

ภาคผนวก ก

บทความที่ได้รับการตีพิมพ์เผยแพร่

รายชื่อบทความวิชาการที่ได้รับการตีพิมพ์เผยแพร่

Donjaroennon, N.; Nuchkum, S.; Suddeepong, C.; Leeton, U. Design and Experimental Validation of a Charging Profile Selection System for Electric ATVs Using a Programmable Delta Charger with CANopen and Modbus RTU Communication. *Energies* 2026, 19, 1310. <https://doi.org/10.3390/en19051310>



Article

Design and Experimental Validation of a Charging Profile Selection System for Electric ATVs Using a Programmable Delta Charger with CANopen and Modbus RTU Communication

Natthapon Donjaroenon ¹ , Suphachakan Nuchkum ², Chatchai Suddeepong ² and Uthen Leeton ^{1,*}

¹ School of Electrical Engineering, Institute of Engineering, Suranaree University of Technology, Nakhon Ratchasima 30000, Thailand; d6500467@g.sut.ac.th

² School of Mechatronic Engineering, Institute of Engineering, Suranaree University of Technology, Nakhon Ratchasima 30000, Thailand

* Correspondence: uthenleeton@sut.ac.th

Abstract

This paper presents the design and experimental validation of a hardware-enforced charging profile selection framework for low-voltage electric all-terrain vehicles (ATVs), implemented on a programmable Delta battery charger operating within a voltage range of 0–120 V and a current range of 0–30 A. Unlike conventional programmable chargers that rely primarily on software-defined configuration or battery management system (BMS)-negotiated parameter setting, the proposed system enforces predefined constant-current–constant-voltage (CC–CV) charging profiles at the hardware execution layer. Vehicle identification is performed using CANopen-based identifiers, while relay-based selection, controlled via Modbus RTU, physically routes the charger output to fixed CC–CV control paths, thereby structurally reducing the risk of misconfiguration and unintended parameter changes. The system integrates layered control using embedded ESP32 nodes, a redPLC supervisory controller, and NodeRED-based orchestration, combined with real-time measurement, logging, and visualization using a time-series database and Grafana dashboards. Experimental validation is conducted using lithium-ion battery packs configured at four nominal voltage levels (24 V, 48 V, 60 V, and 72 V). The results confirm correct automatic profile selection, deterministic relay-based routing, and stable CC–CV charging behavior across repeated charging sessions. Rather than proposing a new charging algorithm, this work contributes a safety-by-design execution-layer charging architecture that complements higher-level smart charging and management protocols and is particularly suited for closed, heterogeneous fleet environments where deterministic behavior, robustness against configuration errors, and transparent verification of charging processes are critical.

Keywords: charging profile selection; programmable charging system; electric all-terrain vehicle; CANopen communication; Modbus RTU communication



Academic Editor: Hwa-Dong Liu

Received: 2 February 2026

Revised: 23 February 2026

Accepted: 27 February 2026

Published: 5 March 2026

Copyright: © 2026 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license.

1. Introduction

The rapid growth of electric vehicles (EVs) and the electrification of industrial service vehicles have significantly increased the demand for reliable, scalable, and safe battery charging systems, particularly in mission-critical applications such as airport ground support equipment (GSE) [1–3]. From an energy conversion perspective, battery charging systems are fundamentally power electronic systems responsible for regulating voltage and current delivered to the battery. In such environments, charging infrastructures must

support continuous operation, high availability, and multiple vehicle types with different operating requirements. Consequently, the performance, robustness, and safety of the charger hardware directly affect energy efficiency, battery lifetime, and operational continuity. In this work, although the target application is airport service vehicles, an electric all-terrain vehicle (ATV) is employed as a representative experimental platform to emulate the charging behavior and operating conditions of this class of industrial electric vehicles.

In practical deployments, industrial electric vehicles and service vehicles are often equipped with battery systems of different chemistries, capacities, and nominal voltage levels, leading to heterogeneous charging requirements [4–7]. This situation is particularly common in airport service fleets, such as baggage tractors and other GSE, where vehicles of different generations and specifications coexist within the same charging facility. From the viewpoint of battery charging principles, most lithium-based batteries rely on constant-current-constant-voltage (CC–CV) charging strategies, and the correctness of voltage and current setpoints is critical. Even a small deviation or an incorrect parameter selection may accelerate battery degradation or cause safety risks [8–10]. Therefore, in multi-voltage and multi-capacity charging environments, the correct selection and enforcement of charging profiles become a key engineering requirement rather than merely a configuration issue.

To manage the growing scale and complexity of charging installations, communication and supervision frameworks have been introduced into EV charging systems [11–14]. In this context, the Open Charge Point Protocol (OCPP) has been widely adopted for charging management, monitoring, and coordination in large-scale deployments. However, these communication-oriented solutions mainly operate at the supervisory and management levels. They do not directly guarantee that the safety-critical voltage and current limits are enforced at the power hardware level. As a result, even in well-managed charging infrastructures, the final responsibility for safe charging behavior still rests on the correct configuration and reliable operation of the charger hardware itself.

In addition to communication frameworks and charging infrastructure management, several studies have examined electric vehicle deployment policies, demand-side management, and coordinated control strategies to support large-scale electrification. These works provide important insights into grid integration, infrastructure planning, and system-level optimization of charging operations. While such approaches address broader planning and operational aspects, they do not directly ensure deterministic enforcement of charging parameters at the execution layer. Therefore, the present work focuses on hardware-level mechanisms that complement these higher-level strategies by providing robust and verifiable charging profile enforcement.

Along with the increasing connectivity and configurability of charging systems, cybersecurity and operational reliability issues have also attracted considerable attention [15–17]. Software faults, communication failures, or human errors during configuration may all result in incorrect charging parameters being applied. From a power electronics perspective, such faults are particularly dangerous because they directly affect the electrical output of the charger. In heterogeneous battery environments, these issues may not only cause immediate unsafe charging conditions but may also lead to long-term battery degradation. This observation highlights that misconfiguration is not merely an IT or management problem but a fundamental engineering safety issue in modern programmable charging systems.

In parallel, many studies have investigated EV charging infrastructures from system-level and operational viewpoints, including station operation, scheduling, and large-scale deployment planning [18–20]. While these studies provide valuable insights into infrastructure-level coordination and energy management, the practical problem of deterministic charging profile selection and hardware-level enforcement for heterogeneous battery systems remains insufficiently addressed. In real charging facilities, battery packs

with different nominal voltages and capacities often coexist, and even a single incorrect parameter setting may directly result in unsafe charging behavior or irreversible battery damage. To address this gap, this work focuses explicitly on the execution layer of the charging process and investigates how charging correctness and robustness can be structurally improved through hardware-enforced design principles. Rather than proposing a new charging algorithm or replacing higher-level smart charging, scheduling, or management frameworks, the proposed approach targets the safety-critical interface between vehicle identification and power delivery. By binding vehicle-side identifiers to relay-selected, hardware-configured constant-current–constant-voltage (CC–CV) charging paths, the system reduces reliance on runtime software configuration, backend negotiation processes and operator intervention for voltage and current enforcement. This execution-layer perspective complements existing supervisory and management-oriented charging architectures by providing deterministic and verifiable enforcement of charging parameters at the physical power interface, which is particularly relevant in closed and heterogeneous fleet environments where misconfiguration risk and operational robustness are primary concerns. Motivated by these challenges, this paper presents the design and experimental validation of a charging profile selection framework that emphasizes deterministic enforcement of charging parameters at the execution layer. An electric all-terrain vehicle (ATV) is used as a representative experimental platform to emulate heterogeneous industrial electric vehicle operating conditions. The proposed system is implemented on a programmable Delta charger and integrates CANopen-based vehicle identification with Modbus RTU-controlled relay selection to physically route the charger output to predefined constant-current–constant-voltage (CC–CV) charging paths. By enforcing voltage and current limits through hardware-configured control paths rather than runtime software configuration alone, the framework reduces susceptibility to misconfiguration, communication faults, and operator errors in multi-voltage charging environments. The platform operates within a voltage range of 0–120 V and a current range of 0–30 A and is experimentally validated using real-time monitoring and verification of charging behavior. The proposed approach is intended to complement higher-level charging management and coordination frameworks by providing a robust and verifiable execution mechanism for safety-critical charging operations.

- The design of a charging profile selection system for programmable battery chargers that integrates CANopen-based vehicle identification and Modbus RTU-based relay control to enforce CC–CV charging parameters at the hardware level.
- The implementation of a hardware-enforced charging profile selection mechanism that significantly reduces the risk of configuration errors and human error when charging battery systems with different nominal voltage levels.
- The experimental validation of the proposed system using an electric ATV as a representative experimental platform, demonstrating its feasibility, stability, and practical effectiveness for multi-voltage battery charging applications.

The remainder of this paper is organized as follows. Section 2 describes the overall system architecture and design principles. Section 3 presents the system implementation and experimental setup. Section 4 discusses the experimental results. Finally, Section 5 concludes the paper and outlines future work.

2. System Architecture and Hardware Implementation

2.1. Overall System Architecture and Charging Operation

Before describing the detailed architecture, it is important to clarify the design rationale underlying the proposed charging profile selection system. The system is intentionally

structured as a layered architecture to separate vehicle identification, supervisory control, and power enforcement functions, thereby reducing coupling between software decision logic and safety-critical power control. Relay-based profile selection is adopted to provide deterministic and physically verifiable routing of charging parameters, ensuring that predefined voltage and current limits remain enforceable even under communication delays, software faults, or configuration errors. This design prioritizes robustness and operational transparency over runtime flexibility, which is particularly advantageous in heterogeneous fleet environments where predictable charging behavior is essential.

Figure 1 illustrates the overall hardware and communication architecture of the proposed programmable charging system designed for electric vehicles with heterogeneous battery voltage levels, represented in this work by electric all-terrain vehicles (ATVs). The system architecture is structured to eliminate manual charger configuration by enabling automatic battery identification, adaptive constant-current-constant-voltage (CC-CV) charging profile selection, and integrated data acquisition through coordinated communication, control, and power subsystems [21–23].

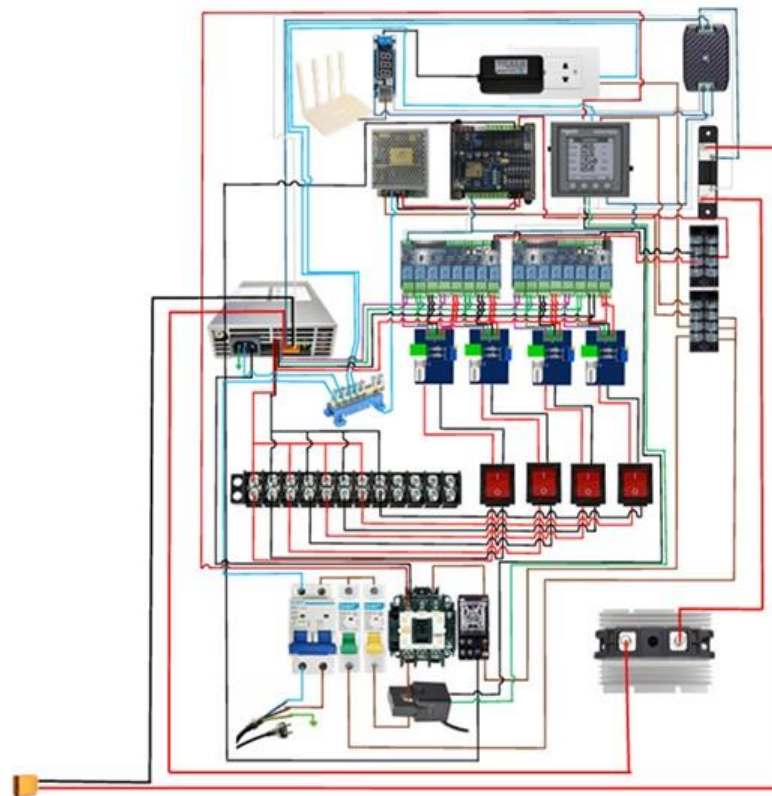


Figure 1. Overall hardware and communication architecture of the proposed automatic charging system for low-voltage electric vehicles. The architecture integrates vehicle-side identification, supervisory control, relay-based CC-CV selection, measurement devices, and data acquisition components. Red and black lines indicate DC power connections; blue and brown lines represent AC supply wiring; all remaining colored lines denote RS485 communication signals.

Experimental validation is conducted using NMC battery packs with nominal ratings of 24 V 5 A, 48 V 5 A, 60 V 5 A, and 72 V 15 A, which are selected as scaled-down representative configurations for experimental validation. Although the Delta programmable charger supports a wide operating range of 0–120 V and 0–30 A, the experiments are intentionally performed at lower current levels to validate the system design, control logic, and hardware-enforced charging profile selection. The target charging current of approximately 20 A at different voltage levels is therefore considered, and the proposed system is directly scalable to higher charging current levels supported by the power conversion hardware.

When the charging cable is connected, communication between the electric vehicle and the charging system is initiated via the CANopen protocol. Each vehicle transmits a predefined CAN identifier (CAN ID) corresponding to a specific CC–CV charging configuration. CANopen is selected due to its deterministic communication behavior, robustness, and widespread adoption in industrial control environments [24,25]. This identifier-based mechanism allows unambiguous battery recognition at the moment of connection without requiring any operator intervention [26].

Upon reception of the CAN ID, the charger-side ESP32 module forwards the identification data to the redPLC controller via MQTT. The redPLC serves as the primary supervisory unit, coordinating system initialization, safety supervision, and communication management [27]. Higher-level control functions, including NodeRED execution, are enabled only after the system verifies that all operational and safety conditions are satisfied. This layered design establishes the redPLC as a supervisory control layer that bridges the charger hardware with software-based configuration and monitoring functions [28].

Once NodeRED is activated, it subscribes to relevant MQTT topics and processes the received CAN ID information using dedicated function nodes. The control logic is explicitly divided into independent branches, each corresponding to a predefined battery charging configuration. Based on the decoded CAN ID, NodeRED triggers the appropriate RS-485 relay modules to select one of four fixed CC–CV control boards connected to the Delta programmable charger. Each board is preconfigured for a specific voltage and current profile, enabling deterministic and electrically isolated output selection [29,30]. This relay-based selection mechanism enhances operational reliability and scalability while reducing the risk of misconfiguration for additional battery configurations in industrial charging environments [31].

