimerDuration;

  // ITECH_LOAD Messages
```

```
message SDO_Client_Load itech_load_CCMode;
message SDO_Client_Load itech_load_LoadMode;
message SDO_Client_Load itech_load_BatteryMode;
message SDO_Client_Load itech_load_Capacity;
message SDO_Client_Load itech_load_Setinitmode;
message SDO_Client_Load itech_load_SOCinit;
message SDO_Client_Load itech_load_Vfull;
message SDO_Client_Load itech_load_Imax_lim;
message SDO_Client_Load itech_load_Imin_lim;

// VSECC Messages
message PECCLimits1 PECCLim1;
message PECCLimits2 PECCLim2;
message PECCLimits3 PECCLim3;
message PECCStatus1 PECCStat1;
message PECCStatus2 PECCStat2;

// VSECC VehicleStatus Signals Variables
float batteryStateOfChargeValue;
float cableCheckVoltageValue;
float chargingStateValue;
float evConnectionStateValue;
float targetContactorsStatusValue;
float targetCurrentValue;
float targetVoltageValue;

// VSECC PECCStatus1 Signals Variables
float contactorsStatusValue;
float drivenCurrentValue;
float drivenVoltageValue;
float isolationStatusValue;
float operationalStatusValue;
float temperatureValue;

// VSECC PECCLimits1 Signals Variables
float limitPowerMaxValue;
float limitPowerMinValue;
float limitVoltageMaxValue;
float limitVoltageMinValue;

// VSECC PECCLimits2 Signals Variables
float limitCurrentMaxValue;
float limitCurrentMinValue;

// VSECC PECCLimits3 Signals Variables
float limitDischargeCurrentMin;
float limitDischargeCurrentMax;
float limitDischargePowerMin;
float limitDischargePowerMax;

// ITECH Battery Parameters
float SOC_init = 20; //[%]
float Capacity = 230000; //[Wh]

// ITECH Generator Limits (EVSE Limits)
float P_max_EVSE = 10000; //W
float P_min_EVSE = -10000; //W
float I_max_EVSE = 5; //A
float I_min_EVSE = -5; //A
float V_max_EVSE = 500; //V
float V_min_EVSE = 0; //V

// ITECH Battery Limits (EV Limits)
float P_max_EV = 5000; //W (sysvar::EV::SysEV_MaxPower)
float V_max_EV = 500; //V
float I_max_EV = 16; //A
float I_min_EV = -16; //A

// Controllo tensione di target
float oldTargetVoltageValue = 0;
float measuredVoltageValue;
float measuredCurrentValue;

// Timer messaggi PECC --> SECC
```

```
  msTimer periodicPECCOutput;

  // Timer sblocco CableCheck
  msTimer CableCheckTimer;
  // Timer disabilita output finito il CableCheck
  msTimer CableCheckOutputTimer;
  float controllo = 0;
}

on stopMeasurement {
    // ITECH_GEN Stop Comunication
    itechStart_gen.byte(0) = 0x02;
    itechStart_gen.byte(1) = 0x01;
    itechStart_gen.dlc = 2;
    output(itechStart_gen);

    // ITECH_LOAD Stop Comunication
    itechStart_load.byte(0) = 0x02;
    itechStart_load.byte(1) = 0x02;
    itechStart_load.dlc = 2;
    output(itechStart_load);

    // ITECH Disable Output
    itech_gen_SetOutput.CmdSpecifier = 0x23;
    itech_gen_SetOutput.Index_LSB = 0x01;
    itech_gen_SetOutput.Index_MSB = 0x30;
    itech_gen_SetOutput.SubIndex  = 0x01;
    itech_gen_SetOutput.byte(4) = 0x00;
    itech_gen_SetOutput.dlc = 8;
    output(itech_gen_SetOutput);
    // ITECH_GEN Set IDC
    itech_gen_SetIDC.CmdSpecifier = 0x23;
    itech_gen_SetIDC.Index_LSB = 0x02;
    itech_gen_SetIDC.Index_MSB = 0x30;
    itech_gen_SetIDC.SubIndex  = 0x05;
    itech_gen_SetIDC.Value_Set = 0;
    itech_gen_SetIDC.dlc = 8;
    output(itech_gen_SetIDC);
}

on start{
  write("Bridge VSECC - ITECH avviato");

    // ITECH_GEN Start Comunication
    itechStart_gen.byte(0) = 0x01;
    itechStart_gen.byte(1) = 0x01;
    itechStart_gen.dlc = 2;
    output(itechStart_gen);

    // ITECH_LOAD Start Comunication
    itechStart_load.byte(0) = 0x01;
    itechStart_load.byte(1) = 0x02;
    itechStart_load.dlc = 2;
    output(itechStart_load);

    // ITECH_GEN Set CV Mode
    itech_gen_CVMode.CmdSpecifier = 0x23;
    itech_gen_CVMode.Index_LSB = 0x02;
    itech_gen_CVMode.Index_MSB = 0x30;
    itech_gen_CVMode.SubIndex  = 0x01;
    itech_gen_CVMode.byte(4) = 0x00;
    itech_gen_CVMode.dlc = 8;
    output(itech_gen_CVMode);

    // ITECH_LOAD Set CC Mode
    itech_load_CCMode.CmdSpecifier = 0x23;
    itech_load_CCMode.Index_LSB = 0x02;
    itech_load_CCMode.Index_MSB = 0x40;
    itech_load_CCMode.SubIndex  = 0x01;
    itech_load_CCMode.byte(4) = 0x01;
    itech_load_CCMode.dlc = 8;
    output(itech_load_CCMode);

    // ITECH_LOAD Set Load Mode
```

```
itech_load_LoadMode.CmdSpecifier = 0x23;
itech_load_LoadMode.Index_LSB = 0x02;
itech_load_LoadMode.Index_MSB = 0x40;
itech_load_LoadMode.SubIndex  = 0x15;
itech_load_LoadMode.byte(4) = 0x01;
itech_load_LoadMode.dlc = 8;
output(itech_load_LoadMode);

// ITECH_LOAD Set Battery Mode
itech_load_BatteryMode.CmdSpecifier = 0x23;
itech_load_BatteryMode.Index_LSB = 0x02;
itech_load_BatteryMode.Index_MSB = 0x40;
itech_load_BatteryMode.SubIndex  = 0x16;
itech_load_BatteryMode.byte(4) = 0x01;
itech_load_BatteryMode.dlc = 8;
output(itech_load_BatteryMode);

// ITECH_LOAD Set Battery Parameters
//SET Capacity
itech_load_SOCinit.CmdSpecifier = 0x23;
itech_load_SOCinit.Index_LSB = 0x04;
itech_load_SOCinit.Index_MSB = 0x40;
itech_load_SOCinit.SubIndex  = 0x06;
itech_load_SOCinit.Value_Set = Capacity/10;
itech_load_SOCinit.dlc = 8;
output(itech_load_SOCinit);
//SET initial mode = SOC
itech_load_Setinitmode.CmdSpecifier = 0x23;
itech_load_Setinitmode.Index_LSB = 0x04;
itech_load_Setinitmode.Index_MSB = 0x40;
itech_load_Setinitmode.SubIndex  = 0x0C;
itech_load_Setinitmode.byte(4) = 0x00;
itech_load_Setinitmode.dlc = 8;
output(itech_load_Setinitmode);
//SET initial SOC
itech_load_SOCinit.CmdSpecifier = 0x23;
itech_load_SOCinit.Index_LSB = 0x04;
```

