

FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

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

Pedro Narciso Serra Campos



Master in Electrical and Computer Engineering

Supervisor: Prof. Agostinho Afonso da Rocha

October 13, 2025

Resumo

Esta dissertação apresenta o projeto, implementação e validação de um protótipo personalizado de um Sistema de Gestão de Bateria, desenvolvido para o veículo elétrico da equipa Formula Student FEUP. O projeto é motivado pela crescente exigência por sistemas de gestão de energia de alto desempenho, fiáveis e adaptáveis em veículos elétricos modernos. O sistema proposto foi desenhado para responder às necessidades específicas da equipa FSFEUP, mantendo a conformidade com os regulamentos da Formula Student.

O processo de desenvolvimento segue o enquadramento do ASPICE e adota as metodologias do modelo em V e da engenharia de sistemas baseada em modelos de simulação (MBSE), garantindo rastreabilidade, decomposição estruturada dos requisitos e verificação rigorosa em cada nível de desenvolvimento. Estes princípios guiaram todo o processo, desde a modelação conceptual inicial até à validação embutida, assegurando que todas as funções do sistema fossem progressivamente verificadas e testadas.

O projeto inicia-se com uma análise abrangente do estado da arte em gestão de baterias, investigando as tecnologias existentes e soluções comerciais disponíveis. Em seguida, é apresentada uma avaliação detalhada do BMS comercial anteriormente utilizado pela equipa. Esta análise identificou limitações importantes ao nível da integração mecânica, monitorização de temperatura, escalabilidade e configurabilidade, motivando assim o projeto de uma nova arquitetura descentralizada do tipo master-slave. A solução proposta introduz aquisição em tempo real de tensão, temperatura e corrente, um algoritmo híbrido de estimativa do estado de carga (SOC), mecanismos de controlo de balanceamento de células mais precisos, limites dinâmicos de corrente adaptáveis aos diferentes cenários da Formula Student, bem como diagnóstico e proteção do sistema.

Todas as simulações e testes físicos realizados ao longo do projeto foram cuidadosamente planeados com objetivos bem definidos. Cada etapa de validação foi pensada para avaliar comportamentos específicos do sistema e os seus desempenhos. Esta abordagem metodológica assegurou que todos os componentes e algoritmos do BMS fossem testados em condições operacionais relevantes, permitindo refinamentos fundamentados e validação robusta do sistema. O resultado final é um protótipo totalmente funcional que cumpre os principais requisitos definidos no início do projeto, demonstrando tanto a viabilidade como a eficácia da arquitetura proposta. Para além do sucesso técnico, o projeto oferece também uma base sólida e contributos relevantes para o desenvolvimento futuro, nomeadamente na adaptação, evolução e expansão desta solução em investigações futuras na área da tecnologia BMS, especialmente no contexto da continuidade deste projeto.

Abstract

The present dissertation presents the design, implementation, and validation of a custom Battery Management System prototype tailored for the Formula Student FEUP electric vehicle. The project is motivated by the increasing demand for high-performance, reliable, and adaptable energy management systems in modern electric vehicles. The proposed system is tailored to match the specific needs of the FSFEUP team while remaining compliant with Formula Student regulations.

The development process follows the ASPICE framework and leverages both the V-model and Model-Based Systems Engineering methodologies to ensure traceability, structured decomposition of requirements, and rigorous verification at each development level. These principles guided the entire process, from early conceptual modeling to embedded validation, ensuring that all system functions could be progressively verified and tested.

The project begins with a comprehensive analysis of the current state of the art in battery management, investigating existing technologies and available solutions. Following this, a detailed evaluation of the commercial BMS previously used by the team was conducted. This analysis identifies key limitations in mechanical integration, temperature monitoring, scalability, and configurability, motivating the design of a new decentralized master-slave architecture. The proposed solution introduces real-time acquisition of voltage, temperature, and current, a hybrid state of charge (SOC) estimation algorithm, precise cell balancing control mechanisms, dynamic current control limits adaptable to the varied scenarios encountered in Formula Student events, diagnostic and protection.

All simulations and physical tests conducted throughout this project were purposefully designed with well-defined objectives. Each validation step was carefully planned to evaluate specific system behaviors and performance metrics. This methodical approach ensured that every component and algorithm of the BMS was tested under relevant operational conditions, allowing for informed refinements and robust validation of the system. The final result is a fully functional prototype that meets the key requirements outlined at the beginning of the project, demonstrating both the feasibility and effectiveness of the proposed architecture. Beyond its technical success, the project also offers a solid foundation and valuable insights for future developments, particularly in adapting, evolving and expanding this solution for future research and development in the field of BMS technology, particularly within the context of the continuation of this project.

UN Sustainable Development Goals

In 2015, the United Nations introduced the 2030 Agenda for Sustainable Development [1], a global initiative consisting of 17 Sustainable Development Goals (SDGs) aimed at addressing critical challenges such as climate change, environmental degradation, and energy access. Engineering projects like this one, focused on battery management systems (BMS) for electric vehicles, play an important role in advancing these objectives through technological innovation and responsible system design.

This dissertation contributes to the following SDGs:

- **SDG 7 – Affordable and Clean Energy:** The proposed BMS architecture promotes efficient energy usage in electric vehicles by improving charge control, thermal management, and state-of-charge estimation. This leads to longer battery lifespan and more reliable vehicle operation, encouraging the broader adoption of clean energy in mobility systems.
- **SDG 9 – Industry, Innovation and Infrastructure:** This project introduces a scalable, modular, and cost-effective battery management solution tailored for electric vehicles, following a decentralized topology. By leveraging embedded systems, model-based design, and rigorous validation, it contributes directly to innovation in electric mobility and infrastructure development.
- **SDG 12 – Responsible Consumption and Production:** Through accurate monitoring and diagnostics, the system maximizes battery usage efficiency and prevents premature degradation. This extends the lifecycle of critical components, reduces waste, and supports a more sustainable and circular use of resources in electric transportation.
- **SDG 13 – Climate Action:** The transition to electric vehicles is key in reducing greenhouse gas emissions. By providing a reliable and customizable BMS tailored for Formula Student prototypes and potentially beyond, this work supports the electrification of transport and contributes to global efforts in climate change mitigation.

Overall, the work presented in this thesis highlights how engineering, focused research and prototyping can directly contribute to the achievement of sustainability goals, particularly in the context of cleaner transportation and smarter energy systems.

Acknowledgements

I would like to express my gratitude to Professor Agostinho Afonso da Rocha, for his guidance, availability, and trust throughout the development of this project. His mentorship played a fundamental role in shaping the direction and quality of this dissertation.

I extend my sincere gratitude to Xelerate for providing the necessary conditions for this project. Special appreciation goes to Daniel Filipe Pereira Magalhães for his patience, technical knowledge, and prior work on the subject.

I would also like to thank investigator Vicente for his practical assistance during the hardware assembly stages and system integration. His help was indispensable in the physical implementation of the prototype.

Lastly, I am grateful to my family and friends, whose presence, motivation, and belief in me have been invaluable throughout this academic journey. Specially to my girlfriend Mariana, for her encouragement, and emotional support during the most intense periods of the project.

To everyone who, in any way, contributed to the completion of this dissertation, a big thank you!

Pedro Campos

“The future belongs to those who prepare for it today.”

Malcolm X

Contents

1	Introduction	1
1.1	Problem and Context Definition	1
1.1.1	Formula Student Context	2
1.1.2	FSFEUP Team Overview	3
1.1.3	Xelerate DigitalCar Context	6
1.2	Project Goals and Development Scope	6
1.3	Thesis Structure	8
2	State of the Art	9
2.1	Battery Management Systems (BMS)	9
2.1.1	Evolution Over the Time	10
2.2	BMS Topologies	11
2.2.1	Distributed Topology	11
2.2.2	Non-distributed Topology	12
2.2.3	Hybrid Topology	14
2.2.4	Topologies Comparison	15
2.3	BMS Features and Functionalities	15
2.4	State Estimation Methods	16
2.4.1	SOC Estimation Methods	17
2.4.2	SOH Estimation Methods	19
2.4.3	Battery Models	22
2.5	Fault Diagnosis	29
2.5.1	Importance of Fault Diagnosis	29
2.5.2	Types of Faults in Li-ion Batteries	29
2.5.3	Fault Diagnosis Techniques	30
2.5.4	Implications of Fault Diagnosis	30
2.6	Current Control	31
2.6.1	Advanced Control Methods	32
2.7	Cell Balancing	33
2.7.1	Introduction to Cell Balancing	33
2.7.2	Balancing Methods	34
2.7.3	Passive Balancing	34
2.7.4	Active Balancing	36
2.7.5	Cell Balancing Circuits Comparison	39
2.7.6	Balancing Control Strategies	40
2.7.7	Based on Automatic Balancing	41
2.7.8	Comparison of Balancing Control Strategies	41
2.8	Characteristics of the BMS's Implemented in the Competition	41

3	Development Process and System Design	43
3.1	Development Strategy	43
3.1.1	Automotive SPICE	44
3.1.2	The V-Model Development Process	44
3.1.3	Model Based System Engineering	45
3.2	System Analysis and Proposed Architecture	46
3.2.1	Rules and Requirement Analysis	46
3.2.2	Previous BMS analysis	49
3.2.3	New BMS configuration	52
3.2.4	Proposed solution description	53
3.3	System Structuring and Development Prioritization	55
3.4	Tools	56
4	Model-Based Simulations	57
4.1	External Communication: BMS to PC Interface	58
4.2	Initial Battery Modeling and Validation	59
4.2.1	Battery Block Parametrization	59
4.2.2	Battery Block validation	62
4.2.3	HMI configuration	66
4.3	Hybrid Estimation Model	66
4.3.1	KiBaM Model Estimation	68
4.3.2	Open-Circuit Voltage (Voc) Estimation	70
4.3.3	Electric Model Estimation	71
4.4	Hybrid Model Validation	72
5	Master–Slave Integration and Communication	76
5.1	Communication Protocol Architecture	77
5.2	Physical Integration and Electrical Adaptations	78
5.2.1	Slave Power-Up and Resistor Ladder Configuration	78
5.2.2	Voltage Compatibility and Signal Conditioning	79
5.2.3	Wake-Up Signal and Jumper Interference	80
5.2.4	Initial Frame Handling and Addressing Strategy	80
5.3	Command Structure and Frame Format	81
5.4	Communication Initialization and Slave Configuration	82
5.4.1	ADC Configuration and Initialization	83
5.4.2	ADC Operation and Data Processing	85
5.4.3	Configuration of the Number of Active Cells	86
5.5	Voltage Acquisition and Validation	87
6	Full Prototype Integration and Acquisition System Validation	90
6.1	Battery Pack Assembly and Real Voltage Acquisition	90
6.1.1	Battery Pack Integration	91
6.1.2	Internal Voltage Acquisition Circuit	92
6.1.3	Firmware Adaptation and Data Processing	93
6.2	Temperature Acquisition System	94
6.2.1	Circuit Setup and Board Configuration	94
6.2.2	Measurement Principle and Software Implementation	95
6.2.3	Component Selection and Final NTC Integration	98
6.3	Current Acquisition System	100

6.3.1	Hardware Configuration and Sensor Selection	100
6.3.2	Physical Integration	101
6.3.3	Software Implementation and Measurement Routine	102
6.4	Sensor Reading Stabilization via Buffered Sampling	102
6.4.1	Timer Configuration and Sampling Strategy	103
6.4.2	Buffering Mechanism and Validation	103
6.4.3	System Validation	106
7	Model-Based Testing and System Validation	108
7.1	Charging and Discharging Infrastructure	109
7.1.1	ISDT T8 BattGO Smart Charger	109
7.1.2	BK Precision 8510 DC Electronic Load	111
7.2	Validation of SOC Estimation Under Full Discharge Test	112
7.2.1	Test Motivation and Objectives	113
7.2.2	Results from Initial Estimation Strategy	114
7.2.3	Relaxation Triggering and One-Time Voc-Based Estimation	118
7.2.4	Time-Based Validation Using Delayed Comparison	123
7.2.5	Robust Relaxation Detection with Timed FIFO Sampling	127
7.2.6	Impact of Voc Estimation Errors on Capacity Calculation	129
7.2.7	Final Estimation Strategy: KiBaM-Only with Voc Initialization	134
7.2.8	Evaluation of Electrical Model Parametrization	137
7.2.9	Hybrid System Validation	148
7.3	Diagnostic and Shutdown Control	149
7.4	Internal Resistance Reestimation	152
7.4.1	Practical Implementation Strategy	153
7.4.2	Impact on V_{out} Estimation and Model Accuracy	153
7.4.3	Experimental Results and Validation	156
7.5	Balancing Test	163
7.5.1	Balancing Capability in the Slave Module	163
7.5.2	Balancing Configuration Steps	164
7.5.3	Balancing Control Strategy	170
7.5.4	Test Results and Analysis	171
7.5.5	Analysis and Conclusion of the Balancing Test	176
8	Conclusions	177
8.1	Project Achievements and Reflections	177
8.2	Future Work and Recommendations	178
	References	181
A	Use Case Diagram	186
B	Work Breakdown Structure and Development Plan	187

List of Figures

1.1	Formula Student events point distribution.	2
1.2	FSFEUP team members, 2025 edition.	4
1.3	FSFEUP01 – The first prototype developed by FSFEUP.	5
1.4	FSFEUP02 – Current development cycle prototype.	5
2.1	Annual demand forecast for EVs in the world, from [7].	10
2.2	BMS topologies.	11
2.3	Distributed BMS topology, adapted from [8].	12
2.4	Modular BMS topology, adapted from [8].	13
2.5	Master-Slave BMS topology, adapted from [8].	13
2.6	Centralized BMS topology, adapted from [8].	14
2.7	FST Hybrid BMS topology, from [9].	15
2.8	BMS main functionalities, adapted from [12].	16
2.9	Typical OCV-SOC curve for a Lithium-Ion battery.	18
2.10	Cycle characteristics from Molicel model INR-18650-P28A with different current discharge conditions.	20
2.11	Battery voltage dynamics under charge/discharge current pulses, from [13].	21
2.12	Different types of battery models, adapted from [20].	23
2.13	Simple battery model, from [21].	24
2.14	Thevenin equivalent electrical circuit (1RC), from [21].	24
2.15	n-th-order generalized circuit, from [21].	25
2.16	Conceptual illustration of the KiBaM, adapted from [23].	27
2.17	Hybrid model fusing KiBaM and Rint model, from [23].	28
2.18	CC-CV charging profile, from [28].	31
2.19	The impact of the SOC imbalance on 4-cell battery pack during charging and discharging scenarios, from [31, 32].	33
2.20	Resume diagram of cell balancing topologies based on the electronic component used, adapted from [33].	34
2.21	Fixed shunt resistor passive balancing circuit with the Zener diode, from [34].	35
2.22	Switched shunt resistor passive balancing circuit, from [34].	35
3.1	V-model process for BMS development, from [10].	45
3.2	Requirement decomposition across SYS, HWE, and SWE levels, adapted from [36].	47
3.3	Orion BMS used in the first-generation FSFEUP electric vehicle, from [38].	49
3.4	Orion BMS used in the second-generation FSFEUP electric vehicle, from [38].	49
3.5	Accumulator configuration of the first FSFEUP car, showing the Orion BMS integration.	50
3.6	Accumulator configuration of the second FSFEUP car, with the updated Orion BMS.	51

3.7	High-level BMS architecture.	53
3.8	Final implemented BMS configuration.	54
4.1	STMicroelectronics STM32F446 Nucleo-64 [41].	57
4.2	USART2 pin mapping for UART communication between the STM32 and the PC.	58
4.3	Molicec INR-18650-28P, from [44].	59
4.4	Datasheet battery block from Simulink, from [45].	60
4.5	Battery model used by the Simulink battery block.	61
4.6	Discharge rate characteristics graph from cell datasheet [43].	63
4.7	Discharge rate characteristics graph from Matlab with C=2800 mAh.	63
4.8	Discharge rate characteristics graph from Matlab with C=2846 mAh.	64
4.9	HMI interface used to configure cell parameters.	66
4.10	Equivalent electrical representation of the KiBaM system, adapted from [20].	67
4.11	Hybrid battery model representation, adapted from [47].	68
4.12	Graphical representation of the KiBaM analogy under discharge and recharge processes.	69
4.13	Discharge curve of the hybrid model in Simulink.	74
4.14	Discharge curve of the hybrid model on STM32.	74
5.1	BQ79616 evaluation module (EVM), from [49].	76
5.2	STM32F446RE UART configuration for USART1 (Master-Slave) and USART2 (PC communication).	77
5.3	Slave configuration using the internal resistor ladder.	78
5.4	Physical implementation of the UART voltage divider between Master and Slave.	79
5.5	Structure of the UART command frame used for Master-Slave communication, from [50].	81
5.6	BQ79616EVM-021 main ADC measurements path, from [50].	83
5.7	ADC_CONF1 register configuration, from [50].	84
5.8	ADC_CTRL1 register configuration, from [50].	84
5.9	Round-Robin ADC sampling process, from [50].	85
5.10	ACTIVE_CELL command and configuration, from [50].	86
5.11	Frame sent and received for VCELL voltage read command.	88
5.12	HMI interface displaying acquired VCELL voltages from the BQ79616EVM-021 slave.	89
6.1	Assembled 6-cell battery pack.	91
6.2	Assembled 6-cell battery pack connected to the slave board.	92
6.3	Internal voltage acquisition circuit of the BQ79616, from [50].	93
6.4	GPIO circuit for NTC measurements.	94
6.5	NTC's placement during the test of the BMS prototype.	95
6.6	Slave board GPIO configuration, from [48].	96
6.7	GPIO_CONF1 register configuration, from [50].	96
6.8	GPIO_CONF2 register configuration, from [50].	97
6.9	NTC Thermistor Murata Electronics NXFT15XH103FEAB035, from [52].	99
6.10	Final implementation of the temperature acquisition circuit with Murata NTCs.	99
6.11	STM32 CubeMX configuration for ADC1_IN10 (PC0).	100
6.12	Allegro current sensor ACS759LCB-050B-PFF-T.	101
6.13	Current sensor pinout diagram, from [53].	101
6.14	TIM2 configuration in STM32CubeMX for 10 ms sampling interval.	103