From a safety perspective, the layered control structure ensures that charging operations are enabled only after successful system initialization and verification of communication integrity across all subsystems. In the event of communication loss, identifier mismatch, or abnormal operating conditions, the system remains in a safe state by preventing unintended relay activation and maintaining previously enforced charging limits. This fail-safe behavior enhances operational reliability and reduces the risk of unintended profile selection during real-world deployment.

2.2. CANopen-Based Vehicle Identification and CC–CV Selection

Figure 2 illustrates the CANopen-based communication architecture between electric vehicles and the charging system. Each ATV is equipped with an ESP32-based CANopen node configured as a transmitter, while the charger-side ESP32 acts as a CANopen receiver. When the charging cable is connected, the vehicle-side node immediately initiates the CANopen boot-up sequence and transmits a predefined CAN identifier (CAN ID) that represents the nominal battery voltage level and the associated CC–CV charging profile [32].

To ensure deterministic and reliable operation in an industrial environment, the CANopen protocol is adopted due to its well-defined communication objects, node management mechanisms, and error-handling features [33,34]. After the physical connection is

established, the charger-side ESP32 waits for a valid CANopen frame within a predefined timeout window. If no valid identifier is received, the system remains in an idle and safe state, preventing any unintended activation of the power stage [35].

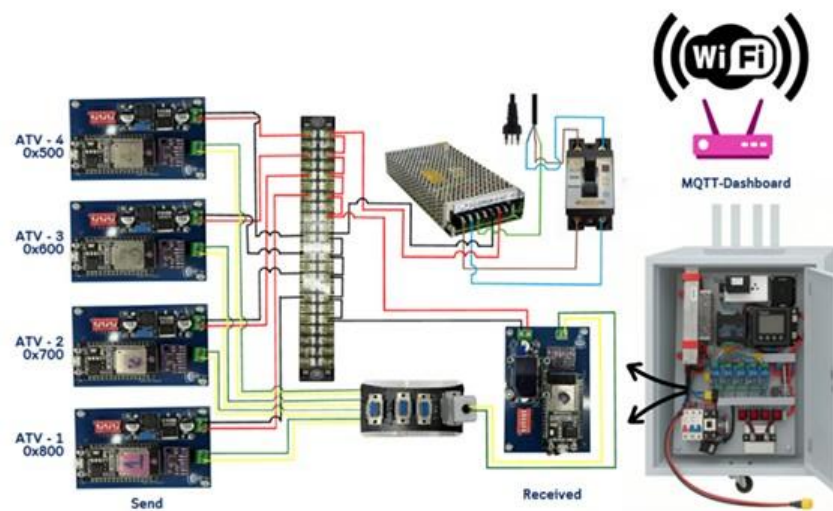


Figure 2. CANopen-based communication architecture between electric vehicles and the charging cabinet. Vehicle-side ESP32 modules transmit predefined CAN identifiers to the charger-side receiver to enable automatic battery identification and charging profile selection. (Red and black lines indicate DC power connections; blue and brown lines represent AC supply wiring; green and yellow lines denote CAN communication signals.)

Once the CAN ID is successfully received and validated, the charger-side ESP32 encapsulates the identification data and forwards it to the red PLC via MQTT. At this stage, the red PLC performs basic safety checks and state verification before enabling higher-level control functions. This two-stage validation process (vehicle-side identification followed by charger-side verification) improves robustness against communication faults and unintended connections [36].

To guarantee unique and traceable identification, each ESP32 installed on an ATV uses its embedded chip ID as a root identifier. During system initialization, this chip ID is mapped to a specific CAN ID corresponding to a predefined charging profile. The complete mapping between ESP32 chip IDs, CAN identifiers, and battery configurations is summarized in Table 1. Although the hardware platform supports a wide operating range of 0–120 V and 0–30 A, only four charging profiles are enabled in this prototype implementation in order to match the experimental validation requirements [37].

Table 1. Mapping between ESP32 chip identifiers, assigned CAN IDs, and corresponding battery configurations used for automatic vehicle and battery identification.

ESP32 Chip ID	Set CAN ID	Battery Configuration
0xF4D1B21AD6B8	0x500	24 V 5 A
0x1CB2F8BD9E7C	0x600	48 V 5 A
0xC0EA9B9EFC08	0x700	60 V 5 A
0x48BDF1BD9E7C	0x800	72 V 15 A

From a system integration perspective, this identifier-based approach enables plug-and-charge functionality. The charging system does not rely on negotiation or manual parameter input but instead deterministically binds the physical vehicle identity to a specific hardware-enforced charging configuration. This design choice is particularly suitable for multi-vehicle environments with heterogeneous battery systems, where operational simplicity and configuration safety are critical [38].

Although the current implementation uses predefined mappings between device identifiers and charging profiles, the architecture is designed to support scalable deployment through configurable identifier registration and modular profile expansion. New vehicle types or battery configurations can be incorporated by extending identifier mappings without modifying the underlying enforcement mechanism, thereby preserving system stability while enabling future growth.

2.3. Fixed CC-CV Configuration Using Relay-Based Hardware Selection

Figure 3 illustrates the relay-based fixed CC-CV control mechanism implemented in the proposed charging system. Instead of dynamically setting voltage and current references through software commands alone, each charging profile is realized using a dedicated CC-CV control board. Each board implements a local control loop that regulates the output either in constant-current (CC) mode or constant-voltage (CV) mode, depending on the instantaneous battery voltage [39].

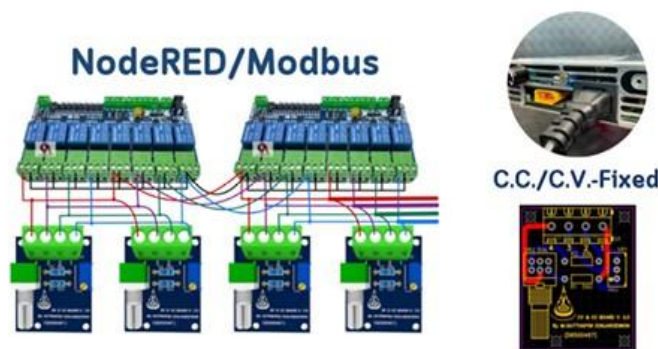


Figure 3. Relay-based fixed CC-CV charging control mechanism using Modbus RTU over RS-485 communication. Each predefined charging profile is implemented with a dedicated CC-CV control board, enabling deterministic voltage and current selection. (Red lines indicate current control connections; purple lines represent voltage control signals; green and blue lines denote DC power connections.)

The use of relay-based hardware selection is motivated by the requirement for deterministic isolation between charging configurations and the need to minimize the risk of unintended parameter switching. Unlike purely software-controlled profile selection, physical switching provides an additional layer of assurance that only the intended configuration is active at any given time. This approach improves traceability and strengthens confidence in safety-critical charging operations.

In the CC phase, the control loop maintains a constant output current until the battery voltage reaches the predefined voltage threshold. Once this threshold is reached, the controller transitions to CV mode and regulates the output voltage while the charging current gradually decreases. This two-stage CC-CV behavior is widely adopted in lithium-based battery charging systems due to its simplicity, safety, and effectiveness in preventing overcharge [40,41].

In the proposed system, the voltage and current limits of each CC–CV control board are physically configured using DIP switches. The correspondence between DIP switch settings and the resulting voltage and current levels is summarized in Table 2, while the full selectable ranges supported by the control boards are listed in Table 3. By fixing these parameters at the hardware level, the system ensures that critical charging limits cannot be modified accidentally or maliciously by software-level commands [42]. This hardware-level parameter enforcement mechanism effectively eliminates misconfiguration risk and provides an additional safety layer for multi-voltage battery charging systems.

Table 2. Fixed CC–CV charging configurations selected through DIP switch settings corresponding to different CAN identifiers.

CAN/CC-CV-Fixed		Voltage [V]	Ampere [A]
0x500	24 V 5 A	001	000
0x600	48 V 5 A	011	000
0x700	60 V 5 A	100	000
0x800	72 V 15 A	101	010

Table 3. Voltage and current levels of the CC–CV charging profiles evaluated in this study.

	Voltage/Diff Switch		Ampere/Diff Switch	
Select Profile CC-CV	12 V	000	5 A	000
	24 V	001		
	36 V	010	10 A	001
	48 V	011		
	60 V	100	15 A	010
	72 V	101		
	84 V	110	20 A	011
	100 V	111		
			25 A	111

The selection among multiple CC–CV boards is performed using Modbus RTU-controlled relay modules over RS-485. Based on the decoded CAN ID, NodeRED commands the appropriate relay channel, which physically connects the selected control board to the Delta programmable charger output. This hardware-level switching mechanism ensures that only one charging profile can be activated at any given time and provides inherent electrical isolation between different configurations [43].

Although relay actuation is initiated by software logic, the actual charging parameters are fully defined by the hardware configuration of the selected CC–CV board. This hybrid control strategy combines the flexibility of software-based system orchestration with the determinism and safety of hardware-enforced parameter definition. As a result, the charging system exhibits predictable and repeatable behavior even in the presence of software faults or communication disturbances [44].

2.4. Control Logic and Communication via Modbus RTU

In addition to CANOpen-based identification and relay-based CC–CV selection, the proposed system employs Modbus RTU as the primary field-level communication protocol for control commands and measurement data acquisition. Modbus RTU is selected due to its simplicity, robustness, and widespread adoption in industrial automation systems, particularly over the RS-485 physical layer [45,46].

In the proposed architecture, the redPLC acts as the Modbus master and periodically polls all connected slave devices according to a predefined scheduling cycle. The slave devices include RS-485 relay modules for CC–CV board selection and power measurement instruments for both the AC input side and the DC output side. Each slave device is assigned a unique slave ID, and its internal registers are mapped to specific control or measurement functions [47].

The polling cycle is designed to ensure deterministic update intervals for both control and monitoring functions. Relay actuation commands are transmitted as discrete coil or register writes, while measurement values are retrieved using holding or input register reads. This master–slave communication structure avoids bus contention and ensures predictable timing behavior, which is essential for reliable operation in industrial control environments [48].

The RS-485 relay modules are configured as Slave ID 1 and Slave ID 2 and are responsible for physically switching the CC–CV control boards. For electrical measurements, a PZEM-017 module is used to measure DC-side voltage, current, power, and accumulated energy [49], while a PM2230 power meter is used for AC-side measurements. The complete Modbus RTU device configuration, including slave IDs and register address ranges, is summarized in Table 4.

Table 4. Modbus RTU device configuration and addressing scheme used for relay control and electrical parameter measurement.

Device Type	Function	Slave ID	Address Range	Description
RS-485 Relay Module 1	CC–CV relay control	1	0–8	Controls CC–CV board selection
RS-485 Relay Module 2	CC–CV relay control	2	0–8	Controls CC–CV board selection
PZEM-017	DC measurement	5	0–6	DC voltage, current, power, energy
PM2230	AC measurement	100	2999–3207	AC voltage, current, power, energy

From a system integration perspective, this Modbus-based fieldbus framework provides a clear separation between high-level supervisory control and low-level device actuation and sensing, thereby improving modularity, maintainability, and fault isolation.

While the proposed architecture introduces additional switching elements, their impact on overall charging efficiency is minimal due to the low contact resistance of industrial relay components. The design emphasizes reliable parameter enforcement and operational safety, with efficiency considerations remaining within acceptable limits for the intended application domain.

2.5. Data Acquisition, Logging, and Real-Time Visualization

Comprehensive data acquisition and monitoring are integral parts of the proposed charging system, as they provide both operational visibility and experimental validation capability. Electrical parameters are continuously measured on both the AC input side and the DC output side, enabling detailed analysis of power conversion efficiency, charging dynamics, and system behavior [50].

On the AC side, the PM2230 power meter provides measurements of voltage, current, active power, and accumulated energy. On the DC side, the PZEM-017 module measures the output voltage, charging current, instantaneous power, and delivered energy. These measurements are periodically collected by the redPLC via Modbus RTU and forwarded to the supervisory software layer [51]. All time-stamped measurement data are stored in

an InfluxDB time-series database, which is optimized for high-frequency data ingestion and efficient temporal queries. This data management approach enables both real-time monitoring and long-term historical analysis. Grafana dashboards are used to visualize key system variables, including voltage and current trajectories, power flow, and charging state transitions [52,53]. The visualization environment supports verification of charging stability, profile selection correctness, and temporal consistency of measured electrical parameters during experimental operation.

Although the present experimental setup focuses on moderate charging currents, the same data acquisition and monitoring framework is directly applicable to higher-power charging systems, such as those required for airport ground service vehicles [54].

3. System Implementation and Experimental Validation

This section describes the system implementation and experimental validation platform of the proposed charging profile selection system. The primary objective is to verify the functional correctness, determinism, and safety of the identifier-based charging mechanism and the hardware-enforced CC–CV configuration under realistic operating conditions, rather than to evaluate charging performance metrics. The experimental platform is implemented using a programmable Delta battery charger and multiple low-voltage lithium-ion battery configurations. An electric all-terrain vehicle (ATV) is employed as a representative experimental platform to emulate the charging behavior of industrial electric vehicles with heterogeneous battery systems. This setup enables practical validation while maintaining general applicability. The experimental implementation is designed to evaluate functional correctness, deterministic profile selection, and robustness of the hardware-enforced charging mechanism under representative operating conditions. The focus of this validation is to confirm reliable coordination between identification, supervisory control, and hardware parameter enforcement rather than to optimize charging performance metrics.

The system implementation follows the architecture presented in Section 2 and reflects the actual signal and control flow of the charging process, from vehicle-side parameter encoding and CANopen-based identifier transmission to charger-side validation, hardware-level profile enforcement, and electrical parameter acquisition. Charging parameters are encoded through physical configuration mechanisms and mapped to predefined CANopen identifiers, enabling deterministic and unambiguous profile selection upon connection. In addition, supervisory control and field-level communication using Modbus RTU are integrated to coordinate relay-based profile selection and measurement acquisition, while preserving strict hardware enforcement of critical charging limits. The following subsections detail the embedded implementation, control logic, and experimental setup [55,56].

3.1. Embedded System Implementation Using ESP32 and CANopen

The proposed charging system employs ESP32 microcontrollers as embedded control units on both the vehicle side and the charger side. The ESP32 platform is selected due to its integrated CAN controller, sufficient computational capability, and suitability for real-time industrial communication and control applications. Vehicle–charger communication is realized using a CANopen-based communication layer operating over the CAN physical bus [57,58].

