FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Design and Implementation of a Battery Management System for a Formula Student

Pedro Ricardo Pinheiro Vieira

Mestrado em Engenharia Eletrotécnica e de Computadores

Supervisor: Prof. Agostinho Afonso da Rocha

Co-Supervisor: Prof. Jorge Pinto

July 26, 2025

© Pedro Vieira, 2025

Resumo

Esta dissertação apresenta o design, desenvolvimento e implementação de um sistema de gestão de baterias (BMS) personalizado para um veículo elétrico (EV) de competição da *Formula Student* (FS), desenvolvido pela equipa da *Formula Student* da Faculdade de Engenharia da Universidade do Porto (FSFEUP), em colaboração com a entidade Vasco da Gama CoLab. O principal objetivo consistiu no desenvolvimento de uma solução otimizada que superasse as limitações dos sistemas comerciais atualmente disponíveis, com foco na estimativa precisa do estado de carga (SoC) das células de lítio (Li-ion) *Molicel P28A* empregadas na bateria do veículo. A abordagem adotada incluiu a modelagem do comportamento dinâmico das células por meio de um circuito equivalente de *Thevenin* de primeira ordem, e a estimativa de SoC foi realizada utilizando um *Extended Kalman Filter* (EKF), proporcionando uma solução adaptada às exigências de alta precisão e desempenho. A validação do sistema foi realizada por meio de testes experimentais, incluindo perfis de carga estáticos e dinâmicos que simularam condições reais de operação. A dissertação também detalha a seleção de componentes, o design do sistema e a integração com as regulamentações da *Formula Student Germany* (FSG), assegurando que o sistema atenda aos rigorosos requisitos técnicos. As principais contribuições deste trabalho incluem o aprimoramento da segurança, da integrabilidade e da confiabilidade do sistema, além de um desempenho superior na gestão e controlo da bateria, demonstrando a viabilidade de soluções BMS personalizadas para ambientes de alta performance e regulamentações desafiadoras.

Palavras-chave: Veículo Elétrico (EV), *Formula Student* (FS), Sistema de Gestão de Bateria (BMS), Estado de Carga (SoC), Modelo de Bateria, *Extended Kalman Filter* (EKF), Integração de Sistema, Design do BMS.

Abstract

This dissertation presents the design, development, and implementation of a custom Battery Management System (BMS) for a Formula Student (FS) electric vehicle (EV), developed by the Formula Student team from the Faculty of Engineering of the University of Porto (FSFEUP), in collaboration with Vasco da Gama CoLab. The main objective was the development of an optimized solution that overcomes the limitations of currently available commercial systems, focusing on the accurate estimation of the State of Charge (SoC) of the Molicel P28A Lithium-ion (Li-ion) cells used in the vehicle's battery. The adopted approach included modeling the dynamic behavior of the cells using a first-order Thevenin equivalent circuit, and the SoC estimation was performed using an Extended Kalman Filter (EKF), providing a solution tailored to the demands for high precision and performance. The system was validated through experimental tests, including both static and dynamic load profiles that simulated real operating conditions. The dissertation also details the selection of components, system design, and integration with Formula Student Germany (FSG) regulations, ensuring that the system meets the rigorous technical requirements. The main contributions of this work include the enhancement of safety, integrability, and reliability of the system, as well as superior performance in battery management and control, demonstrating the feasibility of custom BMS solutions for high-performance environments and challenging regulations.

Keywords: Electric Vehicle (EV), Formula Student (FS), Battery Management System (BMS), State of Charge (SoC), Battery Model, Extended Kalman Filter (EKF), System Integration, BMS Design.

UN Sustainable Development Goals

The United Nations Sustainable Development Goals (SDGs) establish a universal framework to guide efforts towards a more inclusive, equitable, and sustainable future. Comprised of 17 interlinked goals, this global agenda addresses pressing challenges such as poverty, inequality, climate change, clean energy access, and environmental degradation.

This dissertation, through the design and implementation of a custom BMS for an electric race vehicle, aligns most directly with two of these goals: SDG 7 (Affordable and Clean Energy), and SDG 8 (Decent Work and Economic Growth).

SDG 7 Ensure access to affordable, reliable, sustainable and modern energy for all

SDG 8 Promote sustained, inclusive and sustainable economic growth, full and productive employment and decent work for all

The alignment between this work and specific SDG targets is summarized in the table below:

SDG	Target	Contribution	Performance Indicators and Metrics
7	7.1	Development of a custom BMS supports the transition to electric mobility by improving energy efficiency, safety, and battery lifespan in EV systems.	Increase in energy efficiency of the accumulator, successful integration in FSFEUP 02 and validation through dynamic load testing.
	7.a	The project fosters collaboration between academia and industry (Vasco da Gama CoLab), promoting innovation and research in clean energy technologies.	Partnerships established, embedded algorithm development and open-source modular architecture.
8	8.2	In-house development of advanced energy systems strengthens technical expertise and local innovation capacity, reducing reliance on high-cost imported solutions.	45.6% cost reduction versus commercial BMS, fully embedded EKF estimator and new hardware development cycle.
	8.6	The project contributes to youth skills development and engineering training, enabling student involvement in real-world energy systems engineering.	Student-led research, prototype tested and deployed in FSFEUP 02 and thesis-based knowledge transfer.

Acknowledgements

I would like to begin by expressing my heartfelt gratitude to my supervisor and co-supervisor, Prof. Agostinho Rocha and Prof. Jorge Pinto, for their unwavering support and guidance throughout the dissertation. His insightful direction made it possible to complete this project, and I truly appreciate the time and effort he dedicated to helping me.

A special thank you to Vasco da Gama CoLab for being a fundamental support throughout this journey, providing all the necessary resources for the successful completion of my dissertation. A unique thanks to Bernardo André for his availability, patience, and the invaluable knowledge he shared. His attention to detail and dedication to the project were crucial to its success, and I am incredibly grateful for everything.

To my mother, thank you for your constant support, care, and even your stubbornness, which made all the difference in the end. I know I've given you plenty of reasons to worry, but I also know your persistence was the secret ingredient that pushed me to keep going! I can't even imagine where I'd be without you.

To my father, from the bottom of my heart, thank you for the tremendous effort you've made so that I could achieve my goals. You've always been proud of my accomplishments, motivating me every day to do my best, no matter what. You instilled in me the values and principles that helped shape me into the man I am today. You've taught me that a little bit of hard work can make all the difference. You've always been the one to remind me that hard work pays off.

To my sister, the one who has been the biggest influence in shaping me into the person I am today, thank you for always being there, offering the best advice, and pushing me to be better. You've been the driving force in my personal development, helping me stay disciplined and focused. You made me realize that success is worth the sacrifices, and you've truly instilled in me the hunger to succeed. A huge thank you to my 'crybaby' sister, and yes, I know you're going to read this, and I couldn't be more grateful!

A big thank you to my godparents, uncles, and cousins who have supported me and cared about my journey along the way. Your presence has been a comfort and a motivating factor in all that I've done.

To my friends who accompanied me throughout this journey, especially the Arroteia, the Bar da Praia, and the FSFEUP team, thank you for sticking by me through both the highs and the lows. Together, we solved the possible and the impossible. Without you, this journey wouldn't have been the same.

Lastly, I'd like to leave with a final thought, the journey hasn't ended, it's only just begun. It's from the values that were cultivated during this process that I will continue to grow, make my mark, and show the results of all the hard work. Now is when I truly start to become an Engineer!

Pedro Vieira

"For sure it's a clear ambition to show ourselves and the world what we're capable of"

Christian von Koenigsegg

Contents

1 Introduction **3**
- 1.1 Problem and Context . . . 3
- 1.2 FSFEUP Context . . . 4
- 1.3 Vasco da Gama CoLab Context . . . 5
- 1.4 Objectives of the Dissertation . . . 5
- 1.5 Thesis Structure . . . 6

2 State of the Art **7**
- 2.1 BMS in Li-ion batteries . . . 7
- 2.2 System-Level Architecture of a BMS . . . 8
  - 2.2.1 Introduction and Classification Overview . . . 8
  - 2.2.2 Functional Overview . . . 9
  - 2.2.3 Topological Classifications . . . 17
  - 2.2.4 Design Approaches: Off-the-Shelf vs Custom . . . 22
- 2.3 Review and evaluation of existing BMS Solutions . . . 24
- 2.4 SoC estimation techniques . . . 25
  - 2.4.1 Direct Methods . . . 26
  - 2.4.2 Open Circuit Voltage (OCV) . . . 27
  - 2.4.3 Electrochemical Methods (EChMs) . . . 27
  - 2.4.4 Kalman Filters (KFs) . . . 28
  - 2.4.5 Linear KF . . . 28
  - 2.4.6 Extended Kalman Filter (EKF) . . . 30
  - 2.4.7 Unscented Kalman Filter (UKF) . . . 30
  - 2.4.8 Other methods . . . 31
- 2.5 Battery Models . . . 32
  - 2.5.1 Equivalent Circuit Models (ECM) . . . 32
  - 2.5.2 Electrochemical Mechanism Models . . . 33
  - 2.5.3 Data-Driven Models . . . 34
  - 2.5.4 Battery model comparion . . . 34

3 Conceptual Design and Architecture of the BMS **35**
- 3.1 Limitations of Commercial Off-the-Shelf (COTS) Solutions . . . 35
  - 3.1.1 Insufficient Regulatory Compliance and Integration Constraints . . . 35
  - 3.1.2 Limited Functional and Mechanical Integration . . . 37
  - 3.1.3 Estimator Performance and Algorithmic Rigidity . . . 38
  - 3.1.4 Cost-Performance Trade-offs in Feature-Rich BMS Architectures . . . 39
- 3.2 Design Objectives and Constraints . . . 39
- 3.3 System Requirements Specification . . . 41

3.4 Compliance with Formula Student Germany (FSG) Regulations . . . 42
3.5 High-Level Architecture Selection . . . 43
3.6 Hardware Component Selection . . . 44
3.6.1 Monitoring IC . . . 44
3.6.2 Current Sensing . . . 44
3.6.3 PMU Microcontroller and Communication Interface . . . 44
3.7 SoC Estimation Strategy Selection . . . 45
3.8 Integrated System Architecture Diagram and Functional Mapping . . . 46
3.9 Conceptual Comparison Between Orion BMS 2 and the Proposed Custom BMS . . . 47

4 Cell Modeling and Parameter Identification 49
4.1 Model Selection: Thevenin-Based Equivalent Circuit . . . 49
4.2 Parameter Identification Using the HPPC Protocol . . . 51
4.3 Experimental Setup and Data Acquisition . . . 52
4.4 OCV–SoC Curve Identification . . . 53
4.5 Parameter Summary and Model Validation . . . 54

5 SoC estimation using an EKF 56
5.1 Mathematical formulation of the EKF . . . 56
5.1.1 From continuous-time dynamics to discrete-time estimation . . . 57
5.1.2 Prediction and correction in the EKF framework . . . 58
5.1.3 Process noise covariance matrix Q_k . . . 59
5.1.4 Measurement noise covariance matrix R_k . . . 60
5.1.5 Estimation error covariance matrix P_k . . . 60
5.1.6 Selection of the covariance matrices . . . 61
5.2 Experimental validation and performance evaluation . . . 63
5.2.1 Hardware setup . . . 63
5.2.2 Test Procedures . . . 64
5.2.3 Constant Current Discharge at 0.5C . . . 65
5.2.4 Constant Current Discharge at 1C . . . 67
5.2.5 Dynamic Load Validation for FS Autocross Scenario . . . 69

6 Final prototype and integration into the FSFEUP 02 73
6.1 Product overview . . . 73
6.2 Critical evaluation of objective fulfillment . . . 76
6.2.1 Fulfilled objectives . . . 76
6.2.2 Partially fulfilled objectives . . . 78
6.2.3 Unfulfilled objectives . . . 78

7 Conclusions and Future Work 80
7.1 Conclusions . . . 80
7.2 Future Work . . . 81

References 82

A Modeling and Embedded Implementation Code 86
A.1 MATLAB Script for Cell Model Parameterization . . . 86
A.2 C Code for Real-Time EKF Execution on Teensy 4.1 . . . 88
A.3 Python Script for Simulation and Hardware Comparison . . . 91

B Custom Printed Circuit Board (PCB) Schematic and Design **95**
B.1 BMS Slave PCB 95
B.2 BMS Master PCB 102
B.3 Stack PCB 103

List of Figures

1.1 FSFEUP team members, from [4] . . . 4

2.1 Functional architecture of a BMS, from [9] . . . 8
2.2 Categorization of BMSs, adapted from [10] . . . 9
2.3 Functional decomposition of a BMS, from [11]. . . . 10
2.4 Functional decomposition of a BMS, adapted from [12]. . . . 10
2.5 Internal architecture of the LTC6811 battery monitoring IC, schematic excerpted from manufacturer datasheet [15]. . . . 11
2.6 Shunt module from Isabellenhütte (IVT series), from [18] . . . 12
2.7 Reference shutdown circuit architecture as defined by FSG 2025 rules, from [1]. . . . 14
2.8 Simplified schematic of the TS protection architecture, including AIRs, IMD, precharge and discharge circuit, adapted from [12] . . . 15
2.9 Classification of Cell Balancing Methods, from [23] . . . 17
2.10 Centralized BMS topology, adapted from [14], [25] . . . 19
2.11 Modular BMS topology, adapted from [14], [25] . . . 20
2.12 Master–slave BMS topology, adapted from [14], [25] . . . 21
2.13 Distributed BMS topology . . . 22
2.14 Custom BMS developed by Formula Electric Belgium. The slave board is mounted directly on top of the stack module, with the master controller shown on the right, from [27] . . . 24
2.15 Commercial off-the-shelf solution: Orion BMS 2, from [3] . . . 24
2.16 Key milestones in the evolution of battery SoC estimation methods [21]. . . . 26
2.17 Workflow of the LKF for SoC estimation [20]. . . . 29
2.18 Workflow of the EKF for SoC estimation [20]. . . . 30
2.19 Workflow of the UKF for SoC estimation [20]. . . . 31
2.20 Examples of equivalent circuit models. . . . 33
2.21 Typical EIS Spectrum of a Li-ion Cell, from [33] . . . 34

3.1 Accumulator interior of the 02 prototype, illustrating the physical integration burden imposed by the Orion BMS 2 . . . 36
3.2 Internal view of the accumulator highlighting the Orion BMS 2 volume and wiring density . . . 38
3.3 Battery Pack using Molicel P28A cells . . . 42
3.4 System-level architecture of the proposed BMS . . . 47

4.1 Typical HPPC response used for parameter identification, from [38]. . . . 51
4.2 Voltage profile of HPPC test . . . 53
4.3 Zoomed-in HPPC segment showing step, drop, and relaxation. . . . 54

4.4 Fitted fifth-order polynomial curves for the cell parameters and OCV. These functions parameterize the Thevenin model over the SoC range. 55

5.1 Experimental setup used for EKF validation, including Teensy 4.1, LTC6811, and a 6-cell Li-ion pack. 64
5.2 EKF SoC estimation under 0.5C constant current discharge. 65
5.3 EKF SoC estimation vs. simulated under 0.5C constant current discharge. 65
5.4 Percentage error between measured and estimated voltage under 0.5C test conditions. 66
5.5 Comparison between the measured voltage and the estimated voltage for 0.5C test. 66
5.6 Pointwise estimation error for 0.5C test. 67
5.7 EKF SoC estimation under 1C constant current discharge. 67
5.8 EKF SoC estimation vs. reference under 1C constant current discharge. 68
5.9 Percentage error between measured and estimated voltage under 1C test conditions. 68
5.10 Comparison between the measured voltage and the estimated voltage for 1C test. . 69
5.11 Pointwise estimation error for 1C test. 69
5.12 Current profile applied to the power supply to emulate the dynamic load conditions of a FS Autocross event. 70
5.13 Terminal voltage comparison (measured, Teensy-EKF estimate, and simulation) in the non-linear SoC region. 70
5.14 Voltage estimation error in the non-linear SoC region. 71
5.15 Terminal voltage comparison (measured, Teensy-EKF estimate, and simulation) in the linear SoC region. 71
5.16 Voltage estimation error in the linear SoC region. 72

6.1 3D render of the custom-designed PCBs 74
6.2 Final custom hardware developed for the BMS: Master, Slave, and stack interface PCBs. 74
6.3 Integration of the BMS hardware into the FSFEUP battery stack prototype. 75

B.1 BMS Slave Overview Diagram 95
B.2 Voltage Measurement Circuit (Page 1) 96
B.3 Connectors (Page 2) 97
B.4 IC Monitor (Page 3) 98
B.5 Cells Balancing Circuit (Page 4) 99
B.6 Temperature Measurement System (Page 5) 100
B.7 Temperature Auxiliary Measurements (Page 6) 101
B.8 BMS Master PCB Schematic 102
B.9 Stack PCB schematic 103

List of Tables

2.1 Comparison of Current Sensing Technologies in BMS Applications, adapted from [16] . . . 12
2.2 Comparison of SoC Estimation Methods [14], [19] . . . 16
2.3 Comparison of BMS Topologies, adapted from [14] . . . 22
2.4 Comparison between off-the-shelf and custom BMS design approaches . . . 23
2.5 Comparison between commercial BMS solutions (market survey, based on data from [3], [28]–[30]) . . . 25
2.6 Battery model comparison. Adapted from [35]. . . . 34

3.1 BMS Design Objectives and Constraints . . . 40
3.2 Battery Pack Configuration . . . 42
3.3 Comparison Between Orion BMS 2 and the Proposed Custom BMS . . . 48

4.1 Summary of the Identified Thevenin Model Parameters . . . 54

6.1 Summary of BMS objective fulfillment . . . 79

Abbreviations and Symbols

ADC	Analog-to-Digital Converter
AIRs	Accumulator Isolation Relays
ANNs	Artificial Neural Networks
BMS	Battery Management System
CAN	Controller Area Network
CC	Coulomb Counting
CMU	Cell Monitoring Unit
COTS	Commercial Off-the-Shelf
ECM	Equivalent Circuit Models
EChMs	Electrochemical Models
EIS	Electrochemical Impedance Spectroscopy
EKF	Extended Kalman Filter
ESF	Electrical System Form
EV	Electric Vehicle
FL	Fuzzy Logic
FS	Formula Student
FSG	Formula Student Germany
FSFEUP	Formula Student team from the Faculty of Engineering of the University of Porto
GA	Genetic Algorithms
HPPC	Hybrid Pulse Power Characterization
ICs	Integrated Circuits
IMD	Insulation Monitoring Device
isoSPI	Isolated Serial Peripheral Interface
KFs	Kalman Filters
LKF	Linear Kalman Filter
Li-ion	Lithium-ion
MMU	Module Management Unit
NTC	Negative Temperature Coefficient
OCV	Open Circuit Voltage
PCB	Printed Circuit Board

PNGV	Partnership for a New Generation of Vehicles
PMU	Pack Management Unit
PSO	Particle Swarm Optimization
RC	Resistor–Capacitor
RTOS	Real-Time Operating Systems
SDGs	Sustainable Development Goals
SoC	State of Charge
SoH	State of Health
SPI	Serial Peripheral Interface
TS	Tractive System
UART	Universal Asynchronous Receiver/Transmitter
UKF	Unscented Kalman Filter
USB	Universal Serial Bus
ZOH	Zero-Order Hold

Chapter 1

Introduction

This introductory chapter contextualizes the motivation behind the project and outlines the challenges posed by previous hardware limitations, particularly those encountered with commercial off-the-shelf (COTS) solutions. It then situates the work within the organizational scope of FSFEUP and its partnership with Vasco da Gama CoLab. Finally, the chapter defines the technical objectives of the dissertation and summarizes its structure, providing the reader with a road map of the methodological.

1.1 Problem and Context

The rapid expansion of electric mobility has placed unprecedented demands on battery system reliability, safety, and intelligent control. Within this landscape, BMS play an important role, not only as monitoring tools, but as critical enablers of system-level integration, performance optimization, and regulatory compliance. These challenges are amplified in the context of FS, where vehicles operate under strict design constraints and must conform to a rigorous set of safety standards, such as those defined by FSG. Here, the BMS must satisfy specific timing thresholds for fault response, enforce real-time shutdown under thermal or electrical stress, and provide full observability of the accumulator system for inspection and validation purposes [1], [2].

In previous competition cycles, the FSFEUP team adopted a COTS BMS, the Orion BMS 2, as the supervisory unit for its battery system. While this platform provided a degree of maturity and integration convenience, it proved suboptimal in addressing the functional and regulatory requirements of a high-performance, student-designed prototype. Key limitations included a fixed Coulomb Counting (CC) estimation method, which suffered from drift and lacked robustness under dynamic loading; temperature sensing limited to 8 channels, insufficient for the mandated 30% cell coverage, and a centralized architecture that introduced significant harnessing complexity, degraded signal integrity, and complicated mechanical packaging [1], [3].

These shortcomings revealed a critical gap between the capabilities of generic commercial systems and the specific needs of different architecture and a regulation constrained racing application. More importantly, they exposed a lack of adaptability in COTS platforms to incorporate custom

estimation logic, interface with stack-level safety systems, and evolve alongside the team's design iterations. Addressing these limitations required a clean-slate design of a custom BMS architecture: one that could embed advanced estimation algorithms, reduce internal wiring through distributed sensing, meet all regulatory constraints, and adapt to the unique topology of the FSFEUP accumulator.

1.2 FSFEUP Context

FS is a globally recognized engineering competition where university teams are challenged to design, manufacture, and validate a formula-style race car in accordance with strict technical regulations. The competition includes both *Static Events*, which evaluate the engineering design, cost structure, and business strategy, and *Dynamic Events*, which assess the vehicle's performance on track through acceleration, handling, endurance, and energy efficiency tests. For teams developing autonomous-capable vehicles, an additional dynamic challenge is introduced: the *Trackdrive* event, where the car must complete laps without human input, emphasizing reliability, perception, and control system performance.

FSFEUP is the FS team of the University of Porto, founded in 2021 during the COVID-19 pandemic. Initially composed of a small group of motivated students, the team has grown into an 80 member organization spanning various engineering disciplines and academic levels. All departments work toward a common objective: to develop a fully electric race car, designed and built from scratch, with autonomous driving capabilities.

Figure 1.1: FSFEUP team members, from [4]

After two years of iterative development, FSFEUP launched its first fully electric prototype, "FSFEUP 01", in 2023. The car successfully competed in several international events, including FSG, where it made a promising debut. Motivated by this result, the team initiated the development

of a second prototype, "FSFEUP 02", with the ambition to integrate autonomous driving capabilities over a two-year cycle (2024–2025). This step is strategically relevant, as it enhances both static and dynamic event scoring potential, particularly in autonomous-specific events such as *Trackdrive* [1].

The present dissertation supports this evolution by contributing to the development of one of the vehicle's most safety-critical systems: a custom BMS. This effort marks a key milestone in FSFEUP's strategy to internalize core technological competencies and pursue global competitiveness.

1.3 Vasco da Gama CoLab Context

Vasco da Gama CoLab is a Portuguese research and innovation collaborative laboratory dedicated to advancing sustainable energy systems, intelligent mobility, and digital transition. As a key player in bridging academia and industry, the institution promotes applied research projects that align with national and European technological priorities. Its focus on electric mobility and embedded energy management systems makes it a natural partner for academic initiatives seeking to explore advanced battery technologies and vehicle electrification [5].

In the context of this dissertation, the collaboration with Vasco da Gama CoLab provided technical support throughout the modeling, implementation, and validation phases of the custom BMS. The partnership enabled access to laboratory infrastructure, specialized instrumentation, and expert mentoring, which significantly accelerated the development cycle. Furthermore, the collaboration helped ensure that the methods and results achieved were not only academically sound but also aligned with practical constraints, reinforcing the relevance and applicability of the work.

1.4 Objectives of the Dissertation

This dissertation aims to design, implement, and validate a custom embedded BMS that complies with regulations and is specifically tailored to the FSFEUP 02. The work was guided by the following technical objectives:

- Develop a custom BMS;
- Design and implement a real-time SoC estimator, supported by an experimentally parameterized battery model;
- Ensure full compliance with FSG electrical safety regulations;
- Validate the estimator and system behavior experimentally through hardware;
- Integrate the final BMS into the FSFEUP 02 battery pack and evaluate it relative to the previous COTS solution in terms of architecture, scalability and cost.

1.5 Thesis Structure

The following chapters of this dissertation are structured as follows.

Chapter 2 provides a comprehensive review of Li-ion battery systems and BMS technologies. It explores battery characteristics, the functional architecture of BMSs, and critical subsystems including monitoring, protection, estimation, and balancing. Various topological architectures and design methodologies are compared, with an emphasis on the trade-offs between commercial and custom solutions. The chapter concludes with a discussion on SoC estimation techniques and battery modeling strategies, particularly in the context of FS applications.

Chapter 3 details the conceptual design of the proposed BMS, including the rationale behind key design decisions, system architecture, and component selection. It identifies major limitations in the previously used commercial BMS and justifies the adoption of an alternative topology. The chapter also outlines technical requirements derived from FS regulations and explains how the new design addresses challenges related to integration, estimation, and modularity. The proposed solution is positioned as a scalable and regulation-compliant system tailored to the FSFEUP electric prototype.

Chapter 4 focuses on the modeling and parameter identification of the Li-ion cells used in the battery pack. A first-order Thevenin equivalent circuit model is selected for its compatibility with real-time estimation, and its parameters are experimentally determined using the Hybrid Pulse Power Characterization (HPPC) protocol. The model is validated across different SoC levels using empirical data, forming the foundation for the estimator presented in the following chapter.

Chapter 5 presents the development, implementation, and validation of an EKF for real-time SoC estimation. Building upon the Thevenin-based model developed earlier, the EKF is implemented in discrete time for embedded deployment. The chapter discusses tuning methodologies and covariance selection, and experimental validation using static and dynamic load profiles demonstrates the estimator's robustness and accuracy under varying current conditions.

Chapter 6 describes the final implementation of the custom BMS and its integration into the FSFEUP 02 prototype. The complete hardware architecture is presented, developed in a master–slave topology, along with details of mechanical, electrical, and communication integration. A comparative cost analysis highlights the economic benefits of the custom solution over commercial options. The chapter concludes with an evaluation of the project's objectives, confirming the system's compliance, performance, and overall design viability.

And finally, Chapter 7 summarizes the key outcomes of the project, highlighting the successful development and integration of a custom BMS into the FSFEUP 02 prototype. It reflects on the technical and economic advantages of a tailored solution compared to commercial alternatives and confirms the achievement of most defined objectives. Finally, it outlines future research directions, emphasizing the system's modularity and its potential as a foundation for continued innovation in embedded energy systems.

Chapter 2

State of the Art

This chapter provides a historical overview of EVs and the evolution of BMS, emphasizing their role in ensuring safety, performance, and longevity, it highlights the increasing complexity of BMS as a response to the growing demands of modern systems. Following this, it delves into the functionalities of a BMS, discussing trade-offs in implementation and design. Key features such as cell balancing methods, charge and health estimation techniques are reviewed, with particular emphasis on the SoC, due to its crucial role in performance and its planned implementation in this project. Furthermore, the chapter discusses common failure modes in battery systems and the protective mechanisms employed to mitigate these issues. Finally, the application of a BMS in an EV prototype is analyzed, providing context for the specific challenges and design considerations in this competitive environment. This comprehensive review forms the foundation for the subsequent development and implementation of the proposed BMS solution.

2.1 BMS in Li-ion batteries

Among the available electrochemical energy storage technologies, Li-ion batteries are currently regarded as the state-of-the-art for EV applications. This is due to their optimal balance of gravimetric and volumetric energy densities, extended operational lifespan, and high charge-discharge efficiency [6]. These characteristics make them especially suitable for high-performance and space-constrained environments such as FS electric prototypes. However, the electrochemical complexity of Li-ion cells imposes strict operational constraints. Their sensitivity to voltage excursions, excessive current, and thermal variation can lead to rapid degradation, thermal runaway, or even catastrophic failure if not rigorously controlled [6], [7].

Furthermore, cells within a battery pack often exhibit variability resulting from manufacturing tolerances, heterogeneous thermal exposure, and aging. These factors cause charge imbalance and unequal thermal behavior, increasing the likelihood of local overcharge or overdischarge events and, ultimately, reducing overall system efficiency and safety [7]. Given the variability and risk factors inherent to large-scale Li-ion battery assemblies, an intelligent supervisory system becomes essential to ensure stability and performance under all operating conditions.

To address these challenges and unlock the full performance potential of Li-ion technology, the integration of a BMS is indispensable. A BMS consolidates critical functionalities such as electrical parameter sensing, thermal regulation, charge equalization, and real-time fault handling [8]. Modern BMS architectures increasingly employ advanced estimation algorithms, such as Kalman filters (KFs) and neural network-based observers, to compute the SoC, State of Health (SoH), and remaining useful life in real time. These features enable predictive control, enhance operational safety margins, and support preventive maintenance strategies [8], [9].

The conceptual structure of a BMS is illustrated in Figure 2.1. It highlights the integration of sensing, control, estimation, and communication modules necessary to ensure the safe and efficient operation of Li-ion battery packs in EVs.

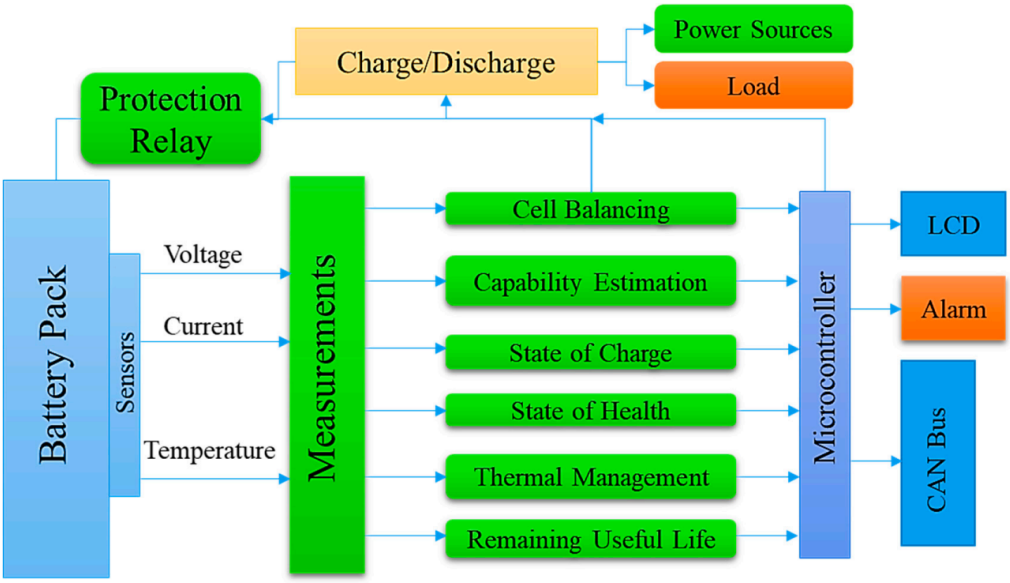


Figure 2.1: Functional architecture of a BMS, from [9]

2.2 System-Level Architecture of a BMS

2.2.1 Introduction and Classification Overview

BMSs are not monolithic components but modular and highly configurable systems whose architecture can vary significantly depending on the application domain, safety requirements, and performance targets. As EVs impose strict constraints in terms of space, reliability, and real-time operation, understanding how BMS architectures are structured becomes essential.

In order to evaluate and compare BMS implementations in a systematic way, it is helpful to classify them along three primary axes: functionality, topology, and design approach. This classification provides a structured foundation to analyze system complexity, modularity, scalability, and suitability for specific use cases.

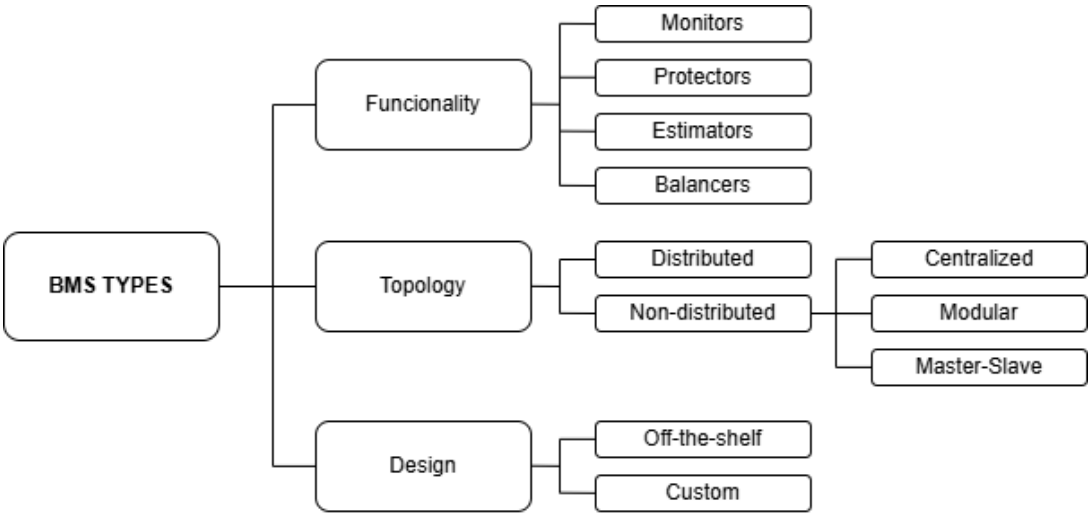


Figure 2.2: Categorization of BMSs, adapted from [10]

The first axis, functionality, refers to the extent of internal features present in the BMS, from simple voltage monitors to fully integrated systems capable of cell balancing, protection, estimators, diagnostics, and closed-loop control. The second axis, topology, distinguishes how the BMS is physically and logically organized across the battery pack: centralized, non-distributed, or fully distributed. The third axis concerns the design philosophy: whether the system is a COTS solution or a fully custom-developed architecture tailored to the application's unique requirements.

In the following sections, each of these dimensions is explored in depth, with a focus on how design choices influence system behavior, flexibility, and integration potential.

2.2.2 Functional Overview

A BMS is composed of several functional blocks that work together to ensure the safe and efficient operation of a Li-ion battery pack. Each block addresses a distinct set of requirements and constraints, encompassing everything from fundamental voltage and temperature monitoring to advanced estimation algorithms and system-level diagnostics. Figure 2.3 illustrates a typical decomposition of these responsibilities.

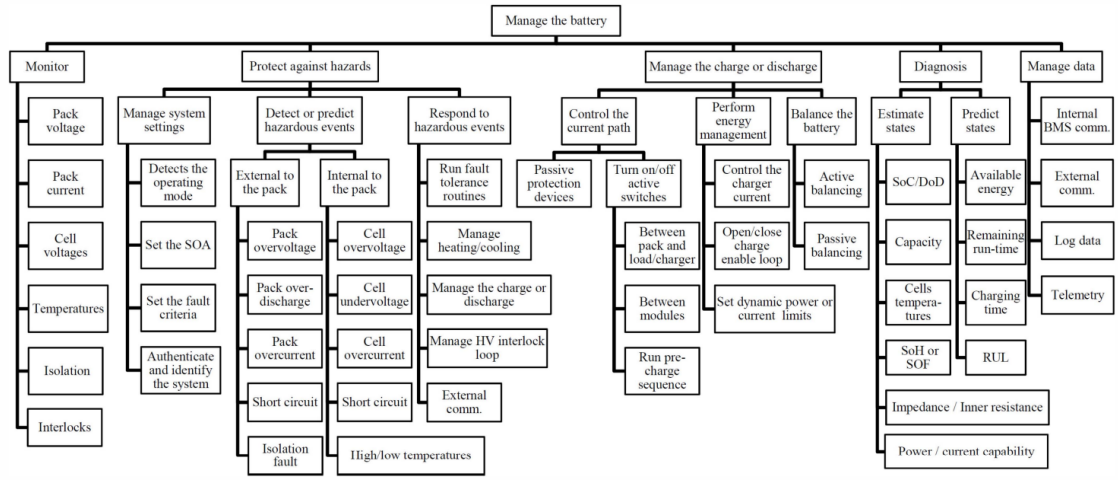


Figure 2.3: Functional decomposition of a BMS, from [11].