6.15	Terminal output showing structured data logging every 200 ms after 20 measurements.	104
6.16	Terminal output showing 200 ms VCell buffer print intervals.	105
6.17	Closer look at VCell readings.	105
7.1	ISDT T8 BattGO Smart Charger, physical layout and ports, from [55].	109
7.2	Mean Well HRP-1000-24 power supply used to power the ISDT T8 charger, from [56].	111
7.3	BK Precision 8510 DC Electronic Load used for programmable discharge testing, from [57].	111
7.4	Battery voltage vs. accumulated capacity from PV8500 software (reference curve).	114
7.5	Battery voltage vs. accumulated capacity from VSCode (reference curve).	115
7.6	Total battery voltage over time during discharge, acquired by the BMS.	115
7.7	Estimated capacity vs. battery voltage (from BMS firmware).	116
7.8	Voltage evolution of each cell over time during the discharge.	117
7.9	SOC minimum and maximum evolution during discharge (initial estimation strategy).	117
7.10	SOC and voltage evolution of the highest-voltage cell during the beginning of the discharge process.	119
7.11	Voltage and current during initial discharge.	120
7.12	Corrected SOC estimation in the beginning of the discharge process.	121
7.13	Current and voltage evolution after logic correction.	121
7.14	Premature Voc estimation using basic FIFO buffer.	122
7.15	Transition from KiBaM to Voc estimation using time-delayed validation logic.	123
7.16	Inconsistent activation of the relaxation flag across multiple cells using the time-based delayed comparison logic.	124
7.17	Zoomed view of the pulsed discharge test, showing the voltage evolution of all cells during a 5 minute discharge and 10 minute relaxation cycle.	125
7.18	Divergent relaxation flag activation across multiple cells during a pulsed discharge scenario.	126
7.19	Evolution of SOC_max and SOC_min during the pulsed discharge test.	126
7.20	Vcells evolution of all cells using the robust relaxation detection strategy.	127
7.21	SOC evolution of all cells using the robust relaxation detection strategy.	128
7.22	SOC_max and SOC_min evolution during the robust relaxation detection test.	128
7.23	Comparison between real extracted capacity and estimated capacity.	129
7.24	Estimation error between real and SOC-based capacity values during the test.	130
7.25	Terminal voltage evolution of all cells during the pulsed discharge in Test 5.	131
7.26	Relaxation state detection during Test 5.	131
7.27	Comparison of measured voltage (V_{cell1}), estimated open-circuit voltage (V_{oc}), and estimated terminal voltage (V_{out}) of Cell 1.	132
7.28	Percentage error between measured voltage and estimated V_{out} of Cell 1.	132
7.29	SOC evolution over time of Cell 1.	133
7.30	Comparison between extracted charge computed from SOC and actual extracted charge.	133
7.31	SOC estimation of Cell 1 using KiBaM-only strategy.	134
7.32	Comparison between extracted charge from SOC estimation and actual charge extracted.	135
7.33	Percentage SOC error between extracted charge from KiBaM estimation and actual extracted charge.	135

7.34	Comparison of V_{cell1} , V_{oc} , and V_{out} of Cell 1, with the final strategy.	136
7.35	Percentage error between V_{cell1} and estimated V_{out} of Cell 1.	136
7.36	Comparison between the SOC function obtained by inverting $V_{OC}(SOC)$ and the fitted logistic $SOC(V_{OC})$ function.	138
7.37	Discrepancy between both functions.	138
7.38	Relative error of the comparison between the 2 electrical model functions.	139
7.39	Extracted SOC–voltage points at the beginning of the discharge, used for comparison against the datasheet curve.	140
7.40	Voltage profile during the reference discharge test and the relaxed state variable. .	141
7.41	SOC points estimated from the $SOC(V_{OC})$ function during relaxed states.	141
7.42	Evolution of the V_{OC} as a function of the extracted charge.	142
7.43	Comparison between the $SOC(V_{OC})$ function parametrization and the fitted datasheet curve from cell datasheet.	142
7.44	Comparison between measured V_{cell} and estimated V_{out} for all six cells during the reference discharge.	144
7.45	Comparison between measured V_{cell} and estimated V_{out} during rest periods ($I = 0A$). .	144
7.46	Absolute voltage error between V_{cell} and V_{out} during rest periods.	145
7.47	Percentage voltage error relative to the total OCV swing during rest periods. . . .	145
7.48	Comparison between measured V_{cells} and estimated V_{out} during active discharge periods ($I \neq 0A$).	146
7.49	Absolute voltage error between V_{cells} and V_{out} during active discharge.	146
7.50	Percentage voltage error relative to the total swing, during active discharge. . . .	147
7.51	Overview of the general BMS state machine.	150
7.52	Normal operation states: Charging, Discharging, and Balancing.	151
7.53	Error handling and shutdown logic.	151
7.54	Evolution of R_{int} for Cell 3 during the discharge test.	154
7.55	Zoomed view of voltage and current variation used to reestimate R_{int} for Cell 3. .	154
7.56	Comparison of V_{cell3} , V_{oc_cell3} , and V_{out_cell3} (model output with R_{int} reestimation). .	155
7.57	Reduction in V_{out} estimation error after enabling R_{int} reestimation.	156
7.58	Measured voltages for all six cells during the four discharge profiles.	157
7.59	Estimated SOC's for all six cells during the four discharge profiles.	158
7.60	Comparison between the real extracted charge and critical cells extracted charge. .	158
7.61	Errors between the real extracted charge and critical cells extracted charge estimations.	159
7.62	Comparison between the V_{cell1} , V_{oc_cell1} , and the V_{out_cell1} , that's a cell that didn't get the R_{int} reestimation.	159
7.63	Error between the V_{cell1} , V_{oc_cell1} , and the V_{out_cell1} , that's a cell that didn't get the R_{int} reestimation.	160
7.64	Comparison between the V_{cell3} , V_{oc_cell3} , and the V_{out_cell3} , that's a cell that get the R_{int} reestimation.	160
7.65	Comparison between V_{cell3} and V_{out_cell} , expressed as an error percentage, with and without the R_{int} reestimation.	161
7.66	R_{int} reestimated for cell 3.	161
7.67	ADC_CTRL3 register configuration, from [50].	165
7.68	Cell balancing time configuration register (CB_CELL_CTRL), from [50].	165
7.69	Cell balancing with 2 consecutive CBFET's on, from [50].	167
7.70	Cell balancing with one internal CBFET, from [50].	167
7.71	Device configuration register (DEV_CONF), from [50].	168

7.72	Cell balancing control register (BAL_CTRL2), from [50].	168
7.73	Cell balancing status register (BAL_STAT), from [50].	169
7.74	Voltage evolution of Cell2 and Cell4 during balancing.	171
7.75	Zoomed view of voltage profile for Cell2 and Cell4 showing periodic balancing steps.	172
7.76	SOC evolution of Cell2 and Cell4 during balancing test.	172
7.77	Voltage evolution of all six cells during full balancing test.	173
7.78	Zoomed voltage view showing interaction between Cell 2, Cell 1, and Cell 3 during balancing.	174
7.79	SOC progression of all six cells during the full balancing cycle.	175
A.1	Use Case Diagram	186
B.1	Integrated WBS and Gantt Chart for BMS Project	188

List of Tables

1.1	Summary of BMS objectives.	7
2.1	Comparison of BMS topologies, adapted from [8].	15
2.2	Comparison of SOC estimation methods, adapted from [14, 15, 16].	19
2.3	Comparison of SOH estimation methods, [13, 18, 19].	22
2.4	Comparison of different balancing circuits characteristics, adapted from [29]. . .	39
2.5	Comparison of balancing control variables, adapted from [13, 35].	41
2.6	Comparison of BMS across different Formula Student teams, adapted from [9]. .	42
3.1	Summary of the most important system requirements.	48
3.2	Estimated cost of Orion BMS units used, from [38].	50
3.3	Development prioritization of system modules based on SBS.	55
3.4	Tools used in the BMS development.	56
4.1	Molicel model INR-18650-P28A datasheet cell characteristics, from [43].	60
4.2	Coefficients used in the $V_{OC}(SOC)$ fitting, from [42].	62
4.3	Error metrics between fitted model and datasheet curve.	62
4.4	Comparison of extracted capacity using $C=2800$ mAh.	64
4.5	Comparison of extracted capacity using $C=2846$ mAh.	64
4.6	Equivalence between KiBaM and electrical circuit parameters, adapted from [20].	67
4.7	Coefficients used in the $SOC(V_{OC})$ fitting function.	71
4.8	Parameters used in the hybrid model validation.	73
4.9	Reference voltage values from datasheet at specific SOC levels.	73
4.10	Error metrics between STM32 implementation and Simulink reference model. . .	74
4.11	Voltage error between STM32 simulation and cell datasheet.	75
6.1	Relevant pin assignments for 6-Cell connection on J15, adapted from [48].	92
6.2	Characteristics of initial NTC thermistors, NTCC-10K from SR PASSIVES, from [51].	99
6.3	Example of UART output message content.	104
6.4	Voltage acquisition error validation results (all 6 cells)	106
6.5	Temperature acquisition error validation results (all 3 sensors)	106
6.6	Current acquisition error validation results	107
7.1	Relevant ISDT T8 charger specifications, from [54, 55].	110
7.2	Electrical model and logistic SOC inversion parameters used.	137
7.3	Initial point values.	140
7.4	Error metrics for all cells during rest and discharge conditions.	147
7.5	Final measured voltages and errors relative to the datasheet reference at 2200 mAh.	149

7.6	Manual estimation of R_{int} based on selected current steps.	155
7.7	Measured cell voltages during balancing sequence.	175
7.8	SOC values of all cells during the balancing process.	175
8.1	Summary of BMS objectives and implementation status.	178

List of Acronyms

AC	Alternating Current
ADC	Analog-to-Digital Converter
AMS	Accumulator Management System
ASPICE	Automotive SPICE
Ah	Ampere-hour
BMS	Battery Management System
CAN Bus	Controller Area Network Bus
CB	Cell Balancing
CBFET	Cell Balancing MOSFET
CC	Constant Current
CC-CV	Constant Current-Constant Voltage
CV	Constant Voltage
DC	Direct Current
DCM	Discontinuous Current Mode
ECU	Electronic Control Unit
EV	Electric Vehicle
FIFO	First-In-First-Out
FS	Formula Student
FSFEUP	Formula Student FEUP
FSG	Formula Student Germany
FST	Formula Student Técnico (Lisboa)
GPIO	General-Purpose Input/Output
HDMI	High-Definition Multimedia Interface
HMI	Human-Machine Interface
HWE	Hardware
isoSPI	Isolated Serial Peripheral Interface
KiBaM	Kinetic Battery Model
LED	Light-Emitting Diode
Li-ion	Lithium-ion
MBSE	Model-Based Systems Engineering
NTC	Negative Temperature Coefficient
NiCd	Nickel-cadmium
OCV	Open Circuit Voltage
P	Proportional
PCB	Printed Circuit Board
PI	Proportional-Integral
PID	Proportional-Integral-Derivative
SOC	State of Charge

SOH	State of Health
SWE	Software
SYS	System
TSAC	Tractive System Accumulator Container
UART	Universal Asynchronous Receiver-Transmitter
UI	User Interface
XDC	Xelerate Digitalcar

Chapter 1

Introduction

This chapter provides a general overview of the dissertation, introducing the motivation, scope, and objectives that guided the development of a custom Battery Management System (BMS) for a Formula Student (FS) prototype. It begins by outlining the context in which the project was conceived, including the growing relevance of electric mobility and the role of the Formula Student FEUP (FSFEUP) team and the partnership with Xelerate, where this thesis was conducted. The chapter also defines the practical and technical goals of the project, highlighting the academic and industrial collaboration that supported its execution. Finally, the structure of the thesis is presented, offering a clear roadmap of the chapters and their contents.

1.1 Problem and Context Definition

The shift toward sustainable and intelligent mobility has accelerated the development of electric vehicles (EVs), establishing them as a viable response to global environmental challenges and fossil fuel dependency. At the heart of these vehicles lies the battery system, whose management determines not only the performance and efficiency of the drivetrain, but also its safety, lifetime, and reliability under varying operational conditions. This evolution has brought BMS to the forefront of modern research and development efforts, particularly in applications that demand high adaptability and real-time decision-making.

A BMS serves as the intelligent layer that oversees the operation of the battery, while also protecting the system against unsafe conditions. As the demands on battery systems increase, particularly in dynamic environments such as electric motorsport, a new generation of flexible, reliable, and configurable BMS architectures is required.

Within this context, FS provides a real-world platform for engineering students to explore these technological challenges. It is a competition that blends technical design, performance validation, and business strategy in the development of fully functional race cars. Beyond the race track, however, it is also an opportunity to validate advanced embedded systems in a controlled but competitive environment, where constraints on safety, packaging, and communication reliability mirror those of industry applications.

This dissertation focuses on the design and implementation of a custom BMS prototype tailored to the FSFEUP electric vehicle, developed in collaboration with Xelerate Digitalcar. The work is part of a multi-phase strategy intended to give the FSFEUP team full control over their battery management strategy, replacing the limitations observed with commercial off-the-shelf solutions. This includes overcoming challenges related to physical integration, temperature acquisition, scalability across battery sizes, and lack of software-level transparency.

The central motivation of this project is to investigate how a custom-developed BMS can become a technically and strategically advantageous solution for the FSFEUP team. Rather than replicating existing systems, the goal is to study and validate how a tailored BMS can surpass the limitations of the commercial off-the-shelf unit previously adopted by the team, while also offering a competitive edge compared to other solutions used by rival teams in FS competitions. This work aims to demonstrate that a custom BMS, when carefully designed around the team's specific architecture, safety, configurability, and monitoring needs, can ultimately become the preferred option for future vehicle prototypes.

1.1.1 Formula Student Context

FS is an international engineering design competition held annually in various countries, designed to challenge university students in developing, building, and testing small-scale formula-style racing cars [2]. Organized by professional engineering institutions, the competition aims to foster innovation, technical excellence, and project-based learning by simulating a real-world engineering scenario where students assume the role of a design team working for a fictional company.



Figure 1.1: Formula Student events point distribution.

The competition spans a complete vehicle development cycle, from concept to manufacturing, under strict technical regulations and safety standards. Each participating team must ensure

that their vehicle complies with the technical rulebook, prioritizing not only performance and efficiency but also safety, manufacturability, and innovation. This process requires extensive interdisciplinary collaboration, time management, and validation strategies, closely mirroring the demands of professional motorsport engineering environments.

It culminates in a series of competitive events where teams showcase their vehicles and engineering rationale before a panel of judges composed of automotive experts, engineers, and industry professionals. These events are broadly divided into two categories: static and dynamic events. Static events focus on evaluating the design process, project management, and economic feasibility of the prototype. These include:

- **Engineering Design (150 points)** – Assesses innovation, engineering justification, and component selection.
- **Cost and Manufacturing Report (100 points)** – Evaluates cost optimization and feasibility of production.
- **Business Presentation (75 points)** – Requires teams to pitch their vehicle as part of a fictional business case to potential investors.

Dynamic events test the physical performance of the vehicle in various operational conditions, such as:

- **Endurance (250 points)** – The most critical event, assessing reliability and energy efficiency across a long-distance track.
- **Efficiency (100 points)** – Evaluates energy consumption throughout the endurance event.
- **Autocross (100 points)** – Measures handling and maneuverability over a short track.
- **Skidpad (50 points)** – Tests lateral grip and chassis setup via tight circular maneuvers.
- **Acceleration (50 points)** – Quantifies raw speed performance over a short straight distance.

These events collectively represent the core of the FS scoring system. While the main scoring breakdown adds up to 875 points, competitions are structured to total 1000 points. The remaining points may be assigned to optional bonus activities, penalties, or event-specific evaluations, depending on the host country's rulebook.