On the vehicle side, the ESP32 encodes the battery voltage and current configuration and transmits this information to the charging system via the CANopen network. During system initialization, a unique hardware identifier derived from the internal chip ID is deterministically mapped to a predefined CANopen identifier (CAN ID), where each CAN ID represents a validated CC–CV charging profile associated with a specific battery configuration. By embedding the charging profile information directly into the CANopen identifier,

the system enables immediate and unambiguous vehicle and battery identification upon physical connection, without requiring manual configuration or software-level negotiation.

After the charging parameters are determined, the vehicle-side ESP32 periodically transmits the selected CANopen identifier and associated data frame over the CAN network. This periodic broadcast ensures that the charger-side controller always receives a valid and up-to-date identification frame prior to any power-stage activation, thereby improving robustness against transient communication disturbances. This deterministic and lightweight CANopen-based identification mechanism forms the foundation for subsequent charger-side validation and hardware-enforced charging profile selection. Detailed descriptions of the vehicle-side parameter encoding and CANopen frame packing procedures are provided in Appendix A.1 and Appendix A.2, respectively.

3.2. Charger-Side Validation and Hardware-Enforced CC–CV Profile Selection

Upon reception of a valid CANopen identifier (CAN ID) at the charger side, the proposed system associates the identifier with a predefined constant-current-constant-voltage (CC–CV) charging configuration. Instead of dynamically generating voltage and current setpoints through software control, the charging parameters are strictly enforced using hardware-configured CC–CV control boards. This design choice follows established hardware-enforced charging principles, which are widely adopted to improve safety and reliability in battery charging systems [59,60].

Each CC–CV control board implements a local analog control loop that regulates the charger output either in constant-current (CC) mode or constant-voltage (CV) mode, depending on the instantaneous battery voltage. The voltage and current limits of each board are physically preconfigured using DIP switches, as summarized in Tables 2 and 3 in Section 2. As a result, critical charging parameters are constrained at the hardware level and cannot be altered by software faults, communication errors, or unintended supervisory commands.

Selection among multiple CC–CV control boards is performed using relay modules controlled via Modbus RTU communication. Based on the decoded CANopen identifier, the supervisory controller activates the corresponding relay channel, physically connecting the selected CC–CV board to the programmable Delta charger output. At any given time, only one relay path is allowed to be active, which guarantees mutual exclusivity among charging profiles, prevents conflicting configurations, and provides inherent electrical isolation between profiles.

During experimental operation, the system behavior was observed to remain stable under normal connection sequences, with charging activation permitted only after successful identifier validation and relay configuration. Any unexpected communication interruption results in the system remaining in a non-energized state, consistent with the fail-safe design principles described in Section 2.

The following simplified equations describe the conceptual CC–CV charging behavior used to illustrate the control logic rather than a detailed electrochemical battery model.

Once the appropriate CC–CV board is selected, the charging process follows the conventional CC–CV strategy. During the constant-current phase, the charging current is regulated to a predefined level I_{CC} , while the battery voltage increases, which can be expressed as

$$I(t) = I_{CC}, \text{ for } V(t) < V_{CV} \quad (1)$$

where $I(t)$ and $V(t)$ denote the instantaneous charging current and battery terminal voltage, respectively. Under this condition, the voltage rise can be approximated by

$$\frac{dV(t)}{dt} = \frac{I_{cc}}{C_{eq}} \quad (2)$$

with C_{eq} representing the equivalent battery capacitance.

When the battery voltage reaches the predefined threshold V_{CV} , the control loop transitions to constant-voltage operation, maintaining

$$V(t) = V_{cv}, \text{ for } t \geq t_{cv} \quad (3)$$

During this phase, the charging current naturally decays according to the battery's internal characteristics, which can be approximated as

$$I(t) = \frac{V_{cv} - V_{oc}}{R_{int}} \quad (4)$$

where $V_{OC}(t)$ is the open-circuit voltage and R_{int} is the internal resistance of the battery. To ensure safe termination of the charging process, a cutoff condition is enforced such that

$$I(t) \leq I_{cutoff} \quad (5)$$

at which point the charger output is disabled. All parameters I_{CC} , V_{CV} , and I_{cutoff} are defined exclusively by the selected hardware CC–CV board and remain invariant throughout the charging process.

By enforcing charging parameters through CANopen identifier mapping and hardware-level CC–CV profile selection, the proposed system achieves deterministic charging behavior, predictable charging trajectories, and enhanced robustness against software faults or network disturbances. This approach is particularly important in charging systems supporting heterogeneous battery configurations, where incorrect parameter selection may lead to unsafe operating conditions, as widely reported in the literature [61]. The charger-side CANopen frame validation and relay-based profile selection logic are detailed in Appendix A.3.

3.3. Supervisory Control and Profile Routing Using NodeRED on redPLC

High-level supervisory control is implemented using NodeRED running on the red-PLC, which serves as the central industrial controller and protocol gateway of the proposed charging system [62,63]. This supervisory layer decouples low-level CANopen communication, handled by the ESP32-based embedded controllers, from high-level orchestration, state management, and system-level supervision executed on the redPLC. Such separation improves modularity, maintainability, and operational robustness.

At the charger side, a local status display provides real-time visualization of the detected CANopen identifier and the corresponding charging profile parameters, allowing immediate verification of correct profile selection before and during operation. The implementation details of this charger-side visualization interface are provided in Appendix A.4. After successful decoding and validation of the CANopen message, the charger-side controller forwards the extracted identification and associated parameters to the supervisory layer via the MQTT protocol. The data are formatted as structured JSON messages and published to predefined MQTT topics subscribed to by the NodeRED-based controller, as detailed in Appendix A.5.

In addition, the supervisory layer enables synchronized observation of charging states and communication events, ensuring that profile selection decisions and relay actions

remain fully traceable during operation. The integration of monitoring and control through a unified platform simplifies system diagnostics and supports systematic verification of experimental outcomes. This arrangement also allows seamless correlation between measured electrical parameters and control events, providing deeper insight into system behavior under different charging conditions. Such capabilities enhance confidence in the operational reliability of the proposed architecture and facilitate future extensions toward larger-scale deployments.

On the supervisory side, NodeRED functions as the central orchestration layer that receives, parses, normalizes, and dispatches incoming MQTT messages to the appropriate control logic blocks. A dedicated function node decomposes each received message into a compact internal representation containing the charging profile identifier, voltage reference, and current limit. This normalization and parsing process is described in Appendix A.6. As illustrated in Figure 4, the supervisory control logic routes each validated request through mutually exclusive logic branches corresponding to predefined charging profiles (0x500, 0x600, 0x700, and 0x800), ensuring deterministic profile routing through explicit interlock mechanisms.

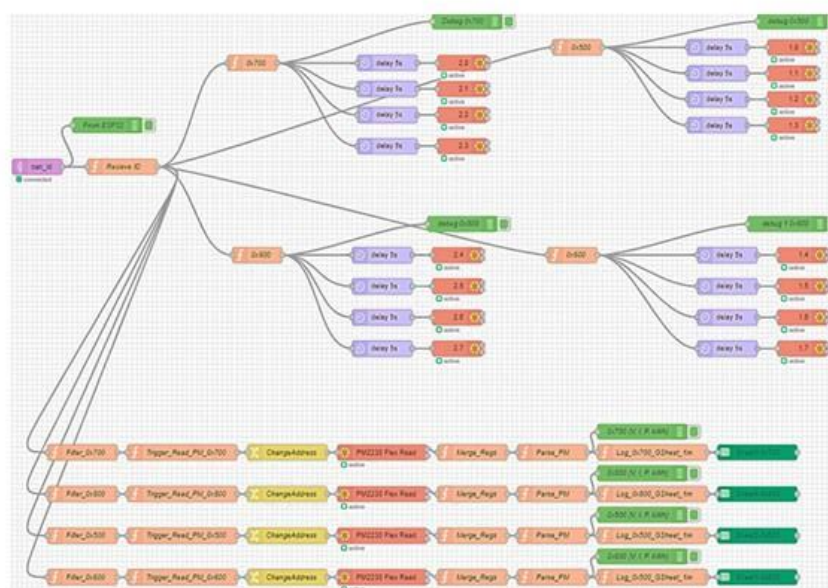


Figure 4. Overview of the NodeRED-based supervisory control flow for CAN ID parsing, charging profile routing, relay actuation via Modbus RTU, and integrated data acquisition and logging.

Once a charging profile is selected, NodeRED issues a predefined sequence of Modbus RTU commands to configure the relay outputs responsible for CC–CV board selection. A short intentional delay is introduced to mitigate relay contact bounce, inrush current, and transient misconfiguration effects. After relay switching is completed, the system transitions into a stable charging state, enabling the charger power stage. This hybrid supervisory–hardware control structure preserves flexibility at the supervisory level while guaranteeing deterministic and fail-safe charging behavior through hardware-enforced parameter selection.

3.4. Industrial Communication Infrastructure via Modbus RTU and redPLC

Industrial communication within the charging cabinet is implemented using Modbus RTU over an RS-485 fieldbus, with the redPLC acting as the central master controller and protocol gateway [64,65]. All field devices are connected to a common multidrop bus and addressed using unique slave identifiers, as summarized in Table 4 in Section 2. This architecture enables deterministic and reliable communication suitable for industrial charging environments.

In the proposed implementation, two relay control modules are configured as Slave ID 1 and Slave ID 2, respectively. Both relay modules operate using a serial communication configuration of 9600 bps baud rate, 8 data bits, no parity, and one stop bit. These relay modules are responsible for selecting the appropriate CC–CV control board, thereby directly determining the physical charging path and enforcing hardware-level charging configuration.

For electrical parameter measurement, a PZEM-017 DC energy meter is configured as Slave ID 5 and operates using the same serial communication parameters. Additionally, a PM2230 AC power meter is configured as Slave ID 100 to measure input-side electrical quantities. The redPLC periodically polls all connected slave devices according to a predefined and deterministic scheduling cycle, issuing write commands to the relay modules and read commands to the measurement instruments.

This master–slave polling mechanism ensures collision-free bus access, predictable communication timing, and reliable data acquisition. By employing the redPLC as both an industrial controller and a protocol gateway, the proposed system supports scalable integration of additional field devices while maintaining strict real-time operation and deterministic system behavior, which are essential requirements for industrial-grade charging infrastructures [66].

The present experimental validation focuses on demonstrating stable operation and correct profile enforcement within the tested voltage and current ranges. Extended stress testing, large-scale fleet deployment evaluation, and long-term reliability assessment are outside the scope of this study but can be supported in future work due to the modular architecture of the proposed system.

3.5. Electrical Parameter Acquisition and Data Logging

Electrical parameters are acquired independently from both the AC input side and the DC output side of the charging system to support operational monitoring and experimental validation [67]. This dual-side measurement architecture enables cross-validation of electrical data and provides detailed insight into power conversion efficiency, charging dynamics, and overall system behavior.

On the DC side, voltage, current, instantaneous power, and accumulated energy are measured directly at the battery interface using a PZEM-017 DC energy meter configured as Modbus Slave ID 5, as summarized in Table 4 in Section 2. As illustrated in Figure 5, the redPLC periodically polls the PZEM-017 via Modbus RTU at a fixed sampling interval. The retrieved register values are parsed and forwarded to the supervisory layer, where they are processed and stored in a time-series database for subsequent analysis.

Within the supervisory system, the acquired voltage, current, power, and energy measurements are normalized and time-stamped using dedicated NodeRED function nodes before being routed to distinct InfluxDB buckets corresponding to each charging profile (24 V 5 A, 48 V 5 A, 60 V 5 A, and 72 V 15 A). This structured data separation enables direct performance comparison among different charging configurations and supports long-term trend analysis of charging behavior and system efficiency.

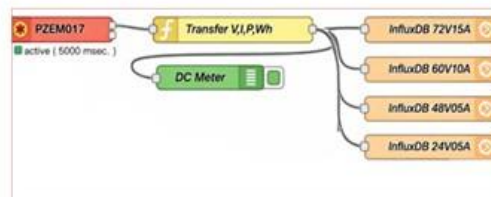


Figure 5. NodeRED flow for DC-side measurement acquisition using the PZEM-017 module and direct data forwarding to the InfluxDB time-series database for real-time monitoring and visualization.

In parallel, on the AC input side, voltage, current, active power, and accumulated energy are measured using a PM2230 industrial power meter configured as Modbus Slave ID 100. All AC-side measurement data are continuously logged to support long-term monitoring of charging sessions, performance evaluation, and cost estimation. Such dual-side measurements and logging architectures are commonly employed to assess charger performance and system efficiency under practical operating conditions [68]. The complete DC-side measurement acquisition and data forwarding pipeline is implemented using the MQTT and NodeRED-based data path detailed in Appendix A.7.

3.6. Electrical Parameter Acquisition and Data Logging

Real-time monitoring and long-term data management of the charging process are implemented using a time-series database and dashboard-based visualization architecture [69,70]. In the proposed system, all DC-side measurement data are written directly to the InfluxDB time-series database without passing through additional middleware layers. This direct data ingestion approach simplifies the overall data pipeline, reduces system complexity, and minimizes potential points of failure.

Figure 6 illustrates an example configuration of the Grafana data source connected to the InfluxDB database. Grafana accesses the database directly using consistent organization, bucket, and authentication token settings. This configuration enables reliable continuous streaming of measurement data for real-time visualization while simultaneously supporting historical data queries for post-process analysis and performance evaluation. The monitoring environment supports verification of charging stability, correctness of profile routing, and temporal consistency of measured electrical parameters throughout the experimental process.

From an architectural perspective, the direct coupling between the measurement layer and the time-series database establishes a clear separation between the control plane and the monitoring plane. The control logic responsible for charging profile selection and hardware actuation operates independently from the data acquisition, storage, and visualization pipeline. Such separation improves overall system robustness and prevents monitoring-related software faults from propagating into safety-critical charging operations [71].

Through this architecture, both real-time and historical trends of voltage, current, power, and accumulated energy can be monitored with high temporal resolution. This capability supports diagnostic analysis, performance evaluation, anomaly detection, and long-term experimental observation of the proposed charging system, which are essential requirements for industrial-grade and safety-oriented charging infrastructures.