```
    itech_load_SOCinit.Index_MSB = 0x40;
    itech_load_SOCinit.SubIndex  = 0x0D;
    itech_load_SOCinit.Value_Set = SOC_init/10;
    itech_load_SOCinit.dlc = 8;
    output(itech_load_SOCinit);

    // ITECH_LOAD Set EV Limits
    //SET I max
    itech_load_Imax_lim.CmdSpecifier = 0x23;
    itech_load_Imax_lim.Index_LSB = 0x04;
    itech_load_Imax_lim.Index_MSB = 0x40;
    itech_load_Imax_lim.SubIndex  = 0x07;
    itech_load_Imax_lim.Value_Set = I_max_EV/10;
    itech_load_Imax_lim.dlc = 8;
    output(itech_load_Imax_lim);
    //SET I min
    itech_load_Imin_lim.CmdSpecifier = 0x23;
    itech_load_Imin_lim.Index_LSB = 0x04;
    itech_load_Imin_lim.Index_MSB = 0x40;
    itech_load_Imin_lim.SubIndex  = 0x08;
    itech_load_Imin_lim.Value_Set = I_min_EV/10;
    itech_load_Imin_lim.dlc = 8;
    output(itech_load_Imin_lim);

    // VSECC Set EVSE Limits
    PECCLim1.limitVoltageMax = V_max_EVSE*10;
    PECCLim1.limitVoltageMin = V_min_EVSE*10;
    PECCLim1.limitPowerMax = P_max_EVSE/10;
    PECCLim1.limitPowerMin = P_min_EVSE/10;
    PECCLim2.limitCurrentMax = I_max_EVSE*10;
    PECCLim2.limitCurrentMin = I_min_EVSE*10;


    // Avvia il timer da 250 ms
    setTimer(periodicPECCOutput, 250);
}
```

```
on timer periodicPECCOutput {
  // VSECC Communication
  output(PECCLim1);
  output(PECCLim2);
  output(PECCLim3);
  output(PECCStat1);
  output(PECCStat2);

  // Riavvio del timer per il prossimo ciclo
  setTimer(periodicPECCOutput, 250);
}

on timer CableCheckTimer {
  // Sblocco CableCheck (controllo isolation=valid)
  per attivare il PreCharge
  PECCStat1.isolationStatus=1;
  output(PECCStat1);
}

on timer CableCheckOutputTimer{
  // ITECH_GEN Enable Output Timer
  itech_gen_SetOutputTimer.CmdSpecifier = 0x23;
  itech_gen_SetOutputTimer.Index_LSB = 0x01;
  itech_gen_SetOutputTimer.Index_MSB = 0x30;
  itech_gen_SetOutputTimer.SubIndex  = 0x04;
  itech_gen_SetOutputTimer.byte(4) = 0x00;
  itech_gen_SetOutputTimer.dlc = 8;
  output(itech_gen_SetOutput);

  // Avvia il timer per forzare lo sblocco Cable Check
  setTimer(CableCheckOutputTimer, 2500);
}



on message VehicleStatus {
```

```
// VSECC Read VehicleStatus Signals
batteryStateOfChargeValue = this.batteryStateOfCharge;
cableCheckVoltageValue = this.cableCheckVoltage;
chargingStateValue = this.chargingState;
evConnectionStateValue = this.evConnectionState;
targetContactorsStatusValue = this.targetContactorsStatus;
targetCurrentValue = this.targetCurrent;
targetVoltageValue = this.targetVoltage;

// se targetContactorsStatus=closed
&& evConnectionState=e_transfer_allowed
&& chargingState=cableCheck
if (targetContactorsStatusValue==1
&& evConnectionStateValue==2
&& chargingStateValue==1){
  write("first iteration");
  controllo = controllo+1;
  if (controllo<=1){
   write("next interations");
  // Test Isolamento
        // ITECH_GEN Set VDC slope
        (pre-impostare modalità slope = time)
        itech_gen_SetV_RiseSlope.CmdSpecifier = 0x23;
        itech_gen_SetV_RiseSlope.Index_LSB = 0x02;
        itech_gen_SetV_RiseSlope.Index_MSB = 0x30;
        itech_gen_SetV_RiseSlope.SubIndex  = 0x03;
        itech_gen_SetV_RiseSlope.Value_Set = 1.5;//[s]
        itech_gen_SetV_RiseSlope.dlc = 8;
        output(itech_gen_SetV_RiseSlope);
        // ITECH_GEN Set VDC fall
        itech_gen_SetV_FallSlope.CmdSpecifier = 0x23;
        itech_gen_SetV_FallSlope.Index_LSB = 0x02;
        itech_gen_SetV_FallSlope.Index_MSB = 0x30;
        itech_gen_SetV_FallSlope.SubIndex  = 0x04;
        itech_gen_SetV_FallSlope.Value_Set = 1.5;//[s]
        itech_gen_SetV_FallSlope.dlc = 8;
        output(itech_gen_SetV_FallSlope);
        // ITECH_GEN Set VDC
        itech_gen_SetVDC.CmdSpecifier = 0x23;
        itech_gen_SetVDC.Index_LSB = 0x02;
        itech_gen_SetVDC.Index_MSB = 0x30;
        itech_gen_SetVDC.SubIndex  = 0x02;
        itech_gen_SetVDC.Value_Set = V_max_EVSE;
        itech_gen_SetVDC.dlc = 8;
        output(itech_gen_SetVDC);
        // ITECH_GEN Enable Output Timer
        itech_gen_SetOutputTimer.CmdSpecifier = 0x23;
        itech_gen_SetOutputTimer.Index_LSB = 0x01;
        itech_gen_SetOutputTimer.Index_MSB = 0x30;
        itech_gen_SetOutputTimer.SubIndex  = 0x04;
        itech_gen_SetOutputTimer.byte(4) = 0x00;
        itech_gen_SetOutputTimer.dlc = 8;
        output(itech_gen_SetOutput);
        // ITECH_GEN Set Output Timer Duration
        itech_gen_SetOutputTimerDuration.CmdSpecifier = 0x23;
        itech_gen_SetOutputTimerDuration.Index_LSB = 0x01;
        itech_gen_SetOutputTimerDuration.Index_MSB = 0x30;
        itech_gen_SetOutputTimerDuration.SubIndex  = 0x05;
        itech_gen_SetOutputTimerDuration.Value_Set = 5; //[s]
        itech_gen_SetOutputTimerDuration.dlc = 8;
        output(itech_gen_SetOutputTimerDuration);
         ITECH_GEN Enable Output
        itech_gen_SetOutput.CmdSpecifier = 0x23;
        itech_gen_SetOutput.Index_LSB = 0x01;
        itech_gen_SetOutput.Index_MSB = 0x30;
        itech_gen_SetOutput.SubIndex  = 0x01;
        itech_gen_SetOutput.byte(4) = 0x01;
        itech_gen_SetOutput.dlc = 8;
        output(itech_gen_SetOutput);