1.1.2 FSFEUP Team Overview

Founded in July 2021, FSFEUP is a young but ambitious team from the Faculty of Engineering of the University of Porto (FEUP), aiming to become a reference in electric vehicle prototyping and competition performance. Despite its recent creation, FSFEUP has shown remarkable growth and potential, currently composed of over 80 members from diverse engineering disciplines [3]. This

multidisciplinary approach is fundamental to tackling the complex challenges of vehicle development and competition readiness.

The team is structured into specialized departments, each responsible for a subsystem or aspect of the prototype:

- **Powertrain** – Responsible for the design and integration of the battery system, transmission, and cooling subsystems.
- **Electronics and Software** – Focuses on embedded software, control algorithms, low-voltage harnessing, and the development of electronic subsystems.
- **Chassis and Aerodynamics** – Manages the structural frame, aerodynamic package, and driver interface components.
- **Vehicle Dynamics** – Handles suspension geometry, simulation, kinematics, and vehicle control strategies.
- **Suspension** – Develops the wheel assemblies, steering mechanism, braking systems, and related hardware.
- **Autonomous Systems** – Dedicated to the driverless vehicle platform, including perception, state estimation, localization, and motion planning.
- **Research and Development (R&D)** – Promotes academic collaboration by integrating thesis students into real-world technical projects and innovation efforts.
- **Operations** – Oversees sponsorship management, logistics, marketing, and external communications.

Figure 1.2 shows the current FSFEUP team during a general meeting.



Figure 1.2: FSFEUP team members, 2025 edition.

The team's first prototype, FSFEUP01, was actively developed between 2021 and 2024 and competed in events such as Formula Student Germany (FSG) and Formula Student UK (FSUK). It provided a vital learning platform that exposed the team to the challenges of endurance, electrical reliability, and team logistics. FSFEUP01 is shown in Figure 1.3.



Figure 1.3: FSFEUP01 – The first prototype developed by FSFEUP.

The ongoing development of FSFEUP02 shifts the focus of the team toward innovation in autonomous driving technologies and refined electric architecture, based on lessons learned from FSFEUP01. Figure 1.4 presents the current FSFEUP02 design.



Figure 1.4: FSFEUP02 – Current development cycle prototype.

To date, FSFEUP has accumulated over 14 awards, partnered with more than 80 sponsors, and continues to expand its influence through strategic collaborations with academic institutions and industry leaders. These partnerships are crucial to supporting the technical and financial demands of each development cycle, allowing the team to pursue increasingly complex engineering challenges [4].

One such partnership is with Xelerate DigitalCar, a reference in automotive innovation and a key enabler of this dissertation's development. Their contribution not only reinforced the team's capabilities in embedded systems but also opened the door to a multi-phase collaboration focused on designing and validating a custom Battery Management System for the FSFEUP02 prototype.

1.1.3 Xelerate DigitalCar Context

Xelerate DigitalCar (XDC) is a forward-looking company operating at the intersection of mobility, electronics, and digital innovation. Supported by AVL Software and Functions GmbH (AVL), XDC develops cutting-edge solutions for automotive systems, focusing on embedded software, control strategies, and simulation frameworks that accelerate the development of smarter, safer, and more sustainable vehicles.

The partnership between XDC and FSFEUP stems from a shared commitment to advancing electric mobility through education and technological excellence. By providing access to industrial expertise, high-end tools, and dedicated mentorship, XDC empowers student-led projects like this one to tackle real-world engineering problems with professional rigor.

This dissertation was developed as the second stage of a long-term collaboration plan aimed at equipping FSFEUP with a fully tailored and competitive Battery Management System. During the first phase, the work concentrated on system modeling, simulation, and proof-of-concept validation. This phase focused on the design of custom hardware and the deployment of a fully functional prototype. In future work, the aim is to achieve a final version fully integrated into the next generation team race car. Through this ongoing partnership, XDC helps close the gap between academic projects and industry-grade engineering, reinforcing the team's ambition to compete at the highest level of FS.

1.2 Project Goals and Development Scope

The primary goal of this dissertation is to design, implement, and validate a custom BMS prototype tailored to the specific needs of the FSFEUP electric race car. More than just a technical challenge, this project aims to prove the feasibility and advantage of developing a fully customized BMS, as an alternative to relying on commercial off-the-shelf solutions. The motivation behind this approach stems from the desire to create a system that is not only cost-efficient and technically optimized but also scalable, configurable, and strategically aligned with the long-term objectives of the team.

The project scope was carefully defined to ensure meaningful outcomes within the available timeframe, while laying a strong foundation for future development. The present work represents

the second phase of a multi-year collaboration between FSFEUP and Xelerate DigitalCar, and it builds upon the proof-of-concept models created in the first phase. In this phase, the focus shifted to the design of a functional prototype that integrates real hardware acquisition and embedded software execution, capable of replicating industry-level logic and control.

To ensure an informed and structured approach, the initial phase of this project involved an in-depth study of the state of the art in BMS's. This included the analysis of modern BMS architectures, cell monitoring strategies, and functional requirements commonly adopted in electric vehicle applications. Based on this research, the system design was defined, and all component selections and implementation decisions were made according to a clear rationale aligned with performance, safety, and adaptability goals.

To guarantee a focused and traceable development process, the system was structured following Automotive SPICE (ASPICE) development standards and the V-model methodology. This ensured that each phase of the project could be directly linked to clear objectives and validation criteria. Accordingly, the functional architecture was decomposed into seven key modules: data acquisition, communication, state estimation, battery protection, diagnostics, user interface, and balancing.

Table 1.1: Summary of BMS objectives.

Objectives
Avoid centralized architecture
Comply with FS rules and requirements
Create a scalable BMS solution
Create a configurable BMS solution
Ensure low-error acquisition of voltages
Ensure low-error acquisition of temperatures
Ensure low-error acquisition of current
SOC estimation capabilities
SOH estimation capabilities
Safety protocols integrated and functional
Diagnostics integrated and functional
Current control algorithm for charging/discharging
Balancing control algorithm
HMI for external monitoring

These modules were developed and validated following a structured planning methodology, prioritizing features that most directly impact the safety and performance of the vehicle.

Rather than following a fixed template, the system was conceived to evolve with the team's needs, offering a flexible framework that can be scaled, adjusted, and improved in future prototypes. By combining structured methodology with practical experimentation, this work delivers both a functional solution and a replicable development process, reinforcing FSFEUP's ability to innovate and take ownership of its core vehicle systems.

1.3 Thesis Structure

This dissertation is organized into eight chapters, each contributing to a comprehensive understanding of the project's development and results.

Chapter 1 presents the introduction to the work, contextualizing the motivation behind the project, the relevance of BMS's in electric vehicles, and the partnership framework in which the project was developed. It also outlines the main goals and scope of the project.

Chapter 2 provides an overview of the state of the art in BMS technologies. It reviews fundamental concepts, current trends in battery modeling, estimation algorithms, and communication strategies, as well as analyzing existing commercial solutions and other teams solutions.

Chapter 3 presents the system architecture and development methodology adopted throughout the project. It begins by introducing the ASPICE framework and the V-model, which were used to structure the engineering process, ensuring traceability, modularity, and systematic validation. The chapter then details the planning strategy, including the functional decomposition of the BMS into development modules and the tools selected to support each phase of the project. A critical analysis of the commercial BMS previously used by FSFEUP is also included, highlighting its main strengths, limitations, and opportunities for improvement. Based on this assessment, the proposed custom BMS configuration is introduced, outlining its architecture, intended functionalities, and the rationale behind the selected components.

Chapter 4 focuses on the initial battery modeling phase, where key electrical models were simulated and validated using datasheet information and benchmark tools to establish reliable foundations for hardware-software integration.

Chapter 5 details the communication architecture between the master and slave, including Universal Asynchronous Receiver-Transmitter (UART) protocol configuration, command handling, and data decoding. The implementation of the real-time acquisition system is described, including the logic behind sampling routines, buffer management, and sensor data processing. It marks the transition from simulation to real hardware acquisition, validating whether the custom master controller could replicate the results previously obtained in Simulink.

Chapter 6 presents the full prototype integration and system validation, describing how all hardware components and software modules were combined, tested, and optimized to ensure stable and accurate acquisition under real conditions.

Chapter 7 is dedicated to the methodology and analysis of each individual test performed, with the objective of designing and validating the complete software and firmware stack responsible for the control logic of the BMS. It explains the objectives, procedures, and outcomes of every experimental stage, offering a critical evaluation of the system's performance and robustness.

Chapter 8 concludes the dissertation by summarizing the main findings, discussing the impact of the developed solution, and providing suggestions for future improvements and potential directions for continued research and application within the FSFEUP project.

Chapter 2

State of the Art

This chapter provides a comprehensive and technically structured overview of the current state of the art in BMS, with a particular focus on their relevance to electric vehicles and competitive environments. It begins by outlining the historical evolution of BMS and the progressive increase in system complexity driven by advancements in battery technologies. The chapter then examines the main topologies used in BMS design, comparing their topologies, functionalities, and trade-offs. It then delves into essential key features such as state estimation methods for determining State of Charge (SOC) and State of Health (SOH), battery modeling approaches, current regulation strategies during charging and discharging phases, fault diagnosis techniques, and cell balancing strategies are explored in depth. Additionally, the chapter addresses representative BMS solutions implemented by leading FS teams. This comprehensive review serves as the technical foundation for the design decisions made in the present work.

2.1 Battery Management Systems (BMS)

Through the research, it became evident that there is no single, universally accepted definition of a Battery Management System (BMS), as its design varies significantly depending on the specific requirements and applications. Although often hidden from the end user, the BMS plays a critical role in ensuring the safe and efficient operation of the battery. Fundamentally, a BMS acts as the protector of the battery, prioritizing its safety and preserving user well-being [5].

The core functions of a BMS will be explored in greater detail in the following sections of this chapter. However, its primary responsibilities include monitoring the battery by measuring key parameters such as current, voltage, and temperature, safeguarding the battery from potential hazards, estimating its operational state, optimizing performance, and facilitating communication with the user and external systems. These functions can be carried out using a variety of methods with different levels of complexity, which ultimately define the intelligence and sophistication of the BMS [5].

2.1.1 Evolution Over the Time

The development of BMS's has closely followed advancements in rechargeable battery technology, which began gaining prominence in the late 19th and early 20th centuries. During this period, lead-acid batteries were the dominant technology. Basic forms of battery management were employed, primarily focusing on voltage regulation to prevent overcharging, which could lead to permanent damage to the battery [6].

In the 1960s, the introduction of nickel-cadmium (NiCd) batteries brought about new challenges. These batteries had specific charging and discharging profiles, necessitating more advanced management techniques. It was during this era that the concept of overcharge protection mechanisms emerged, laying the foundation for modern BMS designs.

A major leap occurred in the 1990s with the commercial release of lithium-ion (Li-ion) batteries. Offering higher energy densities than previous technologies, Li-ion batteries posed significant risks, such as overcharging and thermal runaway, which demanded the development of sophisticated BMS. These systems provided a higher level of monitoring and control to ensure safe operation, making them indispensable in applications where energy density and safety were paramount [6].

In the 21st century, the rise of electric vehicles (EVs) and the growing demand for renewable energy storage solutions significantly expanded the role of BMS. As illustrated in Figure 2.1, the widespread adoption of EVs drove the need for more complex BMS functionalities. Modern systems now feature cell balancing, battery state estimation (such as SOC and SOH), fault diagnostics, and enhanced communication capabilities. These systems are now integral to a wide array of applications, ranging from portable electronics to large-scale energy storage [7].

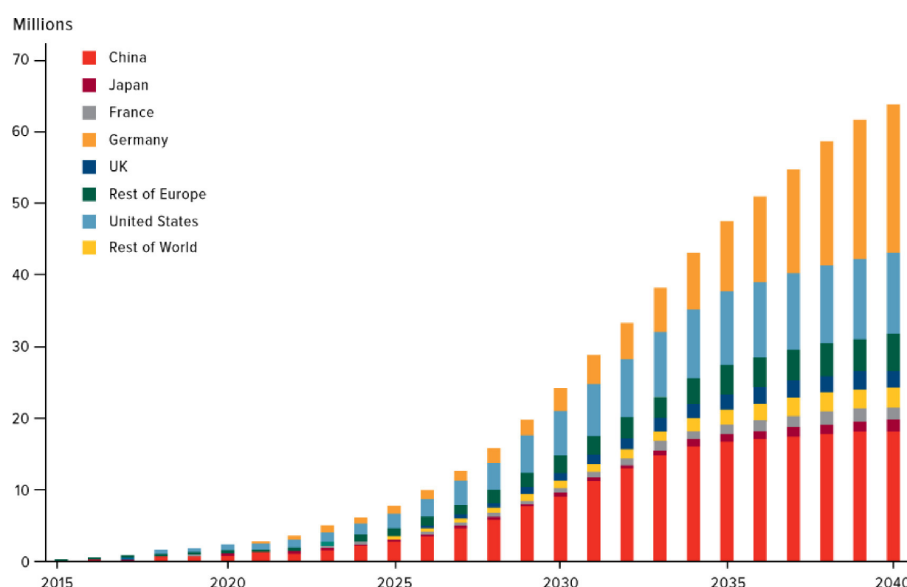


Figure 2.1: Annual demand forecast for EVs in the world, from [7].

2.2 BMS Topologies

BMS topology is defined by the relationship between the system's key components (the monitoring module and the control module) and how they are arranged within the system [8]. The primary distinction in BMS topologies is between distributed, non-distributed and hybrid topologies. Non-distributed topologies are further categorized into centralized and decentralized structures, with the latter including master-slave and modular configurations, as illustrated in Figure 2.2.

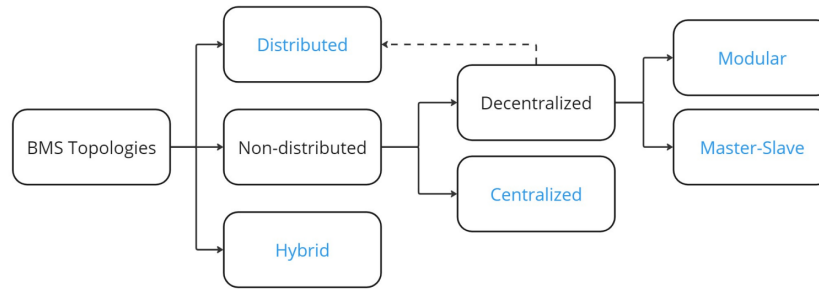


Figure 2.2: BMS topologies.

2.2.1 Distributed Topology

The distributed BMS topology operates with electronics, known as cell boards, directly installed on the battery cells themselves. This distributed approach allows each cell board to be responsible for both monitoring and controlling its specific cell. The cell boards are interconnected, typically through a wired network, facilitating communication and decentralized control across the entire system. This design offers significant advantages in terms of functionality, as it enables precise monitoring and control of each individual cell. By distributing control throughout the system, distributed BMS topologies allow for high scalability. The system can be easily expanded by adding more cells, without needing substantial changes to the core topology [8].

Moreover, this topology is more fault-tolerant because the control responsibilities are shared across multiple nodes. In larger and more complex implementations, a central controller can be added to manage the communication and computation tasks across multiple distributed nodes, further enhancing the system's robustness. However, distributed BMS systems tend to be more expensive due to the need for additional electronics on each cell. Each cell board requires its own set of sensors, controllers, and communication ports, which not only increases cost but also introduces potential complexity in managing multiple communication lines. Despite this, the reduced wiring requirement in a daisy-chain configuration offsets some of the complexity, making it a versatile and scalable solution for large battery systems.

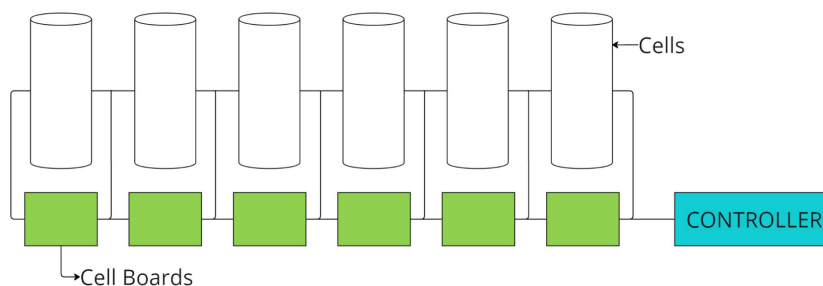


Figure 2.3: Distributed BMS topology, adapted from [8].

2.2.2 Non-distributed Topology

In contrast to the distributed topology, where control and measurement responsibilities are spread across multiple individual cell boards, the non-distributed topology centralizes control within a single module, often referred to as the master module. The following topologies differ from each other in the relationship between the control module and the measurement module.

2.2.2.1 Decentralized Topology

The decentralized topology has an architecture composed by multiple modules, in which one of them is considered the master controlling the rest of the modules. In this perspective, and as seen in Figure 2.2, the distributed topology may also be considered a decentralized topology.

Modular:

The modular BMS topology introduces a decentralized topology where several identical modules are connected to the battery pack. Each module functions as an autonomous unit, responsible for acquiring data and performing measurements for its assigned group of cells. One of these modules is designated as the master, which oversees the other modules, managing data communication and control. This topology allows for greater reliability because if the master module fails, control can be transferred to another module, thus ensuring continuous system operation.