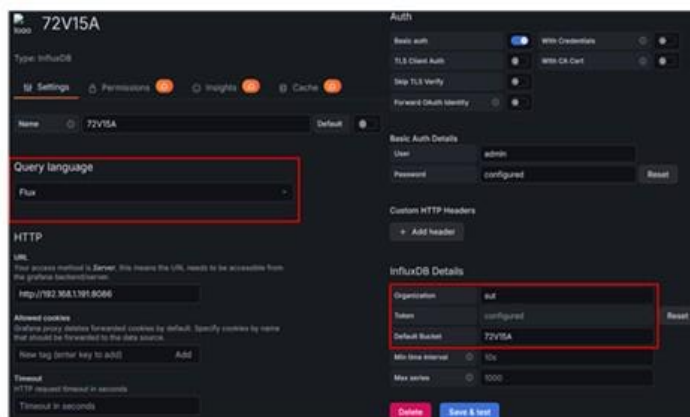


Figure 6. Configuration of the Grafana data source connected to the InfluxDB time-series database for real-time visualization of DC-side charging parameters, including voltage, current, power, and accumulated energy.

4. Experimental Results and Performance Evaluation

This section presents the experimental results and system-level performance evaluation of the proposed charging architecture implemented on an electric all-terrain vehicle (ATV) charging system. The experiments are designed to validate the operational correctness, deterministic behavior, and safety-oriented charging characteristics enabled by the identifier-driven charging profile selection and hardware-enforced CC–CV control architecture.

Rather than focusing on maximum power capability, the objective of this experimental evaluation is to verify the correctness and repeatability of charging behavior under realistic operating conditions, including automatic battery identification, profile routing, CC–CV charging transitions, and energy measurement accuracy. The results demonstrate that the proposed architecture provides a robust and energy-consistent charging framework suitable for industrial and fleet-oriented electric vehicle applications.

4.1. Experimental Setup and Test Conditions

The experimental validation is conducted using a complete electric ATV charging system incorporating the hardware and software architecture described in Section 3. The system consists of ESP32-based CANopen communication units on both the vehicle and charger sides, a redPLC executing supervisory control logic via NodeRED, hardware-enforced CC–CV control boards selected through relay actuation, and Modbus RTU-based measurement and control devices.

On the DC output side, charging voltage, current, instantaneous power, and accumulated energy are measured using a PZEM-017 DC energy meter. On the AC input side, voltage, current, active power, and energy consumption are monitored using a PM2230 industrial power meter. All measurement data are continuously logged to an InfluxDB time-series database and visualized in real time through Grafana dashboards.

Experiments are performed using an electric ATV equipped with lithium-based battery configurations corresponding to predefined charging profiles (24 V 5 A, 48 V 5 A, 60 V 5 A, and 72 V 15 A). For each charging session, the vehicle-side controller broadcasts its CANopen identifier upon physical connection, enabling automatic selection of the

appropriate charging profile without manual intervention. Each session is recorded from the initiation of charging until completion of the CC–CV process.

4.2. Verification of Automatic Charging Profile Selection

This experiment verifies the correctness and determinism of the automatic charging profile selection mechanism based on CANopen identifier detection and supervisory control routing. When the vehicle is connected, the vehicle-side ESP32 broadcasts a predefined CANopen identifier representing the associated battery configuration. The charger-side controller validates the received identifier and forwards it to the supervisory layer for routing and actuation.

Figure 7 illustrates the runtime execution of the NodeRED-based supervisory control flow during a representative charging session. The detected CANopen identifier activates exactly one charging profile branch corresponding to the assigned configuration, while all other branches remain inactive. This mutually exclusive routing behavior ensures that only one CC–CV charging path is enabled at any given time.

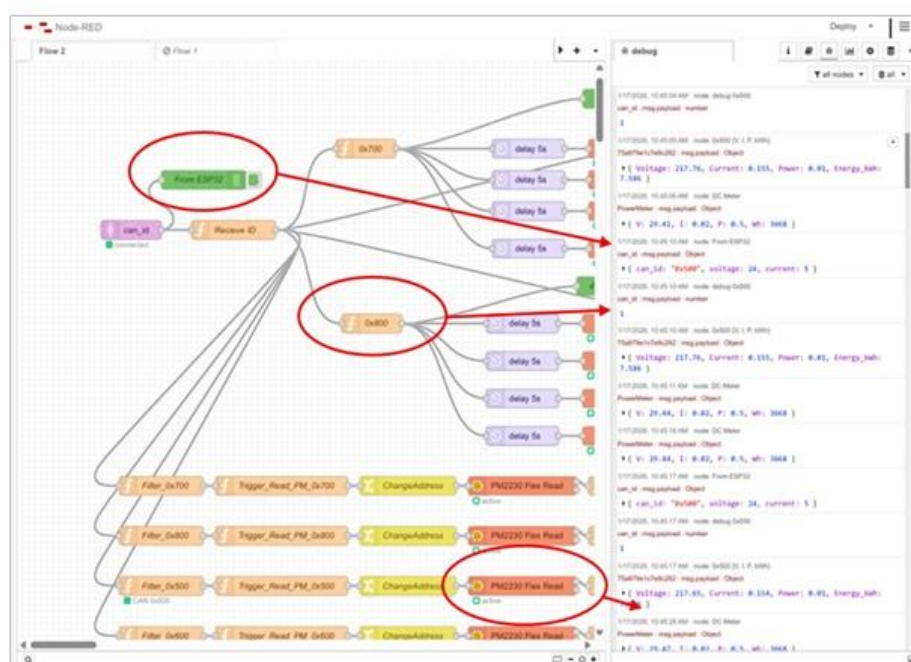


Figure 7. Runtime screenshot of the NodeRED supervisory control flow showing CAN ID–based routing and mutually exclusive charging profile selection. The highlighted path and debug messages confirm that the system correctly activates the charging profile corresponding to the detected battery configuration (CAN ID 0x800 in this example).

After profile selection, the supervisory controller issues a deterministic sequence of Modbus RTU commands to configure the relay outputs, physically connecting the appropriate CC–CV control board to the charger output. No incorrect profile activation, ambiguous routing, or conflicting relay states were observed throughout repeated plug-in and unplug cycles, confirming reliable and repeatable automatic charging profile selection.

4.3. Real-Time CC–CV Charging Behavior and Visualization

Figures 8 and 9 present the real-time charging behavior of the proposed system under a representative 72 V 15 A charging configuration, visualized using the native InfluxDB query interface and the Grafana dashboard, respectively. This charging profile is selected as the representative case because it clearly demonstrates the complete constant-current to constant-voltage (CC–CV) transition under stable operating conditions.



Figure 8. Real-time voltage and current data of the 72 V 15 A charging profile visualized directly using the native InfluxDB query interface. The purple curve represents the battery terminal voltage, while the blue curve denotes the charging current. Although the data are accurately recorded, the visualization is mainly intended for data inspection rather than operator-oriented monitoring.



Figure 9. Real-time CC–CV charging profile of the 72 V, 15 A NMC configuration visualized using Grafana. The upper plots show the voltage evolution, while the lower plots show the current behavior. The transition from constant-current (CC) to constant-voltage (CV) operation is clearly observable.

Figure 8 shows the voltage and current waveforms plotted directly from the InfluxDB time-series database. Although this interface provides accurate and comprehensive access to the recorded measurement data, it is primarily designed for data inspection and historical querying rather than intuitive real-time operational monitoring.

Figure 9 illustrates the same charging session visualized using the Grafana dashboard. The upper plots show the battery terminal voltage, while the lower plots depict the charging current. The transition from constant-current (CC) mode to constant-voltage (CV) mode is clearly observable, confirming that the charging process follows the intended CC–CV charging trajectory.

From the charging behavior shown in Figure 9, the system initially operates in constant-current (CC) mode, regulating the output current at approximately 15 A while the battery voltage increases steadily. Once the voltage reaches the predefined threshold of approximately 78.5 V, the control mode transitions to constant voltage (CV). During this phase, the battery voltage remains nearly constant while the charging current gradually decreases, which is characteristic of lithium-based battery charging processes.

The smooth and stable CC–CV transition confirms the correctness of the hardware-enforced charging control strategy and demonstrates the stability of both voltage and current regulation loops. The absence of voltage overshoot, current oscillation, or abrupt mode switching indicates effective loop design and reliable hardware-level enforcement of charging parameters, which are critical for battery safety and long-term reliability.

Although only the 72 V 15 A charging profile is presented in the main text as the representative case, experimental results for the remaining configurations (24 V 5 A, 48 V 5 A, and 60 V 5 A) exhibit the same CC–CV characteristics. Additional real-time dashboard visualizations for these charging configurations are provided in Appendix B to demonstrate the generality of the proposed architecture across different voltage and current levels.

Voltage deviations remained within $\pm 2\%$, while current deviations were within $\pm 5\%$ during steady-state operation. The deviation values were computed by comparing steady-state measurements recorded in InfluxDB against the predefined CC–CV reference settings over multiple charging sessions.

4.4. Power and Energy Analysis and Charging Session Logging

To evaluate energy transfer behavior and charging efficiency, instantaneous power and accumulated energy are analyzed using the time-series data collected during charging sessions. Figure 10 presents the real-time charging power and cumulative energy profiles visualized through the Grafana dashboard for the 72 V/15 A charging configuration.



Figure 10. Power & Energy Dashboard (Grafana, 72 V 15 A).

The instantaneous power profile reflects the expected behavior of CC–CV charging, with higher power during the CC phase followed by a gradual decrease during the CV

phase. The accumulated energy curve increases monotonically throughout the charging session, confirming consistent energy transfer from the charger to the battery.

Figure 11 provides an example of the energy history and session-level logging extracted from the time-series database. These results demonstrate that the proposed system not only enforces charging parameters at the hardware level but also provides accurate and continuous measurement of energy delivery, which is essential for performance evaluation, traceability, and energy accounting.

	A	B	C	D	E	F	G	H
1	TimeStamp	Voltage [V]	Current [A]	Power [kW]	Energy [kWh]	TOU Type	Rate [THB]	Cost [THB]
674	2026-01-19 19:00:28	0	0	0	8.206	Idle	0	0
675	2026-01-19 19:01:31	212.72	5.233	0.94	8.211	On-peak	5.1135	0
676	2026-01-19 19:02:01	212.7	5.228	0.94	8.219	On-peak	5.1135	0
677	2026-01-19 19:02:36	212.89	5.234	0.94	8.228	On-peak	5.1135	0
678	2026-01-19 19:03:10	212.62	5.252	0.95	8.237	On-peak	5.1135	0
679	2026-01-19 19:04:03	212.73	5.267	0.95	8.25	On-peak	5.1135	0
680	2026-01-19 19:04:36	212.61	5.280	0.95	8.259	On-peak	5.1135	0
681	2026-01-19 19:05:17	212.6	5.295	0.96	8.27	On-peak	5.1135	0
682	2026-01-19 19:05:50	212.41	5.306	0.96	8.279	On-peak	5.1135	0
683	2026-01-19 19:06:56	212.48	5.326	0.96	8.297	On-peak	5.1135	0
684	2026-01-19 19:07:35	212.56	5.334	0.96	8.307	On-peak	5.1135	0
685	2026-01-19 19:08:06	212.64	5.339	0.97	8.315	On-peak	5.1135	0
686	2026-01-19 19:09:01	213.23	4.770	0.86	8.329	On-peak	5.1135	0
687	2026-01-19 19:09:46	213.67	4.366	0.79	8.34	On-peak	5.1135	0
688	2026-01-19 19:10:41	214.24	4.008	0.72	8.351	On-peak	5.1135	0

Figure 11. An example of the energy history (72 V 15 A).

4.5. Summary of Experimental Results and Practical Implications

The experimental results confirm that the proposed charging architecture achieves reliable automatic battery identification, deterministic charging profile selection, and stable CC–CV charging behavior across multiple battery configurations. The integration of CANopen-based identification, hardware-enforced CC–CV control, supervisory routing, and real-time measurement ensures that unsafe or conflicting charging configurations are structurally prevented.

From an energy-system perspective, the proposed architecture provides a clear separation between supervisory logic and safety-critical charging enforcement, ensuring that voltage and current limits are physically constrained by hardware rather than software logic alone. This approach significantly reduces the risk of misconfiguration, human error, and unintended charging behavior in environments where multiple battery types and voltage levels coexist.

Overall, the experimental validation demonstrates that the proposed system offers a robust and energy-consistent charging solution suitable for industrial electric vehicles, fleet charging environments, and other safety-critical charging infrastructures where deterministic behavior, repeatability, and accurate energy monitoring are essential.

No abnormal interruptions or unintended profile activations were observed during the experimental sessions, confirming stable operation. Future work will include long-duration testing and higher power validation to further assess system scalability.

4.6. System-Level Implications and Comparison with OCPP-Based Charging Architectures

Table 5 summarizes the key differences between the proposed identifier-driven, hardware-enforced charging architecture and conventional OCPP-based charging systems. While OCPP-based solutions primarily emphasize interoperability, backend coordination, and software-defined charging management, the proposed approach focuses on determinis-

tic enforcement of voltage and current limits at the physical power interface. By embedding charging profile selection into CANopen identifiers and binding them to relay-selected CC–CV control boards, safety-critical charging parameters are guaranteed by hardware rather than network-dependent software control. This complementary architectural distinction is particularly relevant for industrial and fleet charging environments requiring reliable operation under heterogeneous battery configurations and stringent safety constraints [72–76]. The comparison highlights architectural trade-offs rather than performance superiority claims.

Table 5. Detailed Comparison between OCPP-Based Charging Architectures and the Proposed Hardware-Enforced Charging Framework.

Aspect	Conventional OCPP-Based Charging Architectures	Proposed Identifier-Driven Hardware-Enforced Architecture
System Layer Focus	Operates primarily at the application and backend layers, focusing on charging session control, billing, authorization, and interoperability between charging stations and central management systems via IP-based communication.	Operates at the execution and power-interface layer, focusing on deterministic enforcement of voltage and current limits directly at the charger hardware.
Charging Profile Definition	Charging profiles are dynamically generated, negotiated, or updated via backend software and transmitted over network protocols (e.g., OCPP JSON messages).	Charging profiles are predefined, physically configured, and selected via CANopen identifiers mapped to relay-selected CC–CV control boards.
Parameter Enforcement	Voltage and current constraints are enforced by software commands interpreted by the charger controller, making correct behavior dependent on communication integrity and firmware correctness.	Voltage and current constraints are enforced at the hardware level by preconfigured CC–CV boards, independent of software state or network availability.
Role of Communication Protocols	OCPP enables coordination, monitoring, and management but does not directly constrain the power electronics behavior at the physical interface.	CANopen identifiers act as deterministic selectors, while Modbus RTU performs physical routing and actuation without modifying charging parameters.
Safety-Critical Behavior	Safety depends on correct backend configuration, protocol compliance, and cybersecurity measures; misconfiguration or communication failure may propagate to the charging process.	Safety is guaranteed by design through hardware-level interlocking, preventing unsafe or conflicting charging configurations regardless of software faults.
Handling of Heterogeneous Batteries	Typically relies on backend databases and software logic to associate vehicle types with charging profiles.	Vehicle-side identifiers directly encode battery configuration, enabling immediate and unambiguous profile selection upon physical connection.
Experimental Validation Emphasis	Validation commonly focuses on interoperability, protocol compliance, and smart charging logic rather than physical CC–CV behavior.	Validation emphasizes real-time voltage, current, power, and energy measurements, demonstrating stable CC–CV transitions and deterministic charging trajectories.
Failure Mode Isolation	Software or network failures may affect charging correctness if not properly mitigated.	Hardware-enforced constraints isolate failure modes, preventing unsafe charging even under communication or supervisory control failures.