2.2.2.1 Monitoring Subsystem

As shown in Figure 2.4, the monitoring subsystem forms the foundation of any BMS, as it collects the real-time physical data needed for protection, estimation, and system control. The primary parameters monitored include individual cell voltages, pack-level current, and temperatures at critical points throughout the battery system [12].

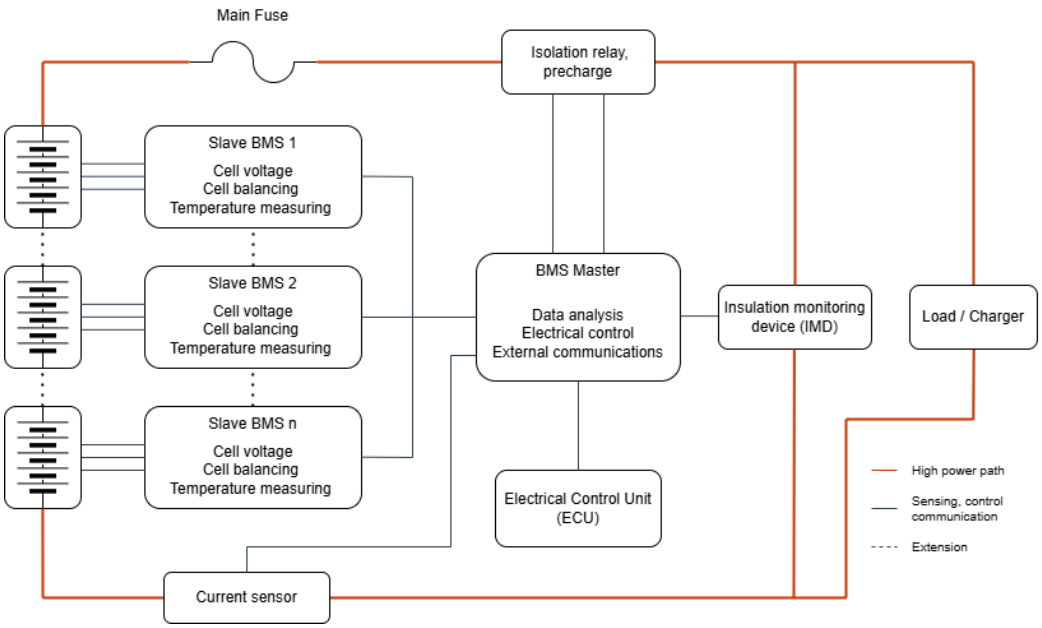


Figure 2.4: Functional decomposition of a BMS, adapted from [12].

Voltage measurement is typically achieved using analog multiplexers connected to a high-resolution Analog-to-Digital Converter (ADC), commonly integrated in specialized battery monitoring Integrated Circuits (ICs) such as the LTC6811 or BQ769x0. These ICs are widely adopted in

both industrial and academic BMS designs, including FS applications, due to their high-resolution ADCs, embedded protection logic, and daisy-chain communication protocols [13].

Given that Li-on cells typically operate between 2.5 V and 4.2 V, the monitoring subsystem must capture voltages within this range with high precision to ensure safe operation and reliable SoC estimation [14]. Voltage signals are commonly scaled through resistor dividers and filtered before being sampled by high-resolution ADCs integrated into monitoring ICs, as shown in Figure 2.5. These ICs, such as the LTC6811, support synchronized sampling and daisy-chaining, which are essential for modular pack configurations.

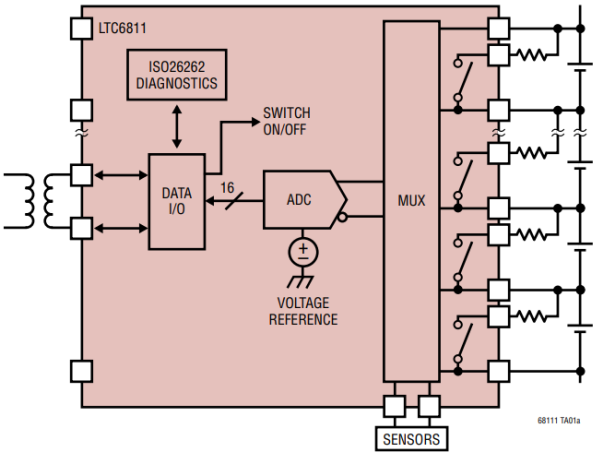


Figure 2.5: Internal architecture of the LTC6811 battery monitoring IC, schematic excerpted from manufacturer datasheet [15].

Accurate SoC estimation based on open circuit voltage (OCV) requires a voltage measurement resolution of at least 10 mV, particularly in the flat regions of the OCV–SoC curve. Although a 9-bit ADC could theoretically provide this resolution over a 5 V range, practical implementations must account for noise, offset and reference drift [14]. Therefore, high-resolution ADCs, typically 12 to 16 bits, are used to ensure reliable estimation across the full cell range.

The current sensor must support bidirectional measurement to account for all current flow directions, including discharging, regenerative charging during braking, and external charging from the accumulator charger. Current sensing is usually performed through either shunt resistors or Hall-effect sensors. Shunt-based designs offer high linearity and low cost, though they introduce thermal drift and lack isolation. Hall-effect sensors provide galvanic isolation and robustness but may suffer from zero-offset and lower precision. The choice depends on system-level trade-offs between cost, accuracy, isolation, integratability and thermal performance [16].

An increasingly adopted alternative in high-accuracy systems is the use of ultralow resistance precision shunt resistors, such as the ISA-WELD and SMx series developed by Isabellenhütte, illustrated in Figure 2.6 [17]. These components exhibit extremely low temperature coefficients, minimal inductance, and tight manufacturing tolerances, enabling high-precision current measurement and long-term thermal stability. Advanced models from the IVT and MPA series integrate

signal conditioning electronics and provide calibrated digital output, with some variants offering built-in Controller Area Network (CAN) Bus communication. This allows direct integration with a BMS without additional analog front-end circuitry [17]. However, due to their significantly higher cost compared to conventional shunts, their use is typically reserved for commercial EVs, aerospace systems, or high-end test benches, making them impractical for cost-sensitive applications such as FS.

Figure 2.6: Shunt module from Isabellenhütte (IVT series), from [18]

To better illustrate the trade-offs between the different current sensing approaches discussed, Table 2.1 summarizes the main characteristics of three commonly used technologies: standard shunt resistors, Hall-effect sensors, and advanced precision shunts such as those developed by Isabellenhütte. The comparison considers key factors such as isolation capability, measurement precision, interface type, and overall cost, each of which has a direct impact on system design and integration complexity. Indicative accuracy values are based on published datasheets and manufacturer application notes [16], [17].

Table 2.1: Comparison of Current Sensing Technologies in BMS Applications, adapted from [16]

Technology	**Isolation**	**Accuracy**	**Interface**	**Cost**
Standard Shunt Resistor	No	Medium	Analog	Low
Hall-Effect Sensor	Yes	Low–Medium	Analog	Medium
Isabellenhütte IVT-S (Smart)	Yes	High	Digital (CAN)	High

Temperature monitoring is typically implemented using Negative Temperature Coefficient (NTC) thermistors located near thermal hotspots or attached directly to cells. In safety-critical environments, including FSG, temperature data is not only used for protection logic but is also explicitly required by the competition's technical rules. Improper placement or sensor failure can result in

rejection during the technical inspection, where live sensor readings are validated through fault injection tests [1], [2].

From a regulatory perspective, the monitoring subsystem must comply with the requirements outlined in the Electrical System Form (ESF), a mandatory document submitted for FSG. The ESF details the number of sensors, their exact placement, the acquisition method, and the BMS response strategy in case of a fault. Technical inspectors verify that the BMS disconnects the accumulator via the shutdown circuit if voltage or temperature limits are exceeded [1], [2].

Furthermore, to ensure valid state estimation and effective protection, synchronised sampling across all channels is essential. This becomes especially relevant under the highly dynamic current profiles encountered in FS applications. Asynchronous measurements can lead to erroneous SoC estimation, misfiring protection triggers, or undetected faults [12], [14].

In short, the quality and reliability of the monitoring subsystem determine not only the BMS's effectiveness but also its compliance with both internal safety standards and external competition rules. The final selection and implementation of the components of this subsystem, reflecting the trade-offs discussed and tailored to the specific constraints of FS, such as packaging, budget and interface compatibility, will be detailed in Chapter 3.

2.2.2.2 Protection Layer

The protection layer of a BMS is responsible for ensuring that the battery pack operates within its defined safe operating area, preventing electrical or thermal conditions that could damage the cells or compromise system safety. A robust BMS must detect and respond to critical events such as overvoltage, undervoltage, overcurrent, and overtemperature with both speed and reliability [14], adopting a fail-safe architecture where safety-critical faults result in immediate system isolation to protect personnel and hardware.

The BMS plays a central role in fault detection and initiates protective actions by controlling or signalling to system-level devices such as the Accumulator Isolation Relays (AIRs), precharge circuit, and Insulation Monitoring device (IMD). Figure 2.7 presents the standard shutdown architecture defined by the FSG 2025 regulations [1], which the BMS must interface with to enforce a safe response upon fault conditions.

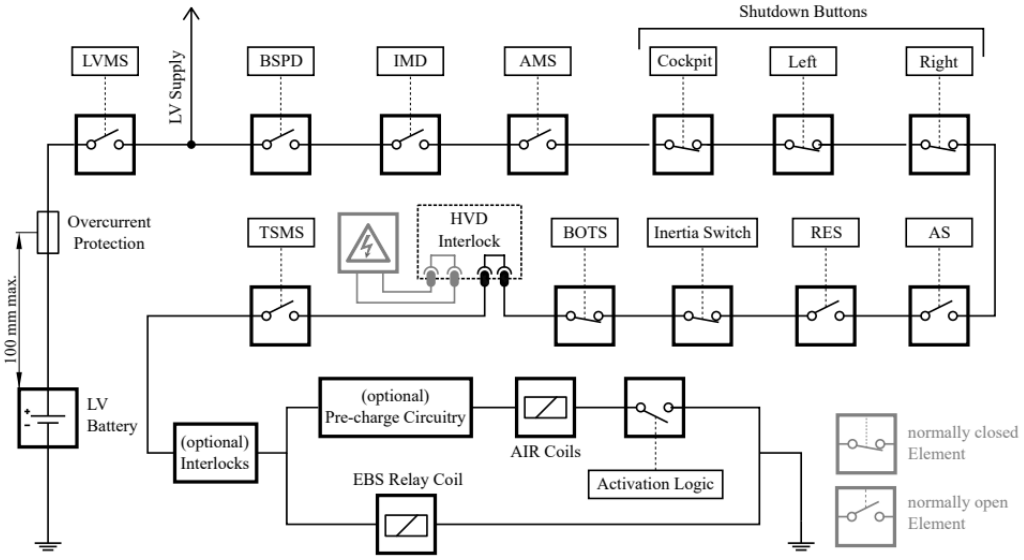


Figure 2.7: Reference shutdown circuit architecture as defined by FSG 2025 rules, from [1].

Protection mechanisms within the BMS are implemented through a hybrid hardware-software strategy, leveraging the deterministic speed of analog hardware with the configurability of software. Hardware-level protections include analog comparators, watchdog timers, and hardware interlocks that directly trigger shutdowns when limits are breached. These are particularly suited for fast-reacting faults such as short circuits. Software protections, on the other hand, allow dynamic fault filtering, state-dependent thresholds, and redundancy logic but are inherently limited by ADC latency and firmware execution time [12]. This architecture prioritizes fast analog shutdown for catastrophic events while reserving software-based logic for trends and non-critical faults.

Voltage protection involves monitoring individual cell voltages and triggering a fault if any cell exceeds the maximum or drops below the minimum allowable limit. A delay or error in voltage fault detection can result in permanent cell damage or thermal runaway [14]. Thermal protection is based on data from thermistors placed at the hottest points in the pack, where the BMS must act upon critical temperature thresholds to derate or disable the system.

Although not directly implemented inside the BMS hardware, other protective components are tightly integrated with BMS logic. For instance, the BMS controls the precharge sequence by verifying voltage equilibration before closing the AIRs, and it may also activate a discharge relay after shutdown to safely de-energize the high-voltage or Tractive System (TS) bus. Figure 2.8 shows a simplified schematic of this system-level coordination, where the BMS acts as the decision-making hub.

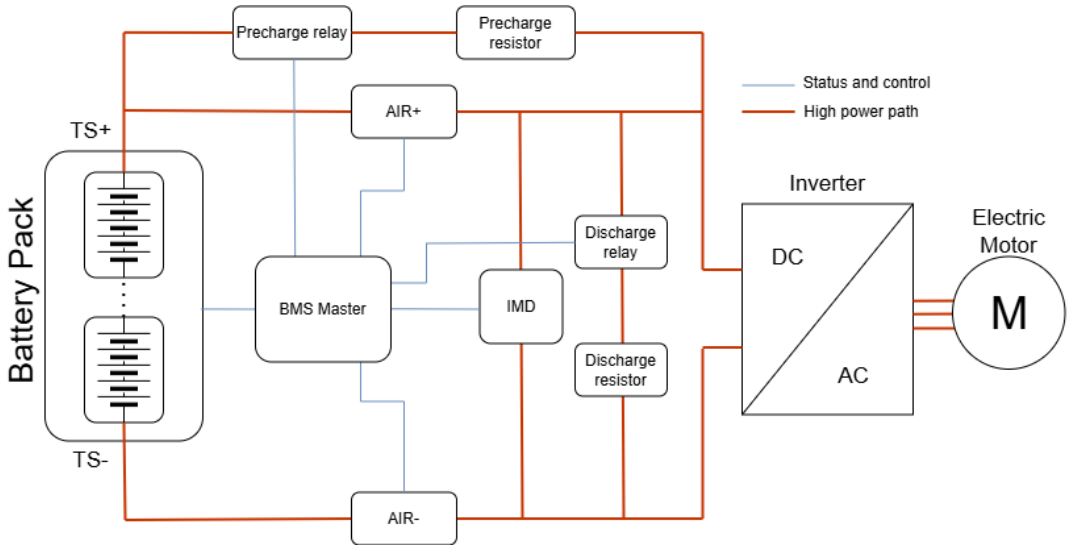


Figure 2.8: Simplified schematic of the TS protection architecture, including AIRs, IMD, precharge and discharge circuit, adapted from [12]

The IMD operates independently of the BMS but communicates with it through a digital fault signal directly routed into the BMS shutdown logic. The IMD is connected to the TS+ and TS-, and to multiple points of the low-voltage ground, the chassis of the vehicle. It works by superimposing a direct current test voltage onto the TS lines. Suppose an unintentional connection exists between the TS and low systems, due to moisture ingress, insulation degradation, or mechanical damage. In that case, a current will flow through a measuring resistor, resulting in a measurable voltage drop. If this drop exceeds a calibrated limit corresponding to a minimum insulation resistance, the IMD issues a fault signal [12]. In FS, the BMS is required to immediately open the shutdown circuit when the measured insulation resistance drops below the mandated threshold (typically 500 Ω/V related to the maximum TS voltage of the battery pack) [1].

Despite the robustness of modern BMS protection logic, its effectiveness depends strongly on sensor accuracy, placement, and system-level integration. In cost-constrained environments like FS, it is necessary to carefully balance protection coverage, redundancy, and board space. A detailed description of the fault thresholds, debounce algorithms, and shutdown logic implemented in this project is presented in Chapter 3.

2.2.2.3 Estimation Algorithms

Accurate estimation of internal battery states is fundamental to BMS performance, particularly in systems relying on Li-ion technology. Among the internal states, SoC and SoH are the most critical. These two parameters serve complementary roles: SoC governs short-term energy availability and power capability, while SoH provides insight into long-term degradation trends and remaining useful life [14].

SoC represents the ratio between the current available charge and the rated full capacity of the battery. As it cannot be measured directly, it is estimated using indirect signals such as current,

voltage, and temperature. Accurate SoC estimation is especially challenging due to nonlinear battery characteristics, temperature dependency, and aging effects. Over the years, SoC estimation has evolved from rudimentary integrative methods to sophisticated model-based and hybrid techniques. These can be broadly classified into three categories: direct, model-based, and data-driven (or hybrid), as shown in Table 2.2.

Table 2.2: Comparison of SoC Estimation Methods [14], [19]

Method	**Input**	**Advantages**	**Limitations**
CC	Current	Simple, low-cost	Drift over time
OCV Lookup	Voltage (rest)	Accurate at rest	Requires long rest
KF	Current, Voltage	Recursive, real-time	Linear model assumption
EKF/UKF	Current, Voltage	Nonlinear capable	Computational burden
AI-Based	Multisensor	Adaptive, robust	Requires training

Direct methods such as CC are straightforward and computationally efficient but susceptible to drift. OCV-based methods offer high accuracy at rest but are unsuitable for dynamic applications. Model-based estimators such as the EKF and Unscented Kalman Filter (UKF) use battery models to infer SoC in real-time, providing a balance between adaptability and computational feasibility. Electrochemical models, while highly accurate, are computationally intensive and best suited for offline analysis. More recently, data-driven approaches, including artificial neural networks, fuzzy logic, and hybrid algorithms, have shown promise, particularly for handling aging effects and dynamic conditions [20], [21].

SoH reflects the condition of the battery relative to its original performance. It is typically quantified in terms of capacity fade (reduction in maximum charge) and power fade (increase in internal resistance). Unlike SoC, SoH changes slowly over the battery lifecycle, but its accurate estimation is vital for predicting end-of-life and planning maintenance, commonly estimated using resistance growth, capacity tracking, impedance spectroscopy, or data-driven degradation models [14], [19]. For many EV and BMS standards, end-of-life is defined when SoH falls below 70–80% of nominal capacity. [22].

SoC and SoH estimation are integral to battery safety, longevity, and performance optimization. This dissertation primarily focuses on SoC estimation. A more in-depth discussion of the various existing SoC estimation techniques is provided in Section 2.4.

2.2.2.4 Cell Balancing Functionality

Cell balancing is a critical function of the BMS, ensuring uniform charge distribution across all cells in a battery pack. Imbalances among cells arise due to variations in manufacturing, differences in internal resistance, and unequal self-discharge rates. Without proper balancing, certain cells may become overcharged or overdischarged, reducing the overall capacity and performance of the battery pack, while also increasing the risk of thermal runaway. By implementing cell balancing

techniques, the BMS optimizes battery performance, enhances safety, and extends the system's operational life [23], [24].

Cell balancing methods are classified into two primary categories: passive and active balancing. Passive balancing involves dissipating excess energy from overcharged cells as heat through resistors. While this method is simple and cost-effective, it is less energy-efficient due to the energy loss as heat. Active balancing, on the other hand, redistributes energy from higher-charged cells to lower-charged ones using components such as capacitors, inductors, or transformers. Although more complex and costly, active balancing offers superior efficiency, faster equalization, and reduced energy losses, making it suitable for high-performance applications [23], [24].

Figure 2.9 illustrates the hierarchical classification of cell balancing techniques, detailing both passive and active methodologies. Active balancing systems are further divided into capacitor-based, inductor/transformer-based, and converter-based methods, each offering specific advantages tailored to different applications, from consumer electronics to EVs. These methods allow the BMS to maintain cell uniformity, optimize energy usage, and ensure the safe operation of Li-ion battery packs.

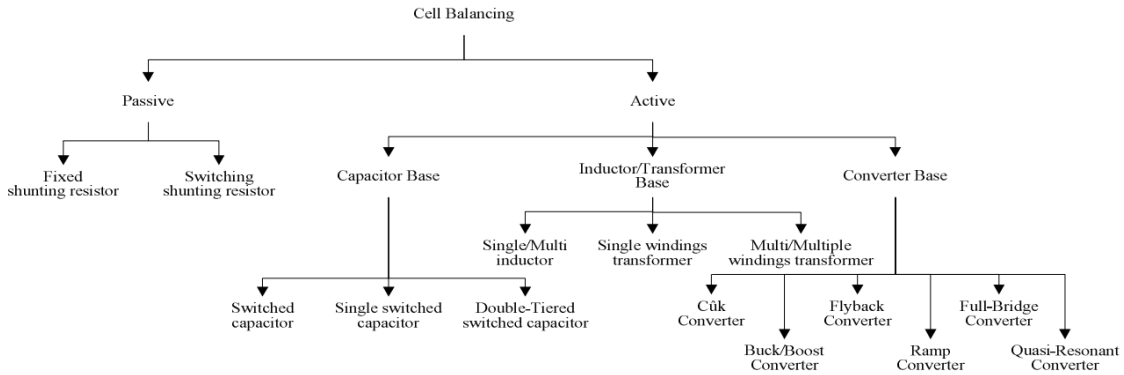


Figure 2.9: Classification of Cell Balancing Methods, from [23]

2.2.3 Topological Classifications

The architecture of a BMS fundamentally impacts its scalability, reliability, system complexity, and integration effort. To describe and compare different topologies in a structured manner, this section adopts a classification based on three functional tiers commonly used in modern BMS architectures:

- Cell Monitoring Unit (CMU): The lowest level, physically or logically associated with each individual cell. It performs cell-level voltage and temperature measurement and local balancing [25].
- Module Management Unit (MMU): The intermediate level, managing a group of CMUs, typically 8 to 12 cells within a module. It performs data aggregation, inter-cell balancing, and module-level fault detection [25].

- Pack Management Unit (PMU): The topmost level, supervising multiple MMUs and managing the overall battery pack. It is responsible for global safety logic, state estimation, current and voltage measurement at pack level, and communication with external systems [25].

This three-tiered classification allows for a consistent and scalable description of various BMS topologies, regardless of whether the functionality of one unit is physically embedded in another. Although CMU, MMU, and PMU are not standardized terms, this abstraction facilitates an objective comparison of architectural strategies across different domains [25].

2.2.3.1 Centralized

In the centralized BMS architecture, all functional layers, cell-level monitoring, balancing, protection logic, and system communication, are consolidated into a single controller. This unit assumes the responsibilities of the CMU, MMU, and PMU simultaneously, forming a monolithic system that directly interfaces with each cell through analog measurement and control lines [14].

This topology offers several advantages in small-scale applications: it simplifies hardware and software design, reduces component count, and minimizes development time and cost. The absence of inter-module communication also limits firmware complexity [14].

However, the limitations of this topology become evident as system size increases. The analog wiring required to connect every cell to the central controller leads to a complex and heavy wiring harness, particularly problematic in high-voltage configurations where galvanic isolation and signal integrity become critical. The centralized structure also presents a single point of failure: any fault in the central controller, its power supply, or a key signal path can disable the entire battery pack [25], [26].

A centralized topology effectively collapses the CMU, MMU, and PMU into a single unit. This eliminates the benefits of functional distribution, such as modular testing, physical isolation, or hierarchical fault containment. Moreover, it provides no support for modular redundancy or localized diagnostics [25].

Figure 2.10 illustrates this architecture. While adequate for compact systems with limited cells, the centralized topology is poorly suited to modern EV applications, where safety, scalability, and modular integration are essential design constraints.

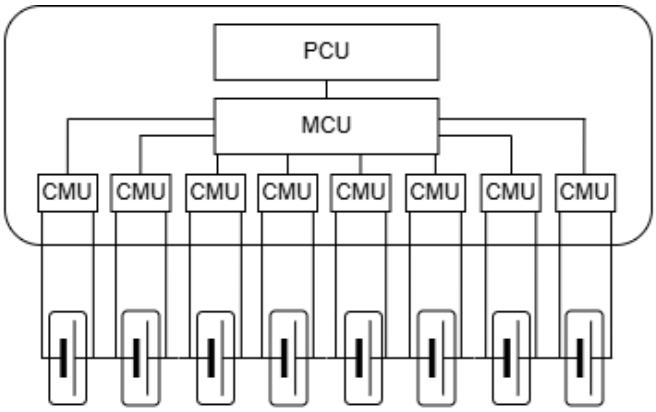


Figure 2.10: Centralized BMS topology, adapted from [14], [25]

2.2.3.2 Modular

In the modular BMS topology, the battery pack is divided into physically identical and functionally independent modules. Each module typically includes a local controller responsible for monitoring cell voltages and temperatures, and for performing cell balancing operations. These controllers combine functionalities typically associated with both the CMU and MMU. One of the modules is designated as the master and additionally performs system-wide coordination, diagnostics, and external communication, effectively assuming the role of the PMU.

This strategy maintains modular independence while consolidating supervisory functions within an existing module, reducing component count without compromising control fidelity. The modules are generally identical in hardware, and the distinction between master and slave roles is configured in software or firmware. The master module collects measurement data from all other modules via a digital communication bus (such as CAN or Serial Peripheral Interface (SPI)), and is responsible for executing the global protection logic, managing system states, and interfacing with higher-level vehicle systems.

By relocating measurement and balancing tasks to the module level, the analog signal paths are shortened, which enhances signal integrity and reduces electromagnetic interference. The modular structure also facilitates system integration and serviceability, as modules can be independently tested, replaced, or upgraded without disturbing the rest of the battery pack.

Although this topology shares some characteristics with centralized architectures, namely, the presence of a central decision point, it differs fundamentally in terms of physical organization, scalability, and fault containment. The system retains the benefits of hierarchical abstraction, where CMU and MMU functionalities are embedded in each module, and PMU responsibilities are consolidated within the designated master module.

Figure 2.11 illustrates this structure. This form of modularity allows for efficient parallel development, simplified diagnostics, and targeted fault isolation, key advantages in fast-paced academic design cycles [14], [25], [26].

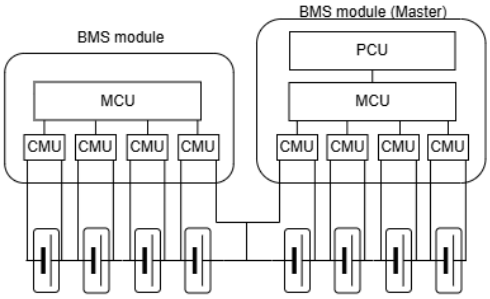


Figure 2.11: Modular BMS topology, adapted from [14], [25]

2.2.3.3 Master–slave

The master–slave BMS topology adopts a hierarchical control structure in which a central master unit supervises a network of slave devices distributed across the battery system. Each slave is typically embedded within a battery module and is responsible for performing cell-level voltage and temperature measurements, as well as executing cell balancing. These slave units incorporate both CMU and MMU functionalities, while the master unit assumes the role of the PMU, responsible for system-level state estimation, coordination, and communication with external controllers [14].

This architecture distributes time-critical tasks, such as cell monitoring and fault detection, to the slave units, thereby offloading the central controller and improving the responsiveness of the system. The master unit collects data from all slaves via a digital communication interface, applies global protection logic, manages contactor states, and transmits diagnostic information to the vehicle's supervisory systems.

One of the key benefits of this topology is the clear segmentation of functionality and wiring. Since analog signal processing is localised within each module, the overall analog wiring is reduced, which improves signal quality and reduces electromagnetic interference. At the same time, the centralised master unit ensures consistent application of system-wide rules and enables coordinated pack-level decision making.

Compared to purely modular topologies, the master–slave approach maintains a tighter integration between the modules and the supervisory logic. It provides greater flexibility in firmware implementation, particularly in systems where modules are not physically identical or where the PMU must execute more complex algorithms for pack balancing, ageing estimation, or fault classification.

Figure 2.12 provides an overview of a typical master–slave configuration. This architecture is widely adopted in commercial EVs and is particularly well suited to FS applications due to its balance of scalability, performance, and diagnostic capability [25], [26].

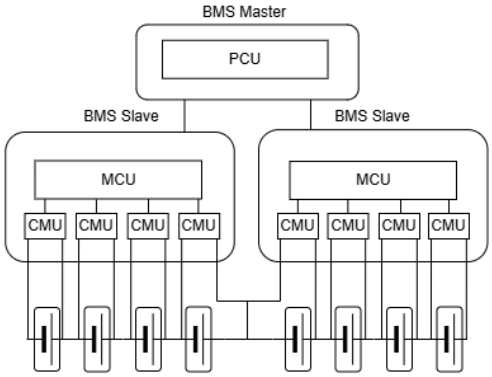


Figure 2.12: Master–slave BMS topology, adapted from [14], [25]

2.2.3.4 Distributed

In a distributed BMS topology, each cell, or small group of cells, is equipped with a local controller capable of performing all fundamental BMS functions: voltage and temperature monitoring, balancing, fault detection, and communication. These distributed nodes effectively combine the roles of CMU, MMU, and, to a limited extent, PMU, within a single unit. As a result, there is no need for a central controller to coordinate pack-wide operations, and each node functions autonomously [25].

This architecture maximises fault isolation and system redundancy, as each unit operates independently and can disengage itself in case of malfunction. It also greatly reduces the complexity of analog wiring, since all measurements and balancing actions are performed locally. Communication between units, when required, is typically digital and minimal, further reducing harness size and weight.

Despite these advantages, distributed topologies present significant system-level challenges. Node synchronisation, communication protocol design, and real-time coordination become critical, especially as the number of cells increases. Additionally, implementing robust safety logic across distributed agents requires sophisticated embedded systems, often involving Real-Time Operating Systems (RTOS) and advanced inter-node communication frameworks. Cost can also become a limiting factor due to the duplication of processing and sensing resources at every node.

From a functional standpoint, this architecture collapses the CMU, MMU, and parts of the PMU into a distributed network of equivalent peers, abandoning the classical hierarchical supervision model. Although this design offers unmatched flexibility and modularity, it is typically adopted only in specialised or experimental systems where fault resilience is prioritised over design simplicity.

Figure 2.13 shows a schematic of this topology. While conceptually attractive, the complexity of distributed architectures often outweighs their benefits in FS applications, where development resources, cost efficiency, and ease of integration are major constraints [25].

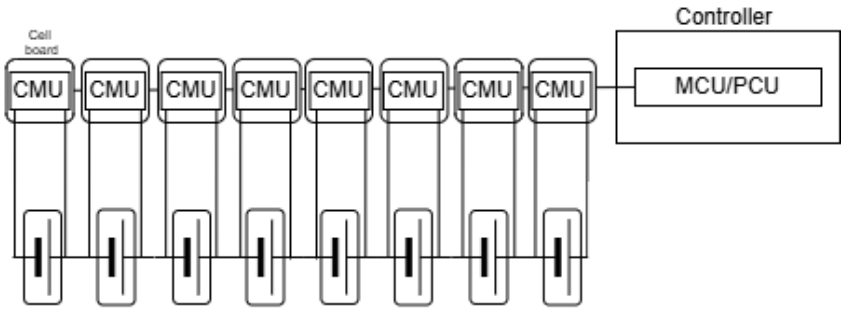


Figure 2.13: Distributed BMS topology

2.2.3.5 Comparative Analysis

Each BMS topology presents a distinct set of trade-offs regarding scalability, integration complexity, cost, reliability, and fault management. These factors must be evaluated against specific application constraints, such as physical space and modularity, all of which are highly relevant in the FS context. Table 2.3 summarizes the comparative characteristics of the four principal architectures, offering a high-level reference to guide architectural decisions in the design of embedded battery systems.

Table 2.3: Comparison of BMS Topologies, adapted from [14]

Criterion	**Centralized**	**Modular**	**Master-Slave**	**Distributed**
Scalability	Low	Medium	High	Very High
Cost	Low	Moderate	Moderate	High
Complexity	Low	Moderate	Moderate	High
Reliability	Low	Medium	High	Very High
Fault Isolation	Poor	Moderate	Good	Excellent
Wiring Requirements	High	Medium	Medium	Low

2.2.4 Design Approaches: Off-the-Shelf vs Custom

The choice between a COTS battery management system and a custom-designed solution is a architectural decision in any BMS development process. Each approach presents distinct trade-offs in terms of flexibility, performance, integration effort, and scalability.

COTS systems such as the Orion BMS 2, previously used in team's prototype, offer rapid deployment, mature safety features, and pre-validated communication protocols. Their appeal lies in ease of integration, reduced development risk, and lower upfront cost. COTS BMSs are often ideal for small to mid-size vehicle manufacturers, consulting firms and other results-driven entities that prioritize cost and reliability over deep customization [14].

However, COTS systems are limited by their general-purpose design. Key constraints include restricted access to firmware, inability to modify estimation algorithms, limited analog/digital I/O,

and inflexible communication interfaces. For teams seeking to implement advanced SoC estimators or to integrate tightly with vehicle-specific control architectures, these limitations pose significant barriers.

Conversely, a custom BMS design offers complete control over sensing architecture, estimator implementation, and hardware integration. It enables high-resolution current and voltage acquisition, custom cell balancing strategies, and advanced telemetry systems. Custom designs benefit institutions that value innovation, precision and a good integration inside the battery pack [14]. Table 2.4 summarizes the principal trade-offs between off-the-shelf and custom BMS design approaches. While the table captures the conceptual differences, the physical implementation of each solution further reinforces these distinctions. Figures 2.14 and 2.15 illustrate the practical embodiment of both design paradigms. The custom BMS, developed by Formula Electric Belgium [27], demonstrates tight integration with the battery module and showcases full access to internal hardware and software components. In contrast, the Orion BMS 2 represents a robust, commercially available solution that encapsulates all functionality within a vendor-defined enclosure, prioritizing reliability and ease of deployment over flexibility and customizability.

Table 2.4: Comparison between off-the-shelf and custom BMS design approaches

Aspect	**Off-the-Shelf Solution**	**Custom Design**
Time to Market	Fast deployment with minimal development time	Longer development cycle due to design and testing efforts
Flexibility	Limited to vendor's features and architecture	Fully adaptable to project-specific requirements
Estimator Integration	Restricted, depends on vendor tools	Full control over estimator models and integration
Initial Cost	Lower upfront cost	Higher due to development, tools, and validation
Software	Vendor-controlled, limited customization	Full ownership of design and code
Best Suited For	Rapid prototyping, system integrators	Research, development, and innovation

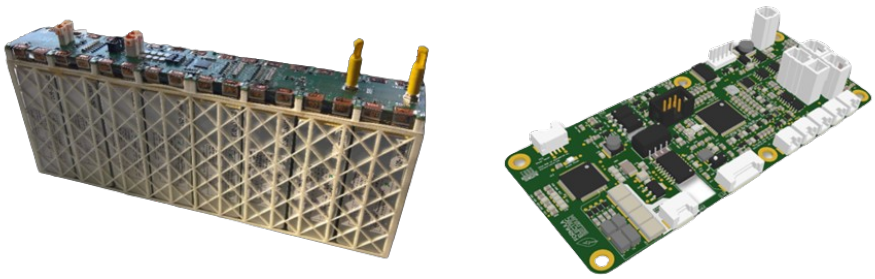


Figure 2.14: Custom BMS developed by Formula Electric Belgium. The slave board is mounted directly on top of the stack module, with the master controller shown on the right, from [27]

Figure 2.15: Commercial off-the-shelf solution: Orion BMS 2, from [3]

2.3 Review and evaluation of existing BMS Solutions

To inform the architecture and development strategy of the BMS for this project, a review of commercially available solutions was carried out. This market survey focuses on systems representative of different BMS topologies, centralized and master-slave. The comparison includes aspects such as the number of voltage inputs per unit, support for thermal sensors, communication protocols, configurability, and pricing. These criteria were selected based on the requirements of the battery, modular battery systems typical of FS vehicles, where real-time measurement access, estimator integration, and communication reliability are essential.