The modular topology excels in terms of scalability. Adding additional modules allows the system to expand easily to accommodate larger battery packs without significantly increasing complexity. This makes it a preferred choice for applications that require flexibility and the potential for future expansion. However, the increased number of modules introduces greater complexity in managing the interaction between them, particularly in assigning and coordinating the master's role. The system also tends to be more expensive than a centralized topology due to the need for additional hardware components in each module.

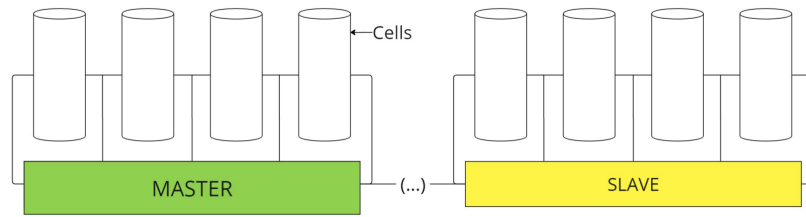


Figure 2.4: Modular BMS topology, adapted from [8].

Master-Slave:

In the master-slave topology, the roles of data acquisition and control are clearly divided. The slave modules are responsible exclusively for collecting data from the battery cells, while the master module handles all control, communication, and decision-making functions. This separation simplifies the system's overall topology by reducing the burden on the slave modules, allowing them to focus solely on gathering information. The master module processes the data and coordinates the actions of the slave units [8].

This design is similar to the modular topology but with a key difference: the master module is irreplaceable. If the master fails, the system loses its ability to control the battery, which reduces the overall reliability compared to a modular setup where the master's responsibilities can be transferred to another module. However, this simplification results in a less complex control structure, making it easier to manage in some applications. The master-slave topology retains scalability, as additional slave units can be added as needed, but it remains less flexible in terms of fault tolerance. The streamlined nature of the master-slave configuration also helps to reduce the complexity of managing multiple modules, making it an efficient solution where the risk of master failure is not a critical concern.

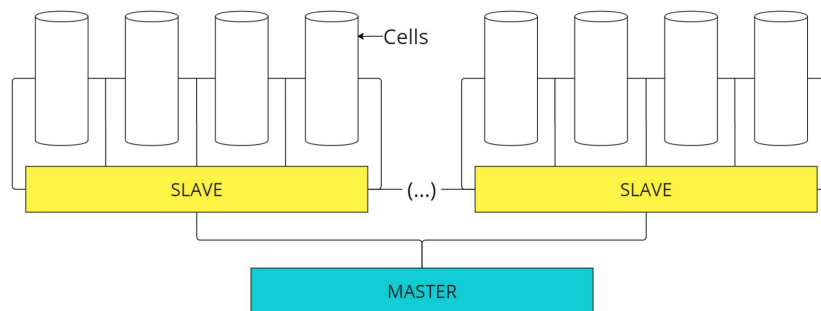


Figure 2.5: Master-Slave BMS topology, adapted from [8].

2.2.2.2 Centralized Topology

In a centralized BMS topology, all the control and monitoring tasks are performed by a single central unit, which is responsible for measuring the parameters of each cell and controlling the

system as a whole. The central module consolidates both the monitoring and control functions into one unit, simplifying the system's overall design. This approach offers a more cost-effective solution, as fewer electronic components are required. The system's simplicity also makes it easier to manage, with lower maintenance needs and a more compact form factor compared to distributed or modular designs.

However, centralized topologies come with certain limitations. One of the most significant challenges is the need for extensive wiring to connect every battery cell to the central controller. This increases the risk of misconnections and can reduce the system's overall reliability. Furthermore, centralized topologies have limited scalability, making it difficult to adapt the system to larger configurations or expand it in the future without major redesigns. Despite these disadvantages, centralized BMS designs are well-suited for applications where simplicity and cost are more critical than scalability.

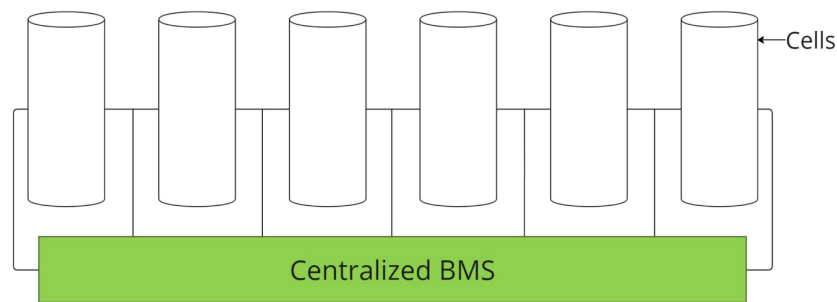


Figure 2.6: Centralized BMS topology, adapted from [8].

2.2.3 Hybrid Topology

Hybrid BMS topologies combine features from multiple topologies to offer a flexible and adaptable solution. By integrating characteristics from distributed, centralized, and modular designs, hybrid systems can be tailored to meet specific operational requirements[8]. For example, a hybrid system might use a distributed approach for high-precision cell monitoring, while adopting a centralized or master-slave configuration for overall control and communication. This allows for the best of both worlds: the scalability and fault tolerance of distributed topologies with the cost efficiency and simplicity of centralized control.

Hybrid topologies are often deployed in applications that require high adaptability and optimization based on the specific use case. This flexibility comes at the cost of increased design complexity, as multiple control strategies must be integrated and managed within a single system. Additionally, hybrid solutions can be more expensive due to the need to accommodate different architectural components. However, the benefits of enhanced scalability, reliability, and the ability to customize the system to meet evolving needs often outweigh these drawbacks, especially in complex and large-scale battery systems. An example of this hybrid adoption is the FST team's BMS.

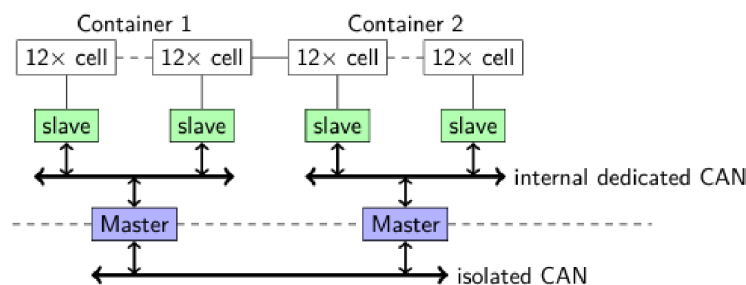


Figure 2.7: FST Hybrid BMS topology, from [9].

2.2.4 Topologies Comparison

The choice of topology is an important decision, as it directly impacts key aspects such as system cost, reliability, complexity, and scalability [8]. Table 2.1 summarizes the main characteristics to be considered in the overall choice of solution and the choice of topology should align with the specific operational requirements and long-term goals of the system, considering the trade-offs between performance, cost, and adaptability.

Table 2.1: Comparison of BMS topologies, adapted from [8].

BMS Topology	Reliability	Scalability	Electronics Cost	Complexity
Distributed	High	High	High	High
Centralized	Medium	Low	Low	Low
Modular	High	High	Medium	High
Master Slave	Medium	High	Medium	Medium

2.3 BMS Features and Functionalities

As mentioned before, the complexity of a BMS can vary, from basic systems to more advanced architectures, depending on the functionalities and components used.

The system is normally composed of two key modules: the monitoring module and the control module. The monitoring module is responsible for gathering vital data from the battery, such as cell voltage and temperature and battery current. This monitoring function is fundamental, as the other BMS features rely on these parameters. The control module processes the collected data and manages the battery system, providing functionalities such as diagnostics, charge and discharge control, and state estimation. It also ensures communication between various system components and typically has a Human-Machine Interface (HMI) [10].

In addition to monitoring, a BMS may provide other crucial functions, such as cell balancing, SOC estimation, and SOH estimation, which are key to the longevity and efficiency of the battery. Advanced BMS systems also implement techniques for controlling current limits during both charge and discharge processes, as well as fault diagnosis methods [11].

Figure 2.8 summarizes the main functionalities of a BMS, illustrating the wide range of tasks, from essential parameter measurement to data management and communication.

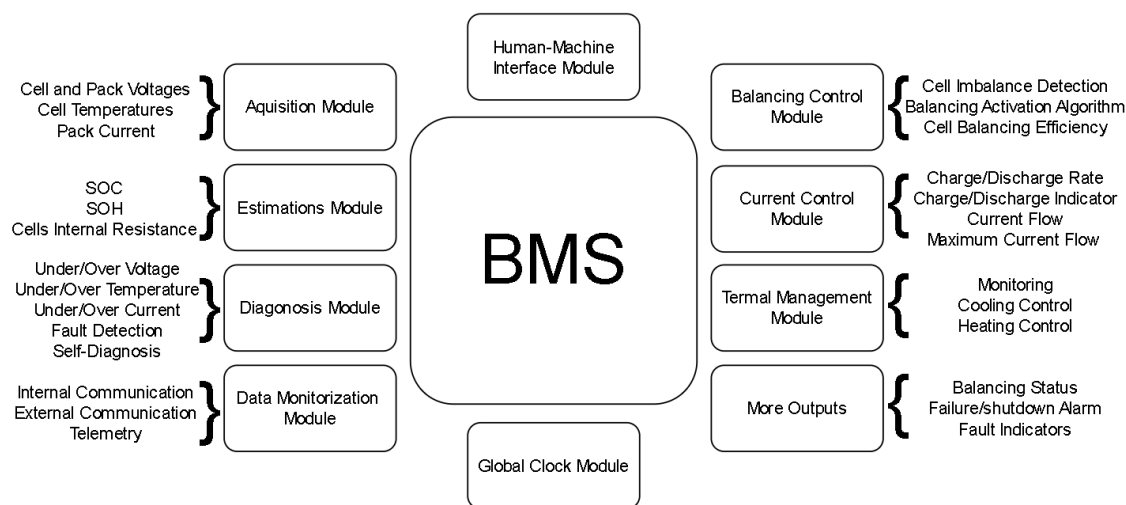


Figure 2.8: BMS main functionalities, adapted from [12].

2.4 State Estimation Methods

As previously discussed, one of the core responsibilities of a Battery Management System (BMS) is to estimate the current state of the battery with high accuracy. Studies made it clear that various techniques have been developed to assess different aspects of the battery's condition [5], the most commonly referenced being SOC, SOH, State of Power (SOP), State of Energy (SOE), and State of Safety (SOS) [13].

Given the highly nonlinear and dynamic behavior of lithium-ion batteries, particularly under varying operational conditions, it is critical not only to estimate internal states but also to accurately predict the battery's response in real-time [13]. To support this, BMS implementations depend heavily on battery models that can simulate and represent the electrochemical and electrical behavior of the cells.

This chapter starts by focuses specifically on SOC and SOH estimation, two of the most essential indicators for battery monitoring and control. The following sections introduce their theoretical definitions, present the most widely used estimation techniques, and analyze their advantages and limitations. Furthermore, different types of battery models are described and compared, highlighting which modeling approaches are best suited for integration into BMS solutions based on application requirements.

2.4.1 SOC Estimation Methods

Although the SOC is conceptually straightforward, representing the ratio of the battery's available charge to its maximum capacity, it cannot be directly measured due to its dependence on the internal charge concentration [14]. Instead, SOC necessitates the use of estimation methods based on measurable parameters such as voltage, current, and temperature[15]. This section outlines several methodologies employed for SOC estimation.

2.4.1.1 Coulomb Counting Method

The Coulomb counting method, often referred to as the Ampere-hour (Ah) counting method, is one of the most widely employed techniques for SOC estimation. The principle underlying this method involves integrating the current entering or leaving the battery over time to determine the change in charge [15]. The SOC at any given time t can be expressed as:

$$SOC(t) = SOC(t_0) - \frac{1}{C_n} \int_{t_0}^t \eta I(t) dt \quad (2.1)$$

where $SOC(t_0)$ represents the initial SOC, C_n denotes the rated capacity of the battery, $I(t)$ is the instantaneous current (negative during charging and positive during discharging), and η is the Coulombic efficiency, defined as the ratio of the charge delivered during discharge to the charge stored during charging.

Despite its widespread use due to its simplicity and ease of implementation, this method is not without limitations. The accuracy of Coulomb counting is highly dependent on the precision of the initial SOC value, which can be challenging to determine accurately [14]. Furthermore, errors in current measurement can accumulate over time, particularly under conditions of fluctuating loads or elevated temperatures, leading to significant deviations from the actual SOC.

2.4.1.2 Open Circuit Voltage Method

The open circuit voltage (OCV) method relies on the well-established relationship between SOC and the battery's open circuit voltage. When the battery is allowed to rest for a sufficient duration, its terminal voltage stabilizes to the OCV. This voltage can then be mapped to the SOC using a predefined OCV-SOC curve, which is typically unique to the battery's chemistry and state of health. Figure 2.9 illustrates a typical OCV-SOC curve for a li-ion battery.

While this method is highly accurate during periods of rest, it is impractical for real-time SOC estimation during active battery operation due to the long relaxation periods required for voltage stabilization. Consequently, the OCV method is often combined with other dynamic estimation techniques to enhance accuracy under operational conditions.

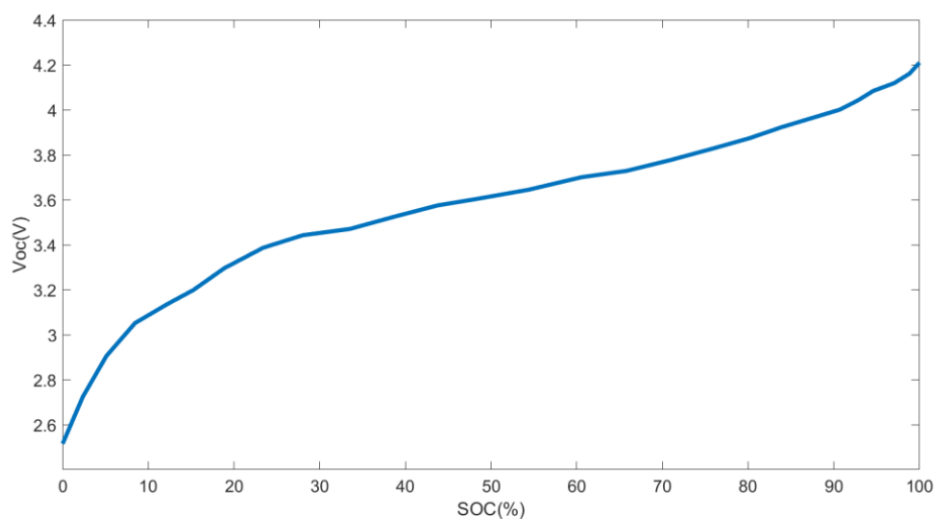


Figure 2.9: Typical OCV-SOC curve for a Lithium-Ion battery.

2.4.1.3 Model-Based SOC Estimation

Recent advancements in battery modeling have facilitated the development of sophisticated model-based SOC estimation techniques [14]. These methods utilize mathematical models to represent the dynamic behavior of the battery, incorporating parameters such as terminal voltage, internal resistance, and temperature. By continuously updating the model parameters through real-time data, these techniques can provide accurate SOC estimations under varying operational conditions.

One prominent approach within this category involves the use of equivalent circuit models, which approximate the battery's behavior using a combination of resistors and capacitors. These models are computationally efficient and can be easily adapted to various battery chemistries [14]. However, their accuracy depends on the fidelity of the parameter identification process.

2.4.1.4 Adaptive Filtering Techniques

Adaptive filters, derived from control theory, have emerged as powerful tools for SOC estimation, particularly in addressing the challenges posed by noise and model inaccuracies. Among these, the Kalman filter and its variants, such as the Extended Kalman Filter and the Unscented Kalman Filter, are widely used [16]. These filters operate by predicting the system's state based on previous measurements and subsequently updating the prediction using new observations [16]. This iterative process ensures convergence toward the true SOC value, even in the presence of significant measurement noise.

Another advanced technique is the Particle Filter, which leverages a probabilistic framework to estimate SOC in nonlinear and non-Gaussian systems [16]. Although computationally intensive, the PF provides robust performance in scenarios where other filtering techniques may falter.

2.4.1.5 Neural Network-Based Estimation

The application of neural networks in SOC estimation has gained traction due to their ability to model complex nonlinear relationships inherent in battery systems[17]. By training on historical data, neural networks can predict SOC with high accuracy, even under dynamic operating conditions. However, this approach requires extensive datasets for training and is computationally demanding, making it more suitable for offline analysis or systems with substantial processing capabilities[17].

2.4.1.6 Comparison of SOC Estimation Methodologies

The selection of an appropriate SOC estimation method depends on the specific requirements of the application, including accuracy, computational efficiency, and real-time applicability. While traditional methods such as Coulomb counting and OCV provide foundational approaches, advanced techniques like adaptive filters and neural networks offer enhanced performance for modern, dynamic battery systems. The integration of these methods within a BMS ensures reliable and efficient battery operation, aligning with the demands of energy storage applications.

The Table 2.2, provides a detailed comparison of SOC estimation methods.

Table 2.2: Comparison of SOC estimation methods, adapted from [14, 15, 16].