        // Avvia il timer per togliere l'output per
        lo sblocco Cable Check
        setTimer(CableCheckOutputTimer, 5000);
}
```

```
}


// se chargingState=preCharge
if (chargingStateValue==2){
  // ITECH_GEN Set VDC slope
  itech_gen_SetV_RiseSlope.CmdSpecifier = 0x23;
  itech_gen_SetV_RiseSlope.Index_LSB = 0x02;
  itech_gen_SetV_RiseSlope.Index_MSB = 0x30;
  itech_gen_SetV_RiseSlope.SubIndex  = 0x03;
  itech_gen_SetV_RiseSlope.Value_Set =
  (targetVoltageValue/10)/1000;//[V/ms]
  itech_gen_SetV_RiseSlope.dlc = 8;
  output(itech_gen_SetV_RiseSlope);
  // ITECH_GEN Set VDC fall
  itech_gen_SetV_FallSlope.CmdSpecifier = 0x23;
  itech_gen_SetV_FallSlope.Index_LSB = 0x02;
  itech_gen_SetV_FallSlope.Index_MSB = 0x30;
  itech_gen_SetV_FallSlope.SubIndex  = 0x04;
  itech_gen_SetV_FallSlope.Value_Set =
  (targetVoltageValue/10)/1000;//[V/ms]
  itech_gen_SetV_FallSlope.dlc = 8;
  output(itech_gen_SetV_FallSlope);
  // ITECH_GEN Set VDC
  itech_gen_SetVDC.CmdSpecifier = 0x23;
  itech_gen_SetVDC.Index_LSB = 0x02;
  itech_gen_SetVDC.Index_MSB = 0x30;
  itech_gen_SetVDC.SubIndex  = 0x05;
  itech_gen_SetVDC.Value_Set =
  targetVoltageValue/10;
  itech_gen_SetVDC.dlc = 8;
  output(itech_gen_SetVDC);
  // ITECH_GEN Enable Output
  itech_gen_SetOutput.CmdSpecifier = 0x23;
  itech_gen_SetOutput.Index_LSB = 0x01;
  itech_gen_SetOutput.Index_MSB = 0x30;
  itech_gen_SetOutput.SubIndex  = 0x01;
```

```
    itech_gen_SetOutput.byte(4) = 0x01;
    itech_gen_SetOutput.dlc = 8;
    output(itech_gen_SetOutput);
    // ITECH_GEN Measure VDC
    itech_gen_MeasureVDC.CmdSpecifier = 0x43;
    itech_gen_MeasureVDC.Index_LSB = 0x03;
    itech_gen_MeasureVDC.Index_MSB = 0x30;
    itech_gen_MeasureVDC.SubIndex  = 0x01;
    itech_gen_MeasureVDC.dlc = 8;
    output(itech_gen_MeasureVDC);
  }


// se chargingState=charge
  if (chargingStateValue==3){
    // ITECH_LOAD Set V @batteryVoc
    itech_load_Vfull.CmdSpecifier = 0x23;
    itech_load_Vfull.Index_LSB = 0x04;
    itech_load_Vfull.Index_MSB = 0x30;
    itech_load_Vfull.SubIndex  = 0x01;
    itech_load_Vfull.Value_Set = targetVoltageValue/10;
    itech_load_Vfull.dlc = 8;
    output(itech_load_Vfull);

    // ITECH_GEN Set CC Mode
    itech_gen_CCMode.CmdSpecifier = 0x23;
    itech_gen_CCMode.Index_LSB = 0x02;
    itech_gen_CCMode.Index_MSB = 0x30;
    itech_gen_CCMode.SubIndex  = 0x01;
    itech_gen_CCMode.byte(4) = 0x01;
    itech_gen_CCMode.dlc = 8;
    output(itech_gen_CCMode);

    // ITECH_GEN Set EVSE Limits
    //SET V high
    itech_gen_Vhigh_lim.CmdSpecifier = 0x23;
    itech_gen_Vhigh_lim.Index_LSB = 0x02;
    itech_gen_Vhigh_lim.Index_MSB = 0x30;
    itech_gen_Vhigh_lim.SubIndex  = 0x08;
    itech_gen_Vhigh_lim.Value_Set = V_max_EV/10;
    itech_gen_Vhigh_lim.dlc = 8;
    output(itech_gen_Vhigh_lim);
    //SET V low
    itech_gen_Vlow_lim.CmdSpecifier = 0x23;
    itech_gen_Vlow_lim.Index_LSB = 0x02;
    itech_gen_Vlow_lim.Index_MSB = 0x30;
    itech_gen_Vlow_lim.SubIndex  = 0x09;
    itech_gen_Vlow_lim.Value_Set = V_min_EV/10;
    itech_gen_Vlow_lim.dlc = 8;
    output(itech_gen_Vlow_lim);
    //SET P high
    itech_gen_Phigh_lim.CmdSpecifier = 0x23;
    itech_gen_Phigh_lim.Index_LSB = 0x02;
    itech_gen_Phigh_lim.Index_MSB = 0x30;
    itech_gen_Phigh_lim.SubIndex  = 0x0E;
    itech_gen_Phigh_lim.Value_Set = P_max_EVSE/10;
    itech_gen_Phigh_lim.dlc = 8;
    output(itech_gen_Phigh_lim);
    //SET P low
    itech_gen_Plow_lim.CmdSpecifier = 0x23;
    itech_gen_Plow_lim.Index_LSB = 0x02;
    itech_gen_Plow_lim.Index_MSB = 0x30;
    itech_gen_Plow_lim.SubIndex  = 0x0F;
    itech_gen_Plow_lim.Value_Set = P_min_EVSE/10;
    itech_gen_Plow_lim.dlc = 8;
    output(itech_gen_Plow_lim);
  }


    if (targetCurrentValue=!0){
    // ITECH_GEN Set IDC
    itech_gen_SetIDC.CmdSpecifier = 0x23;
    itech_gen_SetIDC.Index_LSB = 0x02;
    itech_gen_SetIDC.Index_MSB = 0x30;
```

```
    itech_gen_SetIDC.SubIndex  = 0x05;
    itech_gen_SetIDC.Value_Set = targetCurrentValue/10;
    itech_gen_SetIDC.dlc = 8;
    output(itech_gen_SetIDC);
    // ITECH Measure IDC
    itech_gen_MeasureIDC.CmdSpecifier = 0x43;
    itech_gen_MeasureIDC.Index_LSB = 0x03;
    itech_gen_MeasureIDC.Index_MSB = 0x30;
    itech_gen_MeasureIDC.SubIndex  = 0x02;
    itech_gen_MeasureIDC.dlc = 8;
    output(itech_gen_MeasureIDC);
    }


   if (chargingStateValue==4 || evConnectionStateValue==3){
      // ITECH Disable Output
      itech_gen_SetOutput.CmdSpecifier = 0x23;
      itech_gen_SetOutput.Index_LSB = 0x01;
      itech_gen_SetOutput.Index_MSB = 0x30;
      itech_gen_SetOutput.SubIndex  = 0x01;
      itech_gen_SetOutput.byte(4) = 0x00;
      itech_gen_SetOutput.dlc = 8;
      output(itech_gen_SetOutput);
      // ITECH_GEN Set IDC
      itech_gen_SetIDC.CmdSpecifier = 0x23;
      itech_gen_SetIDC.Index_LSB = 0x02;
      itech_gen_SetIDC.Index_MSB = 0x30;
      itech_gen_SetIDC.SubIndex  = 0x05;
      itech_gen_SetIDC.Value_Set = 0;
      itech_gen_SetIDC.dlc = 8;
      output(itech_gen_SetIDC);
    }
}

on message RespMessage_Generator{
   if (this.CmdSpecifier == 0x43 && this.SubIndex == 0x01){
    // VSECC Receive Measured VDC
```