Table 2.5: Comparison between commercial BMS solutions (market survey, based on data from [3], [28]–[30])

BMS	Type	Voltage Inputs	Temp. Inputs	Protocol	Price
Orion 2 BMS [3]	Centralized	Up to 180	8 (expandable)	CAN	750€–1540€
Stafl Systems [28]	Master-Slave	12 per slave, up to 256	4–12	CAN	N/A
EVAL-ADBMS1818 [30]	Master-Slave	18 per slave	9	isoSPI	210€
EMUS G1 (Distributed) [29]	Distributed	4–80 per slave, up to 2560	External sensors	CAN	N/A
EMUS G1 (Centralized) [29]	Hybrid / Centralized	6–16 per slave, up to 512	Up to 16	CAN	N/A

Although many of the evaluated systems offer scalable architectures and robust communication interfaces, most of them have limitations that are non-trivial in the context of FS. Proprietary firmware and limited access to raw measurements restrict the implementation of custom estimation algorithms such as SOC/SOH observers. Furthermore, pricing and availability vary significantly depending on region, lead time, and integration support.

Systems like Orion 2 [3] and EMUS [29] are mature and well-documented, but offer limited flexibility in terms of software integration. Others, like Analog Devices' evaluation boards [30], provide direct access to low-level data but require additional development to reach production-level functionality. Stafl Systems [28] propose robust master-slave architectures, yet information on configurability and support is sparse without direct vendor engagement.

Overall, the review reveals a clear trade-off between integration effort and design flexibility. While COTS BMSs may accelerate initial prototyping, their constraints in estimator integration, cost, debugging, and hardware adaptation often limit their suitability for custom battery packs. These insights motivate further investigation into custom BMS development, which is explored in subsequent chapters.

2.4 SoC estimation techniques

The concept of SoC estimation has a rich history tied to advancements in rechargeable battery technology. Early systems, such as the single-meter device invented by Heyer in 1938, laid the groundwork for modern techniques. In 1963, Curtis Instruments developed gauges for monitoring SoC in traction vehicles, while Finger introduced a current-flow integrator module in 1975. The evolution continued with Peled's voltage and temperature-based SoC estimation method in 1984

and the formalization of CC by Aylor for lead-acid batteries. More recent innovations include KFs, introduced by Garche et al. in 2000, and hybrid approaches integrating machine learning [21].

These historical milestones reflect the growing complexity and sophistication of SoC estimation, driven by the need to address challenges such as nonlinear battery behavior, aging effects, and dynamic operating conditions. Figure 2.16 illustrates the key advancements in SoC estimation over time.

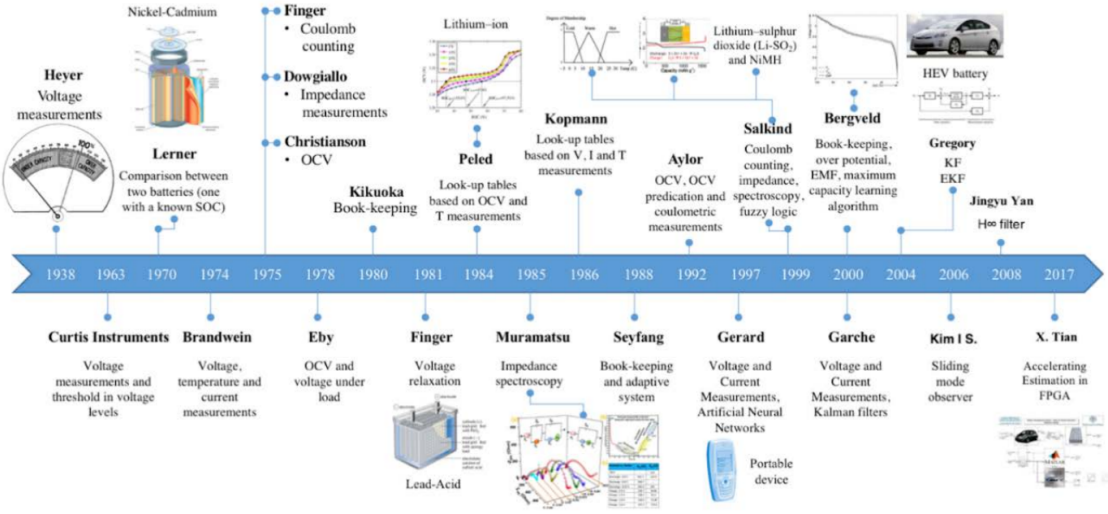


Figure 2.16: Key milestones in the evolution of battery SoC estimation methods [21].

Each SoC estimation method, whether direct or indirect, presents unique trade-offs in terms of performance, efficiency, cost, control, and complexity. The choice of method often depends on the application's specific requirements. Direct methods may be preferable in cost-sensitive, low-power systems, while indirect or hybrid methods are better suited for high-performance applications like EVs, where precision and adaptability are paramount. Understanding these trade-offs is essential for designing effective BMS that balance technical and economic constraints [20], [21], [31].

2.4.1 Direct Methods

Direct methods estimate the SoC using readily measurable battery parameters such as voltage and current. While these approaches are simple to implement and computationally efficient, their accuracy tends to degrade under dynamic operating conditions [20].

2.4.1.1 Coulomb Counting (CC)

The CC method, often referred to as the Ampere-Hour Counting method, is a fundamental technique for estimating the SOC in battery systems [20], [21]. Its popularity stems from its simplicity, computational efficiency, and ease of implementation. The CC method estimates the SoC by integrating the charge or discharge current over time, relative to the nominal capacity of the battery [20].

$$SoC(t) = SoC(t_0) + \frac{1}{C_n} \int_{t_0}^{t} I_{\text{bat}}(t)\, dt \times 100\% \tag{2.1}$$

where $SoC(t_0)$ represents the initial SoC at time t_0, C_n is the nominal capacity of the battery, and I_{bat} is the instantaneous current flowing into or out of the battery. This method operates by accumulating current over time, making it a direct approach for tracking charge levels.

One of the most notable advantages of CC is its low computational complexity, which makes it suitable for applications where processing resources are limited. Additionally, the method can be easily implemented with basic hardware, reducing system costs. However, these strengths are accompanied by inherent limitations that impact long-term accuracy [20], [21].

The measurement of battery current can contain errors and abundant measurement noise. Errors due to noise and the wide range in sensor resolution or rounding lead to accumulated inaccuracies in Equation 2.1, gradually reducing precision over time. Supporting algorithms are therefore essential to mitigate these issues [20], [21].

2.4.2 Open Circuit Voltage (OCV)

The OCV method is a widely adopted technique for estimating the SOC in BMS. This approach is based on the inherent relationship between the OCV and SoC, which is typically characterized through a polynomial function or a look-up table. The SoC is derived from the OCV as follows: [20], [21]

$$SoC = f^{-1}(\text{OCV}) \tag{2.2}$$

where f^{-1} represents the inverse mapping of the OCV curve to the SoC. This method relies on voltage measurements taken when the battery is at rest, ensuring no current flow and allowing the system to achieve electrochemical equilibrium.

The OCV method is renowned for its high accuracy, particularly under stable conditions. By utilizing the voltage-SOC relationship, it provides a reliable estimate of the battery's charge state. However, this approach is not without its challenges. Accurate OCV measurements necessitate a rest period, during which the battery stabilizes. This limitation makes real-time SoC estimation impractical in dynamic conditions. Furthermore, the OCV-SOC relationship varies significantly across different battery chemistries and electrode materials. For instance, Li-ion batteries exhibit a nonlinear and relatively flat OCV curve over a broad SoC range, complicating precise estimations. The OCV-SOC curve is also sensitive to aging and performance degradation, as well as temperature variations. These factors necessitate frequent recalibration or correction mechanisms to maintain accuracy [20], [21].

2.4.3 Electrochemical Methods (EChMs)

EChMs estimate the SoC by simulating the fundamental physicochemical processes occurring in Li-ion batteries, such as ion diffusion, intercalation, and electrochemical reactions. These models

solve complex partial differential equations to account for the interplay of battery dynamics with temperature, aging, and cycling conditions. By directly linking Li-ion concentrations in electrodes to SoC, EChMs provide unparalleled accuracy compared to other estimation methods. However, their computational complexity and the extensive parameterization required make them unsuitable for real-time applications, relegating their use to offline analysis and design optimization of battery materials and configurations [20], [21].

To mitigate these challenges, reduced-order EChMs simplify the mathematical models while retaining key insights into battery behavior. These reduced models, combined with estimation techniques, enable SoC estimation with reduced computational overhead. Despite these advancements, EChMs remain more resource-intensive than other methods, limiting their adoption in cost-sensitive systems. Nevertheless, their ability to incorporate thermal and degradation dynamics makes them a valuable tool for understanding battery performance and informing the design of advanced BMS. [20], [21]. .

2.4.4 Kalman Filters (KFs)

KFs are widely used tools for state estimation in dynamic systems and have been extensively applied in battery management systems for SoC estimation. While the Linear Kalman Filter (LKF) provides the foundation, the nonlinear behavior of Li-ion batteries necessitates advanced variants such as the EKF and the UKF. The following subsections explain each filter, with their workflows depicted in diagrams adapted from [20].

2.4.5 Linear KF

The KF is an optimal estimator for linear systems with Gaussian noise. It uses a state-space representation to model system dynamics, expressed as:

$$x_{k+1} = A_k x_k + B_k u_k + w_k, \tag{2.3}$$

$$y_k = C_k x_k + D_k u_k + v_k, \tag{2.4}$$

where:

- x_k: The state vector representing the internal states of the system at time k (e.g., SoC, voltage).
- A_k: State transition matrix describing how the states evolve from one step to the next.
- u_k: Input vector representing external control actions (e.g., charging or discharging current).
- B_k: Input matrix mapping the control inputs to changes in the system states.
- y_k: Measurement vector, representing the observed quantities (e.g., terminal voltage).
- C_k: Observation matrix that maps internal states to measurable outputs.

- D_k: Direct transmission matrix (often zero in battery systems).
- w_k: Process noise vector, accounting for uncertainties in the model.
- v_k: Measurement noise vector, capturing inaccuracies in observations.

The KF operates iteratively in two phases:

1. Prediction Step: the current state estimate is projected forward in time based on the process model:

$$\hat{x}_{k|k-1} = A_{k-1}\hat{x}_{k-1|k-1} + B_{k-1}u_{k-1}, \tag{2.5}$$

$$P_{k|k-1} = A_{k-1}P_{k-1|k-1}A_{k-1}^T + Q_w, \tag{2.6}$$

   where $P_{k|k-1}$ represents the covariance of the prediction and Q_w is the process noise covariance.

2. Correction Step: the state estimate is updated by incorporating new measurements:

$$K_k = P_{k|k-1}C_k^T(C_kP_{k|k-1}C_k^T + R_v)^{-1}, \tag{2.7}$$

$$\hat{x}_{k|k} = \hat{x}_{k|k-1} + K_k(y_k - C_k\hat{x}_{k|k-1} - D_ku_k), \tag{2.8}$$

$$P_{k|k} = (I - K_kC_k)P_{k|k-1}, \tag{2.9}$$

   where K_k is the Kalman gain, balancing the trust between the model and the measurements.

Although computationally efficient, the KF assumes linearity, which limits its application to Li-ion batteries due to their nonlinear behavior [20]. Figure 2.17 illustrates the workflow of the LKF.

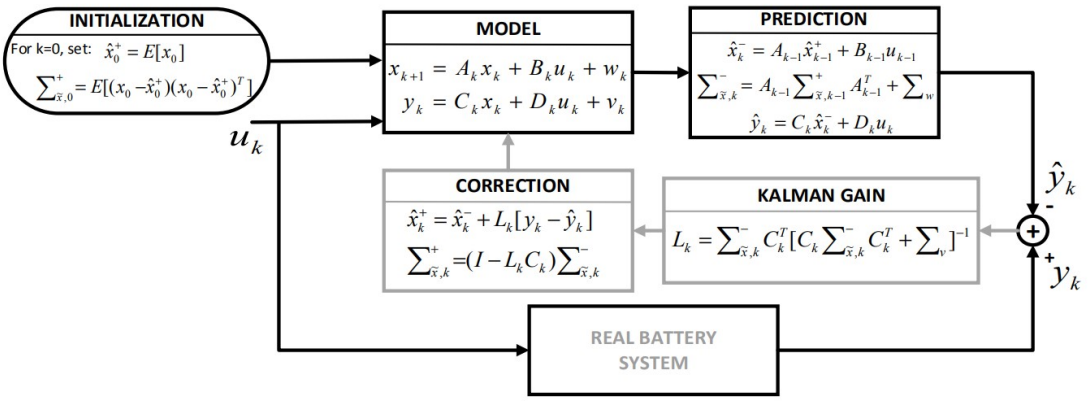


Figure 2.17: Workflow of the LKF for SoC estimation [20].

2.4.6 Extended Kalman Filter (EKF)

The EKF extends the KF framework to nonlinear systems by approximating the nonlinear functions using a first-order Taylor series expansion. The state-space model is described as:

$$x_{k+1} = f(x_k, u_k) + w_k, \tag{2.10}$$

$$y_k = g(x_k, u_k) + v_k, \tag{2.11}$$

where:

- $f(x_k, u_k)$: Nonlinear process function capturing the dynamics of the system.
- $g(x_k, u_k)$: Nonlinear measurement function mapping states to observed outputs.

To incorporate nonlinearities, the Jacobian matrices are computed:

$$\hat{A}_k = \frac{\partial f(x_k, u_k)}{\partial x_k}, \quad \hat{C}_k = \frac{\partial g(x_k, u_k)}{\partial x_k}. \tag{2.12}$$

The EKF workflow follows similar prediction and correction steps, but it linearizes the system around the current state estimate at each time step. Figure 2.18 illustrates the EKF workflow, highlighting its ability to handle nonlinearities more effectively than the KF, albeit with increased computational complexity and potential errors in highly nonlinear systems [20], [32].

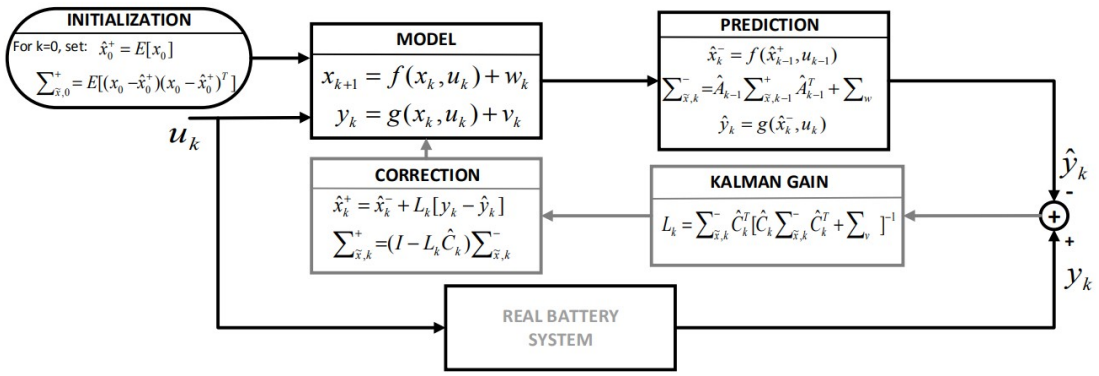


Figure 2.18: Workflow of the EKF for SoC estimation [20].

2.4.7 Unscented Kalman Filter (UKF)

The UKF addresses the limitations of the EKF by avoiding linearization. Instead, it uses sigma points to represent the state distribution and propagates them through the nonlinear functions. The state-space model remains:

$$x_{k+1} = f(x_k, u_k) + w_k, \tag{2.13}$$

$$y_k = g(x_k, u_k) + v_k. \tag{2.14}$$

Key aspects of the UKF include:

- Sigma Points: deterministic samples chosen to capture the mean and covariance of the state distribution.
- Nonlinear Propagation: each sigma point is passed through the nonlinear functions f and g, providing a more accurate estimate of the transformed state and measurement distributions.

Figure 2.19 demonstrates the UKF workflow. By avoiding linearization, the UKF achieves higher accuracy than the EKF, making it suitable for complex battery systems. However, this method requires increased computational resources [20].

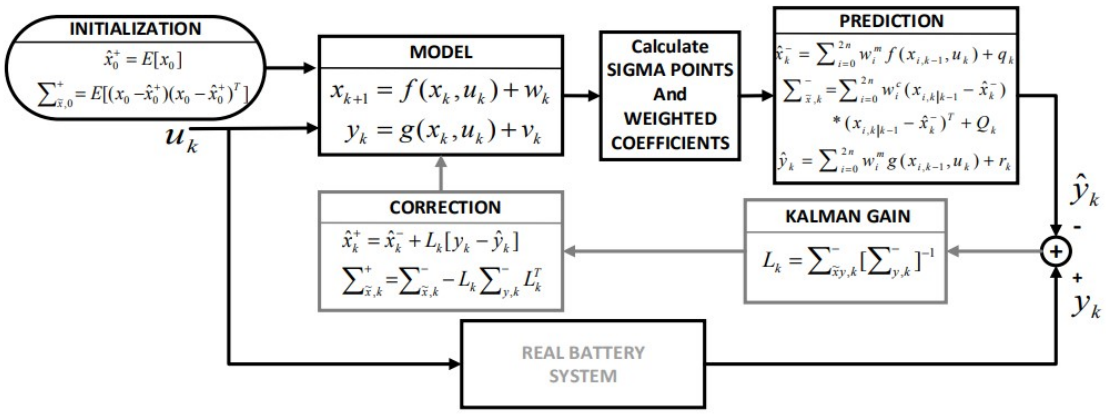


Figure 2.19: Workflow of the UKF for SoC estimation [20].

2.4.8 Other methods

Advanced estimation methods for SoC leverage adaptive techniques such as Fuzzy Logic (FL), Artificial Neural Networks (ANNs), Genetic Algorithms (GA), and Particle Swarm Optimization (PSO) to address the inherent nonlinearity, noise, and dynamic behavior of Li-ion batteries. These methods provide enhanced estimation accuracy by integrating computational intelligence and adaptive frameworks, making them particularly suitable for complex systems like EVs. FL, for instance, excels in managing imprecise and noisy data through a structured process of fuzzification, rule inference, and defuzzification. ANNs, on the other hand, model nonlinear relationships by processing current, voltage, and temperature data through interconnected layers to predict SoC. However, both methods are computationally intensive, require extensive training data, and are application-specific, which limits their generalizability and real-time feasibility [20], [21].

Optimization-based approaches such as GA and PSO optimize model parameters by mimicking natural processes like evolution and swarm intelligence. These methods are often combined with traditional SoC estimation techniques, such as KFs or equivalent circuit models, to enhance accuracy and robustness. Similarly, hybrid techniques like adaptive neuro-fuzzy inference systems combine the strengths of multiple approaches, offering real-time adaptability to system changes.

Despite their potential, these advanced techniques demand significant computational resources and tailored implementations, making them better suited for high-performance applications where precision is prioritized over computational constraints [20], [21].

2.5 Battery Models

The accurate estimation of battery SoC relies on models that can capture the nonlinear and dynamic behavior of batteries under various operating conditions. Battery models are generally categorized into electrochemical mechanism models, equivalent circuit models, and data-driven models, each with distinct advantages and limitations. This section discusses these models while integrating insights from recent reviews [33]–[35].

2.5.1 Equivalent Circuit Models (ECM)

ECMs represent batteries using electrical components, such as resistors, capacitors, and voltage sources, to mimic dynamic behavior. They are widely used for their balance between simplicity and accuracy, making them suitable for embedded systems [35]. Common ECMs include:

- **Rint Model (Figure 2.20a)**: This model consists of an internal resistor (R_0) and an open-circuit voltage (U_{oc}). It is computationally efficient but fails to account for transient behaviors such as polarization [34], [35].

$$V(t) = \text{OCV(SOC)} - I(t) \cdot R_0 \tag{2.15}$$

  This model assumes a fixed internal resistance and neglects dynamic effects such as diffusion or capacitive behavior.

- **Thevenin Model (Figure 2.20b)**: Extends the Rint model by incorporating an RC network to simulate polarization effects. This model offers a good balance between accuracy and complexity, making it a popular choice for practical applications [34].

$$V(t) = \text{OCV(SOC)} - I(t) \cdot R_0 - V_p(t) \tag{2.16}$$

  The addition of an RC branch enables the model to capture voltage recovery during rest periods.

- **Partnership for a New Generation of Vehicles (PNGV) Model (Figure 2.20c)**: Includes additional elements to account for long-term voltage changes during charge-discharge cycles. While more accurate than the Thevenin model, it requires more computational effort [34], [35].

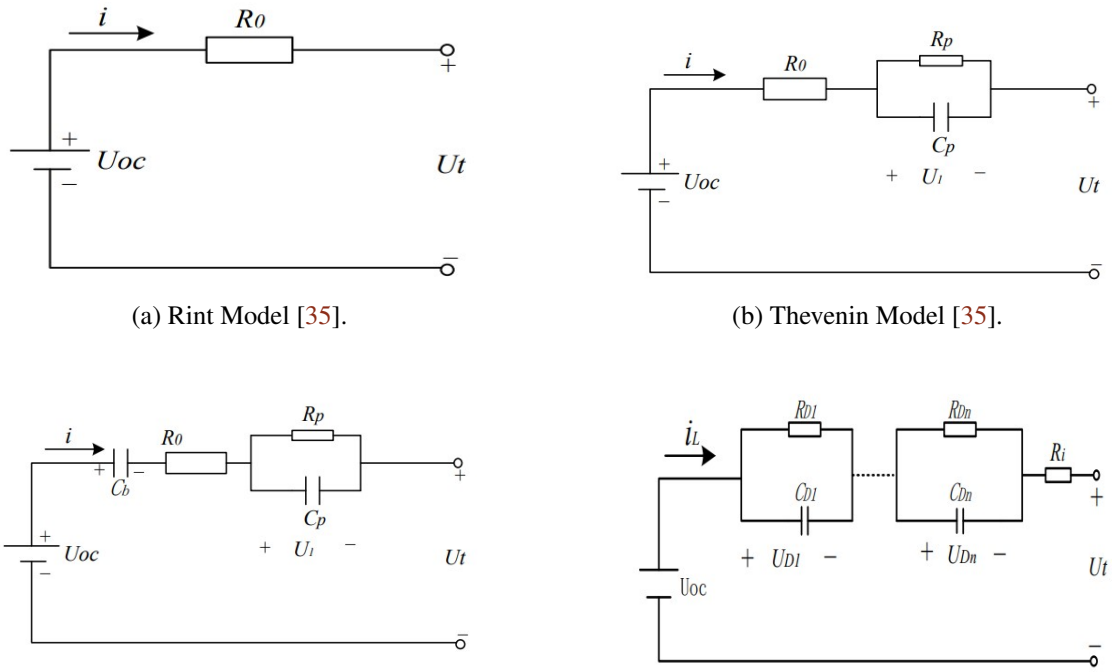
$$V(t) = \text{OCV(SOC)} - I \cdot R_0 - V_b(t) - V_p(t) \tag{2.17}$$

  The PNGV model adds multiple RC branches to capture both short- and long-term polarization effects.

- **Multi-order Model (Figure 2.20d)**: Incorporates multiple RC networks to capture complex dynamic behavior. Although it provides high accuracy, the increased complexity makes it less suitable for real-time applications [33], [34].

$$V(t) = \text{OCV(SOC)} - I \cdot R_0 - \sum_{i=1}^{n} V_{RC_i}(t) \tag{2.18}$$

This structure allows higher accuracy at the cost of increased parameterization and reduced real-time feasibility.



(a) Rint Model [35].

(b) Thevenin Model [35].

(c) PNGV Model [35].

(d) Multi-order Model [35].

Figure 2.20: Examples of equivalent circuit models.

2.5.2 Electrochemical Mechanism Models

Electrochemical mechanism models simulate the physical and chemical processes within batteries, such as ion diffusion, intercalation, and charge transfer reactions. The pseudo-two-dimensional model, developed by Doyle et al., is one of the most comprehensive representations of battery behavior. However, its complexity and high computational demands make it unsuitable for real-time applications [35]. Simplified models, such as the single particle model and the extended single particle model, reduce computational intensity by focusing on solid-phase dynamics while neglecting liquid-phase effects. These models strike a balance between accuracy and computational feasibility, particularly in controlled conditions [34].

Electrochemical Impedance Spectroscopy (EIS) is often used to validate these models and to study battery aging. EIS can characterize impedance changes due to degradation, enabling better

parameter identification for SoC estimation [33]. While highly accurate, electrochemical models require extensive parameterization, which limits their practical deployment in dynamic, real-time environments.

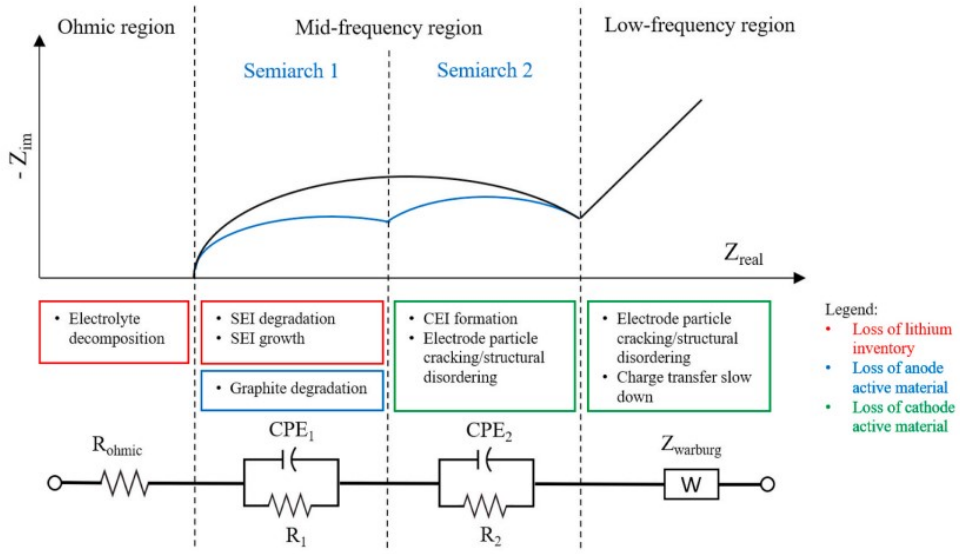


Figure 2.21: Typical EIS Spectrum of a Li-ion Cell, from [33]

2.5.3 Data-Driven Models

Data-driven models use machine learning and statistical techniques to estimate SoC without requiring detailed physical knowledge of the battery. Techniques such as ANN predict SoC based on input variables like current, voltage, and temperature [34]. These models adapt to changing operating conditions and are effective at capturing nonlinear relationships, but their reliance on large datasets and computational intensity limit their applicability in real-time scenarios [33].

2.5.4 Battery model comparion

Table 2.6: Battery model comparison. Adapted from [35].

Type of Model	**Accuracy**	**Complexity**	**Merits and Demerits**
EChMs	Very High	Very High	Accurate but computationally intensive
ECMs	Medium	Medium to Low	Balanced complexity and accuracy
Data-Driven Models	Medium	Medium to High	Adaptive but data-dependent

Electrochemical, equivalent circuit, and data-driven models each address different aspects of battery behavior. The selection of a model depends on application-specific requirements such as real-time feasibility, computational resources, and accuracy needs.

Chapter 3

Conceptual Design and Architecture of the BMS

This chapter outlines the conceptual development of the BMS for the FS electric prototype. Building upon the technical foundations discussed in the previous chapter, this section translates theoretical insights into engineering design decisions. Emphasis is placed on the rationale behind architecture selection, hardware choices, estimation methods, and system-level integration to ensure regulatory compliance, performance, and reliability.

3.1 Limitations of Commercial Off-the-Shelf (COTS) Solutions

The decision to transition from a COTS BMS to a custom-developed architecture is not merely a matter of preference, but a direct consequence of the technical, regulatory, and integration limitations encountered with the Orion BMS 2, a centralized off-the-shelf solution previously adopted by the team. While the Orion BMS 2 offers a mature feature set and reliability proven in generic EV applications, its suitability for high-performance, regulation-constrained environments such as FS is inherently limited [3].

3.1.1 Insufficient Regulatory Compliance and Integration Constraints

Despite offering a commercially mature platform, the Orion BMS 2 fails to satisfy several critical regulatory and integration requirements essential for FS applications. Most notably, its temperature monitoring capability is limited to 8 thermistor channels in its standard configuration, falling short of the minimum 30% cell coverage mandated by EV5.8.3 [1], a constraint that becomes particularly significant in large-scale configurations such as 144s5p, a total of 720 cells. Furthermore, its embedded fault handling logic lacks the flexibility to implement multi-tier shutdown responses as required by rule EV5.8.7, which differentiates timing thresholds for voltage/current anomalies (500ms) and temperature excursions (1s). The system also offers no provision for decoupling or

individually inspecting sensing elements, thereby failing to meet the modular diagnostic requirements established in EV5.8.11 and EV5.8.12. These limitations, although manageable in simpler systems, impose non-trivial barriers to regulatory conformity and inspection readiness.

As a direct consequence of these constraints, several mission-critical functions had to be offloaded to custom-designed subsystems in the previous prototype. A dedicated Printed Circuit Board (PCB) was implemented to interface with the IMD, ensuring real-time ground fault detection and compliance with EV5.6 [1]. A separate shutdown and precharge controller was also developed to manage inrush current and assert traction system disconnection based on independently sensed fault conditions. To meet thermal monitoring requirements, an auxiliary acquisition board was deployed on top of each stack to extend the number of readable cell temperatures beyond Orion's native capabilities. This fragmented implementation, while functionally sufficient, undermined architectural integration, increased harnessing complexity, and introduced additional points of failure. It ultimately contradicted the conceptual role of a BMS as a centralized, safety-critical entity, justifying the pursuit of a custom solution tailored to regulatory demands, system modularity, and platform scalability.

The severity of these limitations is evident in the implementation of the 02 prototype accumulator, shown in Figure 3.1. The image highlights the extent of internal wiring and the presence of multiple auxiliary PCBs, each responsible for functionality that should conceptually reside within the BMS itself. This includes the IMD interface, thermal sensor expansion on top of each stack, and shutdown circuit logic, whose physical separation increases system complexity and compromises maintainability.

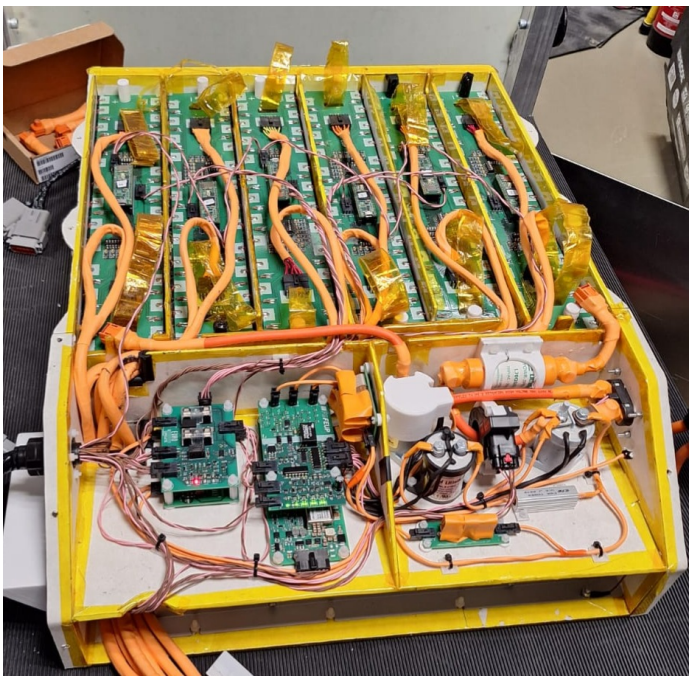


Figure 3.1: Accumulator interior of the 02 prototype, illustrating the physical integration burden imposed by the Orion BMS 2

3.1.2 Limited Functional and Mechanical Integration

The Orion BMS 2 employs a fully centralized architecture, requiring all measurement, control, and communication interfaces to converge on a single external enclosure. While this design may simplify internal firmware management, it introduces significant mechanical and functional integration constraints when applied to high-voltage, high-density battery systems such as those used in FS. Chief among these is the need to route analog signals, including cell voltages and thermistor readings, over long distances from inside the accumulator housing to the centralized BMS unit. These signal paths are highly susceptible to electromagnetic interference and voltage drift, particularly under dynamic loading conditions, thereby degrading measurement reliability.

Additionally, the system offers no provision for embedded or distributed installation within the accumulator structure. As a result, the BMS must be mounted externally and connected via high-density harnesses, leading to substantial internal cabling, increased connector complexity, and a non-negligible thermal footprint. This architectural rigidity adversely affects several aspects of accumulator design: packaging becomes more constrained due to external clearance requirements, total system mass increases due to redundant wiring and mounting hardware, and thermal management is complicated by the need to avoid heat sources near critical measurement points.

Furthermore, core system-level control functions, including precharge sequencing, AIR relay actuation, and IMD handling, are only partially exposed and often governed by proprietary logic. This restricts the ability to integrate safety mechanisms in a manner consistent with FSG guidelines and project-specific design philosophies. The net result is a constrained, monolithic integration strategy that contradicts the design goals of modularity, compactness, and maintainability. Figure 3.2 illustrates the practical implications of this centralized architecture. The image shows the interior of the current accumulator used in the 02 prototype, highlighting the volume occupied by the Orion BMS 2 enclosure and the density of internal cabling required for sensor routing and power distribution. This implementation significantly constrained the layout, increased the system's overall mass, and complicated both assembly and servicing procedures, outcomes that are incompatible with the lightweight, modular ethos that defines FS engineering.