Method	Implementation	Accuracy	Real-Time Applicability	Cost
Coulomb Counting	Easy	Low	Yes	Low
OCV Method	Medium	High (at rest)	No	Low
Battery Model Based	Medium	High	Yes	Moderate
Kalman Filter	Medium	High	Yes	Moderate
Particle Filter	Difficult	High	Yes	Very High
Neural Networks	Difficult	Very High	Yes	Very High

2.4.2 SOH Estimation Methods

The SOH is a critical parameter used to assess the overall condition and aging of a battery. Similar to SOC, SOH must be estimated through indirect methods. It analyze changes in key battery properties such as capacity, internal resistance, and self-discharge rate. These estimations provide insight into the battery's remaining useful life and performance capabilities [14].

2.4.2.1 Cycle Counting Method

This straightforward approach tracks the number of complete charge and discharge cycles the battery has undergone. Each cycle represents a full energy discharge followed by a recharge [13, 18].

While simple to implement, this method assumes a predictable degradation pattern with each cycle, which may not hold true under varying operational conditions, as it is possible to see in the Figure 2.10. For instance, factors such as irregular current loads or temperature fluctuations can significantly impact the degradation rate, reducing the accuracy of this method.

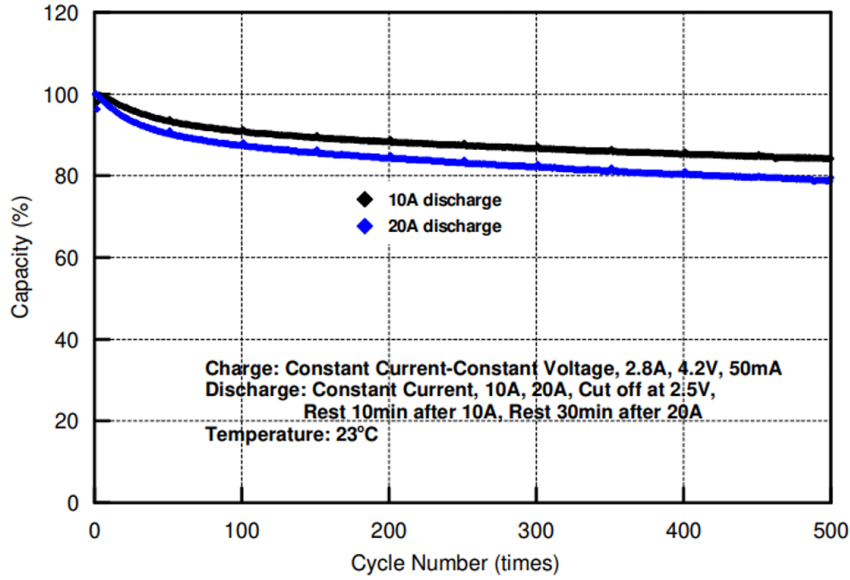


Figure 2.10: Cycle characteristics from Molicel model INR-18650-P28A with different current discharge conditions.

Despite its limitations, cycle counting provides a simpler starting point for SOH estimation in scenarios with consistent usage patterns.

2.4.2.2 Capacity-Based Estimation

Capacity measurement is one of the most reliable methods for determining SOH, particularly under controlled conditions. This method involves fully charging the battery, followed by a complete discharge at a constant current rate until reaching the cutoff voltage [13, 18]. The resulting discharge capacity is then compared to the nominal capacity, with the SOH calculated as:

$$SOH = \frac{CurrentCapacity}{NominalCapacity} \times 100 \quad (2.2)$$

While accurate, this method is not feasible in real-time applications due to its dependence on stable, controlled conditions. It is primarily used in laboratory environments to validate other estimation techniques or to assess the initial health of a battery [13].

2.4.2.3 Internal Resistance Monitoring

Monitoring the internal resistance of a battery is one of the most widely adopted techniques for estimating the SOH, as resistance tends to increase consistently with cell aging due to electrochemical and structural degradation processes [18, 19].

The standard approach involves applying a controlled current pulse and observing the corresponding voltage response. Using Ohm's Law, the internal resistance (R_{int}) is calculated as:

$$R_{int} = \frac{\Delta U}{\Delta I} \quad (2.3)$$

Here, ΔU represents the voltage drop, and ΔI is the applied current change [13]. The Figure 2.11 shows the voltage variations when there is applied charge/discharge current pulses.

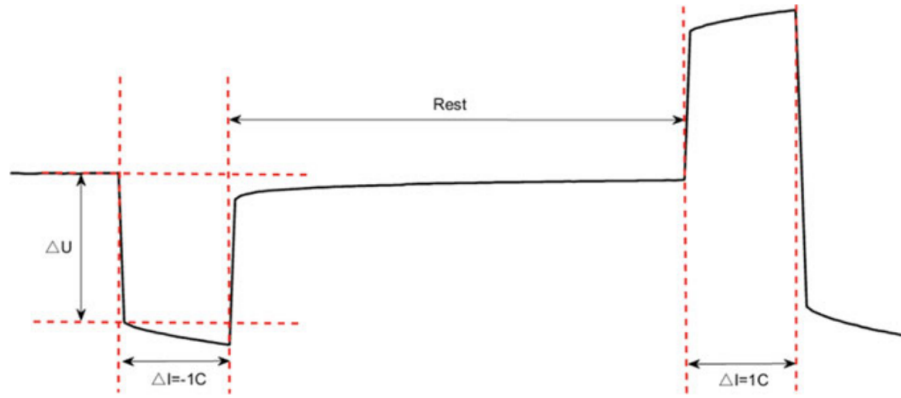


Figure 2.11: Battery voltage dynamics under charge/discharge current pulses, from [13].

As the cell ages, the internal resistance increases due to electrode degradation, electrolyte decomposition, and the growth of the solid electrolyte interphase layer. This increase in resistance leads to higher voltage drops under load, reduced power capability, and increased heat generation, which accelerates battery degradation. With this behavior, it is possible to estimate the SOH using the relationship between the internal resistance of a new cell and that of the aged cell. The SOH can be expressed as:

$$SOH = \frac{R_{new}}{R_{aged}} \times 100 \quad (2.4)$$

where R_{new} represents the internal resistance of a fresh cell, and R_{aged} corresponds to the resistance measured at a given point in the battery's life cycle. Since internal resistance typically increases as the battery deteriorates, SOH decreases accordingly, providing a quantifiable metric for battery aging. This method is particularly effective because resistance growth follows a predictable trend over the battery's lifetime, allowing for degradation tracking and predictive maintenance. However, since temperature significantly affects internal resistance, compensation mechanisms must be employed to ensure accurate SOH estimations, as variations in temperature can lead to temporary fluctuations in measured resistance. This method offers a balance between implementation complexity and real-time applicability, making it well-suited for BMS integration.

2.4.2.4 Model Based Estimation

Model-based methods leverage mathematical representations of battery behavior to estimate SOH. These models integrate data from parameters such as capacity, internal resistance, and self-discharge to provide comprehensive insights into battery health. Techniques such as equivalent circuit models and adaptive filters, including Kalman and Particle Filters, enhance the accuracy of these estimations by updating model parameters in real-time based on observed data [18, 19]. While computationally intensive, model-based methods offer high accuracy and are particularly valuable for applications requiring dynamic monitoring and real-time adjustments.

2.4.2.5 Comparative Analysis of SOH Estimation Methods

The selection of an appropriate SOH estimation method depends on factors such as implementation complexity, accuracy, real-time applicability, and computational requirements. Table 2.3 summarizes the key characteristics of each method.

Table 2.3: Comparison of SOH estimation methods, [13, 18, 19].

Method	Implementation Complexity	Accuracy	Real-Time Application	Computational Cost
Cycle Counting	Low	Low	No	Low
Capacity Measurement	Moderate	High	No	Moderate
Internal Resistance	Moderate	High	Yes	Moderate
Model-Based	Moderate	Very High	Yes	High

In conclusion, each SOH estimation method offers distinct advantages tailored to specific use cases. While simpler methods like cycle counting are suitable for less dynamic applications, model-based approaches provide the highest accuracy and adaptability, making them indispensable for real-time BMS's.

2.4.3 Battery Models

Battery models are mathematical representations that describe the electrical and physical behavior of a battery under various operating conditions. These models are fundamental tools in BMSs, particularly for estimating internal states such as SOC and SOH. An accurate battery model enables the BMS to predict performance, optimize charging and discharging strategies, and prevent unsafe conditions such as overcharging or deep discharging.

Several types of battery models exist, each differing in complexity, computational requirements, and application suitability. Figure 2.12 presents an overview of the main model categories, including electrochemical, empirical, stochastic, mathematical, electrical, analytical, and hybrid approaches.

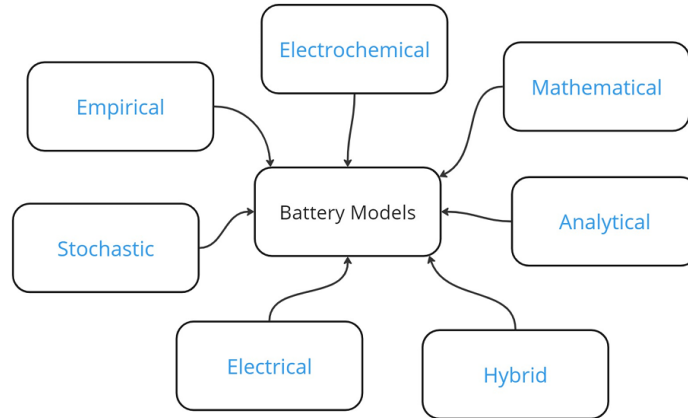


Figure 2.12: Different types of battery models, adapted from [20].

2.4.3.1 Electrochemical Models

Electrochemical models are based on the fundamental chemical reactions occurring within the battery. They offer high accuracy and predictive power but are computationally intensive and require a large set of parameters, many of which are difficult to obtain experimentally. Models like the DualFoil simulate ion diffusion, reaction kinetics, and phase changes in detail, making them ideal for research but impractical for real-time BMS applications due to their complexity and high computational load [20].

2.4.3.2 Empirical Models

Empirical models rely on experimentally derived equations to describe battery behavior. A widely known example is Peukert's Law, which expresses the available capacity as a function of the discharge current:

$$C = L \cdot I^\alpha \quad (2.5)$$

where C is the effective capacity, I the current, L the discharge time, and α a constant that varies with temperature and aging [14, 20]. While easy to implement, empirical models lack adaptability to dynamic conditions and are generally unsuitable for systems with variable loads, such as electric vehicles [20].

2.4.3.3 Stochastic Models

Stochastic models employ probabilistic techniques to characterize battery behavior. For instance, Markov chain models describe the battery as a system transitioning between discrete energy states based on probability distributions. Although they offer insights into qualitative trends like degradation and recovery, they are inadequate for precise voltage or current prediction, limiting their usefulness in control-focused BMS applications [14, 20].

2.4.3.4 Electrical Models

Electrical models simulate battery behavior using equivalent circuits composed of voltage sources, resistors, and capacitors. These models offer an intuitive and practical approach to capturing key battery dynamics and are particularly well-suited for real-time applications in BMSs due to their relatively low computational requirements.

Rint Model:

The Rint (resistance-internal) model is the simplest representation, consisting of an ideal voltage source in series with a single resistance R_o . This configuration accounts for the internal ohmic losses in the battery and models the instantaneous voltage drop that occurs when a current is applied.

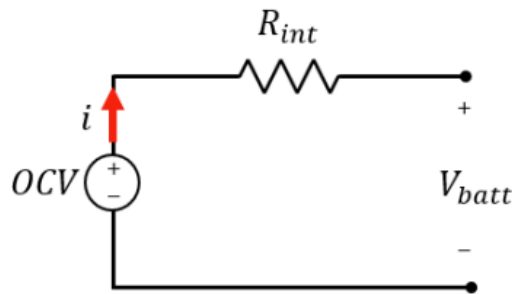


Figure 2.13: Simple battery model, from [21].

Although computationally efficient and easy to implement, the Rint model fails to capture the transient behavior of real batteries, such as relaxation effects after load changes or dynamic voltage delay.

Thevenin Model:

To address the limitations of the Rint model, the Thevenin model introduces a parallel resistor-capacitor (R/C) network in series with the internal resistance. This configuration better captures the first-order transient response of the battery to current variations and allows modeling of voltage relaxation and diffusion effects.

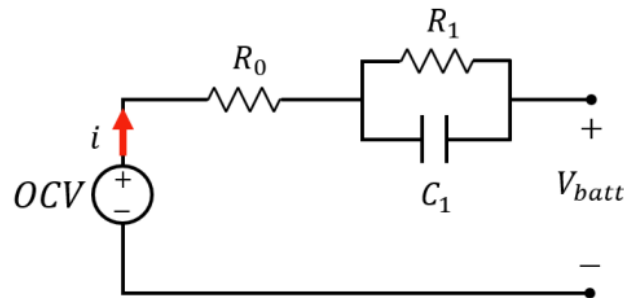


Figure 2.14: Thevenin equivalent electrical circuit (1RC), from [21].

The additional RC branch provides a more accurate description of battery dynamics during load changes while maintaining manageable complexity, making it a common choice in EV applications and academic research.

Dual Polarization (DP) Model:

The Dual Polarization model further enhances the Thevenin structure by incorporating multiple RC branches, typically two or more, each representing distinct polarization mechanisms such as electrochemical and concentration effects. This multi-branch configuration allows the model to reproduce a broader range of dynamic responses, including long-term relaxation and non-instantaneous voltage shifts.

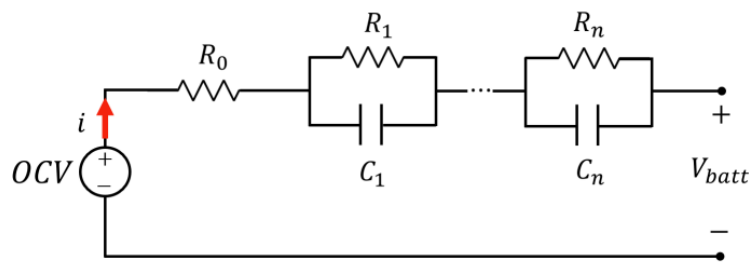


Figure 2.15: n-th-order generalized circuit, from [21].

By adjusting the number and values of the RC branches, the DP model offers a tunable trade-off between accuracy and computational load. It is particularly effective in applications requiring high-fidelity simulation of voltage behavior across varying current profiles. For this reason, it is widely recognized in the literature as a strong compromise between model simplicity and precision, and is commonly adopted in electric vehicle BMS implementations [21].

2.4.3.5 Mathematical Models

Mathematical models describe the voltage behavior of batteries as explicit functions of measurable parameters such as charge/discharge current, SOC, and time. These models are developed by fitting experimental data into predefined analytical expressions, aiming to approximate the dynamic behavior of the battery while maintaining a manageable computational burden. Their primary advantage lies in the ability to estimate both the terminal voltage (V_{out}) and the open-circuit voltage (V_{oc}) through relatively simple formulations, making them suitable for integration into real-time BMS algorithms [20].

Among the most widely adopted mathematical models are the Shepherd model, the Unnewehr Universal model, and the Nernst model. Each model captures the voltage response of the battery during charge and discharge using different levels of complexity and physical representation.

Shepherd Model:

The Shepherd model expresses the terminal voltage as a function of internal resistance and the inverse of SOC, representing the drop in voltage as the battery discharges. It is defined by:

$$V_{out} = E_0 - R_s \cdot I - \frac{k}{SOC} \quad (2.6)$$

$$V_{OC} = E_0 - \frac{k}{SOC} \quad (2.7)$$

where E_0 is the battery's constant voltage, R_s is the internal series resistance, I is the current, and k is an empirical fitting constant. This model reflects the steep voltage decline at low SOC levels but is sensitive to variations in current and may become unstable near full depletion.

Unnewehr Universal Model:

A simplified alternative to the Shepherd model, the Unnewehr Universal model represents the voltage as a linear function of SOC, offering smoother performance at moderate computational cost:

$$V_{out} = E_0 - R_s \cdot I - k \cdot SOC \quad (2.8)$$

$$V_{OC} = E_0 - k \cdot SOC \quad (2.9)$$

While this model lacks the ability to capture non-linear voltage drops seen at extreme SOC values, it performs adequately in mid-range SOC applications and can be useful when real-time implementation constraints are present.

Nernst Model:

The Nernst model, grounded in electrochemical theory, incorporates logarithmic terms to more accurately reflect the non-linear dependence of voltage on SOC:

$$V_{out} = E_0 - R_s \cdot I + k_2 \ln(SOC) + k_3 \ln(1 - SOC) \quad (2.10)$$

$$V_{OC} = E_0 + k_2 \ln(SOC) + k_3 \ln(1 - SOC) \quad (2.11)$$

This model is particularly effective at representing both the initial voltage drop and the sharp rise near full charge, which makes it suitable for applications requiring a more detailed voltage response. However, its implementation requires caution, as numerical instability may arise when SOC approaches 0 or 1.