```
     measuredVoltageValue = this.Value_Resp;
     PECCStat2.measuredVoltage = measuredVoltageValue*10;
     output(PECCStat2);
    }
    if (this.CmdSpecifier == 0x43 && this.SubIndex == 0x02){
     // VSECC Receive Measured IDC
     measuredCurrentValue = this.Value_Resp;
     PECCStat2.measuredCurrent = measuredCurrentValue*10;
     output(PECCStat2);
    }
    write("Measured: V = %.2f  I = %.2f", measuredVoltageValue,
    measuredCurrentValue);
}

on message PECCStatus1 {
  // VSECC Read PECCStatus1 Signals
  contactorsStatusValue = this.contactorsStatus;
  drivenCurrentValue = this.drivenCurrent;
  drivenVoltageValue = this.drivenVoltage;
  isolationStatusValue = this.isolationStatus;
  operationalStatusValue = this.operationalStatus;
  temperatureValue = this.temperature;
}

on message PECCLimits1 {
  // VSECC Read PECCLimits1 Signals
  limitPowerMaxValue = this.limitPowerMax;
  limitPowerMinValue = this.limitPowerMin;
  limitVoltageMaxValue = this.limitVoltageMax;
  limitVoltageMinValue = this.limitVoltageMin;
}

on message PECCLimits2 {
  // VSECC Read PECCLimits2 Signals
  limitCurrentMaxValue = this.limitCurrentMax;
  limitCurrentMinValue = this.limitCurrentMin;
}
```

```
on message PECCLimits3 {
  // VSECC Read PECCLimits3 Signals
  limitDischargeCurrentMinValue = this.limitDischargeCurrentMin;
  limitDischargeCurrentMaxValue = this.limitDischargeCurrentMax;
  limitDischargePowerMinValue = this.DischargePowerMin;
  limitDischargePowerMaxValue = this.limitDischargePowerMax;
}
```

Annex C – Matlab Code for Voltage Disturbance Test

```
% va vb vc variano da Vn a +-50%xVn step 15%xVn

clear all

%% INIZIALIZZAZIONE VARIABILI
Vn = 230;
factor = 0.7;
voltage = 0;

%% INIZIALIZZAZIONE DEL CANALE
fprintf('\n');
input('Premi INVIO per inizializzare il canale CAN...');
canch = canChannel('PEAK-System', 'PCAN_USBBUS1');
configBusSpeed(canch, 250000);
fprintf("Il canale è attivo\n")

%% ATTIVAZIONE DEL CANALE CAN
fprintf('\n');
input('Premi INVIO per comunicare con ITECH IT7900P...');
start(canch);
msg = canMessage(0x000,false,2);
msg.Data = [0x01 0x01];
transmit(canch, msg);
```

```
% lettura conferma attivazione canale
pause(1)
msg = receive(canch, Inf, 'OutputFormat','timetable');
if isempty(msg)
    fprintf("Errore di comunicazione con ITECH IT7900P\n")
else
    fprintf("ITECH IT7900P sta comunicando\n");
end

%% IMPOSTAZIONE MODALITA' TRIFASE
fprintf('\n');
input('Premi INVIO per impostare la modalità trifase...');
msg = canMessage(0x601, false, 8);
msg.Data = [0x23 0x01 0x30 0x02 0x01 0x00 0x00 0x00];
transmit(canch, msg)

% lettura conferma modalità trifase
msg = canMessage(0x601,false,8);
msg.Data = [0x43 0x01 0x30 0x02 0x00 0x00 0x00 0x00];
transmit(canch, msg);
pause(1)
msg = receive(canch, Inf, 'OutputFormat','timetable');
filtered = msg(msg.ID == 1409, :);
if isempty(filtered)
    fprintf('Nessun messaggio trovato con ID 581\n');
else
    lastMsg = filtered(end,:);
    data = lastMsg.Data{1};
    firstByte = data(1);
    fifthByte = data(5);
    if firstByte == 67 && fifthByte == 1
        fprintf("Modalità trifase attivata\n")
    else
        fprintf("Problema attivazione modalità trifase\n")
    end
end
```

```
%% set balance mode: ON
fprintf('\n');
input('Premi INVIO per attivare il bilanciamento delle fasi...');
msg = canMessage(0x601, false, 8);
msg.Data = [0x23 0x03 0x30 0x01 0x01 0x00 0x00 0x00];
transmit(canch, msg);

% lettura conferma balance mode on
msg = canMessage(0x601,false,8);
msg.Data = [0x43 0x03 0x30 0x01 0x00 0x00 0x00 0x00];
transmit(canch, msg);
pause(1)
msg = receive(canch, Inf, 'OutputFormat','timetable');
filtered = msg(msg.ID == 1409, :);
if isempty(filtered)
    fprintf('Nessun messaggio trovato con ID 581\n');
else
    lastMsg = filtered(end,:);
    data = lastMsg.Data{1};
    firstByte = data(1);
    fifthByte = data(5);
    if firstByte == 67 && fifthByte == 1
        fprintf("Modalità bilanciamento delle fasi
        attivata correttamente\n");
    else
        fprintf("Problema attivazione bilanciamento delle fasi\n")
    end
end

%% INPUT SLEW RATE TENSIONE
fprintf('\n');
slewRate = input('Inserire un valore di slew rate
per la tensione (0.0001 - 5000V/ms): ');
while  slewRate < 0.0001 || slewRate > 5000
    fprintf('\n');
    slewRate = input('Il valore inserito non rienta nei limiti,
    inserisci un nuovo valore(0.0001-5000V/ms): ');
```

```
end

msg = canMessage(0x601,false,8);
msg.Data = [0x23 0x03 0x30 0x05 (swapbytes(typecast
(single(slewRate),'uint8')))];
transmit(canch, msg);

% lettura conferma impostazione slew rate
msg = canMessage(0x601,false,8);
msg.Data = [0x43 0x03 0x30 0x05 0x00 0x00 0x00 0x00];
transmit(canch, msg);
pause(1)
msg = receive(canch, Inf, 'OutputFormat','timetable');
filtered =msg(msg.ID == 1409, :);
if isempty(filtered)
    fprintf('Nessun messaggio trovato con ID 581\n');
else
    lastMsg = filtered(end,:);
    data = lastMsg.Data{1};
    firstByte = data(1);
    if firstByte == 67
        value = data(end-3:end);
        realValue = typecast(swapbytes(value) , 'single');
        fprintf('Lo slew rate è stato impostato a
        : %.4f% V/ms\n', realValue);
    else
        fprintf("Problema impostazione dello slew rate\n")
    end
end