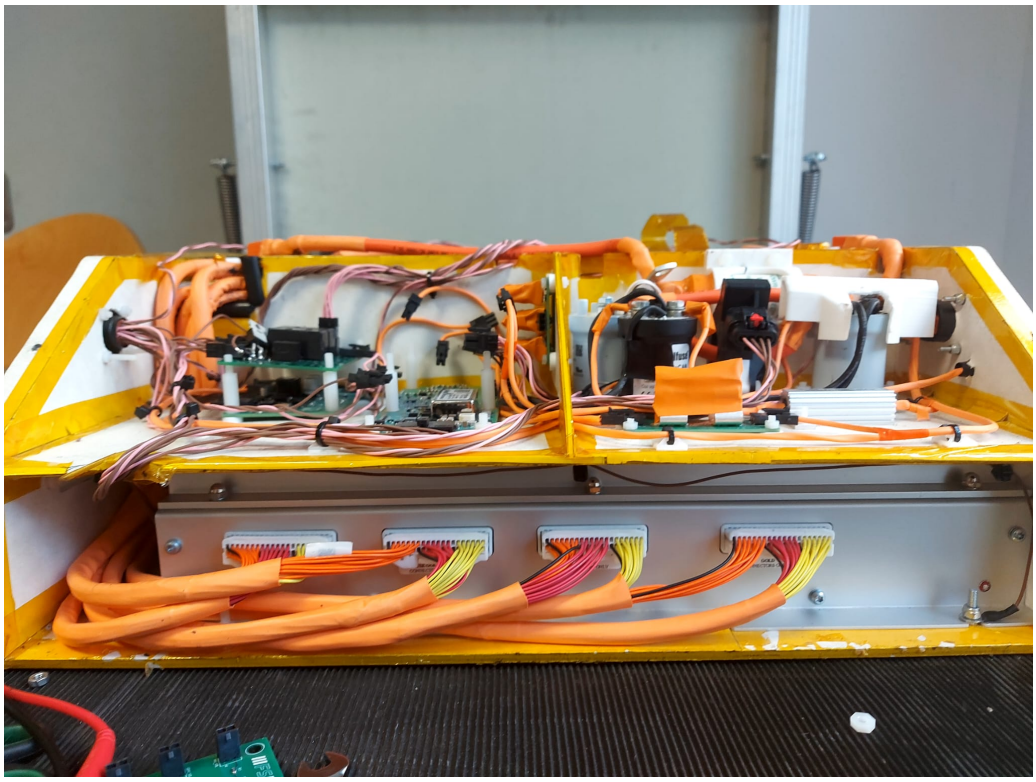


Figure 3.2: Internal view of the accumulator highlighting the Orion BMS 2 volume and wiring density

3.1.3 Estimator Performance and Algorithmic Rigidity

One of the most significant limitations of commercial battery management systems, such as the Orion BMS 2, lies in their restrictive approach to state estimation. The Orion BMS 2 relies exclusively on CC for SoC estimation, a method that, while computationally simple, is known to suffer from cumulative drift and loss of accuracy over time, especially under variable load profiles or extended usage without full charge-discharge resets. This renders the estimator unsuitable for scenarios demanding high reliability and precision over long operational periods.

Compounding this issue is the complete lack of access to the internal estimation algorithms or firmware logic. The Orion BMS operates as a closed system, offering no interface for modifying, tuning, or replacing the default estimation strategy. As a result, users are unable to implement model-based approaches, such as KFs or other adaptive observers, which are widely recognized for their robustness in accounting for nonlinear dynamics, temperature variations, and battery ageing effects. This rigidity severely limits the system's adaptability to application-specific requirements or experimental integration.

A further drawback is the constrained support for different cell chemistries. The Orion configuration software requires users to select a predefined cell profile from a proprietary list. In cases where the desired cell, such as the Molicel P28A used in this project, is not available, the user must request a paid characterization service from the manufacturer. This process introduces additional

cost, lead time, and technical opacity, while preventing direct integration of independently acquired cell data, such as OCV-SOC curves or impedance models derived from laboratory testing.

In the context of this project, where an extended experimental campaign is planned to characterize battery behavior under a range of conditions, the inability to refine or customize the estimation logic is a major drawback. Accurate and adaptable SoC estimation is not only essential for real-time control and safety, but also for validating models, monitoring degradation, and extracting meaningful insights from experimental data. These limitations directly motivate the development of a custom estimation framework tailored to the system architecture and experimental objectives.

3.1.4 Cost-Performance Trade-offs in Feature-Rich BMS Architectures

COTS BMS are often designed to address a wide spectrum of use cases, particularly within the automotive industry. These systems integrate an extensive array of features, ranging from advanced fault diagnostics to multiple communication protocols and configurable protection layers, aimed at maximizing safety and compatibility across various platforms. However, in applications with well-defined and limited functional requirements, such as academic prototypes or custom-built electric vehicles, this breadth of functionality may exceed the actual system demands.

In such contexts, the deployment of a feature-rich BMS can lead to inefficiencies both technically and economically. The Orion BMS 2, for example, includes hardware and software provisions for functions like thermal runaway detection, multiple CAN buses, and extensive logging capabilities, which may remain unused in research-focused or low-volume applications. These unused features not only increase the cost of acquisition but also introduce unnecessary complexity, complicating the integration process and demanding additional configuration time without delivering proportional benefit.

Furthermore, the closed architecture typical of many COTS systems restricts access to low-level firmware and internal estimators, limiting the flexibility needed for advanced algorithm development or hardware repurposing. In scenarios where precise control over current sensing, balancing strategies, or state estimation algorithms is essential, such limitations hinder innovation and adaptability. As a result, while COTS solutions like the Orion BMS 2 offer reliability and rapid deployment, their cost-performance ratio may prove suboptimal when applied to narrowly scoped or highly specialized battery systems.

3.2 Design Objectives and Constraints

The BMS developed in this project aims to overcome the limitations observed in the previous system, which used a centralized off-the-shelf Orion BMS 2. While this system offered rapid deployment and mature safety features, it introduced significant challenges in terms of cabling complexity, integration flexibility, and estimator performance.

To address these issues and align with the goals of the 02 prototype, and potentially future iterations such as the 03 prototype, a custom BMS was proposed. The design seeks to minimize internal wiring, enhance modularity, and allow deeper estimator integration, while also complying

with safety regulations and cost constraints. Table 3.1 summarizes the key objectives and constraints that guide the conceptual development of the new BMS, along with their internal reference codes and development priority.

Table 3.1: BMS Design Objectives and Constraints

Code	Objective / Constraint	Description	Priority
OBJ-01	Minimize internal cabling	Reduce analog wiring and signal congestion inside the accumulator enclosure by avoiding centralized topology.	High
OBJ-02	Avoid centralized architecture	Implement a modular or master–slave topology to improve scalability and signal integrity.	High
OBJ-03	Regulatory compliance	Fully comply with FSG rules regarding monitoring, fault response, and shutdown architecture.	High
OBJ-04	Enhanced SoC estimator	Implement a superior estimator with better accuracy and dynamic performance than Orion's integrated estimator.	High
OBJ-05	Safety protocol integration	Integrate fault management, AIR control, and IMD response in line with shutdown circuit requirements.	High
OBJ-06	Validate SoC estimator in testboard	Compare estimator behavior on hardware with simulation results to confirm real-world accuracy and convergence.	High
OBJ-07	Pack-level safety diagnostics	Continuously evaluate whether the battery is operating within safe voltage, current, and temperature boundaries.	High
OBJ-08	Current measurement capability	Accurately measure and process the battery's charge/discharge current for protection and estimation logic.	High
OBJ-09	Cell model development	Create a reliable electrical model of the selected cell using experimental data to support accurate SoC estimation and parameter tuning.	High

(continued on next page)

Code	Objective / Constraint	Description	Priority
OBJ-10	Uniform stack-mounted PCBs	Ensure all boards mounted on battery stacks are electrically and mechanically identical, promoting hardware uniformity, simplified production, and interchangeable module design.	High
OBJ-11	Lower cost than Orion BMS 2	Achieve a lower final production cost while maintaining equivalent or improved performance.	Medium
OBJ-12	Seamless prototype integration (02)	Ensure compatibility with the mechanical, electrical, and communication interfaces of the current prototype.	Medium
OBJ-13	Medium performance	Ensure that the system supports a feature set at least equivalent to the Orion BMS 2 in terms of accuracy	Medium
OBJ-14	Cell balancing system	Implement a functional cell balancing strategy (passive or active) to maintain voltage uniformity.	Medium
OBJ-15	Multi-protocol communication	Implement different communication protocols for compatibility and versatility with prototype 02.	Medium
OBJ-16	Future adaptability (03)	Design for modular expansion and reuse in the 03 prototype with minimal rework.	Low
OBJ-17	GUI monitoring tool	Develop a user interface to visualize SoC, temperatures, fault flags, and real-time system state.	Low
OBJ-18	SoH estimation support	Provide a basic State of Health estimation algorithm to assess aging and long-term degradation.	Low

3.3 System Requirements Specification

The design of a BMS must align with the specific configuration of the battery pack it manages, ensuring compatibility with voltage, capacity, current limits, and operational constraints. In high-performance environments like FSG, stringent safety and performance regulations must also be met.

The battery pack for this project was configured as follows:

Table 3.2: Battery Pack Configuration

Parameter	**Value**
Cell model	Molicel P28A 18650
Cell configuration	144s5p (24s5p per stack)
Maximum voltage	604.8 V (charged up to 600 V)
Nominal capacity	7.26 kWh
Maximum power	106 kW (limited to 80 kW)
Maximum discharge current	175 A
Maximum charge current	30 A

This configuration uses a series-parallel arrangement (144s5p) to achieve the desired voltage and capacity. The Molicel P28A were selected for their high energy density, robust thermal stability, and excellent performance under high-discharge rates, which are critical for racing applications [16].

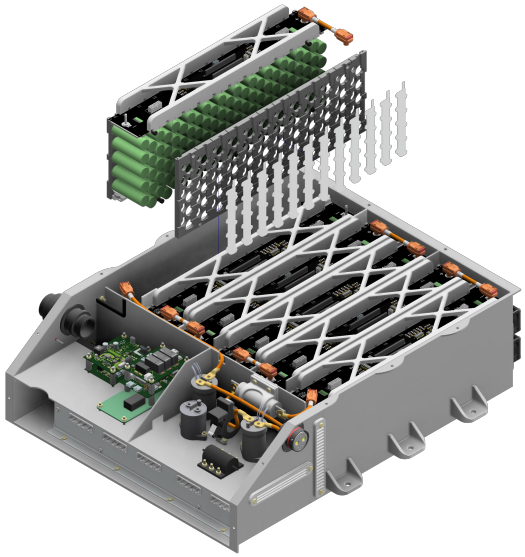


Figure 3.3: Battery Pack using Molicel P28A cells

3.4 Compliance with Formula Student Germany (FSG) Regulations

The BMS must be designed in full compliance with the FSG electrical regulations to ensure system safety, inspection readiness, and eligibility for competition. The following technical requirements must be fulfilled:

- **Continuous monitoring:** Real-time acquisition of all individual cell voltages, TS current, and temperatures of critical cells. At least 30% of lithium-based cells must be monitored for temperature to provide a representative thermal profile (EV5.8.3) [1].

- **Temperature control:** Enforcement of a maximum cell temperature limit of 60°C or the manufacturer-specified maximum, whichever is lower, to prevent thermal runaway and cell degradation (EV5.8.5) [1].
- **Automatic shutdown:** The BMS must be capable of disconnecting the traction system within 500 ms of a voltage or current fault, and within 1 second of a temperature fault (EV5.8.7) [1].
- **Measurement accuracy:** All sensing circuits must offer high-resolution and low-noise data with sampling rates compatible with system dynamics and estimator requirements (EV5.8.8) [1].
- **Inspection accessibility:** All voltage and temperature sensors must be individually disconnectable. The BMS must provide a single diagnostic interface (e.g., laptop interface) showing all measurements in real time for inspection purposes (EV5.8.11, EV5.8.12) [1].

3.5 High-Level Architecture Selection

As established in the design objectives (OBJ-01 and OBJ-02), the proposed BMS must not adopt a centralized architecture. Centralized topologies were excluded due to their excessive cabling requirements, lack of scalability, and poor fault containment, all of which are difficult to implement with the physical and regulatory constraints of accumulator system.

To minimize internal wiring and improve modularity, emphasis was placed on evaluating distributed, modular, and master–slave topologies. The comparative criteria, outlined previously in Table 2.3, include scalability, cost, fault isolation, wiring complexity, and integration effort. These factors were weighed against the specific requirements of this project, such as regulatory compliance, layout constraints, and long-term adaptability.

Although the distributed architecture offers excellent fault isolation and scalability, it was excluded due to its elevated cost and complexity, violating OBJ-09. Distributed systems also demand heterogeneous PCBs, advanced synchronization mechanisms, and complex inter-node communication protocols, which are impractical under the development constraints of this project.

Similarly, a purely modular approach, while offering decent fault containment, was not selected, as it conflicts with OBJ-10, which calls for homogeneous PCB design across all stack segments. Modular architectures typically assign supervisory roles to specific modules, resulting in functional asymmetry that contradicts the project's design philosophy of interchangeable and identical measurement boards.

Taking these considerations into account, the **master–slave topology** emerged as the most appropriate architectural choice. It offers a well-balanced trade-off between cost, integration simplicity, and feature scalability. The master–slave approach enables the deployment of identical slave boards across all segments of the battery stack, each handling voltage, temperature, and balancing tasks locally, while a central master coordinates communication, protection logic, and system-level estimation. This structure supports both the electrical modularity and software adaptability required by the team's long-term objectives.

3.6 Hardware Component Selection

3.6.1 Monitoring IC

The voltage monitoring IC selected for each slave board is the **LTC6811**, a 12-cell monitoring device from Analog Devices. Its selection was based on a combination of technical capabilities and practical design support. The device integrates a 16-bit delta-sigma ADC with high precision, enabling accurate measurement of individual cell voltages across each segment (CMU and MMU). It also includes built-in support for passive balancing and temperature acquisition channels, reducing the need for external components.

A advantage of the LTC6811 is its daisy-chain communication architecture via Isolated Serial Peripheral Interface (isoSPI), which allows up to 31 devices to be chained using just two differential lines. This drastically reduces the wiring required to interface with the master controller, aligning directly with Objective OBJ-01. In the proposed design, one LTC6811 will be embedded per slave board, managing all cell-level measurements and balance control locally.

Additionally, the component's documentation provides detailed layout recommendations and validated design examples, accelerating schematic and PCB development. Its widespread adoption in research and industry further supports its reliability in demanding automotive environments, making it an ideal choice for the electrical and regulatory constraints of FS.

3.6.2 Current Sensing

Current measurement in the proposed BMS is handled by a **LEM DHAB S/118** Hall-effect transducer, selected for its galvanic isolation, analog output, and bidirectional current sensing. This sensor is already integrated in the existing accumulator assembly, originally configured for the Orion BMS 2, making it both a cost-efficient and implementation-ready choice, in line with Objectives OBJ-10 and OBJ-11.

Alternative solutions such as low-resistance shunt resistors were dismissed due to the absence of electrical isolation. Their implementation would require significant changes to the TS architecture, increasing system complexity, cabling, and power dissipation.

Although digital precision shunt modules, as illustrated in Figure 2.6, offer excellent accuracy and CAN interface integration, their cost and sourcing complexity are not justified in a student project context. The LEM DHAB S/118 thus offers the optimal trade-off between performance, availability, and integration simplicity for this application.

3.6.3 PMU Microcontroller and Communication Interface

The master controller for the BMS, herein referred to as the PMU, is based on the STM32F405RGT6, a 32-bit ARM Cortex-M4 microcontroller operating at 168MHz with integrated floating-point support. This microcontroller was selected for its ability to handle computationally intensive tasks, such as real-time state-of-charge estimation, while maintaining deterministic execution of safety-critical logic. Although it does not match the performance of the Teensy 4.1 used in auxiliary

systems of prototype 02, it provides a superior balance between timing determinism, hardware integration, and ecosystem maturity [36].

The STM32F405 integrates all communication protocols required for system coordination. CAN 2.0B enables direct interface with the ECU system; SPI is used to communicate with the daisy-chained LTC6811 acquisition modules via isoSPI bridges; Universal Asynchronous Receiver/Transmitter (UART) provides a low-level debugging channel; and Universal Serial Bus (USB) supports firmware programming and potential PC interfacing. This protocol versatility fulfills OBJ-15 and OBJ-12, ensuring that the PMU is fully compatible with the architecture of both prototype 02 and future designs [36].

The PMU acts as the sole high-level processing node in the system. It performs SoC estimation using cell voltage and current data, oversees all slave communication, manages the shutdown signal logic, and supervises battery safety status in real time. Voltage and temperature data are acquired from the slave acquisition boards via daisy-chained LTC6811 devices, while current sensing is directly interfaced with the PMU, together fulfilling OBJ-08. Continuous evaluation of pack-level operating conditions, including voltage, current, and thermal boundaries, satisfies OBJ-07.

Although the PMU is responsible for orchestrating global safety logic, certain discrete PCBs from prototype 02 will remain in use. These include the hardware responsible for precharge sequencing, AIR actuation, and IMD fault detection. As such, full integration as defined in OBJ-05 will not be achieved in this version. Nonetheless, the PMU will assert the shutdown circuit when internal faults are detected, preserving functional safety.

This hybrid approach preserves development continuity while maintaining compliance with FSG shutdown requirements. To validate the PMU architecture, an extensive test campaign will be conducted. If the implementation proves reliable and functionally complete under both simulated and real-world conditions, future iterations may migrate all safety logic into the PMU, enabling full integration and elimination of redundant subsystems.

3.7 SoC Estimation Strategy Selection

In pursuit of a high-performance SoC estimator tailored to the characteristics of the target battery pack and operating profile, a model-based estimation strategy was selected for integration into the BMS. Among the candidate methods evaluated in Chapter 2, the EKF was identified as the most suitable solution for this implementation stage. Its ability to combine a predictive battery model with real-time voltage measurements provides significant improvements in accuracy and robustness over traditional integration-based methods. Furthermore, the EKF's balance between estimation quality and computational efficiency makes it well-matched to the embedded processing constraints of the PMU.

This choice is particularly appropriate given the constraints imposed on battery operation in FS. The accumulator is never fully charged to its maximum of 604.8V, due to the regulatory ceiling of 600V, nor is it discharged below 20% SoC to prevent undervoltage faults and preserve cell longevity. As a result, the battery operates predominantly in the mid-range region of the OCV–SoC

curve, where voltage response is relatively linear and well-behaved. In this regime, the EKF exhibits fast convergence, low error variance, and strong resilience to sensor noise and model mismatch, making it a highly effective solution under these conditions.

While the UKF was also considered, particularly for its superior handling of strong nonlinearities and faster adaptation to abrupt current transients, it was deemed unnecessarily complex for this initial validation phase. Nonetheless, the BMS and estimator frameworks are designed to support future migration to a UKF, ensuring a scalable path for performance enhancement once the EKF implementation has been validated both in simulation and in hardware.

The development of an EKF-based estimator directly supports OBJ-04. However, its successful implementation depends on the completion of two foundational tasks. First, a cell model must be derived from experimental data obtained using the Molicel P28A, the Li-ion cell used in the accumulator. This fulfills OBJ-09, which requires constructing a electrical model capable of capturing voltage response under variable load and temperature. Second, the estimator must be validated on hardware via a dedicated testboard (OBJ-06), using representative cells under controlled discharge scenarios. Experimental results will be compared against simulated EKF behavior to verify estimator accuracy, error convergence, and robustness to model uncertainty. This validation campaign is critical for establishing the estimator's suitability for deployment in the operational BMS.

3.8 Integrated System Architecture Diagram and Functional Mapping

To consolidate the architectural rationale and engineering decisions from Sections 3.5 through 3.7, Figure 3.4 presents a unified diagram of the BMS architecture. It depicts the PMU, daisy-chained acquisition modules based on the LTC6811, the current sensing system, and the retained auxiliary safety boards from prototype 02. Communication protocols (CAN, SPI/isoSPI, UART) and data pathways are explicitly shown to clarify real-time data flow for fault detection, state estimation, and shutdown actuation.

The diagram also maps how each subsystem contributes to key design objectives, such as reduced wiring complexity (OBJ-01), modularity (OBJ-02), estimator validation (OBJ-06), and shutdown compliance (OBJ-05). It serves as both a synthesis of the system architecture and a validation checkpoint against defined technical goals, while also supporting testing and future development.

Given the 24-series cell configuration per stack and the 12-cell limit of the LTC6811, each stack includes two identical slave boards. This ensures full voltage and temperature acquisition, fulfilling regulatory requirements and uniform PCBs (OBJ-10), which emphasizes hardware uniformity.

To minimize cabling and improve assembly efficiency (OBJ-01), the communication between PMU and slave boards uses isoSPI instead of SPI. Although wire count remains similar, isoSPI's differential signaling enhances noise immunity and supports longer routing distances, crucial in the high-voltage and existing surge currents [15].

This design choice improves signal integrity, supports clearer internal layout, and simplifies debugging and scalability. It represents a deliberate trade-off aligned with modularity, robustness, and regulatory goals, as consolidated in Figure 3.4.

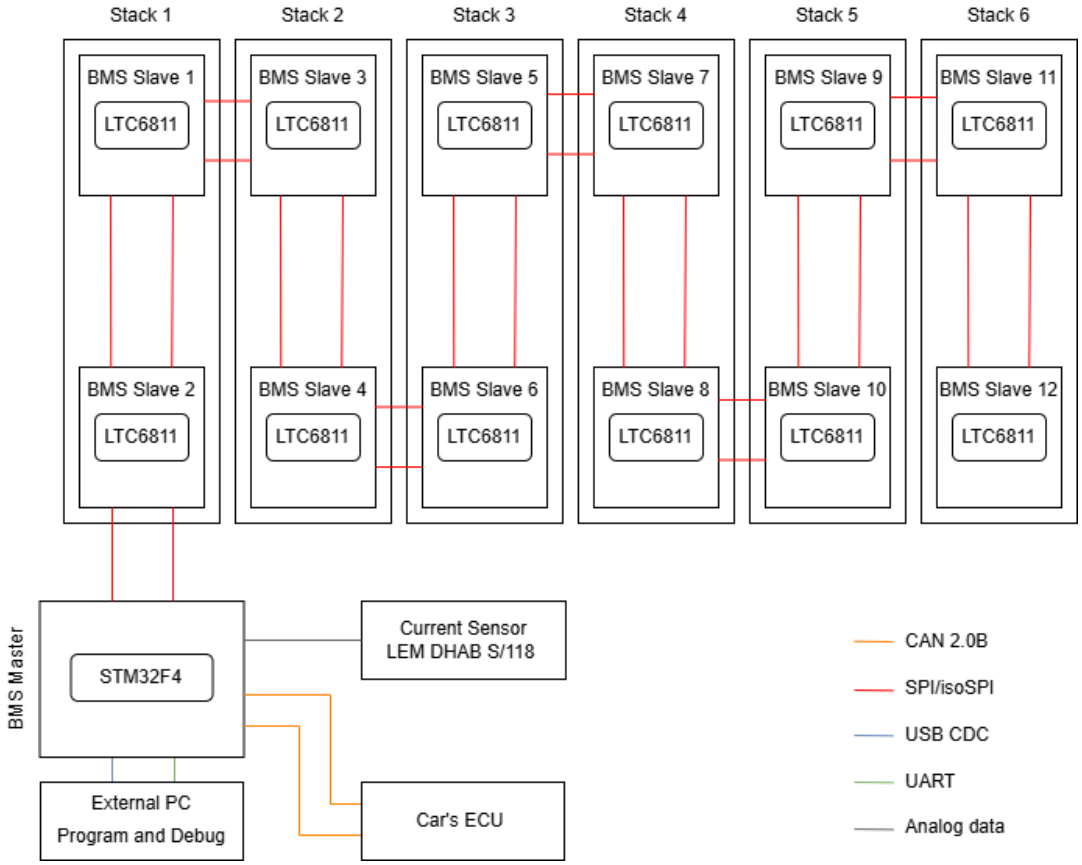


Figure 3.4: System-level architecture of the proposed BMS

3.9 Conceptual Comparison Between Orion BMS 2 and the Proposed Custom BMS

To synthesize the rationale for transitioning from the Orion BMS 2 to a custom-developed architecture, Table 3.3 presents a structured comparison across key design categories. These include topology, sensing capabilities, regulatory compliance, modifiability, and integration strategy, dimensions identified throughout the design process. Quantitative performance metrics and implementation-specific details are further explored in Chapter 6, which addresses real-world integration within the prototype.

Table 3.3: Comparison Between Orion BMS 2 and the Proposed Custom BMS

Feature	**Orion BMS 2 (COTS)**	**Proposed Custom BMS**
Architecture	Centralized	Master–slave
Temperature Sensing	8 channels	Atleast 30% cell coverage (FSG-compliant)
Cell Compatibility	Requires paid profile for P28A	Full support for custom cell models
SoC Estimation Method	CC (closed firmware)	EKF implementation (custom, tunable)
Safety Logic	Proprietary shutdown logic	Custom shutdown logic with modular enforcement
Regulatory Compliance	Partial (EV5.8.3, EV5.8.11 not met)	Compliance ensured
Integration in Accumulator	External mounting, high cabling	Embedded PCBs, reduced harnessing
Cost Efficiency	High unit cost, unused features	Cost aligned with system needs, unused functionalities avoided
Communication Protocols	Fixed CAN, limited UART	CAN, SPI, UART, USB supported
Modifiability	Closed-source, no algorithm access	Fully open and adaptable firmware

Chapter 4

Cell Modeling and Parameter Identification

Accurate state estimation in Li-ion batteries requires mathematical models that can replicate the cell's dynamic voltage response under varying operating conditions. This chapter presents the experimental development and parameter identification of an electrical model for the Molicel P28A 18650 cell, which serves as the foundation for the SoC estimator implemented in the custom BMS architecture. Following the theoretical survey outlined in Chapter 2, a first-order Thevenin-based ECM was selected. The HPPC procedure was employed to extract the model parameters under realistic dynamic loading. The model is subsequently validated and prepared for integration into the estimator framework discussed in Chapter 5.

4.1 Model Selection: Thevenin-Based Equivalent Circuit

Given the real-time and embedded nature of the BMS application, a first-order Thevenin model, also referred to as a first Resistor–Capacitor (RC) ECM was selected. This choice results from the need to accurately capture both the instantaneous voltage drop and the dynamic relaxation behavior observed in Li-ion cells during load transitions, while maintaining a low computational footprint compatible with embedded microcontrollers.

As detailed in Chapter 2, the Thevenin model strikes an optimal balance between simplicity and accuracy. It improves upon the Rint model by introducing a first-order dynamic branch, without resorting to higher-order models that may hinder implementation feasibility or complicate estimator tuning.

The continuous-time representation of the Thevenin model consists of the following elements:

- A nonlinear OCV(SOC) dependent on the battery's SoC;
- A series resistance R_0 modeling ohmic losses;
- A parallel resistor-capacitor branch (R_1, C_1) capturing first-order polarization and electrochemical lag.

Under constant current discharge, the cell voltage is described by:

$$V(t) = \text{OCV(SoC)} - I \cdot R_0(SoC) - I \cdot R_1(SoC) \cdot \left(1 - e^{-t/\tau}\right), \quad \tau = R_1(SoC)C_1(SoC) \tag{4.1}$$

To enable integration into digital signal processing environments, the continuous-time model must be discretized. This is achieved under the assumption of Zero-Order Hold (ZOH) inputs [37], where current is treated as piecewise constant over each sampling interval. The resulting discrete-time state-space form is:

$$x_{k+1} = A_d x_k + B_d u_k \tag{4.2}$$

$$y_k = C_d x_k + D_d u_k \tag{4.3}$$

For the Thevenin model formulated in state-space representation, the continuous-time state vector $x(t)$ includes the voltage across the RC branch:

$$x(t) = V_{RC}(t)$$

The system dynamics can then be expressed as:

$$\dot{x}(t) = -\frac{1}{R_1 C_1} x(t) + \frac{1}{C_1} I(t) \tag{4.4}$$

$$y(t) = \text{OCV(SOC)} - R_0 I(t) - x(t) \tag{4.5}$$

This leads to the following canonical continuous-time state-space form:

$$A = \left[-\tfrac{1}{R_1 C_1}\right], \quad B = \left[\tfrac{1}{C_1}\right],$$
$$C = \left[-1\right], \quad D = \left[-R_0\right]$$

These matrices define the linear dynamics of the Thevenin model under constant current excitation, and are the basis for the ZOH-based discretization process applied below.

The discrete-time system matrices are derived from the continuous-time model [37]:

$$A_d = e^{AT_s} \tag{4.6}$$

$$B_d = \int_0^{T_s} e^{A(T_s - \tau)} B \, d\tau \tag{4.7}$$

This formulation preserves the system dynamics over each sampling interval T_s, allowing for exact discretization under constant input assumptions, it was adopted in linear control systems and Kalman-based estimation frameworks for its numerical stability and implementation simplicity [37].

This transformation accurately preserves the continuous dynamics over the sampling interval T_s, allowing the system to be implemented in real time with consistent numerical behavior. The formulation also ensures compatibility with recursive estimation techniques, which depend on having linear or locally linearized models in discrete-time form.

4.2 Parameter Identification Using the HPPC Protocol

To identify the electrical parameters (R_0, R_1, C_1) of the Thevenin-based ECM, the HPPC protocol was employed. This standardized test applies a sequence of constant current pulses followed by open-circuit rest periods, enabling the extraction of both ohmic and dynamic electrochemical parameters [38].

Each HPPC segment allows for:

- Estimation of R_0 from the immediate voltage drop at pulse initiation;
- Extraction of R_1 and C_1 from the exponential voltage relaxation during the rest phase;
- Acquisition of the OCV after full relaxation, used to construct the OCV–SoC curve.

Figure 4.1 illustrates a typical HPPC voltage profile, with key voltage transitions annotated.

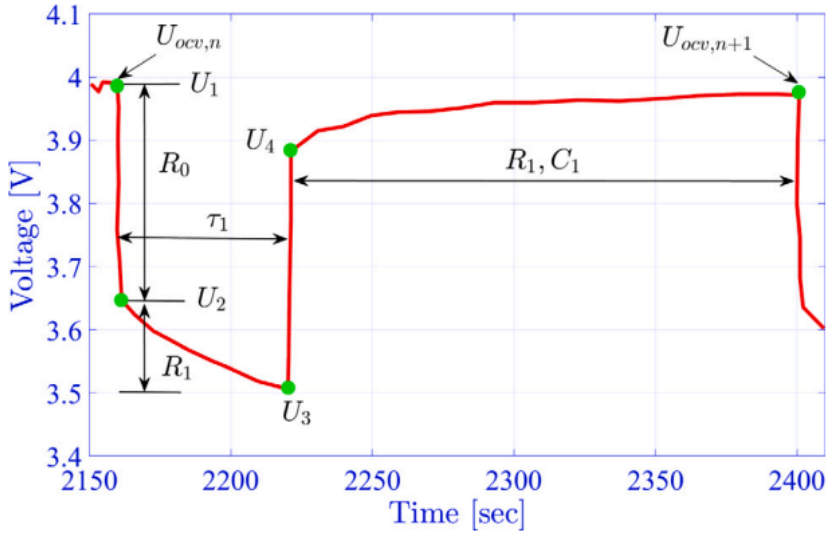


Figure 4.1: Typical HPPC response used for parameter identification, from [38].

The Thevenin model's voltage response under a constant current I is described by:

$$V(t) = \text{OCV(SOC)} - I \cdot R_0 - I \cdot R_1 \cdot \left(1 - e^{-t/\tau}\right), \quad \tau = R_1 C_1 \tag{4.8}$$

This time-domain formulation enables parameter identification by evaluating the cell's response at three distinct instants:

- **At** $t = 0$: The capacitor in the RC branch behaves as a short circuit. The voltage drop is purely ohmic:

$$V(0) = \text{OCV(SOC)} - I \cdot R_0$$

  Thus, the initial drop enables direct estimation of the internal resistance:

$$R_0 = \frac{V_{\text{before pulse}} - V_{\text{after pulse}}}{I}$$

- **At** $t = \tau = R_1 C_1$: The RC network has completed approximately 63.2% of its transition. The terminal voltage becomes:

$$V(\tau) = \text{OCV(SOC)} - I \cdot R_0 - 0.632 \cdot I \cdot R_1$$

  This value can be used to verify the quality of the exponential relaxation fit.

- **As** $t \to \infty$ **(»5**τ**)**: The capacitor becomes an open circuit and the system reaches steady state. The full polarization resistance is expressed in the terminal voltage:

$$V(\infty) = \text{OCV(SOC)} - I(R_0 + R_1)$$

  This allows calculation of R_1:

$$R_1 = \frac{V_{\text{after pulse}} - V_\infty}{I}$$

Once R_1 is known, the time constant τ can be extracted by fitting the voltage recovery during rest to the exponential model:

$$V(t) = V_\infty + A \cdot e^{-t/\tau}$$

Then, the capacitance is obtained as:

$$C_1 = \frac{\tau}{R_1}$$

These three instants provide sufficient information to fully determine the dynamic parameters of the model from measured voltage and current data. Their separation in the HPPC profile enables a structured, physics-informed parameter extraction that is computationally efficient and robust against noise.

4.3 Experimental Setup and Data Acquisition

Prior to the execution of the HPPC protocol, a single Molicel P28A cell was fully charged to its nominal maximum of 4.2 V, corresponding to 100% SoC. Charging was performed under controlled laboratory conditions to ensure thermal stability and repeatability.

After reaching full charge, the cell was subjected to a sequence of 0.1C current discharge pulses, each followed by a rest period sufficiently long to allow voltage stabilization. This procedure enabled the identification of both the instantaneous ohmic behavior (associated with R_0) and the slower polarization effects (modeled via R_1, C_1). Voltage and current were sampled at high resolution using a precision acquisition system, ensuring accurate reconstruction of both fast and slow dynamics.

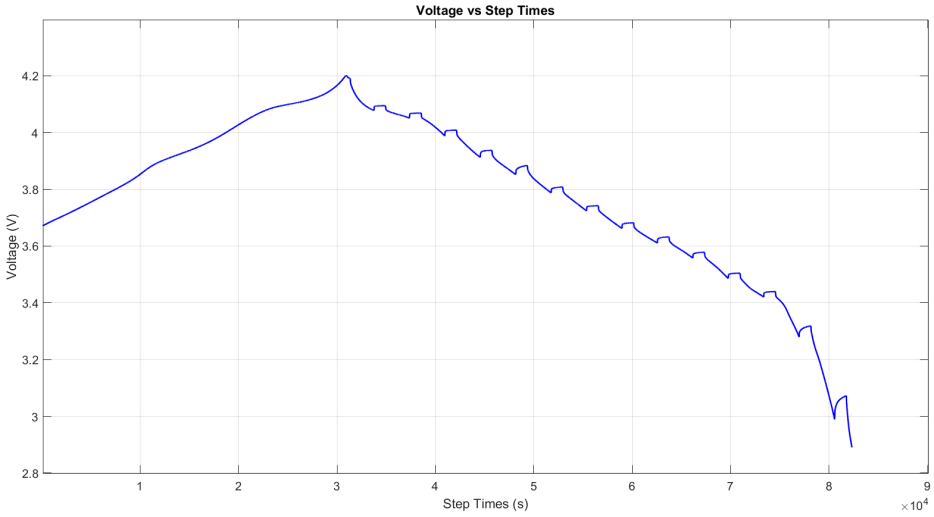


Figure 4.2: Voltage profile of HPPC test

As detailed in Section 4.2, the Thevenin model parameters were extracted from each HPPC segment. The instantaneous voltage drop at the onset of current enabled the computation of R_0, while the exponential voltage recovery during the subsequent rest period was used to extract R_1 and C_1 via curve fitting. This resulted in a discrete mapping of these parameters over the SoC range.