Polynomial and Exponential Fitting:

Beyond these classical models, modern applications often rely on polynomial or exponential curve fitting techniques to model the V_{oc} , SOC relationship with higher precision. One such empirical formulation was proposed in [14, 20], and consists in the following equation:

$$V_{OC}(SOC) = a \cdot \exp(1 - b \cdot SOC) + c \cdot SOC^3 + d \cdot SOC^2 + e \cdot SOC + f \quad (2.12)$$

where a , b , c , d , e , and f are fitted coefficients obtained through regression from experimental data. This model combines the exponential behavior observed at low SOC with polynomial behavior across the remaining charge range, offering high flexibility in capturing realistic voltage profiles.

2.4.3.6 Analytical Models

Analytical models, such as the Kinetic Battery Model (KiBaM) or Rakhmatov and Vrudhula's diffusion model, simulate internal charge dynamics using physical analogies and differential equations. These models are capable of accurately capturing important phenomena such as the recovery effect and the influence of discharge current intensity on battery behavior. Although their implementation is more complex than empirical or mathematical models, due to the need for solving differential equations and performing precise parameter estimation, they offer high accuracy and predictive capability, making them particularly suitable for advanced BMS applications that require detailed insight into the battery's internal processes [22, 23, 24].

Kinetic Battery Model (KiBaM)

The KiBaM, introduced by Manwell and McGowan in 1993, provides an intuitive representation of internal charge behavior by modeling the battery as two interconnected reservoirs (or "tanks"):

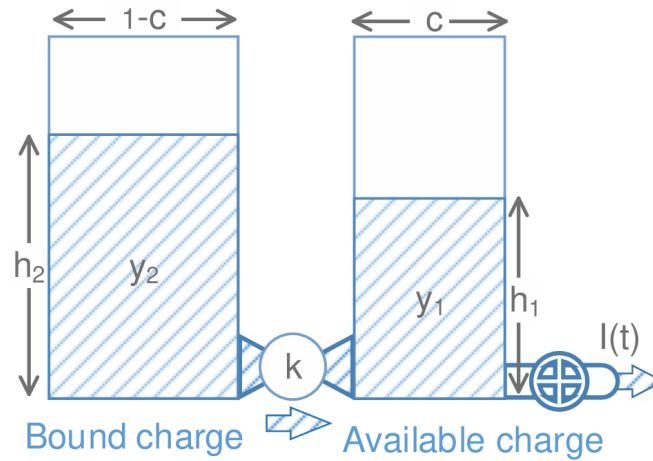


Figure 2.16: Conceptual illustration of the KiBaM, adapted from [23].

$y_1(t)$: the available charge, which is the portion of the battery immediately accessible for supplying or receiving current. During discharge, this is the reservoir that directly powers the load. During charging, it is the first to accumulate energy from the input current.

$y_2(t)$: the bound charge, which exchanges energy with the available charge through a transfer mechanism governed by the conductance parameter k . In discharge conditions, it gradually replenishes $y_1(t)$ as it depletes. In charging conditions, it absorbs part of the incoming energy from $y_2(t)$ over time, modeling delayed internal diffusion.

The flow between the tanks depends on the “height” of charge in each tank, represented by $h_1(t)$ and $h_2(t)$ and is controlled by a conductance constant k [20, 24]. The dynamic behavior of the system is governed by the following differential equations:

$$\frac{dy_1(t)}{dt} = -i(t) + k(h_2(t) - h_1(t)), \quad \frac{dy_2(t)}{dt} = -k(h_2(t) - h_1(t)) \quad (2.13)$$

As shown in Figure 2.16, the model captures the recovery effect, where after intense discharge, the bound charge slowly replenishes the available charge during rest periods. This mechanism reflects the real-world observation that a battery’s usable capacity temporarily increases after a pause in use. Additionally, the model accounts for the rate-capacity effect, meaning that the effective battery capacity decreases with increasing discharge current due to limitations in charge transfer speed.

Thanks to these properties, the KiBaM provides a valuable framework for SOC estimation under varying load conditions, particularly where classical models fail to account for dynamic recharge behavior. However, as its implementation involves differential solvers and careful tuning of parameters like k and the charge distribution factor c , it may be more suited to powerful embedded systems or simulation environments unless simplified variants are used.

2.4.3.7 Hybrid Models

Hybrid models combine elements from multiple modeling approaches to balance accuracy and computational demand. A typical example involves coupling the KiBaM with an electrical equivalent circuit model, for example the Rint model, to simultaneously estimate SOC and terminal voltage. This configuration enhances the model’s ability to capture both chemical and electrical dynamics while maintaining acceptable complexity. Hybrid models are well-suited for real-time BMS applications that require high fidelity in dynamic environments [20, 23]. Figure 2.17 illustrates the coupling of both these models.

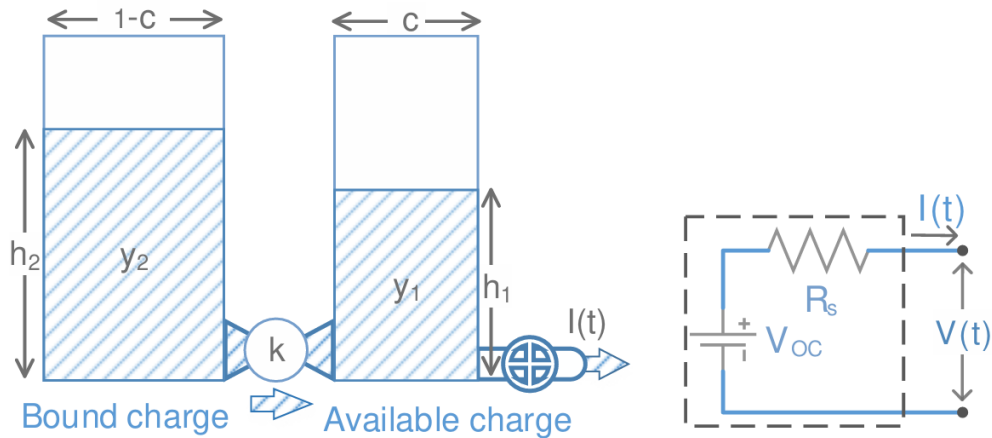


Figure 2.17: Hybrid model fusing KiBaM and Rint model, from [23].

This structure provides a robust estimation platform for real-time Battery Management Systems (BMS), where both energy availability and voltage behavior must be accurately predicted.

Despite their advantages, hybrid models require careful parameter identification, including the charge distribution coefficient c , conductance k , and internal resistance R_{int} . Additionally, the differential equations in the analytical block must be solved efficiently to maintain real-time performance. Nevertheless, the hybrid KiBaM–Rint model represents a powerful and flexible solution for advanced BMS applications, offering higher accuracy than standalone models without excessive computational overhead [20, 23].

2.5 Fault Diagnosis

Fault diagnosis is also one of the essential functionalities of a BMS, enabling the early detection and mitigation of faults that could compromise battery safety or efficiency. For FS vehicles, fault detection becomes even more critical due to the demanding nature of competitions, where the vehicle is often pushed to its operational limits.

2.5.1 Importance of Fault Diagnosis

Fault diagnosis is essential to prevent potential risks such as thermal runaway, reduced efficiency, or permanent damage to battery cells. An effective fault detection mechanism not only ensures the safety of the driver and vehicle but also optimizes the battery's performance and extends its lifespan. By identifying and addressing faults promptly, the BMS can maintain the system's stability and prevent catastrophic failures [25].

2.5.2 Types of Faults in Li-ion Batteries

Several common faults can occur in Li-ion battery systems, each with potential safety and performance implications:

Overcharge and Overdischarge:

Overcharging can result in excessive voltage, leading to thermal runaway and safety hazards, while overdischarging can cause irreversible damage to battery cells, reducing their lifespan.

Overheating and Underheating: Maintaining the battery within its optimal temperature range is crucial. Overheating accelerates degradation and can trigger thermal runaway, while underheating reduces efficiency and capacity.

Short Circuits:

Short circuits pose immediate risks of severe damage, potentially leading to fires or explosions. Rapid detection and isolation are vital to prevent catastrophic failures [25].

Thermal Runaway:

This safety-critical fault involves a self-reinforcing cycle of temperature increase, often caused by overstressing or internal faults, and can lead to battery failure.

Sensor Faults:

Malfunctioning sensors can result in incorrect assessments of the battery's state, leading to improper management and potential safety risks.

SOC and SOH estimations:

Monitoring and maintaining accurate SOC and SOH estimations is essential for avoiding critical failures and ensuring battery longevity. In typical EV applications, the SOC is generally kept within a safe operating window, commonly between 20% and 80%, to prevent deep discharge or overcharge conditions. Similarly, when a cell's SOH drops below approximately 80%, it is often considered to have reached the end of its useful life and should be replaced, as previously discussed.

2.5.3 Fault Diagnosis Techniques

To identify and address these faults, various fault diagnosis methods are employed, each suited to specific applications and requirements:

Direct Sensing:

Sensors monitor key physical parameters such as voltage, current, and temperature. Deviations from expected values indicate potential faults. However, distinguishing between actual faults and random fluctuations can be challenging.

Model-Based Methods:

These methods rely on mathematical models of the battery system to predict expected behavior. Discrepancies between predicted and observed behavior suggest the presence of faults. Although accurate, these methods often require significant computational resources [26].

Data-Driven Methods:

Machine learning and statistical techniques analyze historical data to detect anomalies. These methods excel at identifying complex faults but depend on large volumes of high-quality data [26].

2.5.4 Implications of Fault Diagnosis

Depending on the fault type and severity, the BMS can take corrective actions to mitigate risks. For instance, it might reduce the load on a weak cell, adjust the operating parameters, or shut down the system entirely to prevent further damage. Effective fault diagnosis ensures the protection of the battery system while minimizing disruptions to its application, whether in EVs, FS cars, or other energy storage systems [26].

2.6 Current Control

Efficient and safe control of charging and discharging processes is crucial to maximize the efficiency, safety, and lifespan of Li-ion batteries, especially in high-performance applications like FS. Improper control during these processes can lead to significant safety risks, such as thermal runaway and degradation of battery capacity, which negatively impact both performance and reliability [27].

In charging, the most commonly used method is the Constant Current-Constant Voltage (CC-CV) technique. This approach begins with the CC phase, where the battery charges at a constant current until it reaches a predefined voltage threshold. This phase enables efficient energy transfer and supports quick charging. Once the battery achieves this voltage threshold, the process transitions to the Constant Voltage (CV) phase. During the CV phase, the voltage is maintained at a constant level while the current gradually decreases as the battery nears full capacity, ensuring it is safely charged without surpassing voltage limits.

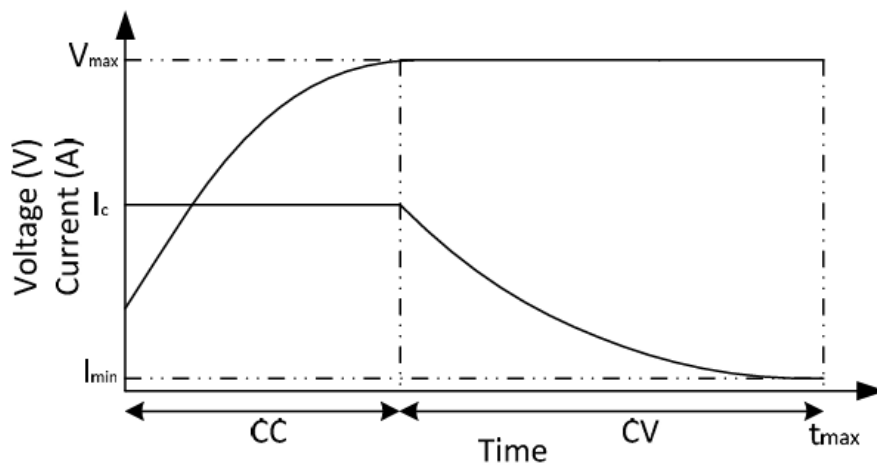


Figure 2.18: CC-CV charging profile, from [28].

To implement this transition and maintain system stability, feedback controllers are applied to regulate the charging current and voltage in real time. These controllers compare a setpoint (target voltage) with the measured output, and adjust the system accordingly. The most commonly used strategies in BMS are:

Proportional (P) Controller:

A simple but effective method where the control action is proportional to the error between the measured and target values. While easy to implement, P control alone may lead to steady-state error, particularly in the CV phase where current needs to gradually decrease toward zero.

Proportional-Integral (PI) Controller:

A widely used solution in BMS. The integral term accumulates past errors, ensuring the output reaches the setpoint without offset. PI control is particularly effective for maintaining constant voltage during the CV phase, where small current adjustments are required as the battery nears full charge. It offers a good trade-off between performance and implementation simplicity, especially for embedded systems.

Proportional-Integral-Derivative (PID) Controller:

For more dynamic applications, PID control provides faster response and better damping of transients by incorporating a derivative term. This helps predict future behavior based on the rate of change of the error, improving performance in systems with rapidly fluctuating current or voltage. PID controllers are especially useful in fast-charging strategies or when precise tapering of current is necessary to protect the cells.

This approach prevents risks associated with overcharging and can help avoid the memory effect, which can hinder battery performance over time [27].

In discharging, control is equally essential to avoid over-discharging, which can cause irreversible damage and reduce battery lifespan. One commonly employed method is the Discontinuous Current Mode (DCM). DCM carefully regulates the discharge current, periodically interrupting it to allow the battery to cool, thereby managing the heat generated and reducing stress on the cells. During discharging, the BMS continuously monitors key parameters such as SOC, temperature, and discharge current, adjusting the discharge rate as needed to prevent deep discharges, balance the load distribution across cells, and protect battery health [29].

2.6.1 Advanced Control Methods

To further optimize the charging and discharging processes, advanced control methods adapt dynamically to real-time changes in battery conditions and environmental factors, thereby enhancing both performance and longevity.

2.6.1.1 Adaptive Control Systems

Adaptive control systems continuously adjust the charging and discharging parameters in response to real-time data on battery conditions, such as SOC, temperature, and environmental influences. This approach reduces stress on the cells and optimizes charging efficiency, which can extend the battery's overall lifespan.

2.6.1.2 Machine Learning Algorithms

Machine learning algorithms analyze historical data to develop and refine optimal charging and discharging profiles. These algorithms learn from the battery's usage patterns and aging process, enabling the BMS to make more precise adjustments over time, which helps maintain battery health.

2.6.1.3 Model Predictive Control

Model Predictive Control uses mathematical models to predict the battery's future states and optimize control actions accordingly. This control can handle multiple objectives and constraints, allowing for robust and precise control, especially in complex systems where performance and safety are critical.

2.7 Cell Balancing

2.7.1 Introduction to Cell Balancing

Ideally, each battery cell produced in a manufacturing line would perform identically throughout its lifetime [30]. However, various factors contribute to the cells in a pack aging at different rates. These factors include variations in the manufacturing process and uneven temperature distribution.

These inconsistencies arise from differences in the manufacturing process, such as cell material impurities and manufacturing tolerances, as well as operational factors like temperature fluctuations within the pack. Variations at the beginning of life in cell parameters like capacity or resistance lead to non-uniform operational characteristics.

Over time, these inconsistencies cause imbalances in SOC, cell depth of discharge, voltage, and current distribution. This results in non-homogeneous performance and varying aging rates across cells, ultimately accelerating the degradation of the entire pack [31]. Consequently, these imbalances negatively impact the overall performance of the battery pack, shortening its lifespan and increasing safety risks due to potential overcharging or over-discharging of individual cells.

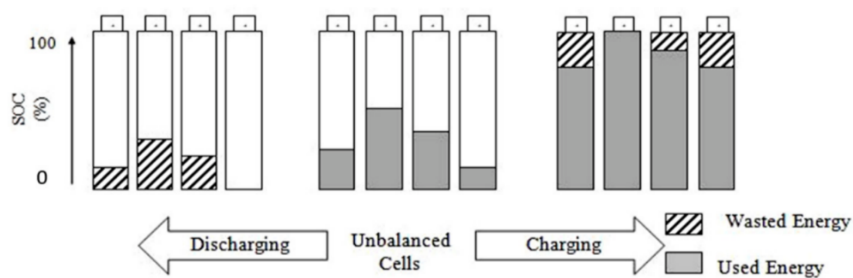


Figure 2.19: The impact of the SOC imbalance on 4-cell battery pack during charging and discharging scenarios, from [31, 32].

In addition, Figure 2.19 provides a visual representation of the unbalanced behavior during the charge and discharge cycles of the pack. When discharging, unbalanced cells with lower capacity deplete their energy faster, leading to wasted energy that cannot be fully utilized. Similarly, during charging, certain cells reach their full SOC before others, causing wasted energy as the remaining capacity in other cells cannot be filled. This imbalance leads to a reduction in the pack's usable capacity, further contributing to inconsistent performance and premature aging of the battery cells.

The figure shows how some cells hit their charge and discharge limits earlier than others, resulting in both wasted energy and an increased risk of overvoltage or undervoltage.

To address these issues, cell balancing methods are crucial for maintaining consistent performance and preventing the accelerated aging of lithium-ion battery packs.

2.7.2 Balancing Methods

There are two main methods of cell balancing: passive balancing, also known as dissipative balancing, and active balancing, also referred to as non-dissipative balancing. These methods differ significantly in their approach to achieving balance within a battery pack [33].