%% ATTIVAZIONE DELL'OUTPUT
fprintf('\n');
input('Premi INVIO per attivare l output...');
msg = canMessage(0x601, false, 8);
msg.Data = [0x23 0x02 0x30 0x01 0x01 0x00 0x00 0x00];
transmit(canch, msg);
```

```
% lettura conferma attivazione dell'uscita
msg = canMessage(0x601, false, 8);
msg.Data = [0x43 0x02 0x30 0x01 0x00 0x00 0x00 0x00];
transmit(canch, msg);
pause(1)
msg = receive(canch, Inf, 'OutputFormat','timetable');
filtered =msg(msg.ID == 1409, :);
if isempty(filtered)
    fprintf('Nessun messaggio trovato con ID 581\n');
else
    lastMsg = filtered(end,:);
    data = lastMsg.Data{1};
    firstByte = data(1);
    fifthByte = data(5);
    if firstByte == 67 && fifthByte == 1
        fprintf("Output attivo\n");
    else
        fprintf("Problema attivazione dell'output\n");
    end
end

%% VOLTAGE DISTURBANCE TEST
fprintf('\n');
input('Premi INVIO per iniziare il voltage disturbance test...');

while factor <= 1.3

      voltage = Vn * factor;

      % IMPOSTAZIONE DELLA TENSIONE
      msg = canMessage(0x601,false,8);
      msg.Data = [0x23 0x03 0x30 0x02 (swapbytes(typecast
      (single(voltage),'uint8')))];
      transmit(canch, msg);

      % lettura conferma impostazione tensione
      msg = canMessage(0x601,false,8);
      msg.Data = [0x43 0x03 0x30 0x02 0x00 0x00 0x00 0x00];
      transmit(canch, msg);
      pause(1)
      msg = receive(canch, Inf, 'OutputFormat','timetable');
      filtered = msg(msg.ID == 1409, :);
      if isempty(filtered)
         fprintf('Nessun messaggio trovato con ID 581\n');
      else
         lastMsg = filtered(end,:);
         data = lastMsg.Data{1};
         firstByte = data(1);
        if firstByte == 67
           value = data(end-3:end);
           realValue = typecast(swapbytes(value) , 'single');
           fprintf('Tensione: %.0f V\n', realValue)
        else
           fprintf("Problema impostazione della tensione\n")
        end
      end
pause(5)
factor = factor + 0.0652173913;
end

%% TERMINE PROVA
fprintf('\n');
input('Premi INVIO per terminare la prova...');

% DISATTIVAZIONE OUTPUT
msg = canMessage(0x601, false, 8);
msg.Data = [0x23 0x02 0x30 0x01 0x00 0x00 0x00 0x00];
transmit(canch, msg);
pause(1)
% lettura conferma disattivazione dell'uscita
msg = canMessage(0x601, false, 8);
msg.Data = [0x43 0x02 0x30 0x01 0x00 0x00 0x00 0x00];
transmit(canch, msg);
pause(1)
```

```
msg = receive(canch, Inf, 'OutputFormat','timetable');
filtered = msg(msg.ID == 1409, :);
if isempty(filtered)
    fprintf('Nessun messaggio trovato con ID 581\n');
else
    lastMsg = filtered(end,:);
    data = lastMsg.Data{1};
    firstByte = data(1);
    fifthByte = data(5);
    if firstByte == 67 && fifthByte == 0
        fprintf("Output disattivato\n");
    else
        fprintf("Problema disattivazione dell'output\n");
    end
end

% IMPOSTAZIONE MODALITA' MONOFASE (imposta
le tensioni a 0V in automatico)
msg = canMessage(0x601, false, 8);
msg.Data = [0x23 0x01 0x30 0x02 0x00 0x00 0x00 0x00];
transmit(canch, msg);

% lettura conferma modalità monofase
msg = canMessage(0x601,false,8);
msg.Data = [0x43 0x01 0x30 0x02 0x00 0x00 0x00 0x00];
transmit(canch, msg);
pause(2)
msg = receive(canch, Inf, 'OutputFormat','timetable');
filtered =msg(msg.ID == 1409, :);
if isempty(filtered)
    fprintf('Nessun messaggio trovato con ID 581\n');
else
    lastMsg = filtered(end,:);
    data = lastMsg.Data{1};
    firstByte = data(1);
    fifthByte = data(5);
    if firstByte == 67 && fifthByte == 0
        fprintf("Modalità monofase attivata\n")
    else
        fprintf("Problema attivazione modalità monofase\n")
    end
end

 % IMPOSTAZIONE di va vb vc a 0
 msg_va = canMessage(0x601,false,8);
 msg_va.Data = [0x23 0x03 0x30 0x02 0x00 0x00 0x00 0x00];
 transmit(canch, msg_va);
 msg_vb = canMessage(0x601,false,8);
 msg_vb.Data = [0x23 0x03 0x30 0x03 0x00 0x00 0x00 0x00];
 transmit(canch, msg_vb);
 msg_vc = canMessage(0x601,false,8);
 msg_vc.Data = [0x23 0x03 0x30 0x04 0x00 0x00 0x00 0x00];
 transmit(canch, msg_vc);
 DISATTIVAZIONE COMUNICAZIONE ITECH IT7900P

% DISATTIVAZIONE COMUNICAZIONE ITECH IT7900P
stop(canch);
% lettura conferma disattivazione canale
pause(1)
msg = receive(canch, Inf, 'OutputFormat','timetable');
if isempty(msg)
    fprintf("Il canale CAN è spento\n");
else
    fprintf("Errore di comunicazione con ITECH IT7900P\n")
end
```

Annex D – Matlab Code for Voltage Unbalance Test

```
% va vb fisse a Vn ----- vc unbalance +-50%xVn step 5%xVn
clear all

%% INIZIALIZZAZIONE VARIABILI
Vn = 230;
factor = 0.05;
step_voltage = factor*Vn;

%% INIZIALIZZAZIONE DEL CANALE
fprintf('\n');
input('Premi INVIO per inizializzare il canale CAN...');
canch = canChannel('PEAK-System', 'PCAN_USBBUS1');
configBusSpeed(canch, 250000);
fprintf("Il canale è attivo\n")

%% ATTIVAZIONE DEL CANALE CAN
fprintf('\n');
input('Premi INVIO per comunicare con ITECH IT7900P...');
start(canch);

% lettura conferma attivazione canale
pause(2)
msg = receive(canch, Inf, 'OutputFormat','timetable');
if isempty(msg)
```

```
    fprintf("Errore di comunicazione con ITECH IT7900P\n")
else
    fprintf("ITECH IT7900P sta comunicando\n");
end
%% IMPOSTAZIONE MODALITA’ TRIFASE
fprintf(’\n’);
input(’Premi INVIO per impostare la modalità trifase...’);
msg = canMessage(0x601, false, 8);
msg.Data = [0x23 0x01 0x30 0x02 0x01 0x00 0x00 0x00];
transmit(canch, msg)