4.4 OCV–SoC Curve Identification

Following each rest phase, the fully stabilized terminal voltage was recorded and associated with the corresponding SoC, which was computed by CC:

$$SOC_k = SOC_{k-1} - \frac{I_k \cdot \Delta t_k}{C_{\text{nom}}}$$

A fifth-order polynomial was then fitted to the collected (SOC, OCV) data using the `polyfit` routine in MATLAB [39]. This operation performs a least-squares regression to minimize the squared deviation between the measured OCV values and the fitted curve:

$$\text{OCV(SOC)} = a_5 \cdot SOC^5 + a_4 \cdot SOC^4 + a_3 \cdot SOC^3 + a_2 \cdot SOC^2 + a_1 \cdot SOC + a_0 \tag{4.9}$$

The polynomial order was selected to balance flexibility and numerical stability, capturing the nonlinearity of the OCV–SoC relationship without introducing oscillations or overfitting artifacts.

Polynomial modeling of the OCV–SoC relationship is not only standard practice in battery state estimation, but also enables efficient integration into nonlinear estimators. Such polynomial models into EKF frameworks improves estimation accuracy and computational robustness, exploits the differentiability of polynomial functions to enable accurate linearization—critical for state-observer convergence [40]. .

The resulting functions, OCV(SOC), $R_0(SOC)$, $R_1(SOC)$, and $C_1(SOC)$, define the parameterized Thevenin model of the Molicel P28A cell. These are subsequently deployed within the estimation framework to support real-time SoC tracking in static and dynamic environments.

4.5 Parameter Summary and Model Validation

The final values obtained for the model parameters are summarized in Table 4.1. These values were integrated into a simulation of the first-order Thevenin model and validated against a segment of experimental data, as shown in Figure 4.3.

Table 4.1: Summary of the Identified Thevenin Model Parameters

Parameter	**Value / Description**
R_0(SoC)	Fifth-order polynomial fit (see Fig. 4.4a)
R_1(SoC)	Fifth-order polynomial fit (see Fig. 4.4b)
C_1(SoC)	Fifth-order polynomial fit (see Fig. 4.4c)
OCV(SoC) Polynomial	Fifth-order polynomial (coefficients below, Fig. 4.4d)
SoC Range Modeled	5% – 100%
HPPC Protocol	0.1C pulses with rest periods (Fig. 4.3)

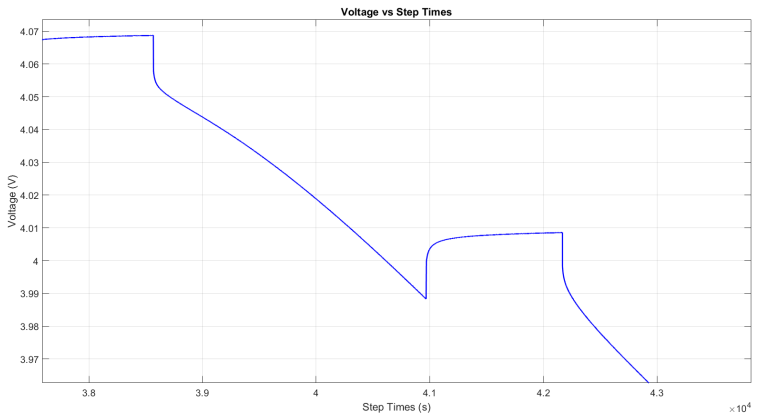
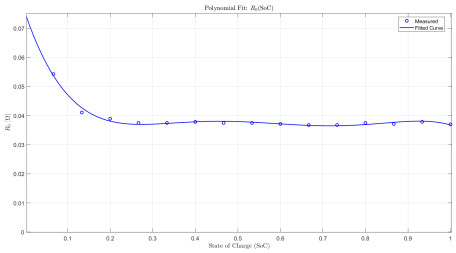
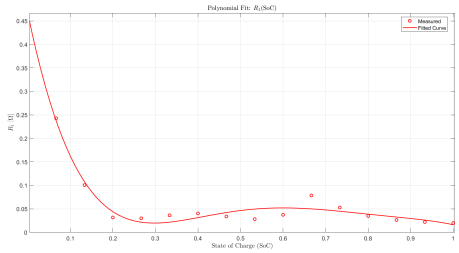


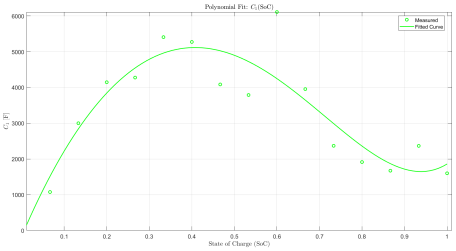
Figure 4.3: Zoomed-in HPPC segment showing step, drop, and relaxation.



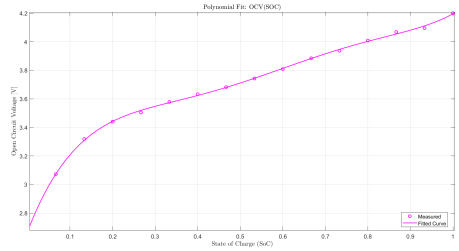
(a) Polynomial fit of R_0(SoC).



(b) Polynomial fit of R_1(SoC).



(c) Polynomial fit of C_1(SoC).



(d) Polynomial fit of OCV(SOC).

Figure 4.4: Fitted fifth-order polynomial curves for the cell parameters and OCV. These functions parameterize the Thevenin model over the SoC range.

The polynomial used to describe the OCV as a function of SoC is defined in Equation 4.9, and was fitted using least-squares regression. The resulting coefficients are:

$$a_5 = 15.8003, \quad a_4 = -47.1367, \quad a_3 = 53.0475,$$
$$a_2 = -27.8100, \quad a_1 = 7.6176, \quad a_0 = 2.6764$$

This function is continuously differentiable, making it suitable for direct implementation in the nonlinear observation model of the EKF, discussed in Chapter 5.

Together with the SoC-dependent polynomial representations of R_0, R_1, and C_1, the OCV–SoC curve completes a physically-grounded and real-time capable model of the Molicel P28A cell. These functions are the output of a robust experimental identification process, which included standardized HPPC testing and model validation against empirical voltage data.

The final fitted parameters were derived from the cell's dynamic response, as characterized in Figure 4.3, where a representative HPPC step is used to illustrate the voltage transient captured by the Thevenin model. Their successful integration into simulation and alignment with measurements confirm the validity of the approach.

These results form the foundation of the estimator developed in the next chapter, where the parameterized model is embedded in an EKF framework to enable real-time SoC tracking under variable current conditions.

Chapter 5

SoC estimation using an EKF

Reliable state estimation in Li-ion batteries requires algorithmic frameworks capable of tracking internal electrochemical variables under nonlinear and time-varying conditions. This chapter presents the formulation, implementation, and validation of an EKF designed to estimate the SoC of the Molicel P28A 18650 cell, using the parameterized Thevenin-based model identified in Chapter 4. Building upon the modeling foundations and nonlinear characteristics introduced in Chapter 2, the EKF was selected for its suitability in embedded systems and its ability to accommodate model nonlinearities through local linearization. The filter is derived from a discrete-time state-space representation of the cell dynamics, and incorporates real-time voltage and current measurements to recursively update the internal SoC estimate. Following its mathematical derivation, the estimator is implemented on a microcontroller-based testboard using a Teensy 4.1 and LTC6811, and is experimentally validated by comparing its performance against a simulated reference across dynamic load conditions [31].

5.1 Mathematical formulation of the EKF

State estimation in battery systems requires inferring internal quantities, such as the SoC, from externally measurable variables like current and terminal voltage. Due to the nonlinear dependence of these measurements, particularly the terminal voltage, on the battery's internal state, linear estimation techniques such as the classic KF are insufficient. The EKF overcomes this limitation by extending the KF framework to nonlinear systems through linearization around the current state estimate. This section derives the EKF equations from the continuous-time nonlinear state-space model, explains their discretization for implementation, and details how each component contributes to recursive state prediction and correction in the context of SoC estimation.

5.1.1 From continuous-time dynamics to discrete-time estimation

The EKF operates on a discrete-time nonlinear system model of the form:

$$x_{k+1} = f(x_k, u_k) + w_k \tag{5.1}$$

$$y_k = h(x_k, u_k) + v_k \tag{5.2}$$

where x_k is the state vector, u_k is the system input, y_k is the measurement output, and w_k, v_k are zero-mean Gaussian noise processes with covariance matrices Q_k and R_k, respectively. In this application, the state vector is defined as:

$$x_k = \begin{bmatrix} \mathrm{SoC}_k \\ V_{RC,k} \end{bmatrix} \tag{5.3}$$

The input u_k corresponds to the applied current I_k, and the output y_k is the measured terminal voltage. The function $f(x_k, u_k)$ captures the nonlinear evolution of the system's state, while $h(x_k, u_k)$ defines the nonlinear measurement relationship. These functions are derived directly from the discrete-time form of the Thevenin equivalent circuit, previously parameterized in Chapter 4.

The evolution of the SoC is governed by CC:

$$\mathrm{SoC}_{k+1} = \mathrm{SoC}_k - \frac{T_s}{C_{\mathrm{nom}}} I_k \tag{5.4}$$

where T_s is the sampling interval and C_{nom} is the nominal capacity of the cell. The second state, $V_{RC,k}$, represents the voltage across the RC network modeling cell polarization, and evolves according to the discrete-time approximation of its continuous-time differential equation:

$$\frac{dV_{RC}(t)}{dt} = -\frac{1}{\tau} V_{RC}(t) + \frac{1}{C_1} I(t) \tag{5.5}$$

with $\tau = R_1(\mathrm{SoC}) C_1(\mathrm{SoC})$. Under the assumption of a ZOH input, the closed-form solution to this equation over a sampling interval T_s leads to the following update rule:

$$V_{RC,k+1} = a_k V_{RC,k} + b_k I_k \tag{5.6}$$

where the coefficients are defined as:

$$a_k = e^{-T_s/\tau_k} \tag{5.7}$$

$$b_k = R_1(\mathrm{SoC}_k) \cdot (1 - a_k) \tag{5.8}$$

The measurement model relates the system's internal state to the terminal voltage, which is expressed as:

$$V_k = \mathrm{OCV}(\mathrm{SoC}_k) - R_0(\mathrm{SoC}_k) \cdot I_k - V_{RC,k} \tag{5.9}$$

This function $h(x_k, u_k)$ is explicitly nonlinear due to the OCV dependence on SoC and the state-dependent resistances, making the application of a nonlinear filter essential.

5.1.2 Prediction and correction in the EKF framework

At each time step k, the EKF operates in two stages: a prediction stage, where the next state is forecasted based on the process model, and a correction stage, where the forecast is updated using the new measurement. The first step uses the known system dynamics to propagate the state forward:

$$\hat{x}_{k|k-1} = f(\hat{x}_{k-1|k-1}, u_{k-1}) \tag{5.10}$$

Simultaneously, the state covariance matrix, which represents uncertainty in the state estimate, is updated as:

$$P_{k|k-1} = F_k P_{k-1|k-1} F_k^\top + Q_k \tag{5.11}$$

Here, F_k is the Jacobian matrix of partial derivatives of the function $f(x,u)$ with respect to the state vector x, evaluated at the current estimate:

$$F_k = \left.\frac{\partial f}{\partial x}\right|_{x=\hat{x}_{k-1|k-1}, u=u_{k-1}} \tag{5.12}$$

The Jacobian F_k captures how small changes in the state vector propagate through the nonlinear dynamics, and is fundamental to linearizing the model locally around the predicted trajectory.

Once the measurement y_k is available, the filter performs the correction step. The predicted output is computed as:

$$\hat{y}_{k|k-1} = h(\hat{x}_{k|k-1}, u_k) \tag{5.13}$$

The difference between the actual and predicted measurements, termed the innovation, is given by:

$$\tilde{y}_k = y_k - \hat{y}_{k|k-1} \tag{5.14}$$

This innovation is used to correct the prediction using a gain factor K_k, known as the Kalman gain, which determines how much the estimate should be adjusted in response to new information:

$$K_k = P_{k|k-1} H_k^\top \left(H_k P_{k|k-1} H_k^\top + R_k \right)^{-1} \tag{5.15}$$

In this expression, H_k is the Jacobian of the measurement function $h(x,u)$ with respect to the state vector x, evaluated at the predicted state:

$$H_k = \left.\frac{\partial h}{\partial x}\right|_{x=\hat{x}_{k|k-1}, u=u_k} \tag{5.16}$$

The matrix H_k encapsulates the sensitivity of the measured voltage to variations in the state variables, which in this case includes both SoC and the dynamic voltage V_{RC}.

The corrected state estimate is then computed as:

$$\hat{x}_{k|k} = \hat{x}_{k|k-1} + K_k \tilde{y}_k \tag{5.17}$$

Finally, the updated covariance matrix is:

$$P_{k|k} = (I - K_k H_k) P_{k|k-1} \tag{5.18}$$

This completes one recursive iteration of the EKF. At each new time step, the process repeats, with the updated state estimate $\hat{x}_{k|k}$ serving as the basis for the next prediction.

5.1.3 Process noise covariance matrix Q_k

The process noise covariance matrix Q plays a critical role in the performance of the EKF, as it encapsulates the uncertainty associated with the system dynamics. In practice, the mathematical model used to describe battery behavior is always an approximation of the real system and cannot capture all its complexities. Within the context of SoC estimation, the Q matrix accounts for several sources of modeling error and unmodeled dynamics. These include phenomena such as temperature variations, cell aging, and hysteresis effects, which are difficult to capture analytically but can significantly influence battery performance over time. Additionally, limitations in sensor resolution or quantization noise, particularly in current measurements, introduce further uncertainty into the state update. Finally, inaccuracies in the parameterization of model curves, such as those describing OCV or resistance as a function of SoC and temperature, also contribute to deviations between the predicted and actual system behavior. By appropriately tuning the Q matrix, the EKF can maintain robust performance in the presence of these real-world uncertainties.

Tuning is a critical and non-trivial task. If its values are set too low, the estimator will place excessive trust in the model predictions, potentially ignoring relevant deviations in the measurements and accumulating significant error over time. On the other hand, if is overestimated, the filter will consider the model highly unreliable, reacting too strongly to every measurement and becoming susceptible to noise [32], [41], [42].

In practice, determining is one of the most complex steps in EKF design. Although some theoretical formulations exist to compute this matrix from parameter uncertainty and system sensitivity, these often require full knowledge of parameter covariance, symbolic derivatives, and real-time adaptation, which may not be practical or computationally feasible in embedded applications [41].

As such, a simplified yet physically informed strategy was adopted in this work. A diagonal structure was assumed, where each diagonal term represents independent process noise acting on each state. The value for the SoC component, reflects how much error accumulates due to current sensor inaccuracy and integration over the sampling period. Likewise, reflects the uncertainty in the RC-network voltage due to modeling simplifications and dynamic transients.

The final values were determined through iterative tuning and validation: starting from conservative initial guesses, the values were refined to achieve a balance between filter responsiveness

and noise robustness. This pragmatic approach is commonly found in the literature and is justified when model identification and test conditions are well controlled.

5.1.4 Measurement noise covariance matrix R_k

The measurement noise covariance matrix R_k encodes the statistical uncertainty associated with the observation model of the system. In the context of this application, where the EKF estimates the battery's internal state based on real-time voltage readings, R_k quantifies the expected variance in the terminal voltage measurement due to sensor imperfections, external interference, and model mismatch in the output equation.

Formally, R_k appears in the EKF measurement update equations as a key factor determining the Kalman gain K_k, and thus governs how much the state estimate should be corrected in response to a new measurement. A small value of R_k indicates high confidence in the measurement, resulting in a large Kalman gain and aggressive correction. Conversely, a large R_k reflects low confidence in the measurement, causing the filter to rely more on the model prediction and less on the observed data.

In practice, R_k is often assumed to be a time-invariant diagonal matrix, especially when the measurements are uncorrelated and the sensor characteristics are well known. [41], where the diagonal elements of R_k are defined as:

$$R_k = k_R \cdot \sigma_V^2 \tag{5.19}$$

Here, σ_i represents the nominal standard deviation of the i-th measurement channel, and $k_R \geq 1$ is a tuning factor introduced to enhance robustness in the presence of unmodeled noise or rare large disturbances. While setting k_R too low can cause the filter to overreact to noise, an excessively large R_k reduces the influence of measurements altogether, potentially causing filter divergence in the presence of model errors.

5.1.5 Estimation error covariance matrix P_k

The estimation error covariance matrix P_k represents the EKF's internal belief about the uncertainty associated with the current state estimate $\hat{x}_k$. Its diagonal elements reflect the variance of individual state components, namely, the SoC and the RC-network voltage V_{RC}, while the off-diagonal terms represent statistical correlations between these errors.

In this application, P_k is initialized at time $k = 0$ as:

$$P_0 = \begin{bmatrix} \sigma_{\text{SoC}}^2 & 0 \\ 0 & \sigma_{V_{RC}}^2 \end{bmatrix} \tag{5.20}$$

This choice assumes that the initial estimation errors in SoC and V_{RC} are statistically independent, which is a reasonable simplification given their different physical origins and the absence of a strong correlation structure in the model.

Unlike the measurement noise covariance R_k, which can often be inferred from datasheets or sensor calibration data, the uncertainty in the initial state $\hat{x}_0$ cannot be measured directly, the filter designer must estimate the degree of confidence in the chosen initial state. If $\hat{x}_0$ is far from the true state x_0 and P_0 is chosen too optimistically, the Kalman gain K becomes too small, and the filter will overly trust its inaccurate model prediction. In such cases, subsequent measurements will have limited corrective influence, and the estimator may converge to the wrong state or even diverge [41].

To avoid this, a recommended approach is to initialize P_0 based on known bounds of the initial state, when exact values are unavailable. This strategy defines:

$$\hat{x}_0 = \frac{x_u + x_l}{2}, \quad P_0 = \text{diag}\left[\left(\frac{x_u - x_l}{2}\right)^2\right] \tag{5.21}$$

where x_u and x_l are the upper and lower bounds, respectively, of each state variable. This ensures that P_0 captures the uncertainty in the best estimate $\hat{x}_0$, without underestimating the possible deviation from the true state.

Additionally, tuning factors $k_{P0,i}$ may be applied to the individual diagonal entries to modulate the filter's confidence in specific states:

$$P_0(i,i) = k_{P0,i} \cdot \left(\frac{x_{u,i} - x_{l,i}}{2}\right)^2 \tag{5.22}$$

In the present work, the initial state $\hat{x}_0$ is determined from a rest condition in which the cell current $I \approx 0$ and the terminal voltage equals the OCV. The OCV is then mapped to SoC using the polynomial relationship derived in Chapter 4. Under these conditions, $V_{RC} \approx 0$, allowing for a highly confident initialization. Consequently, a small diagonal P_0 is justifiable. Nonetheless, this method is only valid in scenarios where initial rest conditions allow accurate OCV-based initialization.

5.1.6 Selection of the covariance matrices

The performance of the EKF is strongly dependent on the tuning of its covariance matrices, which reflect the filter's confidence in the model, in the measurements, and in the initial state. The values of Q_k, R_k, and P_0 were selected based on sensor specifications, model identification accuracy, and the conditions under which the test procedures were performed. Rather than relying on arbitrary values, the design of the estimator was guided by a physically informed strategy tailored to the specific hardware architecture (Teensy 4.1 + LTC6811) and the parameterization methodology presented in Chapter 4.

5.1.6.1 Measurement noise covariance matrix R_k

The matrix R_k quantifies the uncertainty associated with the terminal voltage measurement, which is acquired through the LTC6811 battery monitoring IC. This measurement is subject to quantiza-

tion noise, analog front-end variation, and signal disturbances from the test setup. According to the datasheet, the accuracy of the LTC6811 under ideal conditions is around 1–2 mV, but can degrade to 15 mV in worst-case scenarios.

In the absence of multiple independent measurements, R_k reduces to a scalar variance representing the uncertainty of a single terminal voltage reading. The measurement noise variance is modeled as [41]:

$$R_k = k_R \cdot \sigma_V^2 \tag{5.23}$$

where σ_V is the nominal standard deviation of the measurement noise, and $k_R \geq 1$ is a robustness tuning factor. For this application, a conservative upper-bound estimate of $\sigma_V = 15$ mV was adopted, yielding:

$$R_k = (15 \times 10^{-3})^2 = 2.25 \times 10^{-4} \tag{5.24}$$

5.1.6.2 Initial estimation error covariance matrix P_0

The matrix P_0 defines the initial uncertainty associated with the state estimate $\hat{x}_0$. In contrast to R_k, this matrix cannot be determined experimentally and must be assigned based on the confidence in the initialization procedure. If P_0 is chosen too optimistically while the true state differs from $\hat{x}_0$, the Kalman gain becomes small, and the filter may ignore valuable measurement updates, potentially leading to divergence [41].

In this work, the test protocol involved long rest periods before each dynamic profile, ensuring that:

$$I \approx 0, \quad V_{RC} \approx 0, \quad V = \text{OCV}(\text{SoC}) \tag{5.25}$$

Under these conditions, the initial SoC was inferred from the measured voltage using the OCV–SoC polynomial mapping described in Chapter 4, with high confidence. Consequently, a small initialization covariance is justifiable:

$$P_0 = \begin{bmatrix} 1 \times 10^{-6} & 0 \\ 0 & 1 \times 10^{-6} \end{bmatrix} \tag{5.26}$$

This strategy would not be suitable in real-world BMS scenarios without rest periods. In such cases, a more general and robust initialization would involve defining upper and lower bounds x_u and x_l for each state and computing:

$$P_0 = \text{diag}\left(\left(\frac{x_u - x_l}{2}\right)^2\right) \tag{5.27}$$

This formulation ensures that the initial uncertainty reflects realistic knowledge about the possible deviation from the true state.

5.1.6.3 Process Noise Covariance Matrix Q_k

The matrix Q_k captures uncertainties in the dynamic model, including parameter variability, numerical approximations, and the accumulation of current measurement noise in the SoC integration. The values of Q_k were chosen empirically but guided by physical considerations, such as sensor inaccuracies, ensuring both filter stability and adaptability to unmodeled effects.

In line with standard practice, a diagonal structure was adopted:

$$Q_k = \begin{bmatrix} q_{\text{SoC}} & 0 \\ 0 & q_{V_{RC}} \end{bmatrix} = \begin{bmatrix} 1 \times 10^{-6} & 0 \\ 0 & 1 \times 10^{-6} \end{bmatrix} \tag{5.28}$$

5.2 Experimental validation and performance evaluation

In order to validate the accuracy and robustness of the proposed SoC estimator, a comprehensive experimental campaign was conducted using a custom testbench designed to emulate real-world battery operating conditions. This section describes the experimental setup used for data acquisition and estimator execution, the applied test procedures, and the methodology employed to evaluate the estimation error across different load profiles.

The scripts used for validating the results are detailed in Appendix A. These include MATLAB scripts for simulation, C code to be executed on the real-time microcontroller, and Python code for running the simulations and data analysis.

5.2.1 Hardware setup

To validate the estimator in a real-world setting, an experimental platform was assembled to support the real-time execution of the EKF on embedded hardware. The system architecture was designed to mirror the structure of a production-grade BMS, ensuring both fidelity and practical relevance. At the core of the setup, a Teensy 4.1 microcontroller interfaces with an LTC6811 battery monitoring IC, which acquires the voltage measurements of each individual cell in the battery pack. The battery under test consists of six Molicel P28A 18650 connected in series, forming a 6S configuration.

To apply controlled current profiles for both charging and discharging, the system is connected to an external programmable power supply and an electronic load. These components allow precise modulation of current flow through the pack, enabling the execution of standardized test procedures. The LTC6811 communicates with the microcontroller via the SPI protocol, and both current and voltage data are sampled at a fixed rate. These measurements are used by the EKF to update the internal SoC estimate in real time. The complete hardware configuration is shown in Figure 5.1.

In the simulation environment, the initial SoC is provided as an input parameter, based on prior experimental knowledge of the actual SoC at the start of the test. This ensures that the simulated estimator begins from a realistic operating point, allowing for a meaningful evaluation of its convergence behavior and tracking accuracy over time.

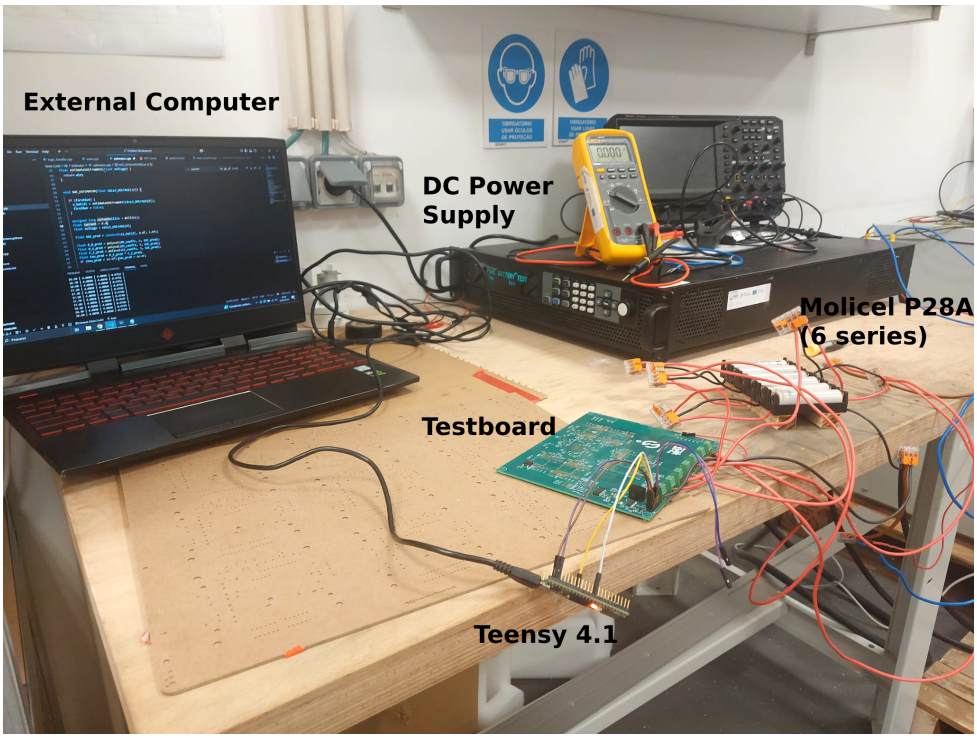


Figure 5.1: Experimental setup used for EKF validation, including Teensy 4.1, LTC6811, and a 6-cell Li-ion pack.

This embedded configuration mirrors the architecture expected for a production BMS, thus ensuring that the implementation and results are both representative and scalable.

5.2.2 Test Procedures

To comprehensively evaluate the estimator's performance under different dynamic conditions, a set of controlled discharge tests was conducted using predefined current profiles. The following load conditions were applied:

- Constant discharge at **0.5C** (1.4A)
- Constant discharge at **1C** (2.8A)
- A dynamic current profile replicating the power demand of a FS Autocross event, including rapid transients from acceleration, braking, and cornering. Unlike standard test profiles, it reflects realistic vehicle behavior under competitive driving.

Each profile was executed using the external electronic load, controlled via software to generate reproducible current patterns. Terminal voltage and applied current were sampled at 5Hz (0.2s) and logged simultaneously with the EKF's real-time SoC estimate.

5.2.3 Constant Current Discharge at 0.5C

This test profile applies a constant discharge current of 0.5C (approximately 1.4A) to the cells, representing a mild operating condition with relatively slow dynamics. The goal is to evaluate the EKF's baseline tracking performance.

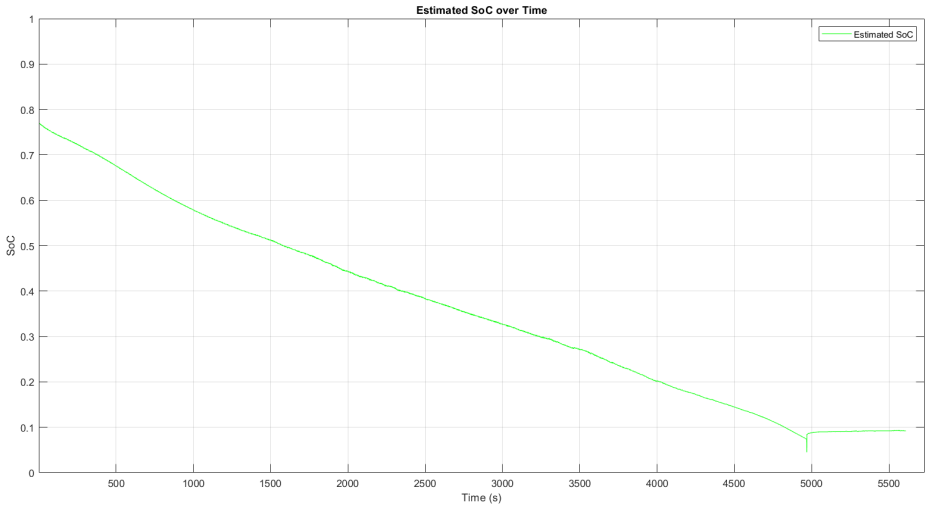


Figure 5.2: EKF SoC estimation under 0.5C constant current discharge.

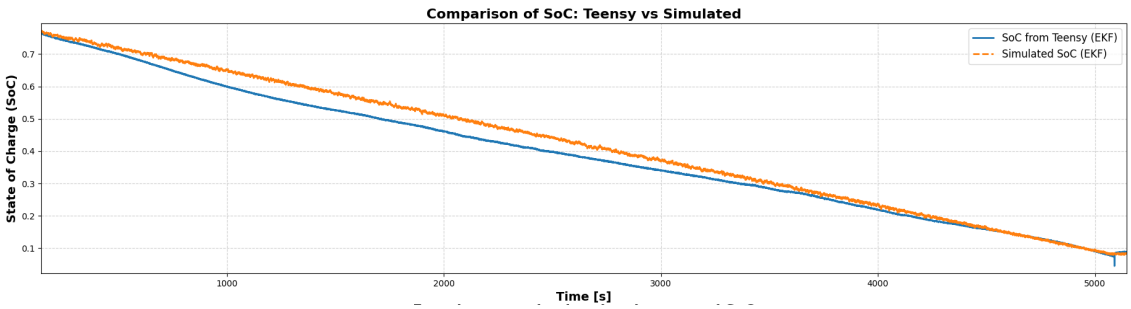


Figure 5.3: EKF SoC estimation vs. simulated under 0.5C constant current discharge.

As shown in Figures 5.2 and 5.3, the EKF estimation closely follows the reference SoC trajectory, demonstrating the filter's effectiveness in estimating the SoC under relatively stable operating conditions. During the test, a constant current discharge was applied within the linear range of the battery's SoC. As the discharge progressed, the voltage dropped steadily, and the SoC decreased predictably, until reaching approximately 10% of the battery's SoC. At this point, the dynamics became nonlinear due to the battery's internal resistance and other factors. After this period, the system was left at rest, allowing the estimation to stabilize. The EKF maintained a smooth and consistent tracking of the SoC trajectory throughout the test, demonstrating its effectiveness in such mild conditions.

Figure 5.4 illustrates the percentage error between the measured and estimated voltage during the 0.5C discharge test. The error remains consistently low throughout the discharge phase, indicating minimal divergence between the estimated and measured voltages. A small increase in error is observed at the end of the test, where the battery was left at rest. This was expected, as the system experiences slight fluctuations when the current was no longer applied. The maximum error reached 0.5% after the current pulse was removed, but the overall error remained within an acceptable range, validating the estimator's robustness.

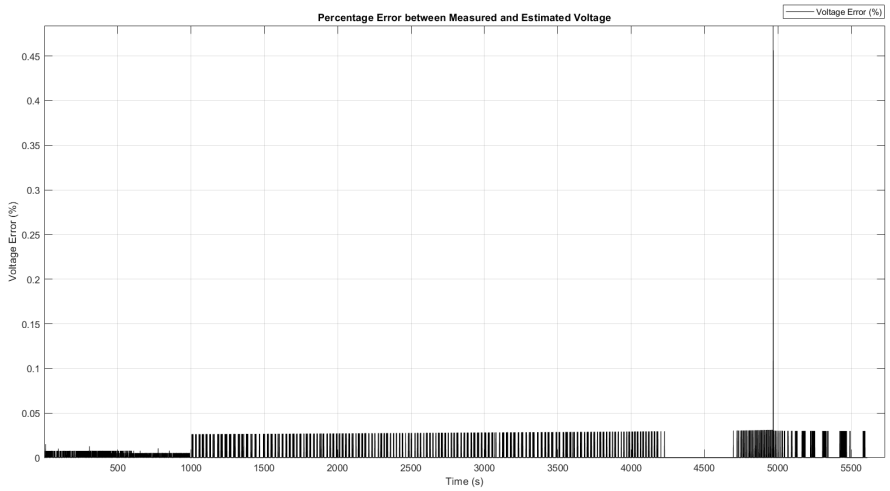


Figure 5.4: Percentage error between measured and estimated voltage under 0.5C test conditions.

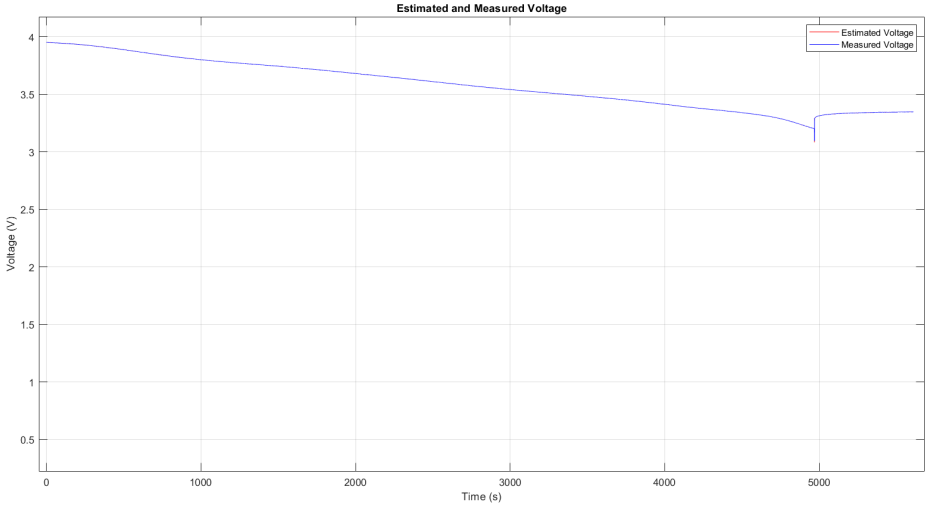


Figure 5.5: Comparison between the measured voltage and the estimated voltage for 0.5C test.

The results in Figures 5.2, 5.4, and 5.5 demonstrate excellent agreement between the EKF estimate and the reference SoC trajectory. A comparison was made between the simulated SoC (calculated in Python) and the real SoC estimated using the embedded system. The comparison

showed that the EKF estimator closely tracked the reference, with a SoC error ranging from 6% to 1%, highlighting the accuracy and robustness of the estimator. The results are in Figures 5.3, 5.6.