In the passive balancing method, excess energy from cells with higher SOC is dissipated as heat through resistive elements, effectively balancing the cells by removing energy. This approach is simpler and less costly, but it has limitations, including energy loss and generation of additional heat that requires thermal management.

In contrast, active balancing transfers energy from cells with higher SOC to those with lower SOC, redistributing charge without significant energy loss using various energy transfer methods. This method offers higher efficiency and minimizes thermal issues, making it ideal for high-performance applications. However, active balancing is more complex and requires additional components, resulting in higher costs and design complexity.

Figure 2.20 illustrates the different topologies used in passive and active balancing methods, based on the electronic components involved in the balancing process.

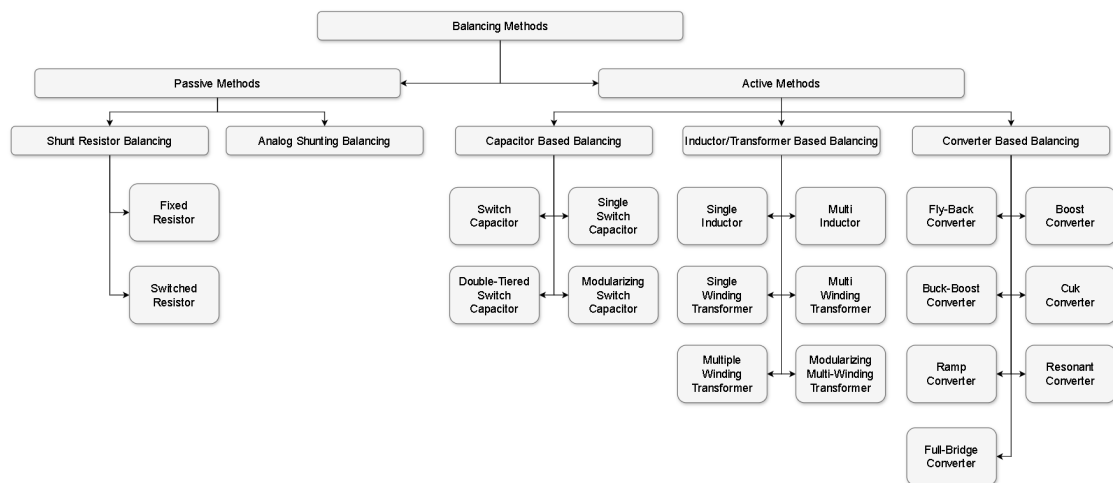


Figure 2.20: Resume diagram of cell balancing topologies based on the electronic component used, adapted from [33].

2.7.3 Passive Balancing

As mentioned before, this technique involves using resistors to dissipate energy from cells with higher SOC levels. Below are the two common methods in passive balancing:

Fixed Resistor Balancing:

Resistor-based balancing is the most widely used form of passive balancing. In this method, resistors are connected to each cell, as shown in Figure 2.21, and when a cell's voltage exceeds a certain voltage, the resistor dissipates excess energy as heat. This method is easy to implement but can result in considerable energy loss and heat generation, requiring effective thermal management.

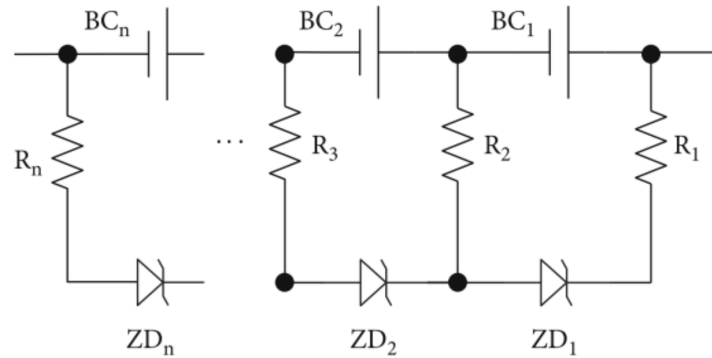


Figure 2.21: Fixed shunt resistor passive balancing circuit with the Zener diode, from [34].

Switched Resistor Balancing:

Switched resistor balancing is the most common technique. It's a variant where resistors are selectively switched on and off to control when balancing occurs. Commonly a MOSFET is used due to control simplicity as shown in Figure 2.22. This method can be more efficient than simple resistor-based balancing, as it allows balancing only when needed, minimizing unnecessary heat generation. However, it still faces challenges with energy wastage and thermal buildup.

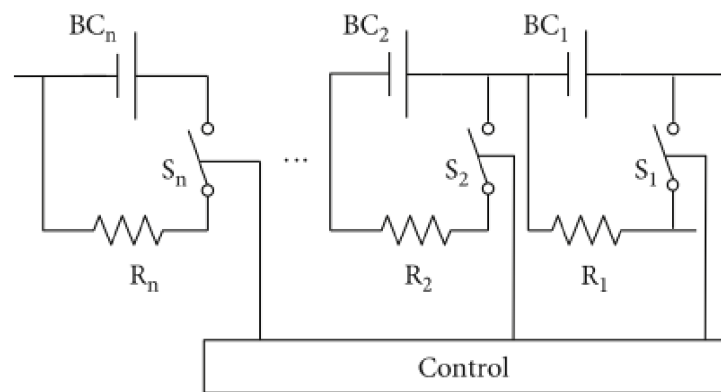


Figure 2.22: Switched shunt resistor passive balancing circuit, from [34].

Analog shunt:

This passive balancing system employs analog components, including an operational amplifier and a transistor. In this configuration, analog circuits are connected in parallel with the electrostatic discharge cell string. As higher-capacity cells become fully charged, excess current flows through resistors associated with the op-amp and transistors, distributing charge until weaker cells reach full capacity. While this system is inexpensive and relatively effective compared to other passive methods, it requires a robust heat management system to handle dissipated energy.

2.7.4 Active Balancing

Active balancing methods are more advanced, involving energy transfer mechanisms that redistribute energy from high-SOC cells to low-SOC cells, reducing energy loss. Here are several methods used in active balancing:

2.7.4.1 Capacitor-Based Balancing

In capacitor-based balancing, energy is transferred from cell to cell or module to module through capacitors, which act as temporary energy storage devices. This process uses switches to transfer energy from cells with higher voltage to those with lower voltage. Voltage sensors continuously monitor each cell's voltage, and based on this information, a controller activates switches to balance the cells. Capacitor-based balancing is straightforward and effective for smaller imbalances but is limited by the capacitor's storage capacity [33, 29].

Switched Capacitor:

In switched capacitor balancing, a network of capacitors and switches transfers charge sequentially between cells. When a cell's SOC reaches a certain threshold, the capacitor connects to the cell, storing excess energy and then discharging to a cell with lower SOC. This method is straightforward and effective for minor imbalances but may be inefficient in large battery packs.

Single Switch Capacitor:

In single-switch capacitor balancing, each cell is connected to a capacitor via a single switch. This approach allows for individual balancing control of each cell, providing more precise balancing but requiring additional control circuitry.

Double-Tiered Switch Capacitor:

The double-tiered switch capacitor approach introduces two levels of switching to handle energy transfer between modules as well as cells. This structure is useful for larger packs, enabling the balancing of groups of cells while minimizing the switching complexity within each module.

Modularizing Switch Capacitor:

In modularized switch capacitor balancing, capacitors are grouped in a modular configuration, allowing energy transfer across multiple cells or modules simultaneously. This method is efficient in balancing larger packs, reducing balancing time and complexity.

2.7.4.2 Inductor/Transformer-Based Balancing

Inductor and transformer-based balancing methods use magnetic fields to store and transfer energy between cells. Energy is temporarily stored in the magnetic field of an inductor or transformer and then directed to cells with lower SOC. This approach allows efficient energy transfer with minimal losses, making it ideal for larger energy imbalances and high-performance applications [33, 29].

Single Inductor:

Single inductor balancing utilizes one inductor to transfer energy sequentially between adjacent cells. By controlling the timing, the inductor can charge from a high-SOC cell and discharge to a low-SOC cell, balancing the SOC efficiently for nearby cells.

Multi Inductor:

Multi-inductor balancing improves balancing speed by using multiple inductors for simultaneous energy transfer across several cells. This approach increases the complexity of the circuit but is beneficial for faster balancing in larger battery packs.

Single Winding Transformer:

Single winding transformer balancing transfers energy between cells using a transformer with a single winding. This method enables the direct transfer of charge from one cell to another and provides galvanic isolation, making it safe for high-voltage applications.

Multi Winding Transformer:

Multi-winding transformers allow simultaneous balancing across multiple cells. Each cell connects to a winding, permitting efficient energy redistribution. This method is highly effective for large battery packs but requires complex control systems.

Multiple Winding Transformer:

Multiple winding transformer balancing uses transformers with multiple windings, where each winding connects to a different cell or cell group within the battery pack. This configuration allows simultaneous energy transfer across various cells, increasing the speed and efficiency of the balancing process. This method provides galvanic isolation and is highly effective for large-scale battery packs with significant SOC imbalances. However, it requires a more complex control system and careful management of transformer core losses.

Modularizing Multi-Winding Transformer:

Modularizing a multi-winding transformer allows energy transfer across modules rather than individual cells. This modular approach is efficient for large-scale balancing and can minimize the physical size and complexity of the circuitry required.

2.7.4.3 Converter-Based Balancing

Converter-based balancing involves direct current (DC-DC) converters that transfer energy from high-SOC cells to low-SOC cells. This approach is highly efficient and allows precise control, ideal for applications requiring extensive balancing with minimal energy loss [33, 29].

Flyback Converter:

The flyback converter transfers energy via a transformer, isolating high-SOC cells from low-SOC cells and providing galvanic isolation. This approach is ideal for high-voltage packs, as it ensures safe energy transfer but requires complex transformer circuitry.

Boost Converter:

Boost converters increase the voltage from high-SOC cells and transfer the energy directly to cells with lower SOC. This method is straightforward and efficient, especially suitable for smaller packs where voltage boosting is required.

Buck-Boost Converter:

The buck-boost converter can either increase or decrease voltage as needed, transferring energy from high-SOC cells to low-SOC cells at various voltage levels. This versatility makes it ideal for packs with significant variations in SOC across cells.

Cuk Converter:

The Cuk converter balances the cells by creating an intermediate energy storage stage, allowing smoother energy transfer between cells with minimal ripples. It is efficient for large packs where high stability in energy transfer is required.

Ramp Converter:

The ramp converter is a specialized type of converter that adjusts the rate of energy transfer between cells based on their SOC levels. It gradually increases or "ramps up" the energy transfer as the SOC difference grows, offering smoother and controlled balancing. This method minimizes stress on the cells by avoiding sudden energy shifts, making it ideal for sensitive applications requiring gradual, controlled balancing. Although the ramp converter is efficient, it requires precise control algorithms and is generally more costly and complex to implement.

Resonant Converter:

Resonant converters utilize resonance circuits to transfer energy, reducing switching losses and increasing efficiency. This method is complex but ideal for high-performance applications where energy efficiency is crucial.

Full-Bridge Converter:

Full-bridge converters allow high-power energy transfer across cells, using four switches to control current flow. This converter is efficient for large battery packs but requires significant space and advanced control circuitry.

2.7.5 Cell Balancing Circuits Comparison

Table 2.4 provides an overview of various balancing techniques, categorized by their methods and characteristics. This comparison highlights key factors such as balancing time, component count, control complexity, voltage and current stress, efficiency, power loss, physical size, cost, and notable advantages and disadvantages. Each technique has specific strengths and trade-offs, making some more suitable than others depending on the application's requirements for efficiency, cost, or control complexity.

Table 2.4: Comparison of different balancing circuits characteristics, adapted from [29].

Balancing Technique	Method	Time	Elements (n cells)	Complexity	Efficiency	Loss	Cost
Fixed Shunt	Passive	Slow	n resistors	Simple	Poor	Very High	Very Cheap
Switch Shunt	Passive	Slow	n switches, n resistors	Simple	Low	Very High	Very Cheap
Analog Shunt	Passive	Slow	n switches, Op-amps, $3n$ resistors, capacitors	Simple	Low	High	Cheap
Single Switch Capacitor	Active	Medium	$n + 5$ switches, 1 capacitor	Complex	Good	Minor	Medium
Switch Capacitor	Active	Medium	$2n$ switches, $n - 1$ capacitors	Medium	Good	Minor	Medium
Double-Tiered Capacitor	Active	Medium	$2n$ switches, $2n - 3$ capacitors	Complex	Good	Minor	Medium
Modular Capacitor	Active	Medium	$M(n + 2)$ switches, capacitors	Complex	Good	Minor	Medium
Single Inductor	Active	High	$2n - 2$ switches, inductor, diodes	Complex	High	Low	Costly
Multi Inductor	Active	High	$n + 1$ switches, $n - 1$ inductors	Complex	High	Low	Large
Single Winding Transformer	Active	Medium	$n + 6$ switches, transformer	Complex	Good	Low	Costly
Multi-Winding Transformer	Active	Medium	2 switches, n diodes, transformer	Complex	Good	Low	Costly
Flyback Converter	Active	Medium	$2n$ switches, inductors	Medium	Good	Low	Costly
Buck-Boost Converter	Active	Very High	$2n - 2$ switches, $n - 1$ inductors	Complex	High	Minor	Medium
Resonant Converter	Active	High	$2n - 2$ switches, capacitors	Complex	High	Very Low	Medium

To conclude this comparison, passive balancing methods, particularly the switched shunt resistor, stand out as a promising approach for applications where simplicity, cost-effectiveness, and compact size are essential. Compared with more complex active methods, the switched shunt resistor achieves a reasonable balancing speed and reliability while maintaining reduced control complexity and thermal management requirements. These factors make it a viable choice for systems prioritizing lower cost and efficiency without the need for high-precision balancing.

2.7.6 Balancing Control Strategies

Balancing control strategies are crucial for managing the efficiency and longevity of battery packs, especially in EVs. The correct state variable choice can make the balancing algorithm simpler and more effective, taking in consideration the needs of the solution. There are two primary approaches for balancing control strategies: control based on key variables like SOC, pack capacity, and terminal voltage, and automatic balancing, which operates without direct monitoring. Each of these strategies has unique advantages and limitations [13].

Based on Voltage Balancing:

Voltage-based balancing systems aim to equalize the voltage of each cell within the battery pack. This approach is relatively simple to implement as it involves identifying the cells with maximum and minimum voltages and adjusting them to achieve balance. Voltage-based systems provide fast sampling speeds, which can benefit applications needing frequent balancing. However, due to cells' uneven internal resistances, voltage-based balancing can lead to over- or under-equalization [13, 35]. Consequently, this method may fail to accurately reflect the battery's state of charge, especially in systems with significant discrepancies in internal resistance.

Based on Pack Capacity:

Capacity-based balancing aims to equalize each cell's capacity within the battery pack to achieve an optimal and accurate reflection of system inconsistencies. By ensuring that each cell has a similar capacity, this method maximizes the battery pack's usable energy, especially during discharge cycles. However, capacity-based balancing is challenging to implement because it requires accurate and precise estimation of each cell's capacity, which involves complex calculations. This method is known for accurately reflecting system inconsistencies but suffers from poor real-time performance due to its computational demands. Additionally, capacity estimation is difficult to measure directly and requires frequent recalibration, making it challenging to implement in dynamic applications like EVs [13, 35].

Based on State of Charge:

SOC-based balancing is focused on equalizing cells according to their state of charge. By matching the SOC of the cells with the highest and lowest values, this method can theoretically maximize the available capacity of the battery pack. SOC-based balancing accurately reflects system inconsistencies and effectively characterizes the system's status, making it suitable for applications that require close monitoring of each cell's charge level. However, SOC-based control demands precise SOC estimation for each cell, resulting in a large amount of computation, which can reduce real-time performance [13]. This complexity makes it challenging for SOC-based balancing to be implemented in real-time applications without sophisticated algorithms and computational resources [35].

2.7.7 Based on Automatic Balancing

Automatic balancing utilizes an open-loop control system that operates based on the natural energy flow between cells. Rather than relying on monitoring equipment, energy flows passively from more charged to less charged cells, balancing the system as voltage differences drive the current flow. This method offers the advantage of simplicity, as it does not require intensive calculation or active cell monitoring. However, it may result in longer balancing times and is less precise, as it depends on the cells' natural balancing properties. Automatic balancing is suitable for scenarios where minimizing control complexity is more critical than achieving high accuracy in balance.

2.7.8 Comparison of Balancing Control Strategies

When comparing balancing control strategies, factors such as the ease of obtaining the control variable, accuracy, and real-time performance must be considered. Table 2.5 summarizes the main characteristics of each control strategy based on these aspects. SOC-based and voltage-based control stand out as they align well with the core balancing requirements. SOC-based control offers high precision in representing system inconsistencies, especially those arising from uneven internal resistance. In contrast, voltage-based balancing is more accessible and provides moderate real-time performance but lacks the precision needed to address deeper system discrepancies.

Table 2.5: Comparison of balancing control variables, adapted from [13, 35].