% lettura conferma modalità trifase
msg = canMessage(0x601,false,8);
msg.Data = [0x43 0x01 0x30 0x02 0x00 0x00 0x00 0x00];
transmit(canch, msg);
pause(2)
msg = receive(canch, Inf, ’OutputFormat’,’timetable’);
filtered = msg(msg.ID == 1409, :);
if isempty(filtered)
    fprintf(’Nessun messaggio trovato con ID 581\n’);
else
    lastMsg = filtered(end,:);
    data = lastMsg.Data{1};
    firstByte = data(1);
    fifthByte = data(5);
    if firstByte == 67 && fifthByte == 1
        fprintf("Modalità trifase attivata\n")
    else
        fprintf("Problema attivazione modalità trifase\n")
    end
end

% IMPOSTAZIONE DI va vb vc a Vn
voltage = Vn;
msg = canMessage(0x601,false,8);
msg.Data = [0x23 0x03 0x30 0x02
(swapbytes(typecast(single(voltage),’uint8’)))];
```

```
transmit(canch, msg);

% lettura conferma impostazione tensione
msg = canMessage(0x601,false,8);
msg.Data = [0x43 0x03 0x30 0x02 0x00 0x00 0x00 0x00];
transmit(canch, msg);
pause(2)
msg = receive(canch, Inf, 'OutputFormat','timetable');
filtered = msg(msg.ID == 1409, :);
if isempty(filtered)
    fprintf('Nessun messaggio trovato con ID 581\n');
else
    lastMsg = filtered(end,:);
    data = lastMsg.Data{1};
    firstByte = data(1);
    if firstByte == 67
        value = data(end-3:end);
        realValue = typecast(swapbytes(value) , 'single');
        fprintf('Va = %.2f V, Vb = %.2f V, Vc = %.2f V\n',
        realValue, realValue, realValue);
    else
        fprintf("Problema impostazione della tensione\n")
     end
end

%% set balance mode: OFF
fprintf('\n');
input('Premi INVIO per disattivare il bilanciamento delle fasi...');
msg = canMessage(0x601, false, 8);
msg.Data = [0x23 0x03 0x30 0x01 0x00 0x00 0x00 0x00];
transmit(canch, msg);

% lettura conferma balance mode off
msg = canMessage(0x601,false,8);
msg.Data = [0x43 0x03 0x30 0x01 0x00 0x00 0x00 0x00];
transmit(canch, msg);
pause(2)
```

```
msg = receive(canch, Inf, 'OutputFormat','timetable');
filtered = msg(msg.ID == 1409, :);
if isempty(filtered)
    fprintf('Nessun messaggio trovato con ID 581\n');
else
    lastMsg = filtered(end,:);
    data = lastMsg.Data{1};
    firstByte = data(1);
    fifthByte = data(5);
    if firstByte == 67 && fifthByte == 0
        fprintf("Modalità bilanciamento delle fasi
        disattivata correttamente\n");
    else
        fprintf("Problema disattivazione bilanciamento delle fasi\n")
    end
end
%% ATTIVAZIONE DELL'OUTPUT
fprintf('\n');
input('Premi INVIO per attivare l output...');
msg = canMessage(0x601, false, 8);
msg.Data = [0x23 0x02 0x30 0x01 0x01 0x00 0x00 0x00];
transmit(canch, msg);

% lettura conferma attivazione dell'uscita
msg = canMessage(0x601, false, 8);
msg.Data = [0x43 0x02 0x30 0x01 0x00 0x00 0x00 0x00];
transmit(canch, msg);
pause(2)
msg = receive(canch, Inf, 'OutputFormat','timetable');
filtered =msg(msg.ID == 1409, :);
if isempty(filtered)
    fprintf('Nessun messaggio trovato con ID 581\n');
else
    lastMsg = filtered(end,:);
    data = lastMsg.Data{1};
    firstByte = data(1);
    fifthByte = data(5);
    if firstByte == 67 && fifthByte == 1
        fprintf("Output attivo\n");
    else
        fprintf("Problema attivazione dell’output\n");
    end
end
%% VOLTAGE UNBALANCE TEST

Va_total = [];
Vb_total = [];
Vc_total = [];


% IMPOSTAZIONE DI vc unbalance +50%
fprintf('\n');
input('Premi INVIO per iniziare il ciclo +50% di unbalance...');
voltage_unb = Vn;
while voltage_unb < 1.5*Vn
     voltage_unb = voltage_unb + step_voltage;
     msg = canMessage(0x601,false,8);
     msg.Data = [0x23 0x03 0x30 0x04
     (swapbytes(typecast(single(voltage_unb),'uint8')))];
     transmit(canch, msg);

     % lettura tensione vc
     msg = canMessage(0x601,false,8);
     msg.Data = [0x43 0x03 0x30 0x04 0x00 0x00 0x00 0x00];
     transmit(canch, msg);
     pause(2)
     msg = receive(canch, Inf, 'OutputFormat','timetable');
     filtered = msg(msg.ID == 1409, :);
     if isempty(filtered)
        disp('Nessun messaggio trovato con ID ',num2str(581));
     else
        lastMsg = filtered(end,:);
        data = lastMsg.Data{1};
        firstByte = data(1);
```

```
   if firstByte == 67
       value = data(end-3:end);
       Vc = typecast(swapbytes(value) , 'single');
   else
       disp("Problema impostazione della tensione")
   end

% lettura tensione va
msg = canMessage(0x601,false,8);
msg.Data = [0x43 0x03 0x30 0x02 0x00 0x00 0x00 0x00];
transmit(canch, msg);
pause(2)
msg = receive(canch, Inf, 'OutputFormat','timetable');
filtered = msg(msg.ID == 1409, :);
if isempty(filtered)
   disp('Nessun messaggio trovato con ID ',num2str(581));
else
   lastMsg = filtered(end,:);
   data = lastMsg.Data{1};
   firstByte = data(1);
   if firstByte == 67
       value = data(end-3:end);
       Va = typecast(swapbytes(value) , 'single');
   else
       disp("Problema impostazione della tensione")
   end
end

% lettura tensione vb
msg = canMessage(0x601,false,8);
msg.Data = [0x43 0x03 0x30 0x03 0x00 0x00 0x00 0x00];
transmit(canch, msg);
pause(2)
msg = receive(canch, Inf, 'OutputFormat','timetable');
filtered = msg(msg.ID == 1409, :);
if isempty(filtered)
   disp('Nessun messaggio trovato con ID ',num2str(581));
```

```
     else
        lastMsg = filtered(end,:);
        data = lastMsg.Data{1};
        firstByte = data(1);
        if firstByte == 67
            value = data(end-3:end);
            Vb = typecast(swapbytes(value) , 'single');
        else
            disp("Problema impostazione della tensione")
        end
     end

     fprintf('[UNBALANCE TEST] Step: %+3.0f%% Vn ->
     Va = %.2f V | Vb = %.2f V | Vc = %.2f V\n',
     (voltage_unb/Vn-1)*100, Va, Vb, Vc);
     end
end

Va_total = [Va_total Va];
Vb_total = [Vb_total Vb];
Vc_total = [Vc_total Vc];