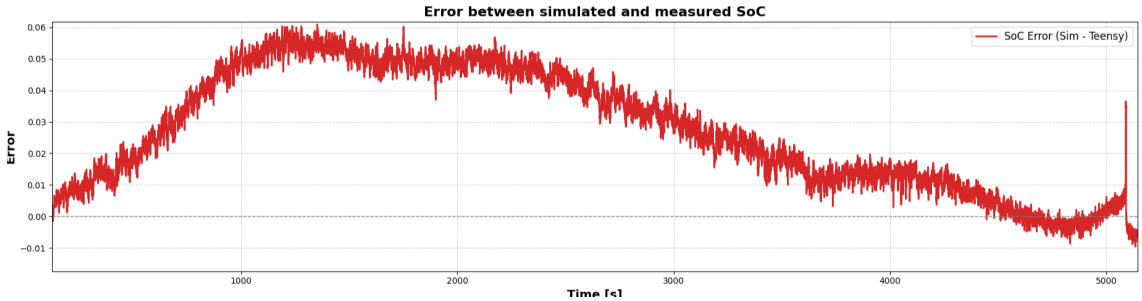


Figure 5.6: Pointwise estimation error for 0.5C test.

5.2.4 Constant Current Discharge at 1C

This test profile applies a constant discharge current of 1C (approximately 2.8A), representing a moderate operating condition with a faster discharge rate. The goal is to evaluate the EKF's performance in tracking the SoC during more dynamic operating conditions. As shown in Figures 5.7 and 5.8, the EKF estimation closely follows the reference SoC trajectory, demonstrating the filter's effectiveness in estimating the SoC under moderate operating conditions. During the test, a constant current discharge was applied to the battery around 80% SoC. The voltage decreased steadily, and the SoC decreased predictably. As the discharge progressed, the system reached a point where the battery's internal resistance became more significant, affecting the dynamics. After this period, the system was left at rest, allowing the estimation to stabilize. The EKF maintained a smooth and consistent tracking of the SoC trajectory throughout the test, demonstrating its robustness under moderate conditions.

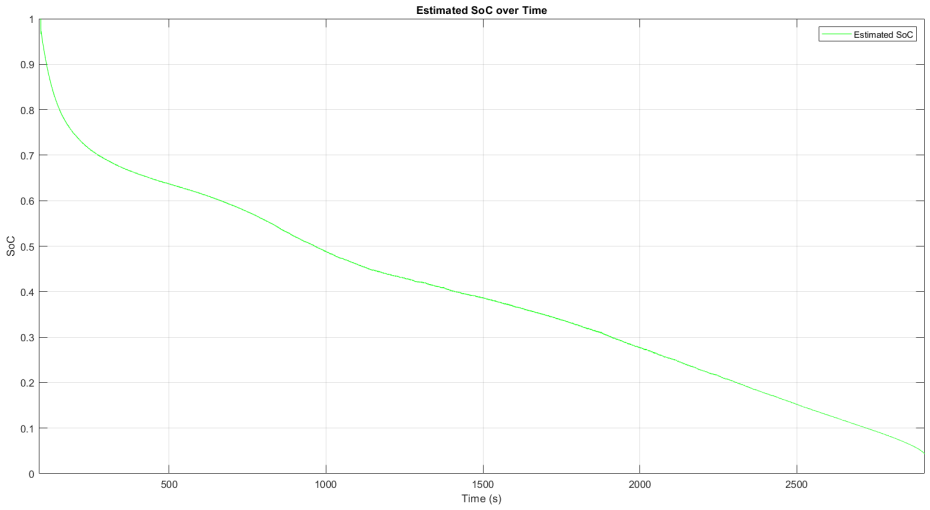


Figure 5.7: EKF SoC estimation under 1C constant current discharge.

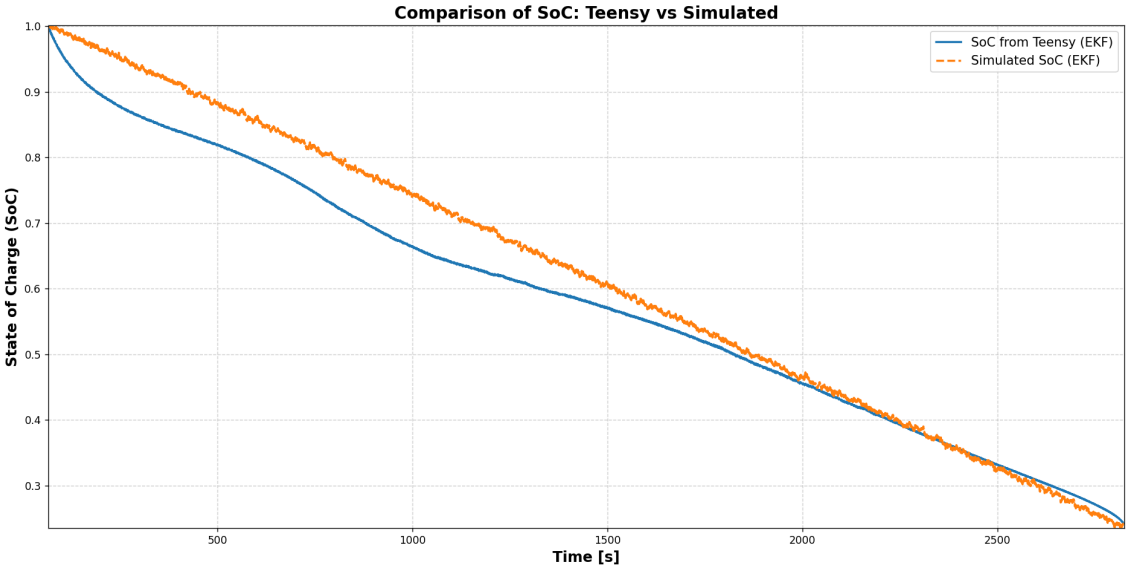


Figure 5.8: EKF SoC estimation vs. reference under 1C constant current discharge.

Figure 5.9 illustrates the percentage error between the measured and estimated voltage during the 1C discharge test. The error remains consistently low throughout the discharge phase, indicating minimal divergence between the estimated and measured voltages. A small increase in error is observed at the end of the test when the battery was left at rest. This behavior was expected, as the system experiences slight fluctuations when no current is applied. The maximum error reached 2% after the current pulse was removed, but the overall error remained within an acceptable range, validating the estimator's accuracy.

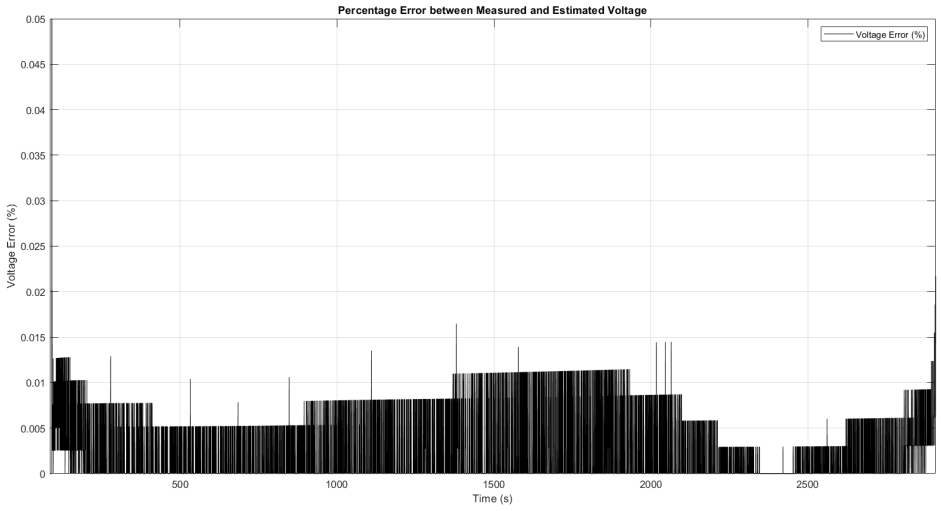


Figure 5.9: Percentage error between measured and estimated voltage under 1C test conditions.

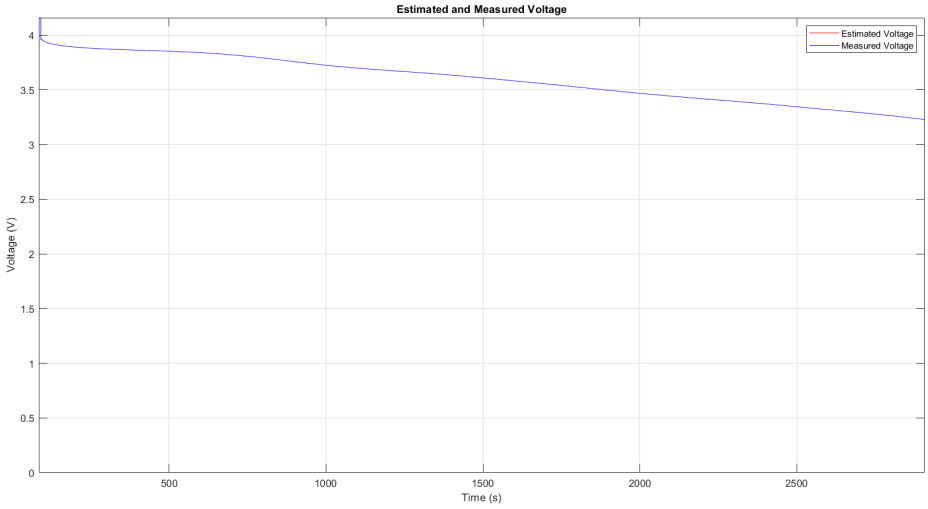


Figure 5.10: Comparison between the measured voltage and the estimated voltage for 1C test.

The results in Figures 5.7, 5.9, and 5.10 show excellent agreement between the EKF estimate and the reference SoC trajectory. The EKF estimator closely tracked the reference, with a SoC error ranging from 9% to 2%, highlighting the accuracy and robustness of the estimator. The results are presented in Figures 5.8, 5.11.

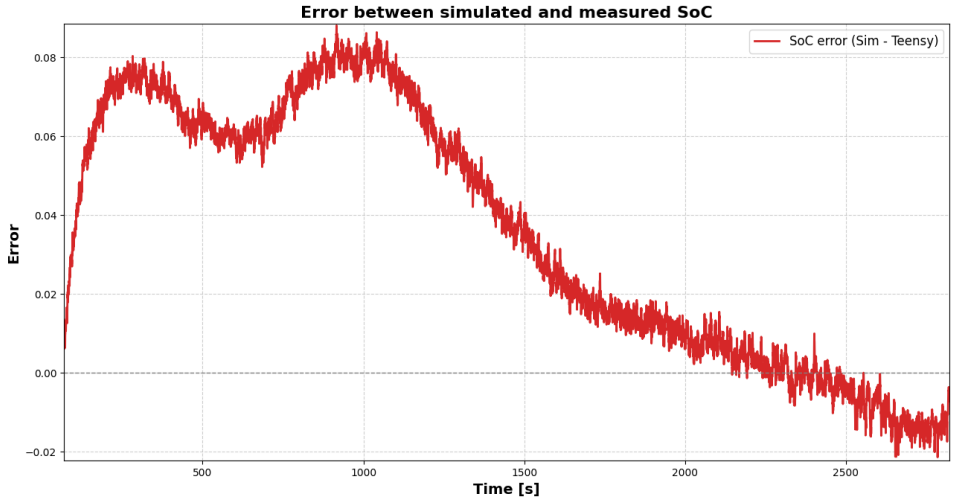


Figure 5.11: Pointwise estimation error for 1C test.

5.2.5 Dynamic Load Validation for FS Autocross Scenario

In order to validate the performance of the EKF estimator under realistic operating conditions, a dynamic current profile was implemented to emulate the behavior of a FS vehicle during an

Autocross event. The profile aimed to replicate the transient and non-linear current demands that occur during actual competition runs.

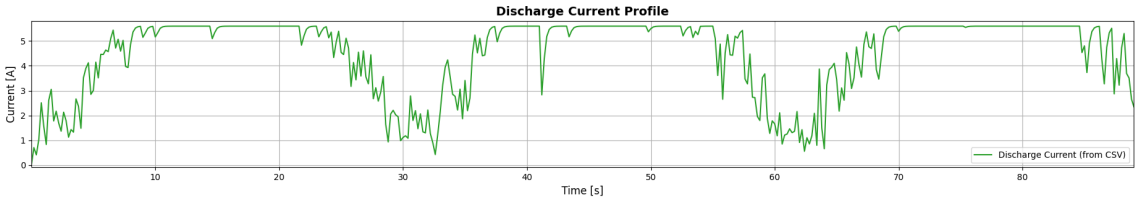


Figure 5.12: Current profile applied to the power supply to emulate the dynamic load conditions of a FS Autocross event.

The discharge current illustrated in Figure 5.12 presents fast and irregular variations, effectively emulating the dynamic behavior observed during a FS race scenario. This current profile was specifically designed to evaluate the robustness of the EKF in estimating both the terminal voltage and the SoC under conditions of rapidly changing load demands.

To thoroughly assess the estimator's performance, two distinct test cases were conducted. The first test focused on the upper SoC region, above 80%, where the cell voltage response is highly non-linear. This region is particularly challenging for state estimation due to the reduced voltage sensitivity to changes in SoC. The second test was carried out in the linear region of the cell's discharge curve, where the terminal voltage exhibits a more direct and proportional relationship with the SoC. These complementary test cases enabled a comprehensive evaluation of the EKF's accuracy across different operating conditions.

5.2.5.1 Voltage Tracking Performance: Non-linear SoC Region

To further challenge the estimator, a similar test was conducted in the high SoC region (around 90%), where the voltage response to SoC changes is inherently weak. This test is particularly important because the observability of SoC in this region is significantly reduced, making estimation more susceptible to error.

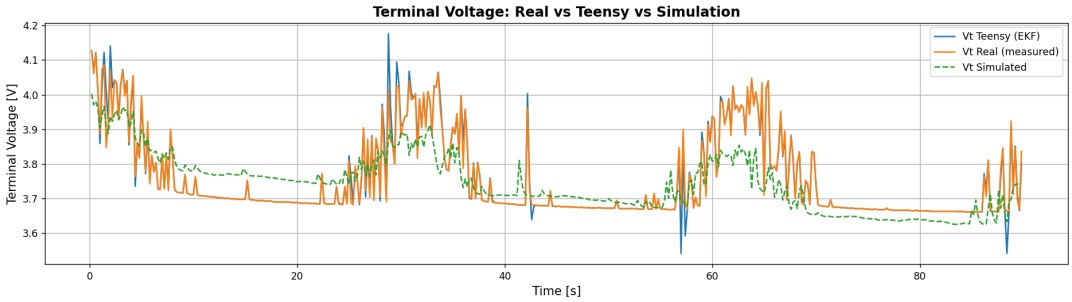


Figure 5.13: Terminal voltage comparison (measured, Teensy-EKF estimate, and simulation) in the non-linear SoC region.

Figure 5.13 shows the voltage response in this challenging operating region. The EKF estimator deployed on the Teensy microcontroller continues to follow the trend of the measured

voltage, though with slightly greater deviation than observed in the linear test. These deviations are expected due to the reduced voltage sensitivity to SoC changes, which lowers the system's observability.

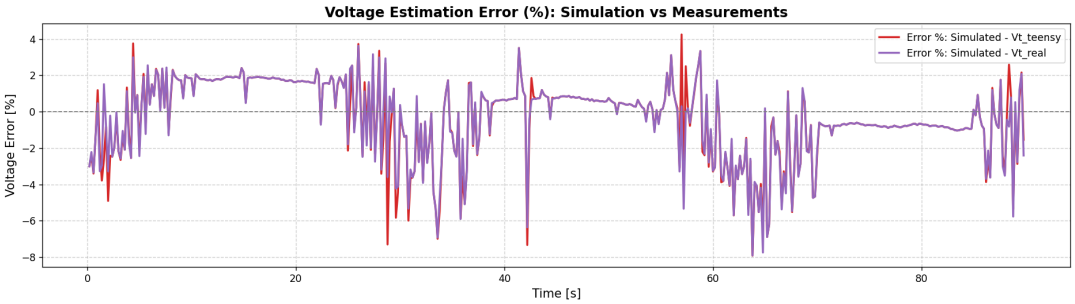


Figure 5.14: Voltage estimation error in the non-linear SoC region.

As illustrated in Figure 5.14, the voltage estimation error remains controlled, despite a slight increase in magnitude during rapid current transitions. The EKF maintained overall stability and tracking capability, demonstrating its robustness even in non-linear and low-observability scenarios.

The performance of the EKF estimator is highly dependent on the selection of its internal covariance matrices, particularly the process noise covariance Q and the initial error covariance P. These matrices govern how the estimator balances model dynamics against measurement information, directly influencing convergence speed and estimation accuracy. However, as discussed previously, they are challenging to tune due to their indirect relationship with measurable quantities and the variability of real-world operating conditions. To improve estimation performance and reduce voltage and SoC errors, several configurations of the Q and P matrices were tested iteratively. This tuning process involved empirical adjustment and simulation-based validation, allowing for gradual refinement of estimator behavior across different operating scenarios. The final configuration used in the dynamic load tests provided a stable trade-off between responsiveness and noise rejection, which proved essential in both linear and non-linear SoC regions.

5.2.5.2 Voltage Tracking Performance: Linear SoC Region

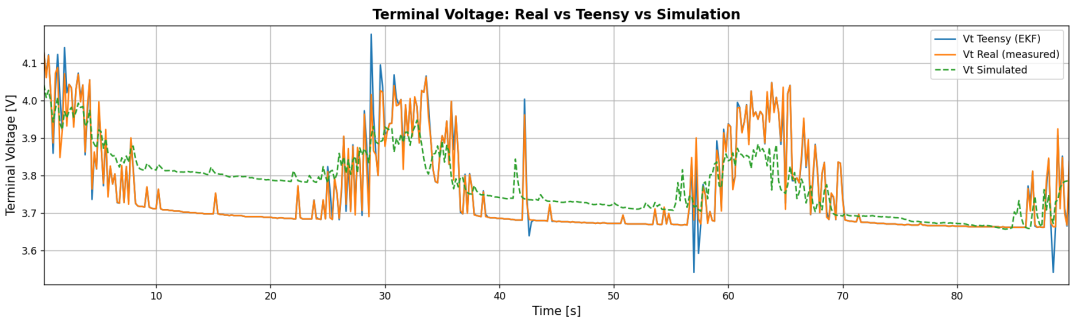


Figure 5.15: Terminal voltage comparison (measured, Teensy-EKF estimate, and simulation) in the linear SoC region.

Figure 5.15 presents a comparison of the measured terminal voltage, the estimation provided by the Teensy, which was implemented with the EKF, and the simulation output. The Teensy 4.1 estimation tracks the measured voltage closely throughout the test duration. Simulated results also align well, although some discrepancies arise during high-frequency current changes, as expected. This agreement demonstrates both the validity of the EKF implementation and the adequacy of the underlying battery model.

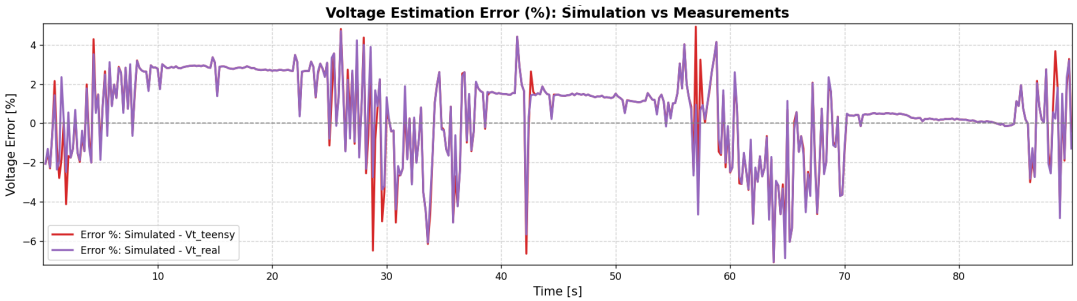


Figure 5.16: Voltage estimation error in the linear SoC region.

The corresponding voltage estimation error is shown in Figure 5.16. The percentage error remains within acceptable bounds, with brief peaks observed during sharp load transients. Overall, the EKF displays high accuracy and responsiveness, validating its effectiveness in regions where the system exhibits near-linear dynamics.

Chapter 6

Final prototype and integration into the FSFEUP 02

This chapter presents the culmination of the development process, in which the custom BMS reaches its final physical form and is integrated into the FSFEUP 02 electric race car. The chapter begins by describing the hardware components that constitute the complete BMS, followed by the integration strategy adopted to ensure mechanical, electrical, and communication compatibility with the vehicle. A comparative cost analysis is then presented, demonstrating the economic advantage of the custom solution over commercial alternatives. Finally, a critical evaluation of each previously defined objective is conducted, providing a structured assessment of the technical, economic, and strategic success of the project. This comprehensive overview not only confirms the practical viability of the developed system but also highlights the key innovations and trade-offs involved in its design.

6.1 Product overview

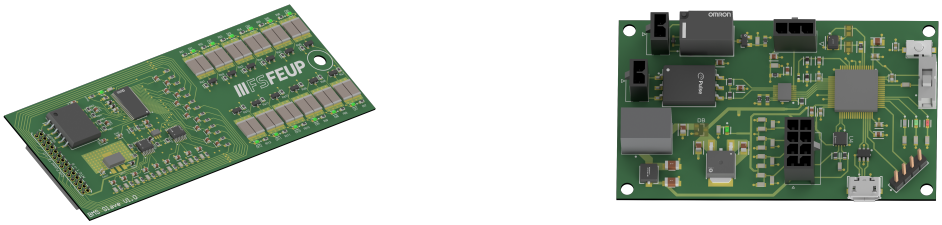
The final BMS architecture was designed to provide robust and scalable battery monitoring capabilities tailored to the FSFEUP 02 vehicle architecture. At its core, the system adopts a master-slave topology, with local sensing and processing performed on stack-mounted Slave units, and centralized supervision handled by a Master unit. This division allows for reduced internal cabling, improved EMI resilience, and modularity across hardware and software layers. The system includes real-time SoC estimation based on an embedded EKF, passive balancing circuitry, safety and fault-handling logic, and multi-protocol communication interfaces. The following sections present the physical implementation of each of these subsystems and their integration into the full battery assembly.

The completed system comprises three primary hardware elements: the BMS Slave, the BMS Master, and a set of interconnection PCBs positioned directly atop each battery stack segment. A total of fourteen BMS Slave boards were manufactured, twelve of which are used in the final battery pack configuration, each monitoring 12 individual cells, covering the full 144-cell series

architecture of the FSFEUP battery. Two additional Slave boards were produced as functional spares, enabling testing and validation procedures outside of the integrated system.

To ensure a modular and stackable architecture compatible with the physical constraints of the vehicle's battery design, seven intermediary PCBs were developed. These interface boards serve both mechanical and electrical functions, facilitating the connection between the Slave boards and the battery terminals. Six of these boards are deployed across the battery stacks, while one remains as a redundant spare. At the top of the hierarchy, a single BMS Master board aggregates the data received from all Slave modules, processes safety logic, and communicates with the vehicle's main control unit via the CAN protocol.

The complete set of PCBs was designed in-house and fabricated externally, then assembled and tested under operational conditions. Figure 6.1 presents a 3D render of the developed PCBs prior to physical fabrication. The outcome of this design process is shown in Figure 6.2, which illustrates the final versions of each individual board. In turn, Figure 6.3 demonstrates the physical integration of the BMS onto a real battery stack, validating the electromechanical compatibility and system completeness achieved in this work.



(a) BMS Slave PCB (b) BMS Master PCB

Figure 6.1: 3D render of the custom-designed PCBs

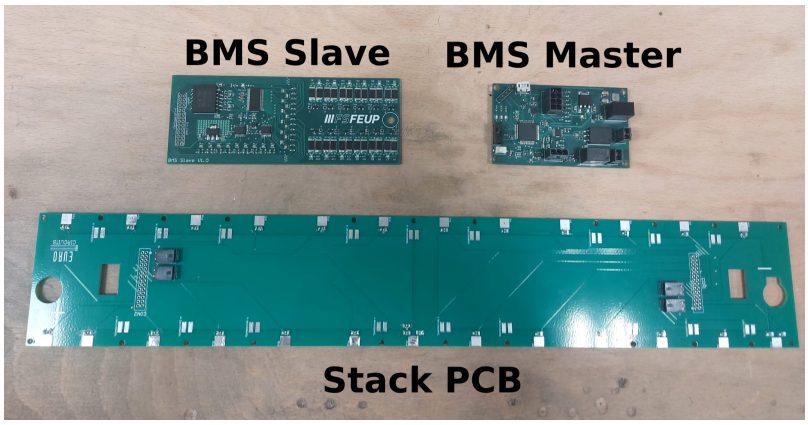


Figure 6.2: Final custom hardware developed for the BMS: Master, Slave, and stack interface PCBs.

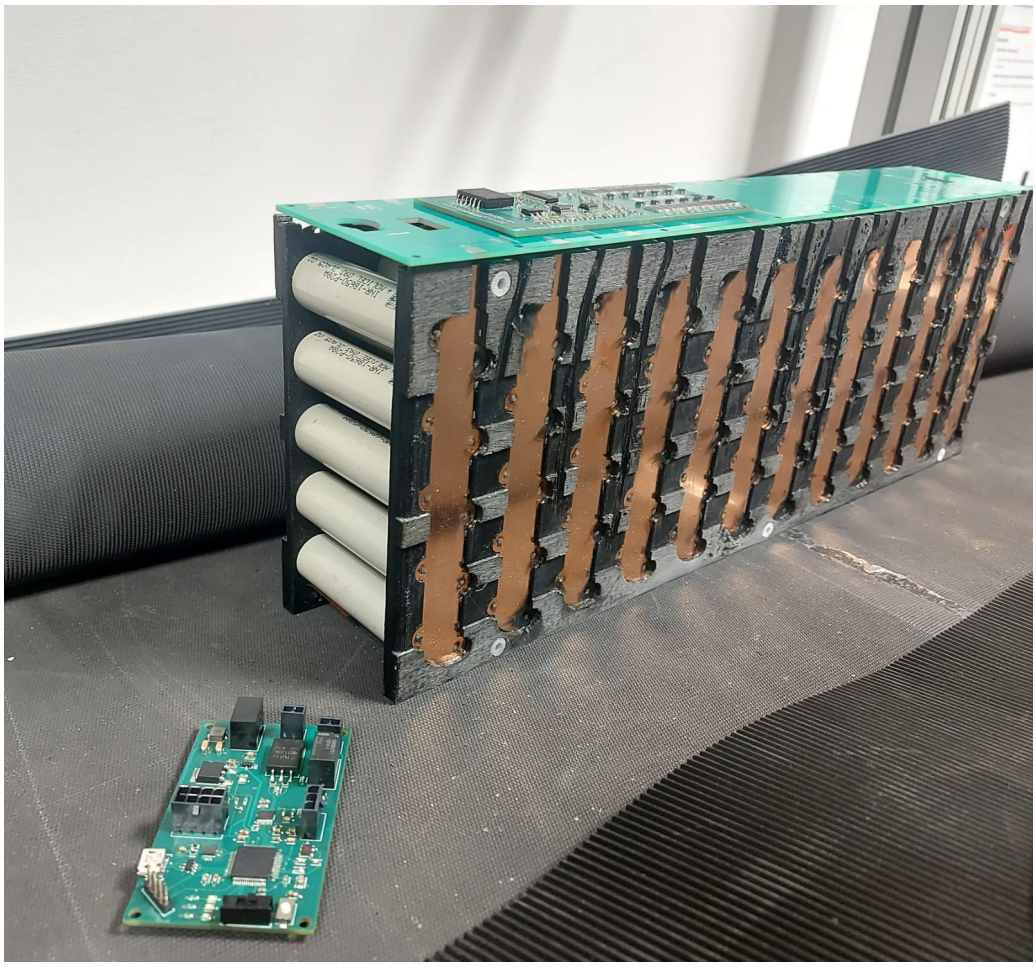


Figure 6.3: Integration of the BMS hardware into the FSFEUP battery stack prototype.

From an economic perspective, a key objective of the project was to deliver a solution that reduced system cost without compromising reliability or performance. This objective, defined in Chapter 5 as OBJ-11, was successfully achieved. The total production cost of the full BMS, including all components and PCBs, amounted to €992.26. This figure encompasses the entire hardware suite: fourteen Slave boards, seven stack interface PCBs, and one Master board, all fabricated and assembled using standard industrial processes.

For comparative purposes, the commercial Orion 2 BMS, configured for 144 cells and including all necessary accessories, such as the wiring harness, CAN adapter, configuration tools, and shipping costs, would amount to a total of €1825.70. The cost reduction is significant, especially when scaled to multiple prototypes or in a competition environment where budget constraints are critical. More importantly, the custom solution was specifically designed to fit the FSFEUP battery pack's topology and constraints, offering a level of flexibility and customization that off-the-shelf systems often cannot provide.

The successful reduction in system cost, without sacrificing safety or estimation performance, demonstrates the viability and competitiveness of in-house development for advanced vehicular subsystems. It confirms that the strategic investment in modeling, embedded algorithm design, and

custom PCB development can lead to high-impact, application-specific solutions that outperform generic alternatives in both cost and integration.

In the next section of this chapter, the degree to which each design objective has been met is systematically evaluated. This evaluation is based on the results obtained throughout the dissertation, from modeling and estimation accuracy to real-time execution and hardware integration, and cross-referenced with the objectives and constraints previously established in Table 3.1. Each goal is assessed not only in terms of binary completion but also with respect to the rationale and empirical evidence that support its fulfillment. For further details on the design and layout of the hardware components, including the BMS Slave and Master boards, as well as the interconnection PCBs, please refer to the PCB schematics provided in the Appendix B. These schematics offer detailed views of the circuit designs for each PCB.

6.2 Critical evaluation of objective fulfillment

This section presents a structured and critical evaluation of the design objectives defined at the outset of the BMS development process, as listed in Table 3.1. The degree to which each objective was fulfilled is assessed based on the results obtained across all stages of the project, including modeling, estimator implementation, hardware integration, and validation. This analysis serves not only to measure the success of the work, but also to highlight its strategic focus, technical boundaries, and priorities adopted throughout the development cycle.

6.2.1 Fulfilled objectives

A significant number of objectives, particularly those classified as high priority, were successfully achieved. Among the most fundamental was OBJ-01, which sought to reduce internal analog cabling within the accumulator enclosure. This was effectively addressed through a master-slave architecture that eliminates the need for centralized sensing and long wiring harnesses. By embedding measurement and control functionalities directly onto the stack, the system benefits from reduced electrical noise susceptibility and improved maintainability.

Closely related to this was OBJ-02, which proposed the adoption of a non-centralized. This requirement was fully realized via a scalable master–slave architecture, with identical Slave units mounted on each 24-cell stack, and a Master coordinating communication and control. This design not only enhances signal integrity but also simplifies expansion, testing, and replacement procedures.

Regulatory compliance, as defined by OBJ-03, was another high-priority goal. All relevant safety mechanisms required by FSG regulations, including individual cell monitoring, thermal protection, fault-triggered shutdown, and compatibility with IMD integrated and validated. The BMS Master was designed to interface directly with the vehicle's safety circuit, ensuring timely response to critical fault conditions.

From a control and estimation standpoint, OBJ-04 was a cornerstone of this work. It required the development of a superior State of SOC estimator, exceeding the accuracy of commercial systems such as the Orion BMS 2. This objective was met through the implementation of the EKF, derived from a parameterized Thevenin model and validated across multiple dynamic test profiles.

The successful deployment of the EKF in an embedded environment also confirms the fulfillment of OBJ-06. The algorithm was implemented on a Teensy 4.1 microcontroller and tested under real-time constraints, validating both convergence behavior and computational feasibility on hardware representative of automotive-grade embedded systems.

Additional objectives focused on system safety and monitoring were equally fulfilled. OBJ-05, concerning the integration of fault detection and AIR control, was fully realized through the BMS Master firmware. OBJ-07, which mandated continuous pack-level diagnostics, was achieved through the implementation of real-time voltage, current, and temperature monitoring on each stack. OBJ-08, related to precise current measurement for both protection and estimation, was also addressed through high-resolution ADC acquisition and integration within the EKF framework.

The theoretical foundation for estimation, namely, the development of a cell model as per OBJ-09, was completed through empirical testing and nonlinear curve fitting. The resulting Thevenin model captured dynamic polarization behavior and supported the derivation of discrete-time state equations with state-dependent parameters.

In terms of hardware uniformity, OBJ-10 was achieved by designing identical PCBs for each stack, facilitating manufacturing, assembly, and interchangeability. Furthermore, the economic objective defined in OBJ-11 was successfully reached. The total cost of the custom BMS prototype represented a 45.6% reduction when compared to the Orion BMS 2 solution configured for equivalent capabilities. This result validates the economic viability of in-house BMS development.

System integration was equally successful. OBJ-12, which demanded mechanical and electrical compatibility with the FSFEUP prototype, was fulfilled as demonstrated by the final installation of the BMS into the battery stack. OBJ-13, which aimed to match the core functionality of commercial systems, was likewise achieved. The system offers cell-level monitoring, configurable protection logic, and real-time SoC estimation, aligning with or surpassing the features typically found in industry-standard platforms.

Finally, while defined as medium and low priority, objectives OBJ-15 and OBJ-16 were also successfully accomplished. The communication architecture of the BMS was designed to support multiple protocols, enabling flexible integration with a variety of subsystems and external tools. In addition to the primary CAN 2.0B interface used for communication with the car's ECU, the system supports UART and SPI communication links, as well as micro-USB for development and diagnostic purposes. Furthermore, the stack-level BMS Slaves interface via isoSPI, ensuring robust and isolated communication across the full battery pack. This multi-protocol capability enhances compatibility with different hardware configurations and facilitates debugging, programming, and future upgrades.

From a design perspective, the modularity of both hardware and firmware supports the fulfillment of OBJ-16, which called for forward adaptability to future prototypes, specifically the FS-

FEUP 03 vehicle. The BMS components were developed with well-defined electrical, mechanical, and communication interfaces, allowing them to be reused or expanded with minimal reengineering. The firmware structure is equally modular, with configurable layers for protocol management, estimator parameters, and fault logic, enabling streamlined adaptation to future requirements or changes in the vehicle's architecture.

6.2.2 Partially fulfilled objectives

One objective was partially fulfilled due to scope and resource allocation decisions. OBJ-14 targeted the implementation of a cell balancing mechanism, a vital feature for long-term pack health. A passive balancing strategy was fully designed and integrated at the hardware level, including the necessary circuitry on each BMS Slave. However, due to the project's emphasis on estimator validation and system safety, this functionality was not subjected to thorough testing under operational conditions. While the feature exists and is theoretically functional, its behavior under repeated cycling remains to be validated in future iterations.

6.2.3 Unfulfilled objectives

Two objectives, OBJ-17 and OBJ-18, were not pursued in this work, and their exclusion is grounded in deliberate prioritization choices. OBJ-17, which proposed the development of a GUI for monitoring system state, was ultimately abandoned to focus effort on critical real-time estimation and protection functionalities. While a GUI would enhance usability and visual accessibility, it was deemed non-essential for the technical validation of the BMS core functions.