In addition, the proposed architecture reduces reliance on continuous network connectivity and backend decision-making during active charging, thereby mitigating risks associated with communication delays, packet loss, or cybersecurity incidents that could otherwise affect software-controlled systems. The hardware-level binding between vehicle identification and charging configuration establishes a deterministic execution path, ensuring that voltage and current limits remain enforced regardless of supervisory network conditions. This characteristic is particularly beneficial in industrial environments where charging operations must continue safely under intermittent connectivity or degraded communication scenarios [72–76]. Furthermore, by separating high-level monitoring from low-level enforcement, the system maintains operational safety even if supervisory services experience temporary interruptions, since critical constraints are governed locally at the power interface. The explicit mapping between CANopen identifiers and relay-selected CC–CV boards also improves traceability and auditability of charging events, facilitating root-cause analysis and compliance verification in regulated deployments. From a system assurance perspective, enforcing limits through fixed hardware configurations simplifies validation and reduces the risk of unintended parameter changes introduced by software updates or configuration errors. While OCPP-based platforms provide flexibility for large-scale network coordination, billing integration, and remote management, the proposed approach prioritizes predictable behavior, fail-safe operation, and robustness against operational uncertainties. This architectural emphasis is especially relevant for fleet and industrial applications where heterogeneous battery systems coexist and strict safety margins must be preserved. Consequently, the comparison underscores complementary design philosophies, highlighting how hardware-enforced control can coexist with higher-level coordination frameworks to achieve both safety and operational efficiency in practical deployments.

Unlike OCPP-based architectures that primarily address interoperability and backend coordination, the proposed system introduces a hardware-enforced execution layer where charging safety, correctness, and determinism are guaranteed at the physical power interface. This complementary design is particularly suitable for industrial and fleet charging infrastructures where heterogeneous battery configurations, unattended operation, and strict safety requirements are critical.

5. Conclusions and Future Work

This study presented an identifier-driven and hardware-enforced charging architecture designed to ensure safe, deterministic, and energy-consistent operation for electric vehicles with heterogeneous battery configurations. By binding CANopen-based vehicle identification directly to relay-selected fixed CC–CV charging paths, the proposed system enforces voltage and current limits at the physical power interface rather than through software-defined negotiation. Experimental validation demonstrated stable CC–CV charging behavior, smooth current–voltage transitions, accurate electrical parameter acquisition, and reliable energy logging across multiple voltage levels. From an energy systems perspective, this approach structurally prevents unsafe power transfer, minimizes configuration-induced losses or instability, and provides a robust charging framework well suited for industrial, fleet-based, and safety-critical electric vehicle applications where repeatability and correctness of energy delivery are paramount.

While the experimental validation demonstrates stable and deterministic operation under representative conditions, the current study is conducted at moderate power levels and on a prototype-scale deployment. Further validation under higher power ratings and large-scale multi-charger environments remains an important direction for future investigation.

Looking forward, the proposed architecture provides a scalable foundation for future energy-oriented charging infrastructures. Ongoing and future work will focus on extending the system to higher power levels and broader battery voltage ranges while preserving hardware-level enforcement of energy constraints. Further integration with battery management systems will enable health-aware and energy-optimized charging strategies, while advanced analysis of time-series power and energy data will support long-term performance assessment, degradation monitoring, and system-level optimization. These developments aim to position the proposed charging framework as a practical and extensible execution layer for next-generation energy-aware charging systems in connected, autonomous, and industrial electric vehicle ecosystems.

Author Contributions: Conceptualization, N.D. and U.L.; methodology, N.D.; software, N.D.; validation, N.D., C.S. and U.L.; formal analysis, N.D.; investigation, N.D.; resources, N.D.; data curation, N.D.; writing—original draft preparation, N.D. and S.N.; writing—review and editing, N.D.; visualization, N.D.; supervision, U.L.; project administration, N.D. and U.L.; funding acquisition, S.N. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by Suranaree University of Technology (SUT), Thailand, under the Student Fundamental Research Fund for Universities, Grant No. IRD-thesis-1.

Data Availability Statement: The data supporting the findings of this study are available from the corresponding author upon reasonable request. The data are not publicly available due to their use as part of an ongoing research project and to prevent potential misuse of system-level configuration details.

Acknowledgments: The authors would like to thank the Suranaree University of Technology, Faculty of Engineering, for their support in terms of budget, equipment, and technical guidance for the thesis, as well as for their valuable advice and close supervision of this work. The authors would also like to express their sincere gratitude to Cherdchai Corporation Co., Ltd. for providing industrial insights and practical guidance toward the future development of large-scale charging stations for airport baggage tractor and ground support vehicle applications.

Conflicts of Interest: The authors declare no conflicts of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript; or in the decision to publish the results.

Abbreviations

The following abbreviations are used in this manuscript:

ATV	All-Terrain Vehicle
EV	Electric Vehicle
CANopen	Controller Area Network Open Protocol
CC-CV	Constant-Current/Constant-Voltage

Appendix A. Vehicle-Charger CANopen Communication Layer

This appendix provides implementation-level details to support reproducibility and to clarify how deterministic profile selection and hardware-enforced safety mechanisms are realized in practice. The included code excerpts illustrate representative configuration, communication, and validation logic used during experimental validation, while the charging behavior itself remains governed by hardware constraints described in the main manuscript.

Looking forward, the proposed architecture provides a scalable foundation for future energy-oriented charging infrastructures. Ongoing and future work will focus on extending the system to higher power levels and broader battery voltage ranges while preserving hardware-level enforcement of energy constraints. Further integration with battery management systems will enable health-aware and energy-optimized charging strategies, while advanced analysis of time-series power and energy data will support long-term performance assessment, degradation monitoring, and system-level optimization. These developments aim to position the proposed charging framework as a practical and extensible execution layer for next-generation energy-aware charging systems in connected, autonomous, and industrial electric vehicle ecosystems.

Author Contributions: Conceptualization, N.D. and U.L.; methodology, N.D.; software, N.D.; validation, N.D., C.S. and U.L.; formal analysis, N.D.; investigation, N.D.; resources, N.D.; data curation, N.D.; writing—original draft preparation, N.D. and S.N.; writing—review and editing, N.D.; visualization, N.D.; supervision, U.L.; project administration, N.D. and U.L.; funding acquisition, S.N. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by Suranaree University of Technology (SUT), Thailand, under the Student Fundamental Research Fund for Universities, Grant No. IRD-thesis-1.

Data Availability Statement: The data supporting the findings of this study are available from the corresponding author upon reasonable request. The data are not publicly available due to their use as part of an ongoing research project and to prevent potential misuse of system-level configuration details.

Acknowledgments: The authors would like to thank the Suranaree University of Technology, Faculty of Engineering, for their support in terms of budget, equipment, and technical guidance for the thesis, as well as for their valuable advice and close supervision of this work. The authors would also like to express their sincere gratitude to Cherdchai Corporation Co., Ltd. for providing industrial insights and practical guidance toward the future development of large-scale charging stations for airport baggage tractor and ground support vehicle applications.

Conflicts of Interest: The authors declare no conflicts of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript; or in the decision to publish the results.

Abbreviations

The following abbreviations are used in this manuscript:

ATV	All-Terrain Vehicle
EV	Electric Vehicle
CANopen	Controller Area Network Open Protocol
CC-CV	Constant-Current/Constant-Voltage

Appendix A. Vehicle-Charger CANopen Communication Layer

This appendix provides implementation-level details to support reproducibility and to clarify how deterministic profile selection and hardware-enforced safety mechanisms are realized in practice. The included code excerpts illustrate representative configuration, communication, and validation logic used during experimental validation, while the charging behavior itself remains governed by hardware constraints described in the main manuscript.

Appendix A.1. Vehicle-Side Configuration Encoding and CANopen Identification

On the vehicle side, the ESP32-based embedded node is responsible for encoding the battery configuration parameters and transmitting the corresponding identification information to the charging cabinet via the CANopen communication layer over the CAN bus. The nominal battery voltage and current limits are selected using physical DIP switch settings and mapped to predefined discrete levels. This hardware-based parameter selection mechanism ensures that only validated and predefined charging profiles can be generated, eliminating software ambiguity and reducing the risk of misconfiguration in automated charging environments. No runtime voltage or current parameters are written or negotiated over the communication interface.

Each vehicle is uniquely identified by a predefined CANopen identifier (CAN ID), which is automatically assigned by the ESP32 controller during system initialization based on its internal chip identifier. In the present implementation, four CANopen identifiers (0x500, 0x600, 0x700, and 0x800) are used to represent different vehicle or battery configurations. This identifier is subsequently used by the charger-side supervisory system to select the corresponding charging profile. The CANopen identifier serves solely as a profile selector rather than a command-based parameter update mechanism.

By binding the battery configuration and charging profile information to a hardware-defined CANopen identifier, the proposed approach enables immediate and unambiguous vehicle and battery identification upon physical connection, without requiring any manual software configuration or negotiation process. This design choice enhances safety, repeatability, and deterministic behavior, and is particularly suitable for unattended and automated operation in connected charging infrastructures. The presented configuration logic reflects the experimental prototype and is intended to demonstrate the feasibility of identifier-based parameter enforcement rather than prescribe a production-ready implementation. Additional validation layers, such as redundancy checks and extended diagnostics, can be incorporated in larger-scale deployments.

Box A1. Presents the vehicle-side logic used to encode the predefined voltage and current settings into a CANopen identifier based on the physical configuration.

```
#define S1 25
#define S2 26
#define S3 27
#define S4 14
#define S5 12
#define S6 13
int readVoltageSetting() {
  int b1 = digitalRead(S1);
  int b2 = digitalRead(S2);
  int b3 = digitalRead(S3);
  int val = (b1 << 2) | (b2 << 1) | b3;
  switch (val) {
    case 0b000: return 12;
    case 0b001: return 24;
    case 0b010: return 36;
    case 0b011: return 48;
    case 0b100: return 60;
    case 0b101: return 72;
    case 0b110: return 84;
    case 0b111: return 100;
    default: return -1;
  }
}
```

Box A1. *Cont.*

```

int readCurrentSetting() {
  int b4 = digitalRead(S4);
  int b5 = digitalRead(S5);
  int b6 = digitalRead(S6);
  int val = (b4 << 2) | (b5 << 1) | b6;
  switch (val) {
    case 0b000: return 5;
    case 0b001: return 10;
    case 0b010: return 15;
    case 0b011: return 20;
    case 0b100: return 25;
    default: return -1;
  }
}

// CANopen ID assignment based on ESP32 chip ID
uint64_t chipID; uint32_t can_id; chipID = ESP.getEfuseMac();
if (chipID == 0xF4D1B21AD6B8) {
  can_id = 0x500;
} else if (chipID == 0x1CB2F8BD9E7C) {
  can_id = 0x600;
} else if (chipID == 0xC0EA9B9EF0C8) {
  can_id = 0x700;
} else if (chipID == 0x48BDF1BD9E7C) {
  can_id = 0x800;
} else {
  can_id = 0x000;
}

```

Appendix A.2. Vehicle-Side CANopen Frame Packing and Transmission

After determining the voltage and current configuration, the vehicle-side ESP32 packages the selected charging profile information into a fixed-format CANopen data frame. The CANopen identifier is used as the primary means of vehicle and battery identification, while the data payload provides additional consistency information for charger-side validation. These values are not interpreted as runtime control commands but are used for identification consistency and validation purposes only.

The CANopen frame is transmitted periodically using the ESP32 TWAI (Two-Wire Automotive Interface) over the CAN physical layer at a bit rate of 125 kbps. This lightweight and deterministic communication mechanism ensures that the charger-side controller always receives a valid and up-to-date identification frame prior to any power-stage activation. Periodic broadcasting, rather than event-driven transmission, improves robustness against transient communication disturbances and guarantees predictable system behavior during automated charging operations.

The use of fixed-format frame packing and periodic transmission aligns with the deterministic design philosophy of the proposed charging system, where charging parameters are strictly enforced at the hardware level and communication is used solely for identification and verification. This approach is particularly important in connected charging infrastructures, where reliable and unambiguous identification is a prerequisite for safe and unattended operation.

Box A2. Illustrates the low-level CANopen frame packing and periodic transmission procedure implemented on the vehicle-side controller.

```
float voltage = (float)readVoltageSetting();
float current = (float)readCurrentSetting();

twai_message_t tx_message;
tx_message.identifier = can_id;
tx_message.extd = 1;
tx_message.rtr = 0;
tx_message.data_length_code = 8;

// Pack voltage and current into CANopen payload
memcpy(&tx_message.data[0], &voltage, 4);
memcpy(&tx_message.data[4], &current, 4);

// Transmit CANopen frame
if (twai_transmit(&tx_message, pdMS_TO_TICKS(100)) == ESP_OK) {
  Serial.print("Sent CAN ID: ");
  Serial.println(can_id, HEX);
}
```

Appendix A.3. Charger-Side CANopen Reception and Identifier Validation

On the charger side, an ESP32-based CANopen interface continuously monitors the CAN bus and receives incoming frames transmitted by the vehicle-side controller. Each received frame is subjected to an initial validation process that checks both the payload length and the CANopen identifier. Only messages with a valid data format and one of the predefined CANopen identifiers corresponding to supported charging profiles are accepted for further processing.

Upon successful reception, the CANopen identifier is used to confirm the selected charging profile prior to any hardware-level action. Payload information contained in the received frame is parsed solely for identification consistency checking and charger-side validation purposes. These values are not used to dynamically configure charging parameters, which remain strictly enforced by the selected hardware CC–CV control board.

Any CANopen frame that does not satisfy the validation criteria is immediately discarded, and the charger output remains inhibited. This fail-safe filtering mechanism prevents unintended or unsafe charging operations and ensures that only explicitly authorized messages can propagate to the hardware selection and control layers.