% pausa la somma delle pause in lettura delle tensioni


% Risettaggio tensione a Vn
voltage = Vn;
msg = canMessage(0x601,false,8);
msg.Data = [0x23 0x03 0x30 0x02
(swapbytes(typecast(single(voltage),'uint8')))];
transmit(canch, msg);


% IMPOSTAZIONE DI vc unbalance -50%
voltage_unb = Vn;
fprintf('\n');
input('Premi INVIO per iniziare il ciclo -50% di unbalance...');
```

```
while voltage_unb > 0.5*Vn
     voltage_unb = voltage_unb - step_voltage;
     msg = canMessage(0x601,false,8);
     msg.Data = [0x23 0x03 0x30 0x04
     (swapbytes(typecast(single(voltage_unb),'uint8')))];
     transmit(canch, msg);

      % lettura tensione vc
     msg = canMessage(0x601,false,8);
     msg.Data = [0x43 0x03 0x30 0x04 0x00 0x00 0x00 0x00];
     transmit(canch, msg);
     pause(2)
     msg = receive(canch, Inf, 'OutputFormat','timetable');
     filtered = msg(msg.ID == 1409, :);
     if isempty(filtered)
        disp('Nessun messaggio trovato con ID ',num2str(581));
     else
        lastMsg = filtered(end,:);
        data = lastMsg.Data{1};
        firstByte = data(1);
        if firstByte == 67
            value = data(end-3:end);
            Vc = typecast(swapbytes(value) , 'single');
        else
            disp("Problema impostazione della tensione")
        end

     % lettura tensione va
     msg = canMessage(0x601,false,8);
     msg.Data = [0x43 0x03 0x30 0x02 0x00 0x00 0x00 0x00];
     transmit(canch, msg);
     pause(2)
     msg = receive(canch, Inf, 'OutputFormat','timetable');
     filtered = msg(msg.ID == 1409, :);
     if isempty(filtered)
        disp('Nessun messaggio trovato con ID ',num2str(581));
     else
```

```
            lastMsg = filtered(end,:);
            data = lastMsg.Data{1};
            firstByte = data(1);
            if firstByte == 67
                value = data(end-3:end);
                Va = typecast(swapbytes(value) , 'single');
            else
                disp("Problema impostazione della tensione")
            end
        end

        % lettura tensione vb
        msg = canMessage(0x601,false,8);
        msg.Data = [0x43 0x03 0x30 0x03 0x00 0x00 0x00 0x00];
        transmit(canch, msg);
        pause(2)
        msg = receive(canch, Inf, 'OutputFormat','timetable');
        filtered = msg(msg.ID == 1409, :);
        if isempty(filtered)
            disp('Nessun messaggio trovato con ID ',num2str(581));
        else
            lastMsg = filtered(end,:);
            data = lastMsg.Data{1};
            firstByte = data(1);
            if firstByte == 67
                value = data(end-3:end);
                Vb = typecast(swapbytes(value) , 'single');
            else
                disp("Problema impostazione della tensione")
            end
        end

        fprintf('[UNBALANCE TEST] Step: %-3.0f%% Vn
        Va = %.2f V | Vb = %.2f V | Vc = %.2f V\n',
        (voltage_unb/Vn-1)*100, Va, Vb, Vc);
        end
end
```

```
Va_total = [Va_total Va];
Vb_total = [Vb_total Vb];
Vc_total = [Vc_total Vc];

% pausa la somma delle pause in lettura delle tensioni

%% TERMINE PROVA
fprintf('\n');
input('Premi INVIO per terminare la prova...');

% DISATTIVAZIONE OUTPUT
msg = canMessage(0x601, false, 8);
msg.Data = [0x23 0x02 0x30 0x01 0x00 0x00 0x00 0x00];
transmit(canch, msg);
% lettura conferma disattivazione dell'uscita
msg = canMessage(0x601, false, 8);
msg.Data = [0x43 0x02 0x30 0x01 0x00 0x00 0x00 0x00];
transmit(canch, msg);
pause(2)
msg = receive(canch, Inf, 'OutputFormat','timetable');
filtered =msg(msg.ID == 1409, :);
if isempty(filtered)
    fprintf('Nessun messaggio trovato con ID 581\n');
else
    lastMsg = filtered(end,:);
    data = lastMsg.Data{1};
    firstByte = data(1);
    fifthByte = data(5);
    if firstByte == 67 && fifthByte == 0
        fprintf("Output disattivato\n");
    else
        fprintf("Problema disattivazione dell'output\n");
    end
end

% IMPOSTAZIONE MODALITA' MONOFASE (imposta le tensioni a
0V in automatico)
msg = canMessage(0x601, false, 8);
msg.Data = [0x23 0x01 0x30 0x02 0x00 0x00 0x00 0x00];
transmit(canch, msg);

% lettura conferma modalità monofase
msg = canMessage(0x601,false,8);
msg.Data = [0x43 0x01 0x30 0x02 0x00 0x00 0x00 0x00];
transmit(canch, msg);
pause(2)
msg = receive(canch, Inf, 'OutputFormat','timetable');
filtered =msg(msg.ID == 1409, :);
if isempty(filtered)
    fprintf('Nessun messaggio trovato con ID 581\n');
else
    lastMsg = filtered(end,:);
    data = lastMsg.Data{1};
    firstByte = data(1);
    fifthByte = data(5);