Similarly, OBJ-18 which suggested including a SoH estimation module, was not considered appropriate within the scope of this dissertation. This decision is justified by the specific operating conditions of the FSFEUP battery pack. Unlike long-life commercial systems, these batteries are designed for short-duration, high-performance use, with operational lifespans limited to approximately 500 km and low cycle counts. The inclusion of SoH estimation, relevant in contexts involving hundreds or thousands of charge–discharge cycles, was considered a non-priority feature offering limited practical benefit in this setting.

Table 6.1 provides a condensed overview of the fulfillment status of each objective. As shown, all high-priority goals were successfully achieved, and only a small number of lower-priority features were either partially completed or excluded from scope, this reflects a focused and technically sound development strategy, aligned with the specific requirements of the FSFEUP prototype.

Table 6.1: Summary of BMS objective fulfillment

Code	Objective / Constraint	Status
OBJ-01	Minimize internal cabling	Fulfilled
OBJ-02	Avoid centralized architecture	Fulfilled
OBJ-03	Regulatory compliance	Fulfilled
OBJ-04	Enhanced SoC estimator	Fulfilled
OBJ-05	Safety protocol integration	Fulfilled
OBJ-06	Validate SoC estimator in testboard	Fulfilled
OBJ-07	Pack-level safety diagnostics	Fulfilled
OBJ-08	Current measurement capability	Fulfilled
OBJ-09	Cell model development	Fulfilled
OBJ-10	Uniform stack-mounted PCBs	Fulfilled
OBJ-11	Lower cost than Orion BMS 2	Fulfilled
OBJ-12	Seamless prototype integration (02)	Fulfilled
OBJ-13	Medium performance	Fulfilled
OBJ-14	Cell balancing system	Partially fulfilled
OBJ-15	Multi-protocol communication	Fulfilled
OBJ-16	Future adaptability (03)	Fulfilled
OBJ-17	GUI monitoring tool	Not fulfilled
OBJ-18	SoH estimation support	Not fulfilled

Chapter 7

Conclusions and Future Work

7.1 Conclusions

The work developed throughout this dissertation led to the successful design, implementation, and validation of a custom BMS tailored to the FSFEUP 02 electric race car. The project addressed the need for a modular, accurate, and cost-effective alternative to commercial BMS solutions, aligning with both the technical constraints of FS competition and the specific integration requirements of the team's accumulator design.

The proposed system followed a master-slave architecture based on stack-mounted Slave units and a centralized Master controller, enabling reduced wiring complexity, improved signal integrity, and enhanced scalability. At its core, an EKF was developed for SoC estimation, based on a parameterized Thevenin model and experimentally validated under multiple static and dynamic load conditions. The estimator was implemented in real-time on embedded hardware, confirming both computational feasibility and convergence accuracy.

The BMS was physically integrated into the FSFEUP 02 prototype, with validation of communication, sensing, and control functionalities under real operation scenarios. A comparative cost analysis demonstrated that the system achieved a 45.6% reduction in overall cost compared to the Orion BMS 2 commercial solution, proving the viability of in-house BMS development.

Out of the 18 objectives initially defined, 15 were fully achieved, one partially completed, and 2 excluded by design. This level of fulfillment not only confirms the technical soundness of the work but also its strategic alignment with the vehicle's requirements. Overall, the system developed in this dissertation lays the foundation for a robust and extensible BMS platform, one that is not only functional and integrated but also economically and academically valuable.

Beyond its immediate application in the FSFEUP 02 vehicle, the system demonstrates the feasibility of designing custom BMS architectures that are both technically robust and financially competitive. The methodology followed, rooted in experimental modeling, embedded estimation, and modular hardware design, can be extended to other electric mobility platforms, making this work a relevant contribution to the broader field of energy systems engineering.

7.2 Future Work

Although the current prototype represents a significant advancement in the team's battery management capabilities, it was intentionally designed as a foundational platform to support future research, iterative refinement, and continuous development in both hardware and software.

An expected continuation of this work is the exploration and comparison of alternative state estimation algorithms. For example, a future project could implement an UKF and compare its performance to the EKF developed in this dissertation. Integrating SoH estimation would also enable long-term monitoring of cell degradation and support predictive maintenance, especially valuable in endurance or high-cycle applications.

Furthermore, the passive balancing mechanism integrated into the BMS hardware was not experimentally validated due to project priorities. Future work could characterize its behavior, effectiveness, and energy efficiency under realistic conditions. Since the system supports more advanced strategies, implementing and validating active balancing architectures is another promising direction for improving energy distribution and cell longevity.

Another important area for investigation concerns the influence of initialization matrices, specifically the state error covariance P_0 and process noise covariance Q, on EKF performance after system startup. These matrices strongly affect convergence and estimator confidence. Studying their impact after rest periods or power cycles could provide insights into robustness under practical conditions, where frequent reinitialization is common.

Temperature is another key factor influencing estimation accuracy. Since battery behavior is highly sensitive to temperature, affecting parameters such as internal resistance, capacity, and OCV. Inadequate thermal modeling or limited sensing can lead to significant estimation errors. Future developments should incorporate thermal compensation techniques and enhanced temperature sensing to improve estimator performance during both initialization and dynamic operation.

Given the modular and reconfigurable nature of the developed BMS platform, another avenue for future work is adapting the system to alternative cell chemistries. This would require revising voltage ranges, thermal response characteristics, and electrochemical modeling, along with recalibration of estimation and protection algorithms.

Ultimately, the system developed in this dissertation was never intended as a fixed solution. A BMS must evolve continuously to meet changing performance goals, safety standards, and design constraints. This work should be viewed not as a conclusion, but as a solid foundation for future academic and engineering progress. Its modular architecture, open-source nature, and comprehensive documentation make it a strong platform for subsequent development cycles, where both hardware and software can be expanded and optimized to meet emerging technological challenges. In this sense, the system is not a final product, but a catalyst for ongoing innovation.

References

[1] "FSG: Rules & Documents." Accessed: Nov. 2, 2024. (2024), [Online]. Available: https://www.formulastudent.de/fsg/rules/.

[2] F. S. Germany, *Fsg25 electrical inspection sheet – electric vehicles*, https://www.formulastudent.de/rules/, Accessed: May 2025, 2025.

[3] Orion BMS, *Orion bms 2 datasheet*, https://www.orionbms.com/products/orion-bms-standard/, Accessed May 2025, 2023.

[4] FS FEUP, *FS FEUP 02*, https://formulastudent.fe.up.pt/fsfeup02/, Accessed: Jun. 22, 2025, 2025.

[5] Vasco da Gama CoLAB, *Vasco da Gama CoLAB - Energy Storage*, https://www.vgcolab.com/pt, Accessed: Jun. 22, 2025, 2025.

[6] Y. Ding, Z. P. Cano, A. Yu, J. Lu, and Z. Chen, "Automotive li-ion batteries: Current status and future perspectives," *Electrochemical Energy Reviews*, vol. 2, no. 1, pp. 1–28, Mar. 2019. DOI: 10.1007/s41918-018-0022-z.

[7] F. M. N. U. Khan, M. G. Rasul, A. S. M. Sayem, and N. K. Mandal, "Design and optimization of lithium-ion battery as an efficient energy storage device for electric vehicles: A comprehensive review," *Journal of Energy Storage*, vol. 71, p. 108 033, Nov. 2023. DOI: 10.1016/j.est.2023.108033.

[8] M. K. Hasan, M. Mahmud, A. K. M. A. Habib, S. M. A. Motakabber, and S. Islam, "Review of electric vehicle energy storage and management system: Standards, issues, and challenges," *Journal of Energy Storage*, vol. 41, p. 102 940, Sep. 2021. DOI: 10.1016/j.est.2021.102940.

[9] A. K. M. A. Habib, M. K. Hasan, G. F. Issa, D. Singh, S. Islam, and T. M. Ghazal, "Lithium-ion battery management system for electric vehicles: Constraints, challenges, and recommendations," *Batteries*, vol. 9, no. 3, p. 152, Mar. 2023. DOI: 10.3390/batteries9030152.

[10] J. Barreras, "Practical methods in li-ion batteries for simplified modeling, battery electric vehicle design, battery management system testing and balancing system control," Ph.D. dissertation, May 2017.

[11] C. Fleischer, D. U. Sauer, J. V. Barreras, E. Schaltz, and A. E. Christensen, "Development of software and strategies for battery management system testing on hil simulator," in *2016 Eleventh International Conference on Ecological Vehicles and Renewable Energies (EVER)*, IEEE, Apr. 2016, pp. 1–12. DOI: 10.1109/EVER.2016.7476438.

[12] K. W. See, G. Wang, Y. Zhang, *et al.*, "Critical review and functional safety of a battery management system for large-scale lithium-ion battery pack technologies," *International Journal of Coal Science Technology*, vol. 9, no. 1, p. 36, May 2022. DOI: 10.1007/s40789-022-00494-0.

[13] F. O. A. P. Teixeira, *Development of a battery management system for electric vehicles applied to formula student*, https://repositorio-aberto.up.pt/handle/10216/162371, Accessed: May 12, 2025, Oct. 2024.

[14] D. Andrea, *Battery Management Systems for Large Lithium-ion Battery Packs*. Artech House, 2010, in EBL-Schweitzer. [Online]. Available: https://books.google.pt/books?id=o-QpFOR0PTcC.

[15] Analog Devices, *Ltc6811-1/ltc6811-2 - multicell battery monitor*, https://www.analog.com/media/en/technical-documentation/data-sheets/ltc6811-1-6811-2.pdf, Accessed: May 2025, 2021.

[16] C. Xiao, L. Zhao, T. Asada, W. G. Odendaal, and J. D. van Wyk, "An overview of integratable current sensor technologies," in *38th IAS Annual Meeting on Conference Record of the Industry Applications Conference, 2003.*, Oct. 2003, 1251–1258 vol.2. DOI: 10.1109/IAS.2003.1257710.

[17] I. H. G.
bibinitperiod C. KG, *Precision and power resistors – product overview*, https://www.isabellenhuette.de/en/precision-and-power-resistors, Accessed: May 2025, 2024.

[18] Isabellenhütte USA, *IVT-S Series*, https://www.isabellenhuetteusa.com/product/ivt-s-series/, Accessed: May 12, 2025, 2025.

[19] B. Pattipati, C. Sankavaram, and K. R. Pattipati, "System identification and estimation framework for pivotal automotive battery management system characteristics," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 41, no. 6, pp. 869–884, Nov. 2011. DOI: 10.1109/TSMCC.2010.2089979.

[20] "Soc estimation for lithium-ion batteries: Review and future challenges," *Electronics*, vol. 6, no. 4, p. 102, 2017, Accessed: Jan. 13, 2025. [Online]. Available: https://www.mdpi.com/2079-9292/6/4/102.

[21] "State of the art of lithium-ion battery soc estimation for electrical vehicles," *Energies*, vol. 11, no. 7, p. 1820, 2018, Accessed: Jan. 13, 2025. [Online]. Available: https://www.mdpi.com/1996-1073/11/7/1820.

[22] M. Etxandi-Santolaya, L. C. Casals, and C. Corchero, "Extending the electric vehicle battery first life: Performance beyond the current end of life threshold," *Heliyon*, vol. 10, no. 4, e26066, Feb. 2024. DOI: 10.1016/j.heliyon.2024.e26066.

[23] M. Daowd, N. Omar, P. V. D. Bossche, and J. V. Mierlo, "Passive and active battery balancing comparison based on matlab simulation," in *2011 IEEE Vehicle Power and Propulsion Conference*, 2011, pp. 1–7. DOI: 10.1109/VPPC.2011.6043010.

[24] A. Nath and B. Rajpathak, "Analysis of cell balancing techniques in bms for electric vehicle," in *2022 International Conference on Intelligent Controller and Computing for Smart Power (ICICCSP)*, 2022, pp. 1–6. DOI: 10.1109/ICICCSP53532.2022.9862513.

[25] , "High-voltage battery management systems (bms) for electric vehicles," in *Advances in Battery Technologies for Electric Vehicles*, Elsevier, 2015, pp. 265–282. DOI: 10.1016/b978-1-78242-377-5.00011-x.

[26] A. Hauser and R. Kuhn, "High-voltage battery management systems (bms) for electric vehicles," in *Advances in Battery Technologies for Electric Vehicles*, Elsevier, 2015, pp. 265–282. DOI: 10.1016/B978-1-78242-377-5.00011-X.

[27] Eurocircuits. "Formula electric belgium – battery management & safety." Accessed: May 22, 2025. (2025), [Online]. Available: https://www.eurocircuits.com/student-projects/formula-electric-belgium-own-electronic-control-unit-with-eurocircuits-pcbs-3-2/.

[28] S. Systems, *Battery management systems*, https://www.staflsystems.com/battery-management-systems, Accessed May 2025, 2023.

[29] EMUS, *Emus g1 bms overview*, https://emusbms.com/, Accessed May 2025, 2024.

[30] Analog Devices, *Eval-adbms1818 evaluation board*, https://www.analog.com/en/resources/evaluation-hardware-and-software/evaluation-boards-kits/eval-adbms1818.html, Accessed June 2025, 2024.

[31] H. S. Ramadan, M. Becherif, and F. Claude, "Extended kalman filter for accurate state of charge estimation of lithium-based batteries: A comparative analysis," *International Journal of Hydrogen Energy*, vol. 42, no. 48, pp. 29 033–29 046, Nov. 2017. DOI: 10.1016/j.ijhydene.2017.07.219.

[32] Z. Huang, P. Du, D. Kosterev, and B. Yang, "Application of extended kalman filter techniques for dynamic model parameter calibration," in *2009 IEEE Power & Energy Society General Meeting*, IEEE, 2009, pp. 1–8. DOI: 10.1109/PES.2009.5275423.

[33] P. Iurilli, C. Brivio, and V. Wood, "On the use of electrochemical impedance spectroscopy to characterize and model the aging phenomena of lithium-ion batteries: A critical review," *Journal of Power Sources*, vol. 505, p. 229 860, Sep. 2021. DOI: 10.1016/j.jpowsour.2021.229860.

[34] P. Shrivastava, T. K. Soon, M. Y. I. B. Idris, and S. Mekhilef, "Overview of model-based online state-of-charge estimation using kalman filter family for lithium-ion batteries," *Renewable and Sustainable Energy Reviews*, vol. 113, p. 109 233, Oct. 2019. DOI: 10.1016/j.rser.2019.06.040.

[35] W. Zhou, Y. Zheng, Z. Pan, and Q. Lu, "Review on the battery model and soc estimation method," *Processes*, vol. 9, no. 9, Sep. 2021. DOI: 10.3390/pr9091685.

[36] "STM32F405RG - High-performance foundation line, Arm Cortex-M4 core with DSP and FPU, 1 Mbyte of Flash memory, 168 MHz CPU, ART Accelerator - STMicroelectronics." Accessed: Jun. 04, 2025. (2025), [Online]. Available: https://www.st.com/en/microcontrollers-microprocessors/stm32f405rg.html.

[37] S. N. M. Azam, "Linear Discrete-time State Space Realization of a Modified Quadruple Tank System with State Estimation using Kalman Filter," en, *Journal of Physics: Conference Series*, vol. 783, no. 1, p. 012 013, Jan. 2017, Publisher: IOP Publishing, ISSN: 1742-6596. DOI: 10.1088/1742-6596/783/1/012013. [Online]. Available: https://dx.doi.org/10.1088/1742-6596/783/1/012013 (visited on 06/10/2025).

[38] T. M. N. Bui, M. F. Niri, D. Worwood, T. Q. Dinh, and J. Marco, "An advanced hardware-in-the-loop battery simulation platform for the experimental testing of battery management system," in *2019 23rd International Conference on Mechatronics Technology (ICMT)*, Oct. 2019, pp. 1–6. DOI: 10.1109/ICMECT.2019.8932115.

[39] *Polyfit - polynomial curve fitting - matlab*, https://www.mathworks.com/help/matlab/ref/polyfit.html, Accessed: Jun. 10, 2025, MathWorks, 2025.

[40] B. Haus and P. Mercorelli, "Polynomial augmented extended kalman filter to estimate the state of charge of lithium-ion batteries," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 2, pp. 1452–1463, 2020. DOI: 10.1109/TVT.2019.2959720.

[41] R. Schneider and C. Georgakis, "How to not make the extended kalman filter fail," *Industrial & Engineering Chemistry Research*, vol. 52, no. 9, pp. 3354–3362, 2013. DOI: 10.1021/ie300415d.

[42] MathWorks, *How to tune kalman filters*, https://www.mathworks.com/help/control/ug/kalman-filter.html, 2022.

Appendix A

Modeling and Embedded Implementation Code

This appendix presents the complete set of scripts and source code developed to support battery modeling, estimator design, and real-time implementation. Three main tools were used throughout the project. First, a MATLAB script was employed to import experimental data from the HPPC protocol, fit the cell model parameters as functions of SoC, and generate the corresponding visualization plots. Second, a C implementation was developed to run the EKF estimator in real time on a Teensy 4.1 microcontroller, interfaced with an LTC6811 battery monitoring IC. Finally, a Python script was created to compare the EKF outputs measured in hardware with simulated ground-truth values under ideal test conditions, enabling cross-validation of the embedded implementation.

A.1 MATLAB Script for Cell Model Parameterization

The first script was written in MATLAB and used for the processing and analysis of data collected during the HPPC test. It reads voltage and current profiles, segments the pulses, and extracts the electrical parameters across different SoC levels. Polynomial fitting is then applied to each parameter to obtain continuous functions of SoC, which are used in both simulation and embedded estimation. This script also generates the figures shown in Chapter 4.

```
% MATLAB Script for Cell Parameter Extraction and Fitting

% Import CSV Data
opts = delimitedTextImportOptions("NumVariables", 25, "Encoding", "UTF-8");
opts.DataLines = [2, Inf];
opts.Delimiter = ";";
opts.VariableNames = {"DataPoint", "DateTime", "TestTimes", "StepTimes", "
    CycleIndex", "StepIndex", "TC_Counter1", "TC_Counter2", "CurrentA", "VoltageV",
     "PowerW", "ChargeCapacityAh", "DischargeCapacityAh", "ChargeEnergyWh", "
    DischargeEnergyWh", "CapacityAh", "mAhg", "ACROhm", "dVdtVs", "
    InternalResistanceOhm", "dQdVAhV", "dVdQVAh", "MetaCode_MV_UD1", "
    MetaCode_MV_UD2", "VarName25"};
```

```
opts.VariableTypes = repmat({"double"}, 1, 25);
opts.ExtraColumnsRule = "ignore";
opts.EmptyLineRule = "read";
opts = setvaropts(opts, {"DateTime", "dVdtVs"}, "WhitespaceRule", "preserve");
opts = setvaropts(opts, {"DateTime", "dVdtVs"}, "EmptyFieldRule", "auto");
opts = setvaropts(opts, "TestTimes", "InputFormat", "MM/dd/yyyy HH:mm:ss.SSS", "
    DatetimeFormat", "preserveinput");

DataMolicel = readtable("C:/Users/Pedro/Desktop/Data_Molicel.csv", opts);

% Plot Voltage vs Step Times
figure;
plot(DataMolicel.StepTimes, DataMolicel.PowerW, 'b-');
xlabel('Step Times (s)'); ylabel('Voltage (V)');
title('Voltage vs Step Times'); grid on;

% Raw parameter data
R0 = [0.037; 0.037857143; 0.037142857; 0.0375; 0.036785714; 0.036785714;
    0.037142857; 0.0375; 0.0375; 0.037857143; 0.0375; 0.0375; 0.038928571;
    0.041071429; 0.054285714];
R1 = [0.020; 0.022142857; 0.026428571; 0.035; 0.052857143; 0.078571429; 0.0375;
    0.028214286; 0.033928571; 0.040357143; 0.036428571; 0.03; 0.031785714;
    0.100714286; 0.242857143];
C1 = [1600; 2366.451613; 1672.432432; 1914.285714; 2368.648649; 3954.363636;
    6109.333333; 3788.860759; 4085.052632; 5272.920354; 5407.843137; 4276.666667;
    4146.516854; 2999.574468; 1076.352941];
SoC = [1; 0.9333388889; 0.8666833333; 0.8000222222; 0.7333611111; 0.6667;
    0.6000388889; 0.5333777778; 0.4667166667; 0.4000555556; 0.3333944444;
    0.2667333333; 0.2000722222; 0.1334111111; 0.06675];
OCV = [4.2; 4.0947; 4.0687; 4.0086; 3.9376; 3.8842; 3.8083; 3.7423; 3.6821; 3.6326;
     3.5785; 3.5046; 3.4399; 3.3188; 3.0726];
T1 = [20, 52.4, 44.2, 67.0, 125.2, 310.7, 229.1, 106.9, 138.6, 212.8, 197.0, 128.3,
     131.8, 302.1, 261.4];

% Polynomial fitting (5th degree)
p_OCV = polyfit(SoC, OCV, 5);
p_R0 = polyfit(SoC, R0, 5);
p_R1 = polyfit(SoC, R1, 5);
p_C1 = polyfit(SoC, C1, 5);
dp_OCV = polyder(p_OCV);

% Display coefficients
fprintf('OCV Coefficients:\n'); disp(p_OCV);
fprintf('R0 Coefficients:\n'); disp(p_R0);
fprintf('R1 Coefficients:\n'); disp(p_R1);
fprintf('C1 Coefficients:\n'); disp(p_C1);
fprintf('dOCV/dSoC Coefficients:\n'); disp(dp_OCV.');

% Plot fits
```

```
SoC_fit = linspace(min(SoC), max(SoC), 100);
figure;
subplot(2,2,1); plot(SoC, R0, 'bo', SoC_fit, polyval(p_R0, SoC_fit), 'b-'); title('R0(SoC)'); grid on;
subplot(2,2,2); plot(SoC, R1, 'ro', SoC_fit, polyval(p_R1, SoC_fit), 'r-'); title('R1(SoC)'); grid on;
subplot(2,2,3); plot(SoC, C1, 'go', SoC_fit, polyval(p_C1, SoC_fit), 'g-'); title('C1(SoC)'); grid on;
subplot(2,2,4); plot(SoC, OCV, 'mo', SoC_fit, polyval(p_OCV, SoC_fit), 'm-'); title('OCV(SoC)'); grid on;
;

% Final high-quality plots for inclusion
figure('Units', 'normalized', 'Position', [0.2 0.2 0.6 0.6]);
subplot(2,2,1); plot(SoC, R0, 'bo'); hold on;
plot(SoC_fit, polyval(p_R0, SoC_fit), 'b-');
title('Polynomial Fit: $R_0$(SoC)', 'Interpreter', 'latex');
xlabel('State of Charge (SoC)', 'Interpreter', 'latex');
ylabel('$R_0$ [$\Omega$]', 'Interpreter', 'latex');
grid on;

subplot(2,2,2); plot(SoC, R1, 'ro'); hold on;
plot(SoC_fit, polyval(p_R1, SoC_fit), 'r-');
title('Polynomial Fit: $R_1$(SoC)', 'Interpreter', 'latex');
xlabel('State of Charge (SoC)', 'Interpreter', 'latex');
ylabel('$R_1$ [$\Omega$]', 'Interpreter', 'latex');
grid on;

subplot(2,2,3); plot(SoC, C1, 'go'); hold on;
plot(SoC_fit, polyval(p_C1, SoC_fit), 'g-');
title('Polynomial Fit: $C_1$(SoC)', 'Interpreter', 'latex');
xlabel('State of Charge (SoC)', 'Interpreter', 'latex');
ylabel('$C_1$ [F]', 'Interpreter', 'latex');
grid on;

subplot(2,2,4); plot(SoC, OCV, 'mo'); hold on;
plot(SoC_fit, polyval(p_OCV, SoC_fit), 'm-');
title('Polynomial Fit: OCV(SoC)', 'Interpreter', 'latex');
xlabel('State of Charge (SoC)', 'Interpreter', 'latex');
ylabel('Open Circuit Voltage [V]', 'Interpreter', 'latex');
grid on;
```

Listing A.1: MATLAB script for cell model parameter extraction and fitting

A.2 C Code for Real-Time EKF Execution on Teensy 4.1

The second implementation is written in C and executed on a Teensy 4.1 microcontroller. This code integrates with an LTC6811 battery monitoring IC and performs real-time SOC estimation using

the EKF. The parameterized cell model (with SoC-dependent coefficients) is embedded directly in the code. The estimator predicts and corrects the internal state using real voltage and current measurements sampled from a lithium-ion battery pack. The output includes estimated SoC and terminal voltage, which are printed over serial for analysis. This implementation validates the feasibility of running the EKF under embedded constraints, as discussed in Chapter 5.

```
% EKF Implementation for SoC Estimation
% Target: Real-time execution on Teensy 4.1 with LTC6811 interface

#include <estimator.h>
#include <Arduino.h>
#include <math.h>

#define TS 0.2f                   // Sampling period in seconds
#define cellCapacity 2.8f         // Nominal capacity in Ah

static bool firstRun = true;      // Flag for EKF initialization

// Polynomial coefficients for battery model parameterization
const float R0_coeffs[6] = {-1.0348f, 3.0692f, -3.4282f, 1.7837f, -0.4285f, 0.0753f
    };
const float R1_coeffs[6] = {-5.6695f, 19.3998f, -25.4586f, 15.6483f, -4.3704f,
    0.4666f};
const float C1_coeffs[6] = {116096.61f, -297356.04f, 305302.76f, -174249.65f,
    53918.36f, -1785.91f};
const float OCV_coeffs[6] = {15.8003f, -47.1367f, 53.0475f, -27.8100f, 7.6176f,
    2.6764f};
const float dOCV_coeffs[5] = {79.0017f, -188.5470f, 159.1425f, -55.6201f, 7.6176f};

// Utility function to evaluate a polynomial at point x
float polyval(const float *coeffs, int degree, float x) {
  float result = coeffs[0];
  for (int i = 1; i <= degree; i++) {
    result = result * x + coeffs[i];
  }
  return result;
}

// EKF state vector and covariance matrices
float x_hat[2] = {0.8, 0.0};       // Initial SoC and V_RC state
float P[2][2] = {{1e-6, 0}, {0, 1e-6}};  // Initial error covariance
const float Q[2][2] = {{1e-6, 0}, {0, 1e-6}};  // Process noise covariance
const float R = 2.25e-4;           // Measurement noise variance

// Estimate initial SoC from terminal voltage using OCV inverse mapping
float estimateSOCFromOCV(float voltage) {
  float low = 0.0f, high = 1.0f, mid;
  float tol = 1e-4f;
  int max_iter = 50;
```

```
  for (int i = 0; i < max_iter; ++i) {
    mid = (low + high) / 2.0f;
    float estimated_voltage = polyval(OCV_coeffs, 5, mid);
    if (fabs(estimated_voltage - voltage) < tol) break;
    if (estimated_voltage > voltage) high = mid;
    else low = mid;
  }
  return mid;
}

// Main EKF routine to be called with cell voltage vector
void SOC_ESTIMATOR(float CELLS_VOLTAGE[12]) {
  if (firstRun) {
    x_hat[0] = estimateSOCFromOCV(CELLS_VOLTAGE[0]);
    firstRun = false;
  }

  unsigned long currentMillis = millis();
  float current = 2.8; // Replace with measured current in production
  float voltage = CELLS_VOLTAGE[0];

  float SOC_pred = constrain(x_hat[0], 0.0f, 1.0f);
  float R_0_pred = polyval(R0_coeffs, 5, SOC_pred);
  float R_1_pred = polyval(R1_coeffs, 5, SOC_pred);
  float C_1_pred = polyval(C1_coeffs, 5, SOC_pred);
  float tau_pred = fmaxf(R_1_pred * C_1_pred, 1e-6f);

  // Prediction step
  float A[2][2] = {{1, 0}, {0, exp(-TS / tau_pred)}};
  float B[2] = {-TS / (cellCapacity * 3600.0f), R_1_pred * (1 - exp(-TS / tau_pred)
      )};
  float x_hat_pred[2] = {
    A[0][0] * x_hat[0] + A[0][1] * x_hat[1] - B[0] * current,
    A[1][0] * x_hat[0] + A[1][1] * x_hat[1] - B[1] * current
  };

  float p_pred[2][2];
  p_pred[0][0] = A[0][0] * P[0][0] + Q[0][0];
  p_pred[0][1] = A[0][1] * P[1][1] + Q[0][1];
  p_pred[1][0] = A[1][0] * P[0][0] + Q[1][0];
  p_pred[1][1] = A[1][1] * P[1][1] + Q[1][1];

  // Correction step
  float R_0_corr = polyval(R0_coeffs, 5, x_hat_pred[0]);
  float R_1_corr = polyval(R1_coeffs, 5, x_hat_pred[0]);
  float C_1_corr = polyval(C1_coeffs, 5, x_hat_pred[0]);
  float tau_corr = fmaxf(R_1_corr * C_1_corr, 1e-6f);
  float OCV_corr = polyval(OCV_coeffs, 5, x_hat_pred[0]);
  float dOCV_dSOC = polyval(dOCV_coeffs, 4, x_hat_pred[0]);
```

```
  float H[2] = {dOCV_dSOC, -1};

  float V_R1 = exp(-TS / tau_corr) * x_hat_pred[1] + R_1_corr * (1 - exp(-TS /
      tau_corr)) * current;
  float estimatedOutput_corr = OCV_corr - V_R1 - (current * R_0_corr);
  float error = voltage - estimatedOutput_corr;

  float S = H[0] * (p_pred[0][0] * H[0] + p_pred[0][1] * H[1]) +
            H[1] * (p_pred[1][0] * H[0] + p_pred[1][1] * H[1]) + R;
  float K[2] = {
    (p_pred[0][0] * H[0] + p_pred[0][1] * H[1]) / S,
    (p_pred[1][0] * H[0] + p_pred[1][1] * H[1]) / S
  };

  x_hat[0] = x_hat_pred[0] + K[0] * error;
  x_hat[1] = x_hat_pred[1] + K[1] * error;

  // Covariance update
  for (int i = 0; i < 2; i++) {
    for (int j = 0; j < 2; j++) {
      P[i][j] = p_pred[i][j] - K[i] * (p_pred[0][j] * H[0] + p_pred[1][j] * H[1]);
    }
  }

  // Output: current SoC and estimated voltage
  float SOC = x_hat[0];
  float R_0 = polyval(R0_coeffs, 5, SOC);
  float R_1 = polyval(R1_coeffs, 5, SOC);
  float C_1 = polyval(C1_coeffs, 5, SOC);
  float tau = fmaxf(R_1 * C_1, 1e-6f);
  float OCV = polyval(OCV_coeffs, 5, SOC);

  V_R1 = exp(-TS / tau) * x_hat[1] + R_1 * (1 - exp(-TS / tau)) * current;
  float estimatedOutput = OCV - V_R1 - (current * R_0);

  Serial.print(currentMillis / 1000.0f); Serial.print(" | ");
  Serial.print(estimatedOutput, 4); Serial.print(" | ");
  Serial.print(voltage, 4); Serial.print(" | ");
  Serial.print(SOC, 4); Serial.println(" | ");
}
```

Listing A.2: Real-time EKF estimation code implemented on Teensy 4.1

A.3 Python Script for Simulation and Hardware Comparison

A third script was developed in Python to support validation of the embedded EKF. It simulates the behavior of the estimator using the same input current profiles applied during experimental testing and compares the outputs, such as SoC and estimated voltage, with the real-time data captured

from the Teensy 4.1. The script reads log files from the serial output, aligns them with simulation results, and computes the pointwise error. This cross-validation helps confirm the correctness of the embedded implementation and identify discrepancies due to noise, sampling delays, or hardware-specific behavior.