The charger-side CANopen reception and validation logic is implemented in embedded firmware on the ESP32 controller. Box A3 provides the corresponding implementation code that realizes the identifier validation, payload checking, and fail-safe filtering mechanisms described above. The source code is included for completeness and reproducibility, while the charging behavior itself remains governed by hardware-enforced CC–CV constraints. The validation routine is designed to ensure that only correctly formatted and explicitly recognized messages are accepted, thereby preventing unintended activation of charging configurations due to corrupted frames or communication disturbances. This behavior supports fail-safe operation under realistic industrial communication conditions.

Box A3. Presents the charger-side CANopen frame reception and identifier validation logic implemented in the ESP32 firmware.

```
twai_message_t rx_message;
float rx_voltage, rx_ampere;
uint32_t rx_id = 0;
bool dataReceived = false;
```

Box A3. Cont.

```

if (twai_receive(&rx_message, pdMS_TO_TICKS(100)) == ESP_OK) {
  rx_id = rx_message.identifier;
  if (rx_message.data_length_code == 8) {
    if (rx_id == 0x500 || rx_id == 0x600 || rx_id == 0x700 || rx_id == 0x800) {
      memcpy(&rx_voltage, &rx_message.data[0], 4);
      memcpy(&rx_ampere, &rx_message.data[4], 4);
      dataReceived = true;
    } else {
      dataReceived = false;
    }
  }
}

```

Appendix A.4. Charger-Side Status Display for CANopen-Based Vehicle Identification and Charging Parameter Monitoring

To facilitate system commissioning, debugging, and on-site verification, a local status display is implemented on the charger-side controller. This display provides real-time visualization of the detected CANopen identifier and the corresponding charging profile parameters, allowing operators and engineers to immediately verify correct profile selection prior to and during charging operation.

Specifically, the display presents the following information:

- The detected CANopen identifier (CAN ID) of the connected vehicle;
- The target charging voltage corresponding to the selected profile;
- The target or measured charging current associated with the active CC-CV configuration.

This human-machine interface serves as a lightweight diagnostic and monitoring tool during system setup, maintenance, and troubleshooting. By providing immediate visual confirmation of identifier-based profile selection, the display enhances operational confidence and supports verification that the CANopen-based identification and routing mechanism is functioning correctly before hardware-level enforcement and throughout the charging process.

Box A4. Shows an example implementation of the charger-side display update routine using an SSD1306 OLED module.

```

// Clear display and prepare for update
display.clearDisplay();           // Clear screen
display.setTextSize(1);          // Set text size
display.setTextColor(SSD1306_WHITE);
display.setCursor(0, 0);          // Set cursor to top left
// Display CAN ID
display.print("CAN ID: ");
display.println(can_id, HEX);
// Display voltage reference
display.print("Voltage V: ");
display.println(rx_voltage);
// Display current reference
display.print("Ampere A: ");
display.println(rx_ampere);
// Update display
display.display();
// Small delay for display refresh stability
delay(50);

```

Appendix A.5. Charger-Side MQTT Forwarding of CANopen Identification and Charging Parameters

After successful decoding and validation of the CANopen message at the charger side, the extracted identification and associated charging parameters are forwarded to the supervisory control system using the MQTT protocol. The data are formatted as a structured JSON object containing the CANopen identifier, voltage reference, and current limit.

The formatted message is published to a predefined MQTT topic subscribed to by the NodeRED-based supervisory controller. This forwarding mechanism enables real-time supervisory monitoring and deterministic profile routing while preserving a clear separation between the low-level CANopen communication layer and the high-level supervisory control logic.

By decoupling charger-side embedded firmware from supervisory logic, the proposed architecture improves modularity and allows the control strategy to be extended or modified at the supervisory level without requiring firmware changes on the charger-side microcontroller. Due to space limitations, only the core functional components of the MQTT forwarding logic are presented.

Box A5. Illustrates the charger-side MQTT forwarding logic used to publish validated CANopen identification and charging parameters.

```
if (dataReceived) {
  String payload = "{}";
  payload += "\"can_id\": \"0x\" + String(rx_id, HEX) + \"\", ";
  payload += "\"voltage\": \"\" + String(rx_voltage) + \"\", ";
  payload += "\"current\": \"\" + String(rx_ampere);
  payload += "\"}";

  if (client.publish(mqtt_topic, payload.c_str())) {
    Serial.println("MQTT message sent: " + payload);
  } else {
    Serial.println("Failed to send MQTT message");
  }
}
```

Appendix A.6. Supervisory-Side CANopen Message Parsing and Normalization in NodeRED

After the charger-side controller forwards the decoded CANopen data to the supervisory system via MQTT, NodeRED operates as the central orchestration layer responsible for receiving, parsing, and dispatching incoming messages to the appropriate control logic blocks. The received MQTT message is formatted as a JSON object containing three primary fields: the CANopen identifier (CAN ID), the requested voltage reference, and the requested current limit.

To simplify downstream processing and ensure deterministic routing, a dedicated NodeRED function node—referred to as the Receive ID node—is used to normalize the incoming message payload. This node extracts the relevant fields from the JSON object and maps them into a compact internal representation consisting of three variables:

- x: CANopen identifier (profile selector);
- y: requested voltage reference;
- z: requested current limit.

This normalized representation allows subsequent supervisory logic to remain independent of the original MQTT message format and enables the same processing pipeline to be reused across multiple charging profiles and multiple vehicles. Importantly, this parsing and normalization process does not modify charging parameters; it serves solely to support deterministic profile routing, validation, and supervisory coordination.

Box A6. Shows the NodeRED function node implementation for CANopen message parsing and normalization.

```
msg.payload = {
  x: msg.payload.can_id,
  y: msg.payload.voltage,
  z: msg.payload.current
};
return msg;
```

Appendix A.7. Charging Profile Selection and Relay Actuation via Modbus RTU

After the MQTT message has been received and decomposed into internal variables (x,y,z), as described in Section 3.3, the supervisory control layer selects the appropriate charging profile and initiates hardware configuration of the charger. This process is implemented in NodeRED by combining identifier-based routing logic with Modbus RTU-based relay actuation.

Specifically, the supervisory variable msg.payload.x, which represents the CANopen identifier of the connected vehicle, is evaluated by a set of dedicated function nodes corresponding to predefined charging profiles (0x500, 0x600, 0x700, and 0x800). Each function node performs a strict equality check. If the received CANopen identifier matches the assigned profile, the function output is asserted; otherwise, the message is discarded. This routing mechanism enforces mutual exclusivity among charging profiles and guarantees that only one charging configuration can be active at any given time.

Once a valid charging profile is selected, the corresponding trigger signal is forwarded to a predefined sequence of Modbus RTU write nodes that control the RS-485 relay modules. These relays are responsible for physically connecting the selected CC-CV control board and associated voltage and current configuration inside the charger. A short intentional delay (5 s) is introduced between relay actuation steps to mitigate inrush current, contact arcing, and transient instability during switching. The relay activation sequence is intentionally serialized and guarded by logical checks to avoid simultaneous switching or transient misconfiguration. This sequencing strategy reduces electrical stress and improves operational reliability during repeated profile transitions.

Through this mechanism, the system enforces hardware-level interlocking, ensuring that incorrect or conflicting charging configurations cannot be activated simultaneously. As a result, the abstract software-level selection of a charging profile based on the CANopen identifier is deterministically translated into a concrete hardware configuration via Modbus RTU. After relay actuation is completed, the system enters a stable charging state and proceeds with the measurement and monitoring phase of the charging process.

Box A7. Shows an example NodeRED routing function for the charging profile corresponding to CANopen identifier 0x500.

```
let can = msg.payload.x;
if (can == "0x500") { // Change (0x600, 0x700, 0x800)
  msg.payload = 1; // trigger signal Relay-RS485
  return msg;
}
// If not 0x500, block this path
return null;
```

Appendix B

In the main body of this paper, the experimental results focus on the 72 V 15 A charging configuration as a representative operating condition, where the charging behavior is visualized using a Grafana-based monitoring platform. To further demonstrate the generality and scalability of the proposed charging architecture, three additional charging configurations with different voltage and current ratings were evaluated using the same measurement, logging, and visualization pipeline based on InfluxDB and Grafana.

Figures A1–A3 present the Grafana-based time-series visualizations of voltage and current for the 24 V 5 A, 48 V 5 A, and 60 V 5 A charging systems, respectively. In all cases, the charging process consistently follows the expected constant-current–constant-voltage (CC–CV) profile. During the initial stage, the charging current remains approximately constant while the battery voltage gradually increases (CC mode). After the target voltage is reached, the system transitions smoothly to constant-voltage (CV) operation, during which the voltage is regulated and the charging current decreases progressively as the battery approaches full charge. These results confirm that the proposed CANopen–NodeRED–Modbus–InfluxDB–Grafana-based platform can be directly reused across different battery configurations without modification of the core control logic, while maintaining consistent energy monitoring and visualization behavior. These additional experiments demonstrate consistent behavior of the proposed architecture across multiple battery configurations, supporting its applicability beyond the representative case discussed in the main text.

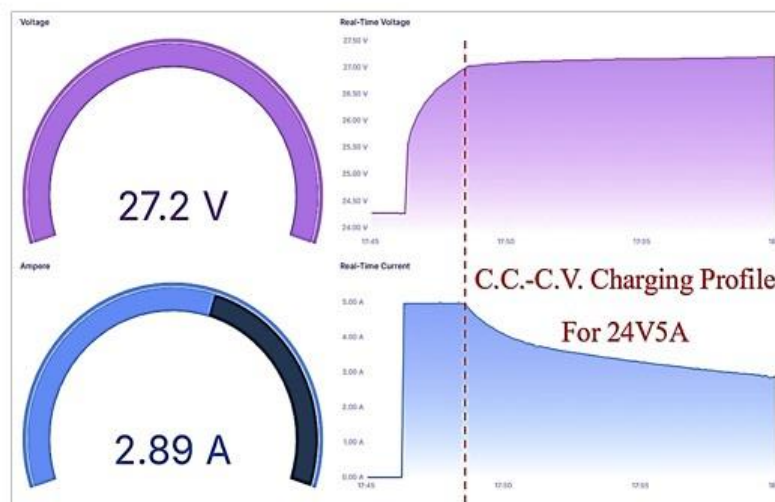


Figure A1. Grafana-based time-series visualization of voltage and current during the charging process for the 24 V 5 A configuration. The red dashed vertical line indicates the transition point from constant-current (CC) mode to constant-voltage (CV) mode.

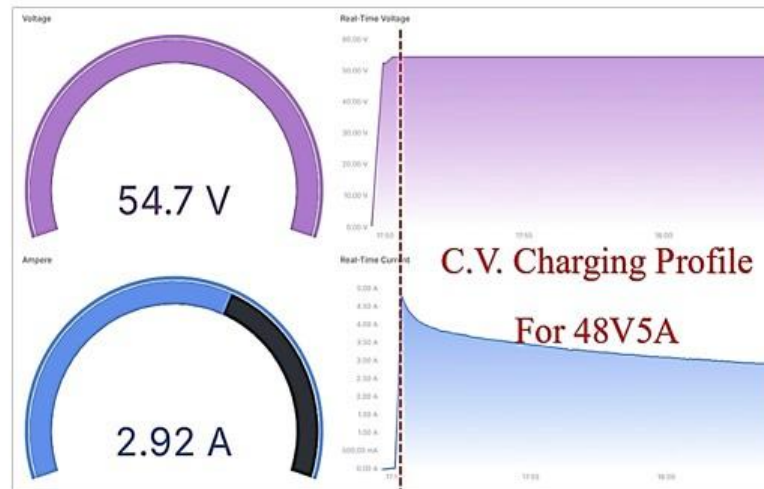


Figure A2. Grafana-based time-series visualization of voltage and current during the charging process for the 48 V 5 A configuration. The red dashed vertical line indicates the transition point from constant-current (CC) mode to constant-voltage (CV) mode.

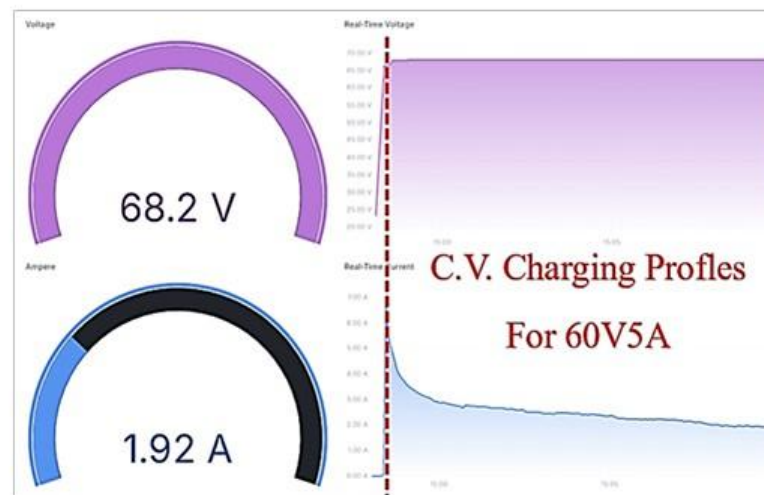


Figure A3. Grafana-based time-series visualization of voltage and current during the charging process for the 60 V 5 A configuration. The red dashed vertical line indicates the transition point from constant-current (CC) mode to constant-voltage (CV) mode.