    if firstByte == 67 && fifthByte == 0
        fprintf("Modalità monofase attivata\n")
    else
        fprintf("Problema attivazione modalità monofase\n")
    end
end
% msg_va = canMessage(0x601,false,8);
% msg_va.Data = [0x23 0x03 0x30 0x02 0x00 0x00 0x00 0x00];
% transmit(canch, msg_va);
% msg_vb = canMessage(0x601,false,8);
% msg_vb.Data = [0x23 0x03 0x30 0x03 0x00 0x00 0x00 0x00];
% transmit(canch, msg_vb);
% msg_vc = canMessage(0x601,false,8);
% msg_vc.Data = [0x23 0x03 0x30 0x04 0x00 0x00 0x00 0x00];
% transmit(canch, msg_vc);
% DISATTIVAZIONE COMUNICAZIONE ITECH IT7900P
% DISATTIVAZIONE COMUNICAZIONE ITECH IT7900P
stop(canch);
% lettura conferma disattivazione canale
pause(2)
msg = receive(canch, Inf, 'OutputFormat','timetable');
if isempty(msg)
    fprintf("Il canale CAN è spento\n");
else
    fprintf("Errore di comunicazione con ITECH IT7900P\n")
end
```

Bibliography

[1] Visiongain. *Vehicle-to-Grid (V2G) Market is projected to grow at a CAGR of 27.6% by 2034*. 2024. URL: https://www.globenewswire.com/news-release/2024/03/08/2843040/0/en/Vehicle-to-Grid-V2G-market-is-projected-to-grow-at-a-CAGR-of-27-6-by-2034-Visiongain.html.

[2] International Energy Agency. *Global EV Outlook 2023*. 2023. URL: https://www.iea.org/reports/global-ev-outlook-2023.

[3] European Commission. *Battery electric vehicles (BEVs)*. 2023. URL: https://single-market-economy.ec.europa.eu/sectors/automotive-industry/environmental-protection/battery-electric-vehicles-bevs_en.

[4] National Renewable Energy Laboratory. *Electrification Futures Study: Operational Analysis of U.S. Electric Power Systems with Increased Electrification and Demand-Side Flexibility*. 2021. URL: https://www.nrel.gov/docs/fy21osti/79226.pdf.

[5] International Renewable Energy Agency. *Innovation Outlook: Smart Charging for Electric Vehicles*. 2019. URL: https://www.irena.org/publications/2019/May/Innovation-outlook-Smart-charging.

[6] International Energy Agency. *Global EV Outlook 2024*. 2024. URL: https://www.iea.org/reports/global-ev-outlook-2024.

[7] International Energy Agency. *Tracking EV Charging Infrastructure*. 2024. URL: https://www.iea.org/reports/global-ev-outlook-2024/tracking-ev-charging-infrastructure.

[8] *IEC 61851-1: Electric Vehicle Conductive Charging System - Part 1: General Requirements*. International Electrotechnical Commission, 2017.

[9] Deltrix. *Charging Modes Explained*. 2025. URL: https://deltrixchargers.com/about-emobility/charging-modes/.

[10] BSI Standards Limited. *CEI EN 62196 - Plugs, socket-outlets, vehicle connectors and vehicle inlets - Conductive charging of electric vehicles - Part 1: General requirements*. cit. on p. 7. 2019. URL: https://webstore.iec.ch/publication/59922.

[11] BSI Standards Limited. *CEI EN 62196 - Plugs, socket-outlets, vehicle connectors and vehicle inlets - Conductive charging of electric vehicles - Part 2: Dimensional compatibility requirements for AC pin and contact-tube accessories*. cit. on p. 7. 2019. URL: https://webstore.iec.ch/publication/64364.

[12] Vector. *Charging Interfaces*. Accessed: 2025-05-08. 2023. URL: https://www.vector.com/it/it/know-how/smart-charging/charging-interfaces/#c323062.

[13] SAE International. *SAE Electric Vehicle Conductive Charge Coupler (J1772)*. 2020. URL: https://www.sae.org/standards/content/j1772_202010/.

[14] Standardization Administration of China. *GB/T 20234.2-2015: Connection set for conductive charging of electric vehicles*. 2015. URL: https://openstd.samr.gov.cn/bzgk/gb/newGbInfo?hcno=0B75C2AD687F6A01E44F3896D760E9D0.

[15] SAE International. *SAE International Announces Standard for NACS Connector*. 2023. URL: https://www.sae.org/news/press-room/2023/06/sae-international-announces-standard-for-nacs-connector.

[16] CharIN Association. *CCS - Combined Charging System*. Accessed: May 2025. https://www.charin.global/technology/ccs-at-a-glance/. 2023.

[17] CHAdeMO Association. *CHAdeMO Protocol Technical Overview*. Accessed: May 2025. https://www.chademo.com/technology/. 2022.

[18] China Electricity Council. *GB/T 27930-2015: Communication Protocol Between Off-board Conductive Charger and Battery Management System.* Accessed: May 2025. https://www.evs-institute.org/standards/. 2015.

[19] CHAdeMO and China Electricity Council. *ChaoJi: Next Generation Charging Interface.* Accessed: May 2025. https://www.chademo.com/chaoji/. 2021.

[20] CharIN. *Megawatt Charging System (MCS).* Accessed: May 2025. https://www.charin.global/technology/mcs/. 2024.

[21] Tesla Inc. *North American Charging Standard (NACS).* Accessed: May 2025. https://www.tesla.com/blog/opening-north-american-charging-standard. 2022.

[22] Statzon. *The Number of EV Charging Stations in Europe Continues to Rise Rapidly.* 2024. URL: https://statzon.com/insights/ev-charging-points-europe.

[23] Mobility Portal Europe. *EV Charging Infrastructure in Europe: 2024 Update.* 2024. URL: https://mobilityportal.eu/ev-charging-infrastructure-europe-2024/.

[24] Sanjeev Arora and altri. "A Comprehensive Review on the Characteristics and Modelling of Lithium-ion Battery Ageing". In: *ResearchGate* (2021).

[25] Björn Nykvist and Måns Nilsson. "Rapidly falling costs of battery packs for electric vehicles". In: *Nature Climate Change* 5 (2015), pp. 329–332.

[26] Robert Bosch GmbH. *CAN Specification Version 2.0.* Available at: https://www.bosch-semiconductors.com/media/ip_modules/pdf_2/canliterature1.pdf. 1991.

[27] Texas Instruments. *Controller Area Network Physical Layer Requirements.* 2008. URL: https://www.ti.com/lit/an/slla270/slla270.pdf.

[28] Hyunjin Sun et al. "Catch ID if You CAN: Dynamic ID Virtualization Mechanism for the Controller Area Network". In: *IEEE Access* 7 (2019). DOI: 10.1109/ACCESS.2019.2950373.

[29] Kvaser AB. *CAN Bit Timing.* Technical article, available at: https://www.kvaser.com. 2020.

[30] *ISO 11898-1:2015 - Road vehicles – Controller area network (CAN) – Part 1: Data link layer and physical signalling.* International Organization for Standardization, 2015. URL: `https://www.iso.org/standard/63648.html`.

[31] CAN in Automation (CiA). *CAN FD protocol and physical layer.* `https://www.can-cia.org`. CiA White Paper. 2015.

[32] IEEE Standards Association. *IEEE Standard for Ethernet (802.3-2022).* 2022.

[33] Charles Spurgeon. *Ethernet: The Definitive Guide.* 2nd ed. O'Reilly Media, 2014.

[34] Robert Metcalfe and David Boggs. "Ethernet: Distributed packet switching for local computer networks". In: *Communications of the ACM* 19.7 (1976).

[35] Natalia Olifer and Victor Olifer. *Computer Networks: Principles, Technologies and Protocols for Network Design.* Wiley, 2010.

[36] Rich Seifert and Jim Edwards. *The All-New Switch Book: The Complete Guide to LAN Switching Technology.* Wiley, 2008.

[37] David C. Plummer. "An Ethernet Address Resolution Protocol: Or Converting Network Protocol Addresses to 48.bit Ethernet Address for Transmission on Ethernet Hardware". In: *RFC 826* (1982).

[38] Thomas Narten et al. "Neighbor Discovery for IP version 6 (IPv6)". In: *RFC 4861* (2007).

[39] Andrew S. Tanenbaum and David J. Wetherall. *Computer Networks.* 5th ed. Pearson, 2011.

[40] HeelpBook. *Networking – Switch Learning and Forwarding.* 2016. URL: `https://www.heelpbook.net/2016/networking-switch-learning-and-forwarding/`.

[41] IEEE Standards Association. *IEEE Standard for Local and Metropolitan Area Networks–Bridges and Bridged Networks (802.1Q-2022).* 2022.

[42] International Electrotechnical Commission (IEC). *Electric vehicle conductive charging system - Part 23: DC electric vehicle charging station.* International Electrotechnical Commission (IEC), 2023.