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd

# ------------------ PARAMETERS FOR CELL AND PACK ------------------
Ts = 0.2  # Sampling time [s]
Q_cell = 2.8 * 3600  # Cell capacity [C]
Q_total = Q_cell  # Total capacity for parallel cells

# ------------------ OCV, R0, R1, Tau1 CURVES ------------------
OCV_data = np.array([4.2, 4.0947, 4.0687, 4.0086, 3.9376, 3.8842, 3.8083, 3.7423,
    3.6821, 3.6326, 3.5785, 3.5046, 3.4399, 3.3188, 3.0726])
R0_data = np.array([0.037, 0.037857143, 0.037142857, 0.0375, 0.036785714,
    0.036785714, 0.037142857, 0.0375, 0.0375, 0.037857143, 0.0375, 0.0375,
    0.038928571, 0.041071429, 0.054285714])
R1_data = np.array([0.020, 0.022142857, 0.026428571, 0.035, 0.052857143,
    0.078571429, 0.0375, 0.028214286, 0.033928571, 0.040357143, 0.036428571, 0.03,
    0.031785714, 0.100714286, 0.242857143])
Tau1_data = np.array([20, 52.4, 44.2, 67.0, 125.2, 310.7, 229.1, 106.9, 138.6,
    212.8, 197.0, 128.3, 131.8, 302.1, 261.4])

# Polynomial fits for OCV, R0, R1, Tau1
ocv_poly = np.poly1d(np.polyfit(SoC_data, OCV_data, deg=5))
r0_poly = np.poly1d(np.polyfit(SoC_data, R0_data, deg=5))
r1_poly = np.poly1d(np.polyfit(SoC_data, R1_data, deg=5))
tau1_poly = np.poly1d(np.polyfit(SoC_data, Tau1_data, deg=5))

# ------------------ LOAD DATA FROM TEENSY ------------------
teensy_df = pd.read_csv("C:/Users/Pedro/Desktop/Pedro/PV/dados_1C.csv", sep=';',
    skiprows=1, header=None)
teensy_df = teensy_df[[1, 3]]
teensy_df.columns = ['Vt_teensy', 'SoC_teensy']

# ------------------ SIMULATION INITIALIZATION ------------------
I_descarga = 2.8  # Constant discharge current per cell [A]
N = 14131  # Total number of samples
I_cell = np.full(N, I_descarga)  # Discharge current

# Initialize SoC and state vectors
SoC = np.zeros(N)
SoC[0] = 1
x_hat = np.array([[0.0], [1]])
P = np.diag([1e-6, 1e-6])
Q_k = np.diag([1e-6, 1e-6])
```

```
R_k = np.array([[2.25e-4]])

# ------------------ SIMULATION LOOP ------------------
for k in range(N - 1):
    # Coulomb counting to estimate SoC
    SoC[k+1] = SoC[k] - (Ts / Q_total) * I_cell[k]
    V_oc = ocv_poly(SoC[k])  # Calculate OCV based on SoC
    R0 = r0_poly(SoC[k])  # Calculate R0 based on SoC
    R1 = r1_poly(SoC[k])  # Calculate R1 based on SoC
    Tau1 = tau1_poly(SoC[k])  # Calculate Tau1 based on SoC

    # Real voltage calculation
    V1_real = x_hat[0, 0]
    Vt_real[k] = V_oc - V1_real - R0 * I_cell[k]
    y_k = Vt_real[k] + np.random.normal(0, np.sqrt(R_k[0, 0]))  # Add noise

    # Kalman filter prediction and update
    A = np.array([[np.exp(-Ts / Tau1), 0], [0, 1]])
    B = np.array([[R1 * (1 - np.exp(-Ts / Tau1))], [Ts / Q_total]])
    x_pred = A @ x_hat + B * I_cell[k]
    P_pred = A @ P @ A.T + Q_k  # Predicted error covariance

    # Measurement update
    dVoc_dSoC = (ocv_poly(SoC[k] + 1e-5) - ocv_poly(SoC[k] - 1e-5)) / (2 * 1e-5)
    C = np.array([[-1, dVoc_dSoC]])
    y_est = ocv_poly(x_pred[1, 0]) - x_pred[0, 0] - R0 * I_cell[k]
    e = y_k - y_est  # Measurement residual

    # Kalman update
    S = C @ P_pred @ C.T + R_k
    K = P_pred @ C.T @ np.linalg.inv(S)
    x_hat = x_pred + K * e
    x_hat[1, 0] = np.clip(x_hat[1, 0], 0, 1)  # Ensure SoC is within bounds
    P = (np.eye(2) - K @ C) @ P_pred  # Update covariance

# ------------------ PLOTTING RESULTS ------------------
plt.figure(figsize=(12, 8))

# Plot 1     SoC comparison (Measured vs Simulated)
plt.subplot(3, 1, 1)
plt.plot(tempo_teensy, teensy_df['SoC_teensy'], label='SoC from Teensy', color='tab
    :blue', linewidth=2)
plt.plot(tempo_sim, SoC_sim_shifted, '--', label='Simulated SoC', color='tab:orange
    ', linewidth=2)
plt.ylabel('State of Charge (SoC)')
plt.xlabel('Time [s]')
plt.title('SoC Comparison: Teensy vs Simulated')
plt.grid(True, linestyle='--')
plt.legend()
```

```
85
86 # Plot 2     Terminal Voltage comparison (Measured vs Simulated)
87 plt.subplot(3, 1, 2)
88 plt.plot(tempo, teensy_df['Vt_teensy'], label='Measured Voltage', linewidth=1.5)
89 plt.plot(tempo_sim, Vt_sim_shifted, '--', label='Simulated Voltage', linewidth=1.5)
90 plt.ylabel('Terminal Voltage [V]')
91 plt.xlabel('Time [s]')
92 plt.title('Terminal Voltage Comparison')
93 plt.grid(True)
94 plt.legend()
95
96 # Plot 3     Error (SoC error)
97 plt.subplot(3, 1, 3)
98 plt.plot(tempo_erro, erro_soc, label='SoC error (Sim - Teensy)', color='tab:red',
       linewidth=2)
99 plt.axhline(0, color='gray', linestyle='--', linewidth=1)
100 plt.ylabel('Error')
101 plt.xlabel('Time [s]')
102 plt.title('Error between simulated and measured SoC')
103 plt.grid(True, linestyle='--')
104 plt.legend()
```

Listing A.3: EKF simulation and Teensy data comparison in Python

Appendix B

Custom Printed Circuit Board (PCB) Schematic and Design

B.1 BMS Slave PCB

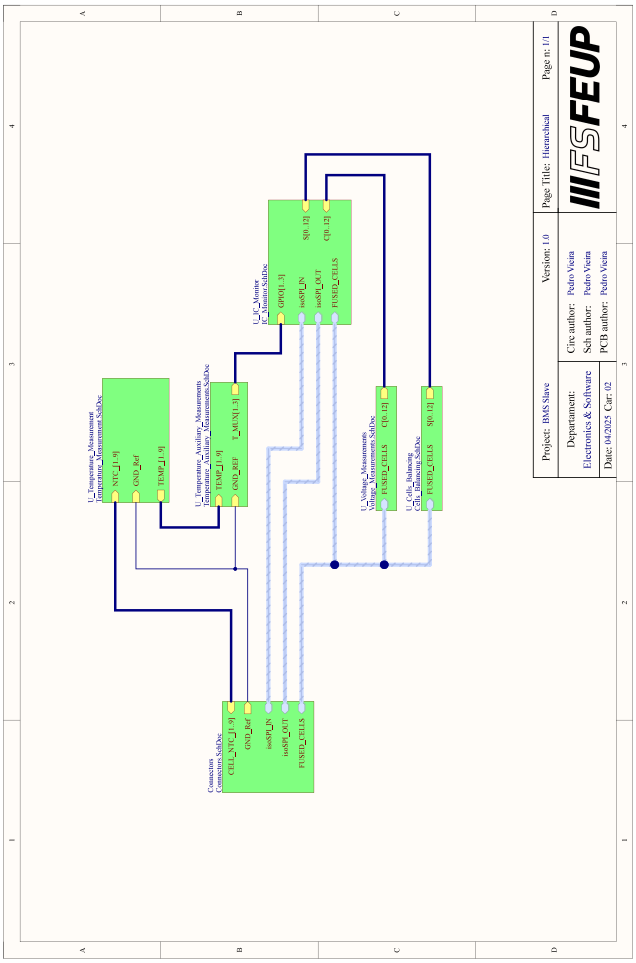


Figure B.1: BMS Slave Overview Diagram

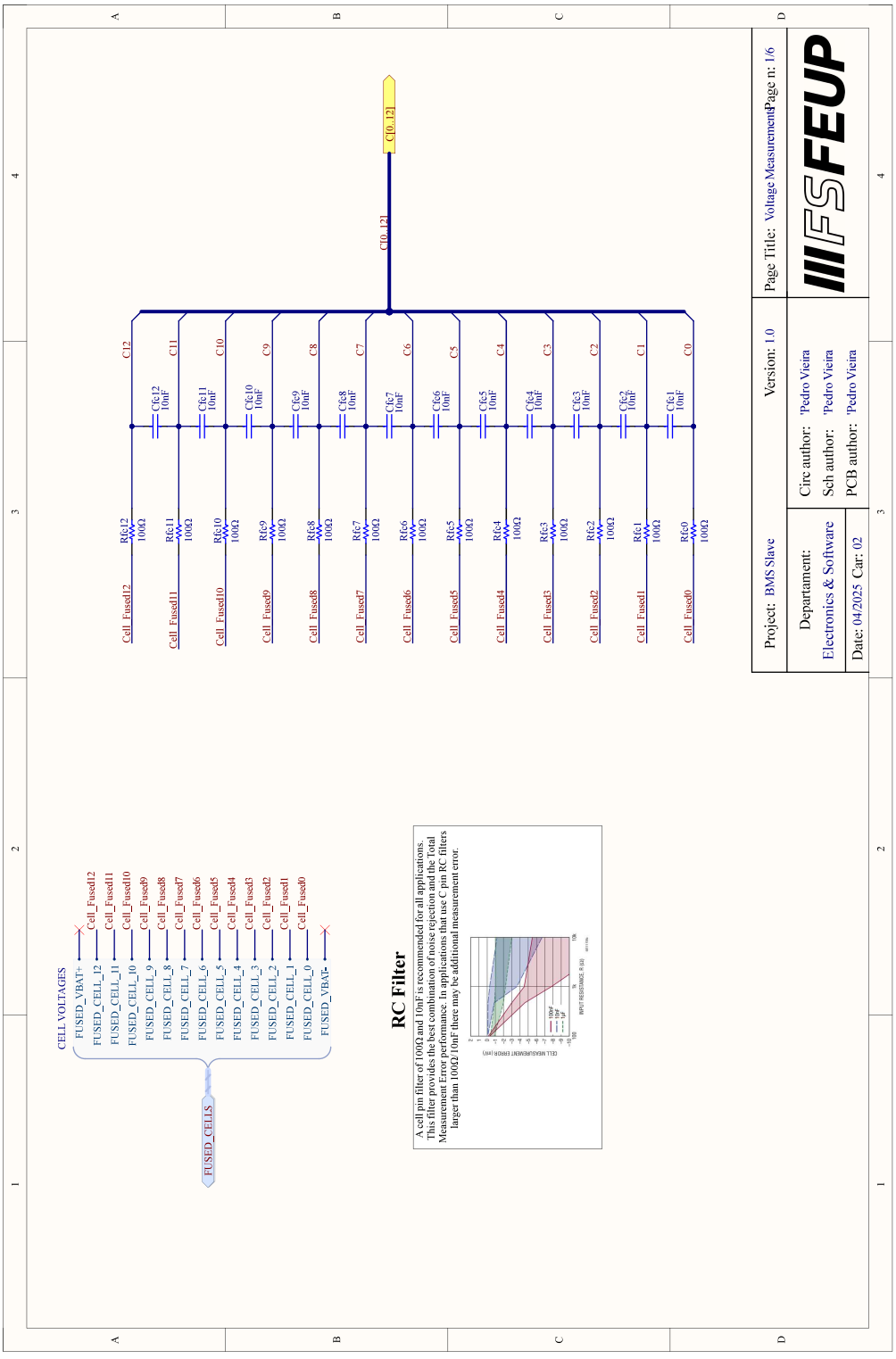


Figure B.2: Voltage Measurement Circuit (Page 1)

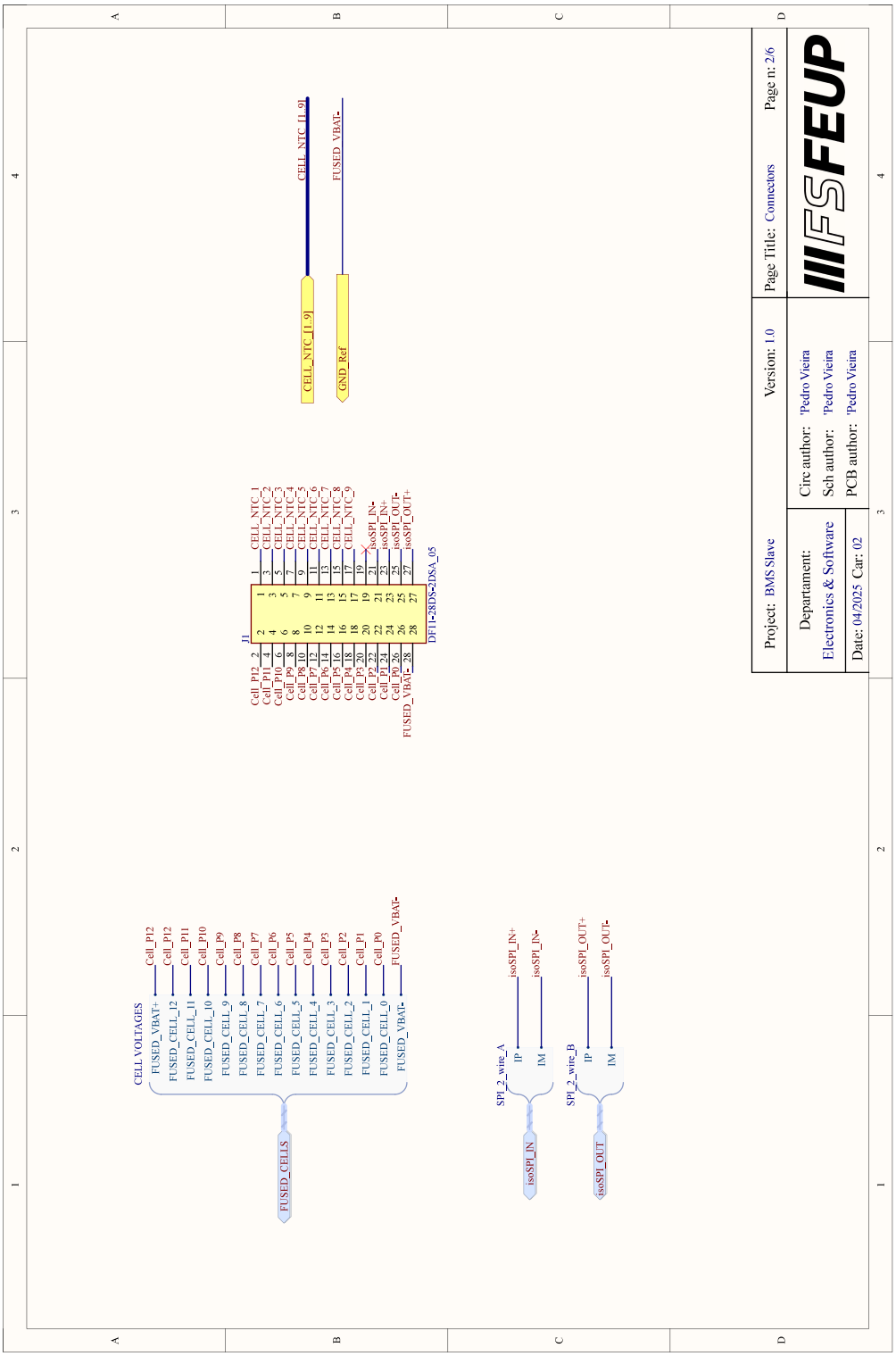


Figure B.3: Connectors (Page 2)

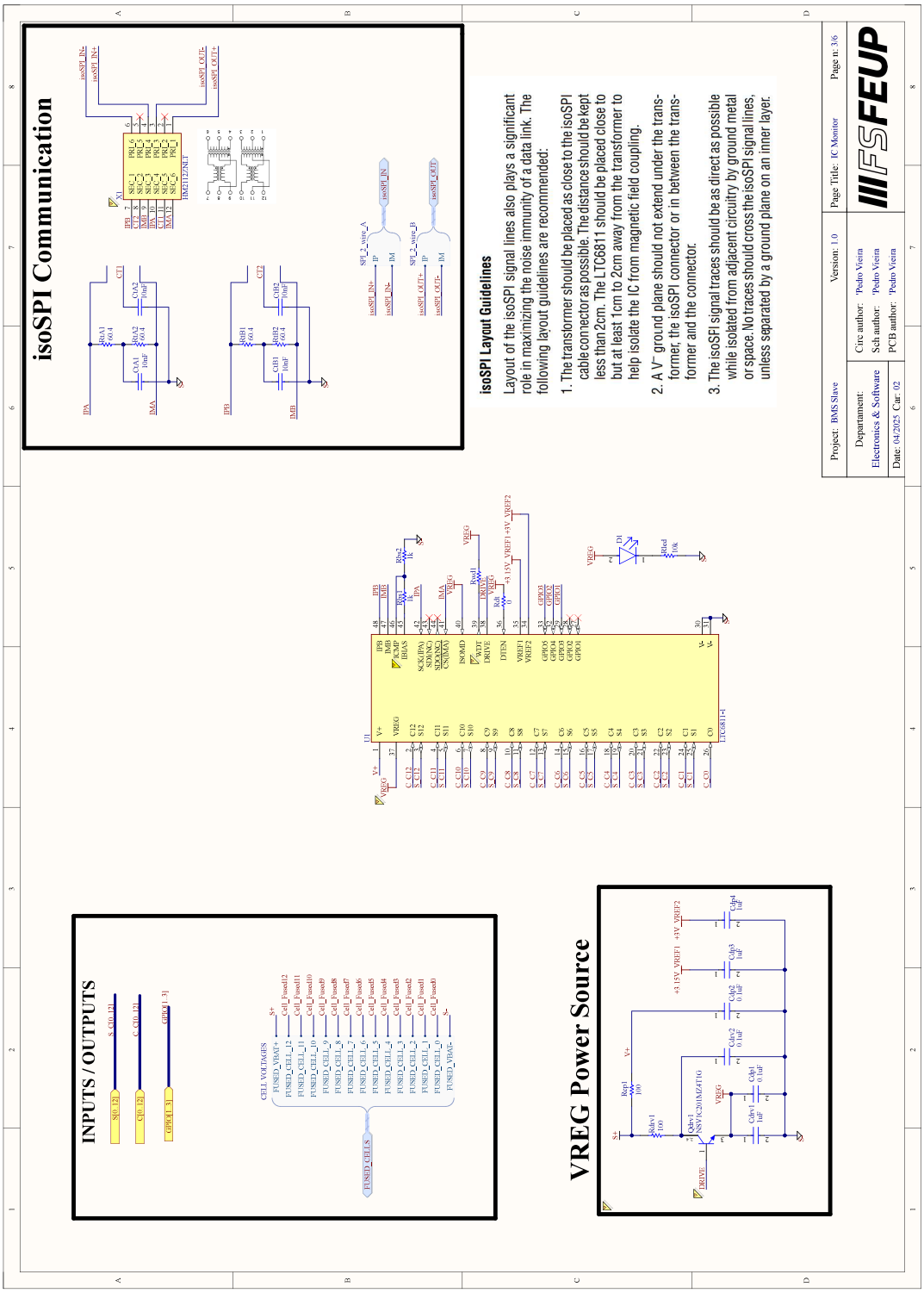


Figure B.4: IC Monitor (Page 3)

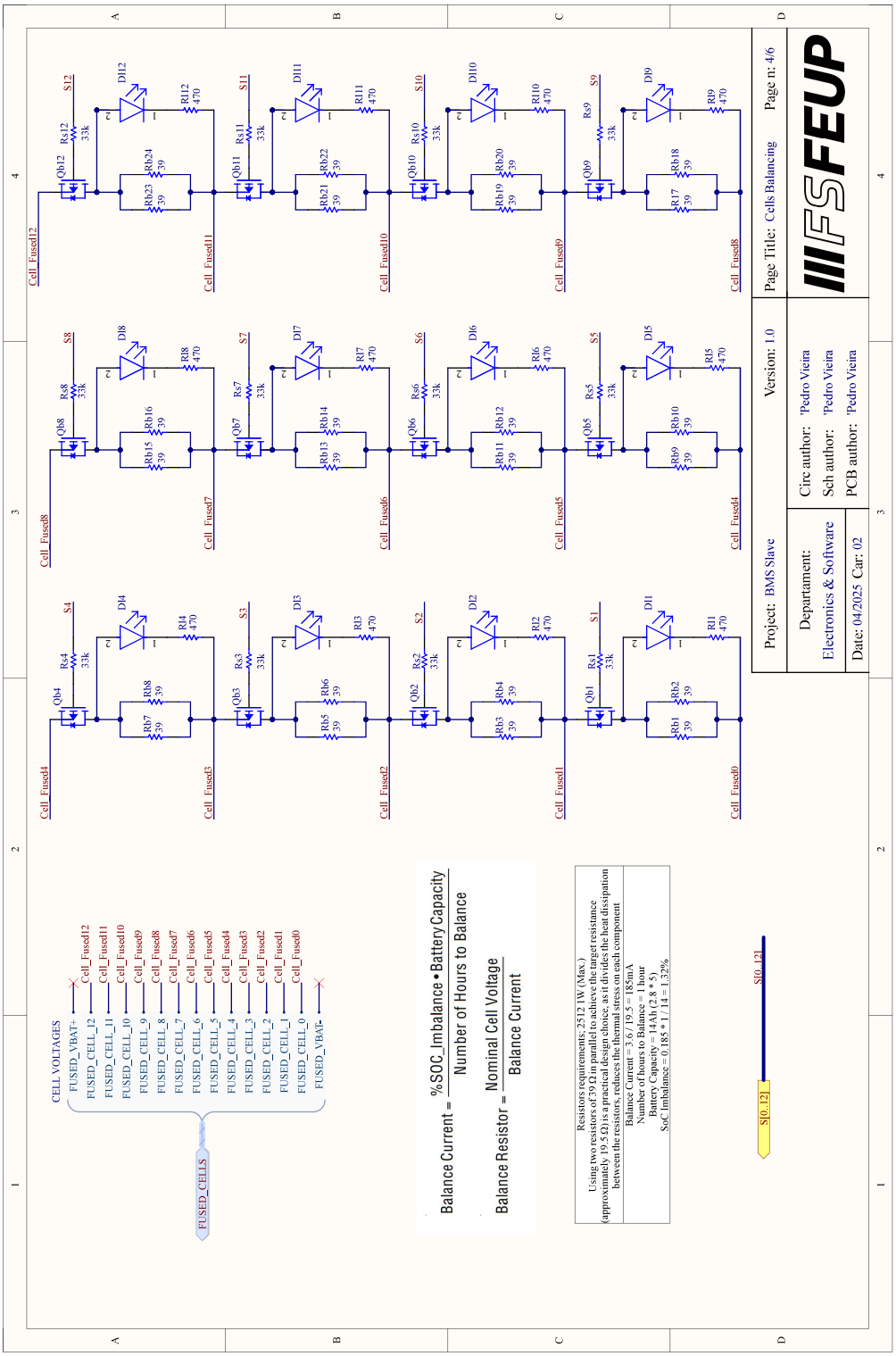


Figure B.5: Cells Balancing Circuit (Page 4)

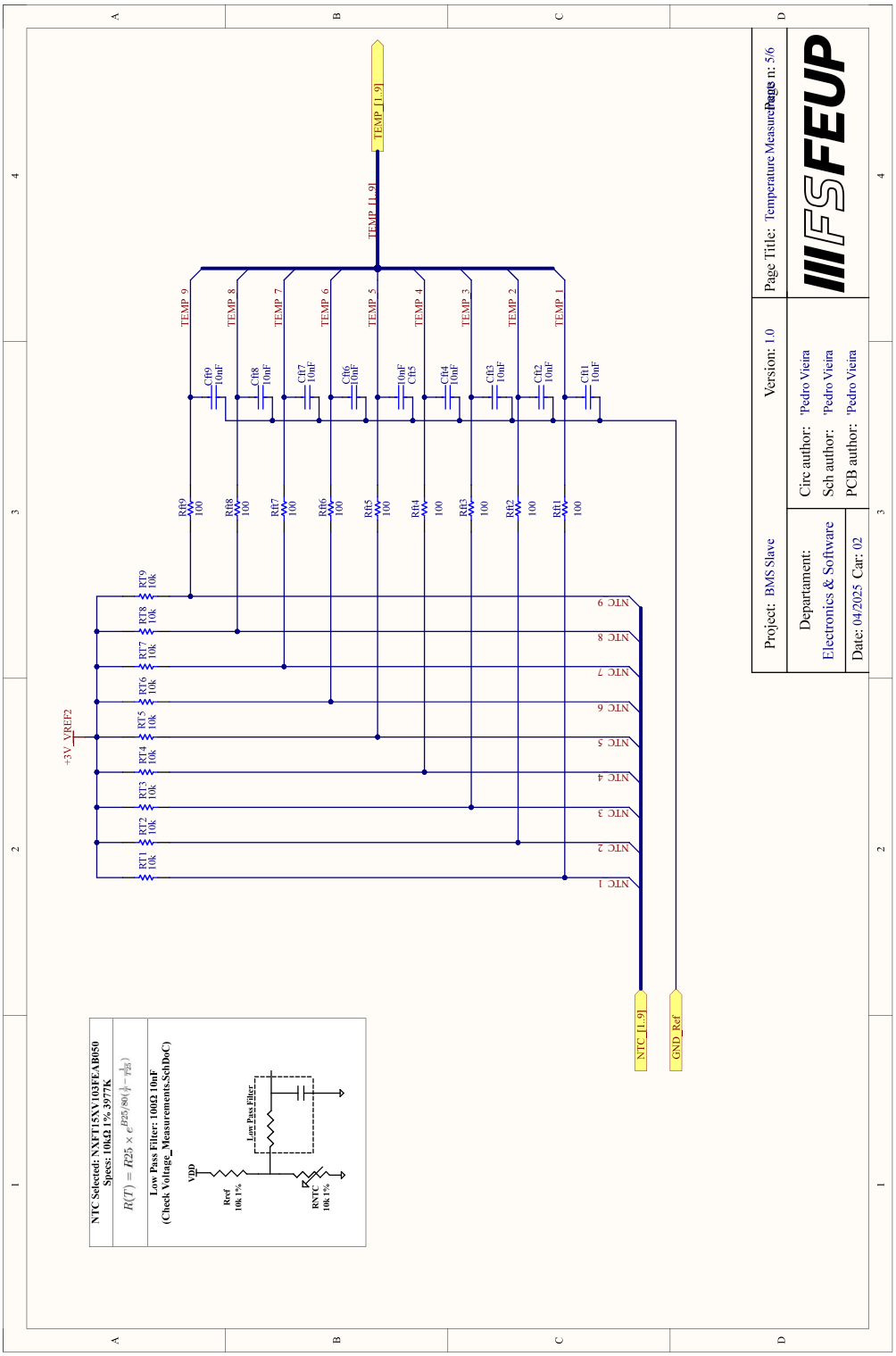


Figure B.6: Temperature Measurement System (Page 5)

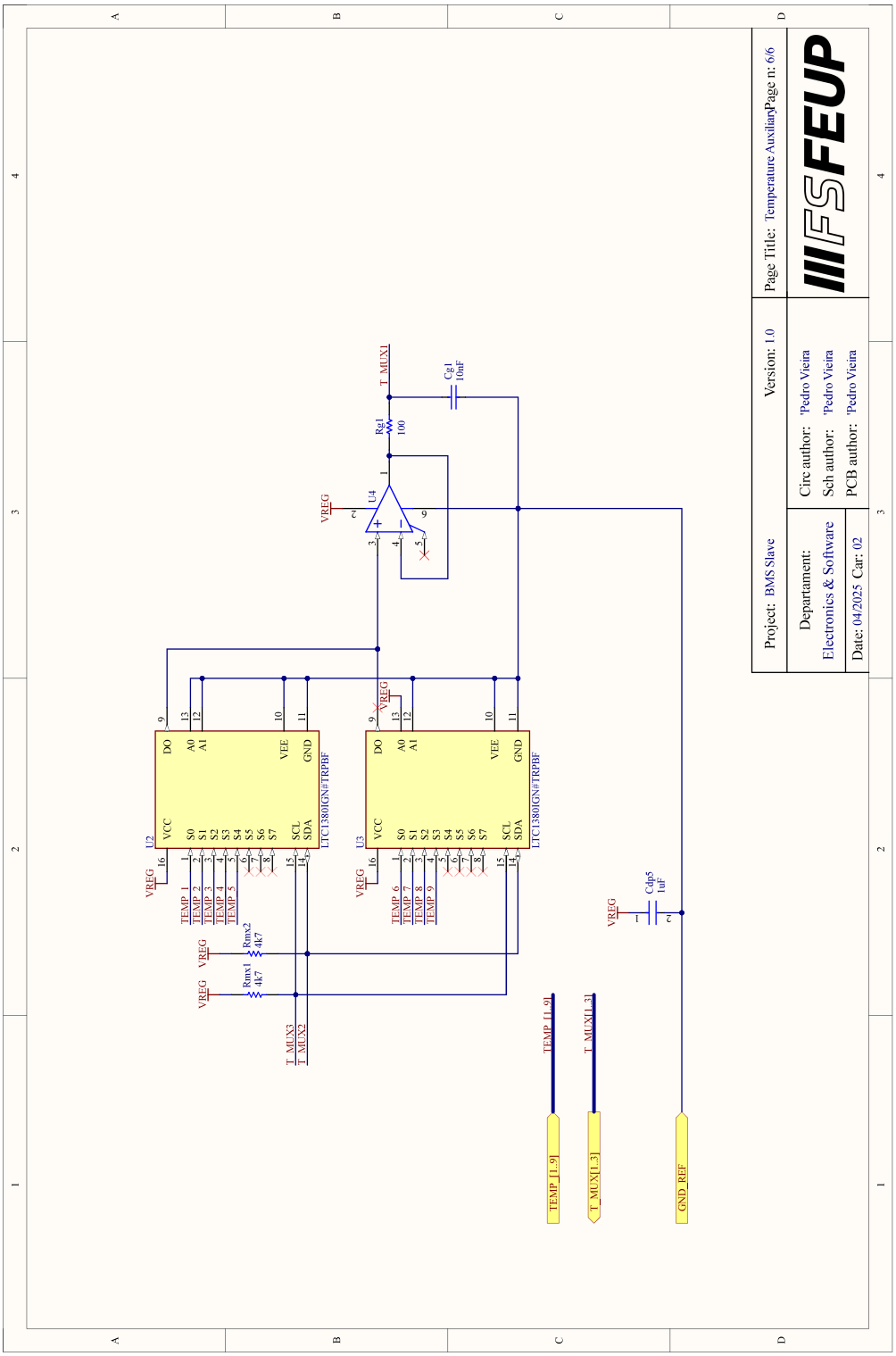


Figure B.7: Temperature Auxiliary Measurements (Page 6)

B.2 BMS Master PCB

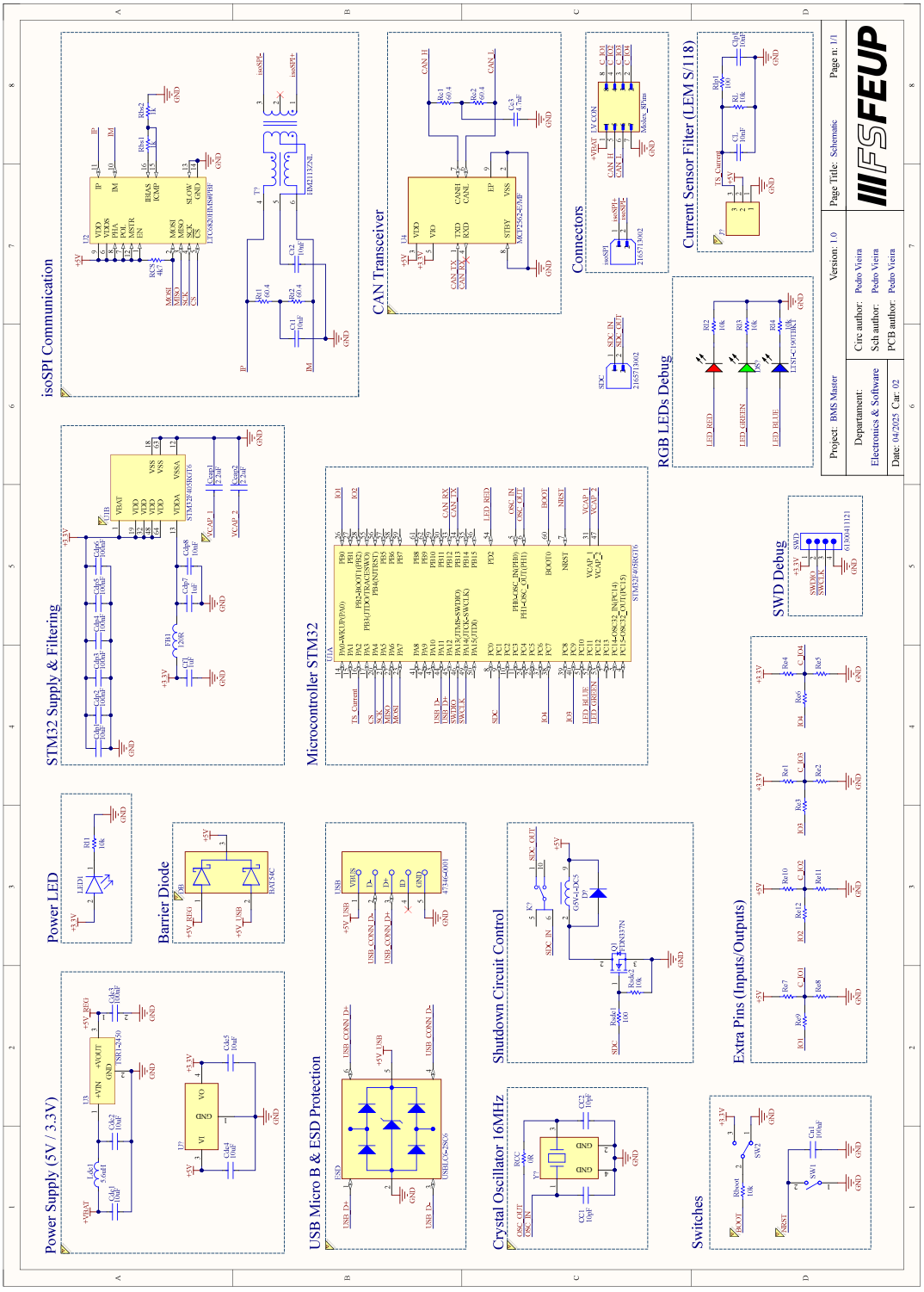


Figure B.8: BMS Master PCB Schematic

B.3 Stack PCB

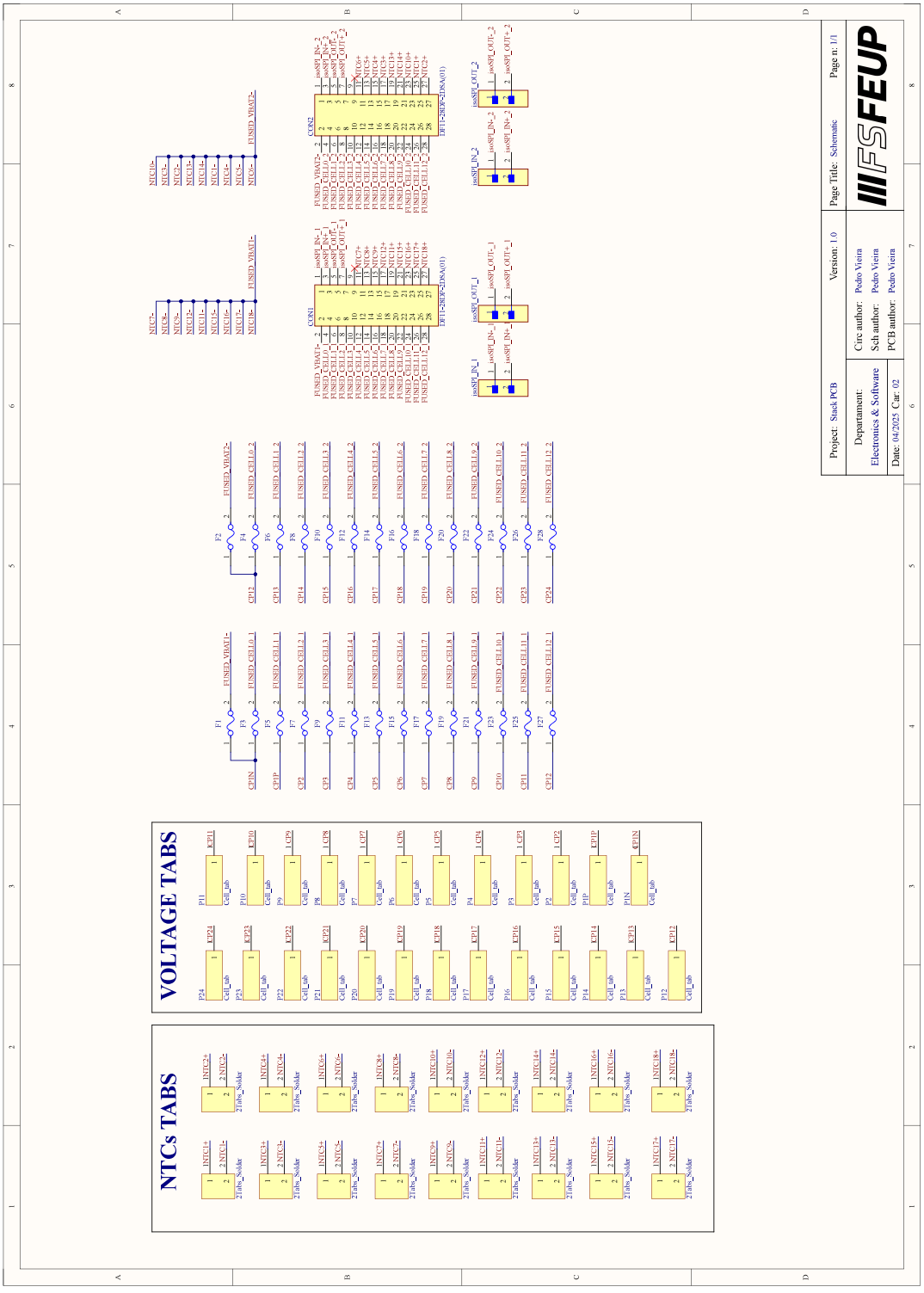


Figure B.9: Stack PCB schematic