References

1. Mateen, S.; Amir, M. Ultra-fast charging of electric vehicles: Power electronics converters, grid stability and optimal battery consideration in multi-energy systems. *Energy Rep.* **2023**, *9*, 1970–1988. [CrossRef]
2. Yenil, V.; Cetin, S. Load independent constant current and constant voltage control of LCC-series compensated wireless EV charger. *IEEE Trans. Power Electron.* **2022**, *37*, 8701–8712. [CrossRef]

3. Dar, A.R.; Haque, A.; Khan, M.A.; Kurukuru, V.S.B.; Mehfuz, S. On-board chargers for electric vehicles: A comprehensive review of technology, challenges, and prospects. *Energies* **2024**, *17*, 4534. [\[CrossRef\]](#)
4. Zentani, A.; Almaktoof, A.; Kahn, M.T. A Comprehensive Review of Developments in Electric Vehicles Fast Charging Technology. *Appl. Sci.* **2024**, *14*, 4728. [\[CrossRef\]](#)
5. Bhosale, P.; Sujil, A.; Kumar, R.; Bansal, R.C. Electric vehicle charging infrastructure, standards, types, and its impact on grid: A review. *Electr. Power Compon. Syst.* **2024**, *52*, 253–271. [\[CrossRef\]](#)
6. Zeballos, M.; Ferragut, A.; Paganini, F. Proportional fairness for EV charging in overload. *IEEE Trans. Smart Grid* **2019**, *10*, 6792–6801. [\[CrossRef\]](#)
7. Rivera, S.; Goetz, S.; Kouro, S. Charging Infrastructure and Grid Integration for Electromobility. *Proc. IEEE* **2023**, *111*, 371–396. [\[CrossRef\]](#)
8. Rafi, M.A.H.; Bauman, J. A Comprehensive Review of DC Fast-Charging Stations with Energy Storage: Architectures, Power Converters, and Analysis. *IEEE Trans. Transp. Electr.* **2021**, *7*, 345–368. [\[CrossRef\]](#)
9. Shahed, M.T.; Harun-ur Rashid, A.B.M. Battery charging technologies and standards for electric vehicles: State-of-the-art review, challenges, and future research prospects. *Energy Rep.* **2024**, *10*, 2212–2239. [\[CrossRef\]](#)
10. Kuperman, A.; Levy, U.; Goren, J.; Zafransky, A.; Savernin, A. Battery Charger for Electric Vehicle Traction Battery Switch Station. *IEEE Trans. Ind. Electron.* **2013**, *60*, 5391–5399. [\[CrossRef\]](#)
11. Shi, J.; Zeng, T.; Moura, S.J. Electric fleet charging management considering battery degradation and nonlinear charging profile. *Energy* **2023**, *283*, 129094. [\[CrossRef\]](#)
12. Maia, O.F.; Gärtner, N.; Stuckenschmidt, H. Data-driven smart charging for heterogeneous electric vehicle fleets. *Energy AI* **2020**, *1*, 100007. [\[CrossRef\]](#)
13. Kirchner, S.R. OCPP Interoperability: A Unified Future of Charging. *World Electr. Veh. J.* **2024**, *15*, 191. [\[CrossRef\]](#)
14. Hamdare, S.; Brown, D.J.; Cao, Y.; Aljaidi, M.; Kumar, S.; Alanazi, R.; Jugran, M.; Vyas, P.; Kaiwartya, O. A Novel Charging Management and Security Framework for the Electric Vehicle (EV) Ecosystem. *World Electr. Veh. J.* **2024**, *15*, 392. [\[CrossRef\]](#)
15. Rojas-Hernandez, I. Electric Vehicle Fleet control through OCPP deployment. 2022 IEEE 40th Central America and Panama Convention (CONCAPAN), Panama City, Panama, 9–12 November 2022; pp. 1–5. [\[CrossRef\]](#)
16. Hamdare, S.; Brown, D.J.; Jha, D.N.; Aljaidi, M.; Cao, Y.; Kumar, S.; Kharel, R.; Jugran, M.; Kaiwartya, O. Cyber defense in OCPP for EV charging security risks. *Int. J. Inf. Secur.* **2025**, *24*, 134. [\[CrossRef\]](#)
17. Tanyildiz, H.; Şahin, C.B.; Dinler, Ö.B.; Migdady, H.; Saleem, K.; Smerat, A.; Gandomi, A.H.; Abualigah, L. Detection of cyber attacks in electric vehicle charging stations. *Sci. Rep.* **2025**, *15*, 92895. [\[CrossRef\]](#)
18. Santos, J.B.; Francisco, A.M.B.; Cabrita, C.; Monteiro, J.; Pacheco, A.; Cardoso, P.J.S. Development and Implementation of a Smart Charging System for Electric Vehicles Based on the ISO 15118 Standard. *Energies* **2024**, *17*, 3045. [\[CrossRef\]](#)
19. Motlagh, S.G.; Oladigbolu, J.; Li, L. A review on electric vehicle charging station operation considering market dynamics and grid interaction. *Appl. Energy* **2025**, *326*, 126058. [\[CrossRef\]](#)
20. Dharangaonkar, S.K.; Patil, S.A. Planning of Electric Vehicle Charging Infrastructure: A Review and a Conceptual Framework Based on Spatial and Predictive Analysis. *Int. J. Sustain. Dev. Plan.* **2025**, *20*, 2317–2330. [\[CrossRef\]](#)
21. Ferretti, F.; De Paola, A. Machine learning identification of Electric Vehicles from charging session data. *Energy AI* **2025**, *20*, 100502. [\[CrossRef\]](#)
22. Pavković, D.; Kasać, J.; Krznar, M.; Cipek, M. Adaptive constant-current/constant-voltage charging of a battery cell based on cell open-circuit voltage estimation. *World Electr. Veh. J.* **2023**, *14*, 155. [\[CrossRef\]](#)
23. Wu, X.; Hu, C.; Du, J.; Sun, J. Multistage CC-CV charge method for Li-ion battery. *Math. Probl. Eng.* **2015**, *1*, 294793. [\[CrossRef\]](#)
24. Wang, W.; Deng, J.; Chen, D.; Wang, Z.; Wang, S. A Novel Design Method of LCC-S Compensated Inductive Power Transfer System Combining Constant Current and Constant Voltage Mode via Frequency Switching. *IEEE Access* **2021**, *9*, 117244–117256. [\[CrossRef\]](#)
25. Nichols, K.M.; Roembke, R.A.; Adamczyk, P.G. Real-Time Motor Control Using a Raspberry Pi, ROS, and CANopen over EtherCAT, with Application to a Semi-Active Prosthetic Ankle. *Actuators* **2025**, *14*, 84. [\[CrossRef\]](#)
26. Santhosh Kumar, P.; Kamath, R.N.; Boyapati, P.; Josephson, P.J.; Natrayan, L.; Shadrach, F.D. IoT battery management system in electric vehicle based on LR parameter estimation and ORMeshNet gateway topology. *Sustain. Energy Technol. Assess.* **2022**, *53*, 102696. [\[CrossRef\]](#)
27. Riza, M.Y.; Aji, P.; Priyasta, D.; Rijanto, E.; Priandana, E.R.; Yusivar, F.; Nurdiana, E. A Design of Smart Charging Architecture for Battery Electric Vehicles. In *Proceedings of the 2023 IEEE PES Innovative Smart Grid Technologies Europe (ISGT Europe 2023)*, Grenoble, France; IEEE: New York, NY, USA, 2023. [\[CrossRef\]](#)
28. Lobato, E.; Prazeres, L.; Medeiros, I.; Araújo, F.; Rosário, D.; Cerqueira, E.; Tostes, M.; Bezerra, U.; Fonseca, W. A Monitoring System for Electric Vehicle Charging Stations: A Prototype in the Amazon. *Energies* **2022**, *16*, 152. [\[CrossRef\]](#)
29. Martins, J.A.; Rodrigues, J.M.F. Intelligent Monitoring Systems for Electric Vehicle Charging. *Appl. Sci.* **2025**, *15*, 2741. [\[CrossRef\]](#)

30. Mohamad Nizam, M.S.; Abdullah, E.; Hidayat, N.M.; Hashim, N.M.Z. Real-Time Energy Monitoring in Renewable EV Charging Stations: An ESP32-Based System Integrating Modbus, MQTT, and ESP-NOW Protocols. In *2024 IEEE 22nd Student Conference on Research and Development (SCoReD)*; IEEE: New York, NY, USA, 2024. [\[CrossRef\]](#)
31. Tayri, A. Grid Impacts of Electric Vehicle Charging: A Review of the Effects of High EV Penetration and Mitigation Strategies. *Energies* **2025**, *18*, 3807. [\[CrossRef\]](#)
32. Mansour, H.S.E.; Samir, M.; Elhady, B.; Maksoud, S.A. Wireless Charging Systems for Electric Vehicles: Review. *Green Energy Intell. Transp.* **2025**, 100371. [\[CrossRef\]](#)
33. Peserico, G.; Morato, A.; Tramarin, F.; Vitturi, S. Functional Safety Networks and Protocols in the Industrial Internet of Things: Integration of Safety and Reliability Features. *Sensors* **2021**, *21*, 6073. [\[CrossRef\]](#)
34. Yanprechaset, C.; Donjaroennon, N.; Nuchkum, S.; Leeton, U. Development of an EV Battery Management Display with CANopen Communication. *World Electr. Veh. J.* **2025**, *16*, 375. [\[CrossRef\]](#)
35. Morrissey, P.; Weldon, P.; O'Mahony, M. Future standard and fast charging infrastructure planning: An analysis of electric vehicle charging behaviour. *Energy Policy* **2016**, *89*, 257–270. [\[CrossRef\]](#)
36. Acharige, S.S.G.; Haque, M.E.; Arif, M.T.; Hosseinzadeh, N.; Hasan, K.N.; Oo, A.M.T. Review of electric vehicle charging technologies, standards, architectures, and converter configurations. *IEEE Access* **2023**, *11*, 41218–41255. [\[CrossRef\]](#)
37. Khekert, E.V.; Malozyomov, B.V.; Klyuev, R.V.; Martyshev, N.V.; Konyukhov, V.Y.; Kukartsev, V.V.; Remezov, I.S. Battery Charging Simulation of a Passenger Electric Vehicle from a Traction Voltage Inverter with an Integrated Charger. *World Electr. Veh. J.* **2025**, *16*, 391. [\[CrossRef\]](#)
38. Johnson, J.; Berg, T.; Anderson, B.; Wright, B. Review of Electric Vehicle Charger Cybersecurity Vulnerabilities and Standard Protocols including ISO 15118 Plug-and-Charge. *Energies* **2022**, *15*, 3931. [\[CrossRef\]](#)
39. Chen, G.-J.; Wang, Y.-H.; Huang, Q.-Y.; Zhang, Z. Evaluation of Charging Methods for Lithium-Ion Batteries. *Electronics* **2023**, *12*, 4095. [\[CrossRef\]](#)
40. Ku, K.; Son, S.-B.; Gim, J.; Park, J.; Liang, Y.; Stark, A.; Lee, E.; Libera, J. Understanding the constant-voltage fast-charging process using a high-rate Ni-rich cathode material for lithium-ion batteries. *J. Mater. Chem. A* **2022**, *10*, 288–295. [\[CrossRef\]](#)
41. Li, W.; Yanyachi, A.; Xia, J.; Kuppan, S.; Zhou, J.; Wang, Y.; Li, Y.; Pianetta, P.; Zhao, K.; Mitlin, D.; et al. Mechanistic Considerations for Battery Charging Protocol Design. *Adv. Energy Mater.* **2025**, *16*, e04499. [\[CrossRef\]](#)
42. Dini, P.; Saponara, S.; Colicelli, A. Overview on Battery Charging Systems for Electric Vehicles: Architectures, Control Strategies, and Safety Considerations. *Electronics* **2023**, *12*, 4295. [\[CrossRef\]](#)
43. Bui, V.-T.; Dow, C.-R.; Huang, Y.-C.; Liu, P.; Thai, V.D. A CANopen-Based Gateway and Energy Monitoring System for Electric Bicycles. *Energies* **2020**, *13*, 3766. [\[CrossRef\]](#)
44. Urrea, C.; Morales, C.; Kern, J. Implementation of error detection and correction in the Modbus-RTU serial protocol. *Int. J. Crit. Infrastruct. Prot.* **2016**, *15*, 27–37. [\[CrossRef\]](#)
45. Yue, G. Design of Intelligent Monitoring and Control System Based on Modbus. In *Proceedings of the 2020 5th International Conference on Communication, Image and Signal Processing (CCISP)*, Chengdu, China, 13–15 November 2020; pp. 149–153. [\[CrossRef\]](#)
46. Gäitan, V.G.; Zagan, I. Enhancing the Modbus Communication Protocol to Minimize Acquisition Times Based on an STM32-Embedded Device. *Mathematics* **2022**, *10*, 4686. [\[CrossRef\]](#)
47. Ungurean, I. Integrating Fieldbus and Data-Centric Middleware: An STM32 Modbus Master Gateway for DDS-Based IIoT Systems. *Technologies* **2025**, *13*, 526. [\[CrossRef\]](#)
48. Xu, Y.; Zhang, H.; Liu, J. *Research on Model Mapping Method Between MODBUS and IEC 61850*; SPIE: Eindhoven, The Netherlands, 2023; p. 125960N. [\[CrossRef\]](#)
49. You, W.; Ge, H. Design and Implementation of Modbus Protocol for Intelligent Building Security. In *Proceedings of the 2019 IEEE 19th International Conference on Communication Technology (ICCT)*, Xi'an, China, 16–19 October 2019; pp. 420–423. [\[CrossRef\]](#)
50. Xylia, M.; Olsson, E.; Macura, B.; Nykvist, B. Estimating charging infrastructure demand for electric vehicles: A systematic review. *Energy Strategy Rev.* **2025**, *59*, 101753. [\[CrossRef\]](#)
51. Tole, S.; Wahono, T.; Jopri, M.H. Monitoring the Charging of a Lithium-Ion Battery from a Photovoltaic Source Using a PZEM-017 Sensor via the ESP32-Based Internet of Things. In *Proceedings of the 2025 12th International Conference on Electrical Engineering, Computer Science and Informatics (EECSI)*, Semarang, Indonesia, 25–26 September 2025. [\[CrossRef\]](#)
52. Wu, L. The Usage of Big Data in Electric Vehicle Charging. *Energies* **2025**, *18*, 5066. [\[CrossRef\]](#)
53. Burgio, A.; Cimmino, D.; Nappo, A.; Smarrazzo, L.; Donatiello, G. An IoT-Based Solution for Monitoring and Controlling Battery Energy Storage Systems at Residential and Commercial Levels. *Energies* **2023**, *16*, 3140. [\[CrossRef\]](#)
54. Rachid, A.; El Fadil, H.; Gaouzi, K.; Rachid, K.; Lassioui, A.; El Idrissi, Z.; Koundi, M. Electric Vehicle Charging Systems: Comprehensive Review. *Energies* **2023**, *16*, 255. [\[CrossRef\]](#)
55. Ucer, E.; Kisacikoglu, M. Development of a Hardware-in-the-Loop Testbed for a Decentralized, Data-Driven Electric Vehicle Charging Control Algorithm. *IEEE Trans. Ind. Appl.* **2024**, *60*, 6622–6633. [\[CrossRef\]](#)

56. Hildermeier, J.; Kolokathis, C.; Rosenow, J.; Hogan, M.; Wiese, C.; Jahn, A. Smart EV Charging: A Global Review of Promising Practices. *World Electr. Veh. J.* **2019**, *10*, 80. [\[CrossRef\]](#)
57. Torregrossa, D.; Le Boudec, J.-Y.; Paolone, M. Model-free computation of ultra-short-term prediction intervals of solar irradiance. *Sol. Energy* **2016**, *124*, 57–67. [\[CrossRef\]](#)
58. Rudnik, R.; Wang, C.; Reyes-Chamorro, L.; Achara, J.; Le Boudec, J.-Y.; Paolone, M. Real-time control of an electric-vehicle charging station while tracking an aggregated power-setpoint. *IEEE Trans. Ind. Appl.* **2020**, *56*, 5750–5761. [\[CrossRef\]](#)
59. Nutkani, L.; Toole, H.; Fernando, N.; Andrew, L.P.C. Impact of EV charging on electrical distribution networks: A comprehensive review. *IET Smart Grid* **2024**, *7*, 485–502. [\[CrossRef\]](#)
60. Hu, X.; Li, S.; Peng, H. A comparative study of equivalent circuit models for Li-ion batteries. *J. Power Sources* **2012**, *198*, 359–367. [\[CrossRef\]](#)
61. Wen, R.C.; Han, S.R.; Liang, L. Design of Electric Vehicle Battery Management System Based on CAN Bus. *Adv. Mater. Res.* **2012**, *546–547*, 272–277. [\[CrossRef\]](#)
62. Nitulescu, I.-V.; Korodi, A. Supervisory Control and Data Acquisition Approach in Node-RED: Application and Discussions. *IoT* **2020**, *1*, 76–91. [\[CrossRef\]](#)
63. Omidi, S.A.; Baig, M.J.A.; Iqbal, M.T. Design and Implementation of Node-Red Based Open-Source SCADA Architecture for a Hybrid Power System. *Energies* **2023**, *16*, 2092. [\[CrossRef\]](#)
64. Găitan, V.G.; Zagan, I.; Găitan, N.C. Modbus RTU Protocol Timing Evaluation for Scattered Holding Register Read and ModbusE-Related Implementation. *Processes* **2025**, *13*, 367. [\[CrossRef\]](#)
65. Sun, C.; Wang, L.; Zheng, J. Design and Development of Modbus/MQTT Gateway for Industrial IoT Applications. In Proceedings of the 2019 IEEE Conference on Automation, Hangzhou, China, 22–24 November 2019. [\[CrossRef\]](#)
66. Sánchez Ocaña, W.E.; Navas-Castellanos, A.; Salazar-Jácome, E.; Chancusig-Muso, C. Validation of the Functionality of an Industrial Network Based on RS-485 and Industrial Ethernet Protocols for Multivariable Processes. *Int. Rev. Autom. Control. (IREACO)* **2023**, *16*, 162–170. [\[CrossRef\]](#)
67. Syamsudin, M.S.; Riza, L.S.; Rasim, R. Energy Management of a Low-Cost Power Meter using ESP8266 and PZEM-016. *Int. J. Reg. Innov.* **2024**, *4*. [\[CrossRef\]](#)
68. Pawar, J.P.; Amirthaganesh, S.; ArunKumar, S.; Satish, K.B. Real-time energy measurement using smart meter. In Proceedings of the 2016 Online International Conference on Green Engineering and Technologies (ICGET), Coimbatore, India, 19 November 2016; pp. 1–5. [\[CrossRef\]](#)
69. Khattach, O.; Moussaoui, O.; Hassine, M. End-to-End Architecture for Real-Time IoT Analytics and Predictive Maintenance Using Stream Processing and Machine Learning Pipelines. *Sensors* **2025**, *25*, 2945. [\[CrossRef\]](#)
70. Garcia, A.; Oregui, X.; Franco, J.; Arrieta, U.; Ferreres, J.; Valencia, J.A. Time Series Manufacturing Data Edge Monitoring and Visualization to Support Industrial Maintenance Teams. *SN Comput. Sci.* **2023**, *5*, 131. [\[CrossRef\]](#)
71. Grzesik, P.; Mrozek, D. Comparative Analysis of Time Series Databases in the Context of Edge Computing for IoT and Smart Systems. In *Computational Science—ICCS 2020; Lecture Notes in Computer Science*; Springer: Berlin/Heidelberg, Germany, 2020. [\[CrossRef\]](#)
72. Alcaraz, C.; Cumplido, J.; Trivino, A. OCPP in the spotlight: Threats and countermeasures for electric vehicle charging infrastructures 4.0. *Int. J. Inf. Secur.* **2023**, *22*, 1395–1421. [\[CrossRef\]](#)
73. Hsaini, S.; Ghogho, M.; Charaf, M.E.H. An OCPP-Based Approach for Electric Vehicle Charging Management. *Energies* **2022**, *15*, 6735. [\[CrossRef\]](#)
74. Garofalaki, Z.; Kosmanos, D.; Moschyiannis, S.; Kallergis, D.; Douligeris, C. Electric Vehicle Charging: A Survey on the Security Issues and Challenges of the Open Charge Point Protocol (OCPP). *IEEE Commun. Surv. Tutor.* **2022**, *24*, 1504–1533. [\[CrossRef\]](#)
75. Alcaraz, C.; Lopez, J.; Wolthusen, S. OCPP Protocol: Security Threats and Challenges. *IEEE Trans. Smart Grid* **2017**, *8*, 2452–2459. [\[CrossRef\]](#)
76. Adamska, P.; Jużwin, M.; Wierzbicki, A. Picking Peaches or Squeezing Lemons: Selecting Crowdsourcing Workers for Reducing Cost of Redundancy. In *Computational Science—ICCS 2020; Krzhizhanovskaya, V., Závodszky, G., Lees, M.H., Dongarra, J.J., Sloot, P.M.A., Brissos, S., Teixeira, J., Eds.; Lecture Notes in Computer Science*; Springer: Cham, Switzerland, 2020; p. 12138. [\[CrossRef\]](#)

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.

ภาคผนวก ข

ซอร์สโค้ดสำหรับการสื่อสารข้อมูลผ่าน CANopen

การตั้งค่า Platform IO ของ Visual Studio Code

```
[env:esp32doit-devkit-v1]    // กำหนด Environment สำหรับบอร์ด ESP32
platform = espressif32       // เลือกแพลตฟอร์ม ESP32 ของ Espressif
board = esp32doit-devkit-v1  // ระบุรุ่นบอร์ดที่ใช้งาน
framework = arduino          // ใช้ Arduino Framework ในการพัฒนาโปรแกรม
monitor_speed = 115200       // กำหนด Baud rate สำหรับ Serial Monitor ใช้ดูค่า Debug
upload_speed = 115200        // กำหนดความเร็วในการอัปโหลดโปรแกรมลงบอร์ด
upload_port = COM4           // ระบุพอร์ตที่เชื่อมต่อกับ ESP32
lib_deps =                   // กำหนด Library ที่ต้องติดตั้งเพิ่มเติม
  adafruit/Adafruit SSD1306  // Library สำหรับควบคุมจอ OLED SSD1306
  adafruit/Adafruit GFX Library // Library สำหรับแสดงผลกราฟิก
  knolleary/PubSubClient      // Library สำหรับสื่อสาร MQTT
```

```
uint64_t chipID; // ตัวแปรเก็บหมายเลข Chip ID ของ ESP32
uint32_t can_id; // ตัวแปรเก็บหมายเลข CAN ID
//----- ตั้งค่าเริ่มต้น -----//
void setup() {
  // อ่าน Chip ID เพื่อกำหนด CAN ID ในการเลือกโปรไฟล์การชาร์จ
  chipID = ESP.getEfuseMac(); Serial.print("Chip ID: "); Serial.println(chipID, HEX);
  // ตรวจสอบ Chip ID เพื่อกำหนดหมายเลข CAN ID เพื่อประจำกล่องโปรไฟล์การชาร์จ
  if (chipID == 0xF4D1B21AD6B8) {
    can_id = 0x500;
  } else if (chipID == 0x1CB2F8BD9E7C) {
    can_id = 0x600;
  } else if (chipID == 0xC0E4B09EFC08) {
    can_id = 0x700;
  } else if (chipID == 0x48BFD1BD9E7C) {
    can_id = 0x800;
  } else {
    can_id = 0x000; // กรณีไม่พบ Chip ID ที่กำหนด
  }
}
```

//----- อ่านค่าแรงดันจาก Dip Switch -----//

```
int readVoltageSetting() {
  int b1 = digitalRead(S1); // อ่านสถานะสวิตช์ S1
  int b2 = digitalRead(S2); // อ่านสถานะสวิตช์ S2
  int b3 = digitalRead(S3); // อ่านสถานะสวิตช์ S3
```

```

// รวมค่าบิตจาก Dip Switch
int val = (b1 << 2) | (b2 << 1) | b3;
// กำหนดค่าแรงดันตามตำแหน่งสวิตช์
switch (val) {
    case 0b000: return 12;    case 0b001: return 24;
    case 0b010: return 36;    case 0b011: return 48;
    case 0b100: return 60;    case 0b101: return 72;
    case 0b110: return 84;    case 0b111: return 100;
    default: return -1; // กรณีอ่านค่าไม่ได้
}
}

```

```

//----- อ่านค่ากระแสจาก Dip Switch -----//
int readCurrentSetting() {
    int b4 = digitalRead(S4); // อ่านสถานะสวิตช์ S4
    int b5 = digitalRead(S5); // อ่านสถานะสวิตช์ S5
    int b6 = digitalRead(S6); // อ่านสถานะสวิตช์ S6
    // รวมค่าบิตจาก Dip Switch
    int val = (b4 << 2) | (b5 << 1) | b6;
    // กำหนดค่ากระแสตามตำแหน่งสวิตช์
    switch (val) {
        case 0b000: return 5;    case 0b001: return 10;
        case 0b010: return 15;   case 0b011: return 20;
        case 0b100: return 25;   default: return -1; // กรณีอ่านค่าไม่ได้
    }
}

```

```

//----- การส่งค่า CAN ID ด้วยโปรโตคอล CANopen -----//
float voltage = 0.00; // ตัวแปรเริ่มต้นเก็บค่าแรงดันไฟฟ้า
float current = 0.00; // ตัวแปรเริ่มต้นเก็บค่ากระแสไฟฟ้า

void loop() {
    //----- สร้างข้อความ CAN -----//
    twai_message_t tx_message;
    tx_message.identifier = can_id; // กำหนด CAN ID
    tx_message.extd = 1;           // ใช้ Extended ID ขนาด 29-bit CAN ID
    tx_message.rtr = 0;           // ส่งข้อมูลแบบปกติ (Data Frame)
}

```

```

tx_message.data_length_code = 8; // ส่งข้อมูลจำนวน 8 Byte

//----- จัดเก็บข้อมูลลง Data -----//
memcpy(&tx_message.data[0], &voltage, 4); // เก็บค่าแรงดัน (Voltage) 4 ไบต์
memcpy(&tx_message.data[4], &current, 4); // เก็บค่ากระแส (Current) 4 ไบต์

//----- ส่งข้อมูลผ่าน CAN -----//
if (twai_transmit(&tx_message, pdMS_TO_TICKS(100)) == ESP_OK) {
    Serial.print("Sent CAN ID: 0x");
    Serial.print(can_id, HEX);
    Serial.print(" V=");
    Serial.print(voltage);
    Serial.print(" A=");
    Serial.println(current);
} else {
    Serial.println("Failed to transmit"); // กรณีส่งข้อมูลไม่สำเร็จ
}
}

```

```

void loop() {
//----- การรับค่า CAN ID ด้วยโปรโตคอล CANopen -----//
twai_message_t rx_message;
float rx_voltage = 0.0; // ตัวแปรเก็บค่าแรงดันที่รับมา
float rx_ampere = 0.0; // ตัวแปรเก็บค่ากระแสที่รับมา
uint32_t rx_id = 0; // ตัวแปรเก็บ CAN ID

//----- ตรวจสอบการรับข้อมูล -----//
if (twai_receive(&rx_message, pdMS_TO_TICKS(100)) == ESP_OK) {
    rx_id = rx_message.identifier; // อ่านค่า CAN ID
    if (rx_message.data_length_code == 8) {
// แปลงข้อมูลจาก Data เป็น float
        memcpy(&rx_voltage, &rx_message.data[0], 4);
        memcpy(&rx_ampere, &rx_message.data[4], 4);
// ตรวจสอบ CAN ID ของแต่ละกล่อง
        if (rx_id == 0x500 || rx_id == 0x600 || rx_id == 0x700 || rx_id == 0x800) {

```

```

        Serial.print("Received from CAN ID: 0x");
        Serial.print(rx_id, HEX);
        Serial.print(" V=");
        Serial.print(rx_voltage);
        Serial.print(" A=");
        Serial.println(rx_ampere);
    } else {
        Serial.print("Unknown CAN ID: 0x");
        Serial.println(rx_id, HEX);
    }
} else {
    Serial.println("Invalid data length");    // ข้อมูลไม่ครบ 8 Byte
}
} else {
    Serial.println("No data received");        // ไม่พบข้อมูลที่รับเข้ามา
} delay(2000);                                // หน่วงเวลาก่อนวนลูปรอบถัดไป
}

```

//----- ตั้งค่าการเชื่อมต่อ WiFi และ MQTT -----//

```

void setup() {
    setupWiFi(); // เชื่อมต่อเครือข่าย WiFi
    // กำหนด MQTT Server และ Port
    client.setServer(mqtt_server, 1883);
}

void loop() {
    //----- ตรวจสอบการเชื่อมต่อ MQTT -----//
    if (!client.connected()) {
        reconnectMQTT(); // เชื่อมต่อ MQTT ใหม่กรณีหลุด
    }
    client.loop(); // รักษาสถานะการเชื่อมต่อ MQTT

    //----- ตรวจสอบข้อมูล CAN -----//
    if (dataReceived) {
        // สร้างข้อความ JSON สำหรับส่ง MQTT
        String payload = "{";
        payload += "\"can_id\": \"" + String(rx_id, HEX) + "\", ";
        payload += "\"voltage\": " + String(rx_voltage) + ", ";
    }
}

```

```

    payload += "\"current\":" + String(rx_ampere);
    payload += "};

//----- ส่งข้อมูลไปยัง MQTT Broker -----//
    if (client.publish(mqtt_topic, payload.c_str())) {
        Serial.println("MQTT message sent");           // ส่งข้อมูลสำเร็จ
    } else {
        Serial.println("Failed to send MQTT message"); // ส่งข้อมูลไม่สำเร็จ
    }
}

delay(5000); // หน่วงเวลา 5 วินาทีก่อนวนลูปรอบถัดไป
}

```