Control Variable	Acquisition complexity	Precision	Real-time suitability
Terminal Voltage	Low	Low	Moderate
Battery Capacity	High	High	Low
State of Charge	Moderate	High	Moderate

In summary, each balancing control strategy offers unique advantages, but as shown in the Table 2.5 and considering the requirements of this project, the SOC-based method is the one that align better because of its accuracy in reflecting system inconsistencies and its ability to effectively manage the state of charge across cells. Although SOC-based control is computationally intensive, its precise tracking of each cell's charge level makes it well-suited for applications that demand reliable balancing and efficient battery management, aligning closely with the goals of this project.

2.8 Characteristics of the BMS's Implemented in the Competition

BMSs are crucial components in electric vehicles, ensuring the safety, efficiency, and longevity of battery packs. In the context of FS competitions, teams develop or select BMS solutions tailored to their specific vehicle designs and performance goals. This section provides an overview of the BMS configurations employed by various teams, highlighting key parameters such as battery configuration, cell chemistry, topology, balancing methods, current sensing, communication protocols, and associated costs.

Table 2.6 provides a comparison of BMS implementations by some prominent teams, showcasing its features.

Table 2.6: Comparison of BMS across different Formula Student teams, adapted from [9].

University	TU Lisbon	ETH Zürich	UAS Augsburg	ETSEIB Barcelona	FEUP	NTNU	Karlsruhe IT
Team	Projecto FST	AMZ Racing	StarkStrom Augsburg	ETSEIB MotorSp	FSFEUP	Revolve NTNU	KA Racing
Vehicle	FST-04e	Grimsel	Elinor	CAT07e	FSFEUP01	Lyra	Kit23
Year	2011	2014	2014	2014	2019	2023	2023
Batt. Config.	48s5p	112s4p	96s3p	140s2p (2x70s2p)	120s5p	-	-
Cell Chem.	LiFePO ₄	-	LiCoO ₂	LiCoO ₂	LiFePO ₄	-	LiPo
Cell Type	Cylindrical	Pouch	-	Pouch	Pouch	-	-
Model	Self-Developed	Joint-Developed	Li-BMS v3	Elithion Lithiumate Pro	Orion BMS	Self-Developed	Self-Developed
Topology	Distributed	Distributed	Centralized /Distributed	Distributed	Master-Slave	Master-Slave	Master-Slave
Balancing	Passive	Passive	Passive	Passive	Passive	Passive	Passive
Curr. Sensor	Hall Effect	Shunt Resistor	Shunt Resistor	Hall Effect	Hall Effect	Shunt Resistor	IVT-S300
Comm. Protocols	CAN	4xCAN	CAN, RS232, I2C	CAN, RS232	CAN, isoSPI	CAN-FD, isoSPI	CAN
Cost (€)	1100	1500	N/A	4200	2000	N/A	N/A

The analysis reveals that most teams prefer passive balancing for its simplicity and cost-effectiveness, despite its lower efficiency compared to active balancing methods [19]. Additionally, a shift from distributed to master-slave topologies is observed in recent years, offering advantages in modularity and scalability. Controller area network bus (CAN bus) remains the standard communication protocol, with some teams adopting higher-speed variants like isolated serial peripheral interface (isoSPI) to accommodate increased data transmission requirements.

Chapter 3

Development Process and System Design

The design and development of a BMS for an electric vehicle prototype requires a rigorous and structured engineering approach, given the system's complexity and critical role. This chapter outlines the complete development methodology adopted for the BMS prototype presented in this dissertation, covering both technical strategies and project planning decisions.

It begins by detailing the engineering frameworks that guided development, including Automotive SPICE (ASPICE), the V-model, and MBSE. These methodologies provided a formal foundation for requirement definition, architecture design, traceability, and verification, ensuring that all functional and safety objectives were systematically addressed.

The chapter then presents a thorough analysis of regulatory and technical requirements and compares the limitations of the team's previous commercial BMS (Orion®) to highlight the motivations behind designing a custom solution. This analysis leads directly to the proposal of a new modular BMS architecture, tailored to meet team needs, reduce system complexity, and support future scalability.

Finally, the planning and prioritization strategies adopted to structure development are described, including the project scheduling, work breakdown, and supporting tools used throughout the project. This integrated strategy ensured that despite resource and time constraints, the most critical modules were prioritized and implemented efficiently, laying the groundwork for the results in the following chapters.

3.1 Development Strategy

The following section describes the system engineering methodology that served as the foundation for the development of this BMS. It begins by outlining the Automotive SPICE (ASPICE) framework, which establishes a structured approach to software, hardware, and system engineering. Within this context, requirements engineering is highlighted as a key component, ensuring that all functional and safety constraints are systematically defined, analyzed, and validated.

Following this, the V-model process, a widely recognized methodology within ASPICE, is characterized in detail. This model provides a structured development and verification process, ensuring that each stage of the system implementation is accompanied by a corresponding validation step.

In addition to the development framework, the definition of system requirements played a crucial role in ensuring the BMS met both FS regulations and FSFEUP team expectations. The competition imposes strict constraints on battery safety, fault detection, and monitoring, which directly influenced the system's core functionalities. A structured requirement analysis and prioritization process was applied, allowing for an optimized balance between regulatory compliance and performance enhancements.

Lastly, the MBSE approach is introduced as a fundamental pillar of this project. MBSE played a critical role in the early-stage validation of system behavior. The benefits of this methodology, particularly for the initial conceptualization and validation of the BMS, are explored in depth, reinforcing its importance in modern automotive development processes.

3.1.1 Automotive SPICE

The development of this BMS is fundamentally rooted in system engineering principles, structured according to the ASPICE framework. Systems engineering provides a methodical approach to product development, ensuring that all stakeholder requirements are formally captured, analyzed, and systematically validated throughout the development cycle.

ASPICE, in particular, serves as an industry standard framework for assessing and structuring development processes in the automotive sector. It emphasizes traceability, modularity, and iterative validation, particularly in the domains of software, hardware, and system integration. By aligning with ASPICE process models, this project ensured that the BMS architecture, requirement definition, and verification phases were developed in a structured and methodical manner.

One of the key principles enforced by ASPICE is requirements engineering, which involves defining and organizing all system-level, hardware, and software requirements before development begins. In the context of the V-model process, this ensures that each requirement is systematically validated through corresponding verification stages, guaranteeing that the final implementation aligns with the safety, performance, and regulatory constraints established for the FS competition.

3.1.2 The V-Model Development Process

The V-model process is a structured systems engineering methodology that ensures the systematic development and validation of complex systems such as BMS. This model follows a structured approach, where each development phase is directly associated with a corresponding verification or validation step, forming a V-shaped process structure.

At the beginning of the process, high-level system requirements are defined, breaking them down into subsystem and component-level specifications. Each of these requirements must then undergo progressive verification and validation, ensuring that no phase of development proceeds without appropriate testing and validation.

A core principle of the V-model, closely linked to ASPICE, is the integration validation rule. This rule dictates that integration testing can only be performed after individual unit-level tests have been successfully completed. This sequential verification methodology minimizes the risk of propagating errors during system integration, ensuring a more robust and reliable implementation.

Figure 3.1 illustrates the V-model adapted for this BMS development process, highlighting the relationship between development phases and validation steps at each stage.

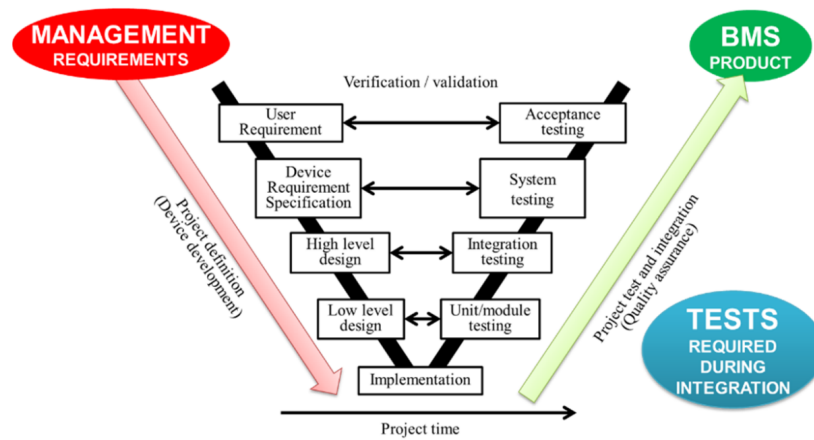


Figure 3.1: V-model process for BMS development, from [10].

3.1.3 Model Based System Engineering

In modern automotive and embedded system development, MBSE plays a crucial role in enhancing system design and verification processes. MBSE provides a structured methodology that integrates system modeling and simulation to reduce design uncertainties before physical implementation.

This approach was fundamental in the early stages of the BMS development, as it enabled precise definition of system behavior before hardware deployment. In addition to improving testing efficiency and safety, MBSE enables structured management of system complexity, ensuring that all components align with specified requirements and function as intended [10]. By leveraging MBSE principles, the BMS development was optimized, ensuring systematic approach to integrating hardware, software, and control strategies, better traceability of requirements, and a structured verification process.

Throughout this dissertation, different graphical representations derived from the MBSE methodology will be introduced in their respective contexts.

3.2 System Analysis and Proposed Architecture

This section is structured into three key parts. First, the regulatory and technical requirements are analyzed and categorized, forming the basis for all subsequent design decisions. Then, the current commercial solution used by the team, the Orion BMS, is critically reviewed, highlighting its limitations and the motivations for replacing it. Finally, the proposed architecture is presented, designed to overcome the shortcomings of the existing system while aligning with both competition regulations and the FSFEUP team's goals.

Before designing a new BMS, it's not only essential to analyse not only the problems but also the good things from the previous BMS but also to rigorously analyze system requirements, both from competition regulations and team specific needs, and assess the limitations of the current solution. This subsection builds on the V-Model approach (see Figure 3.1), where high-level requirements are decomposed into subsystem specifications and directly linked to their respective validation steps, as previously mentioned before.

3.2.1 Rules and Requirement Analysis

The analysis and definition of system requirements mark the first and most critical step in the design process of any safety-critical system. Aligned with the left-hand branch of the V-model framework (Figure 3.1), this phase ensures that every system function is traceable, testable, and rooted in actual technical and regulatory demands. In this project, the methodology adopted follows the guidelines established by the Automotive SPICE® Process Assessment Model (ASPICE 4.0) [36], which provides a structured framework for developing automotive-grade systems in a modular and traceable way.

To manage the system's complexity, all requirements were classified into three categories, each corresponding to a different level of the system's architecture:

- **SYS – System Requirements:** These define high-level constraints and global functionalities, such as compliance with FS rules [37], system-level safety, isolation procedures, fault indicators, and general interfaces.
- **HWE – Hardware Requirements:** Covering sensor ranges, temperature tolerance, current/voltage acquisition accuracy, analog response times, thermal limits, and electrical protections.
- **SWE – Software Requirements:** Focused on algorithmic behavior, control loops, timing constraints, state machine transitions, fault detection thresholds, and communication protocols.

It is important to note that all functional, safety, and performance requirements originate from the system-level (SYS) and are then decomposed into hardware (HWE) and software (SWE) requirements depending on their implementation domain.

Each of these categories was further refined into detailed requirement sets: SYS1 through SYS5, HWE1 through HWE4, and SWE1 through SWE6. This hierarchical subdivision was necessary to decompose broad objectives into specific, testable functionalities. Figure 3.2 illustrates the layered structure followed during the development process.

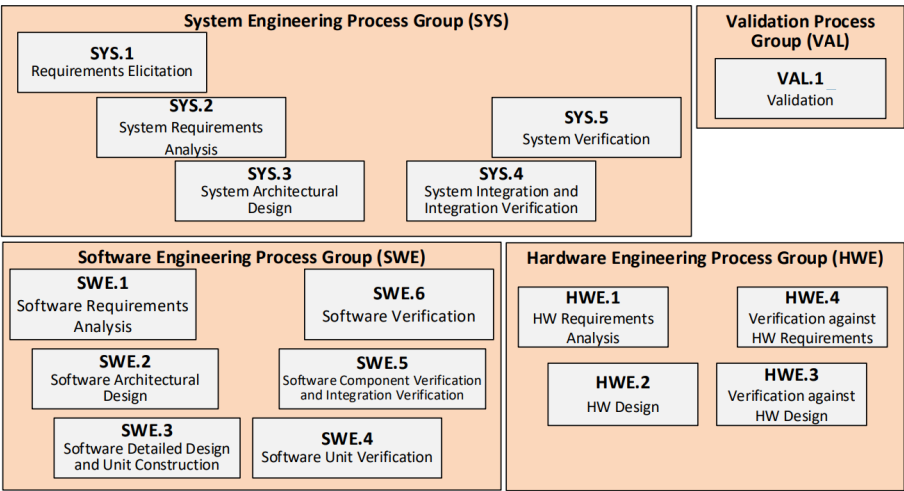


Figure 3.2: Requirement decomposition across SYS, HWE, and SWE levels, adapted from [36].

This structured approach mirrors a "requirement pyramid," where the top layer begins with high-level expectations, often vague or broadly defined, such as a rule from the competition handbook or a team preference. These are progressively analyzed and decomposed into more precise requirements, ultimately converging into detailed technical specifications that are directly linked to implementation and verification tasks.

From a systems engineering perspective, this process represents a transition from a black-box view to a white-box architecture. Initially, the system is viewed as a whole, with only general goals known. As analysis progresses, each subsystem is detailed, interconnected, and tied to validation criteria. On the right side of the V-model, this structure allows the implementation to follow a mirrored verification path: unit-level tests validate individual components, which are then progressively integrated and tested together until the full system is verified as a coherent whole.

Although dozens of requirement entries were documented in the development database, only the most representative examples are presented in the next Table 3.1. These selected table include consolidated constraints that influenced system design choices and will be further referenced in later sections when implementation and testing are discussed. The complete traceability matrix and extended requirement database were used throughout development to ensure compliance, completeness, and verification coverage.

Table 3.1: Summary of the most important system requirements.

Table	Requirement Description	Priority	From
SYS1	The AMS must continuously measure all cell voltages	High	FS Rules
SYS1	The AMS must continuously measure the TS current	High	FS Rules
SYS1	The AMS must continuously measure for lithium-based cells: the temperature of at least 30 % of the cells equally distributed within the TSAC	High	FS Rules
SYS1	Cell temperature must be measured at the negative terminal of the respective cell. The sensor used must be in direct contact with the electrically exposed negative terminal or less than 10 mm along the high current path, away from the terminal in direct contact with the respective busbar. It is acceptable to monitor multiple cells with one sensor if this requirement is met for all cells sensed by the sensor	High	FS Rules
SYS1	The maximum cell temperature is 60 °C or the limit stated in the cell data sheet, whichever is lower	High	FS Rules
SYS1	The loss of any measurement connection, must result in an open SDC	High	FS Rules
SYS1	The AMS must disconnect the Tractive System via the Shut Down Circuit in the event of critical cell voltage, cell temperature, or pack current	High	FS Rules
SYS1	The AMS must be adaptable for different battery configurations	High	FSFEUP Team
SYS1	The AMS must adopt a decentralized and scalable architecture that improves reliability, reduces system complexity, and simplifies hardware implementation compared to the previous design	High	FSFEUP Team
SYS1	The AMS must estimate SOC	High	FSFEUP Team
SYS1	The BMS must update and transmit all monitored values to the user interface within a maximum interval of 200ms	High	FSFEUP Team
SYS1	The AMS must diagnose over/under voltage, over/undercurrent, over/under temperature, and the health of the pack	High	FSFEUP Team
SYS1	The AMS must communicate to the ECU the charging and discharging current limit	High	FSFEUP Team
SYS1	The AMS must have a UI to perform battery configurations	Medium	FSFEUP Team
SYS1	The AMS must monitor the critical cell unbalance and initiate balancing when needed	Medium	FSFEUP Team

It is important to note that, according to FS regulations, the terminology used differs from conventional naming. What is commonly referred to as the Battery Management System (BMS) must be designated as the Accumulator Management System (AMS) in all official documentation and implementation. Similarly, other acronyms such as TS (Tractive System), TSAC (Tractive System Active Circuit), SDC (Shutdown Circuit), and ECU (Electronic Control Unit) are used following the strict definitions set by the competition rules [37].

3.2.2 Previous BMS analysis

Until now, the FSFEUP team has relied on two different commercial BMSs from the Orion® series across the development of their first and second generation electric vehicles [38, 39]. For the first prototype, a smaller battery configuration consisting of 120 cells in series allowed the use of a more compact Orion BMS unit, as shown in Figure 3.3.

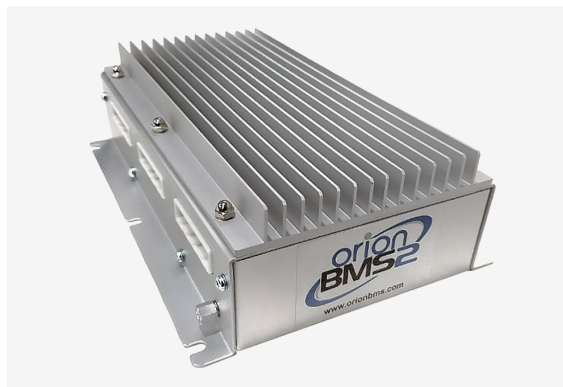


Figure 3.3: Orion BMS used in the first-generation FSFEUP electric vehicle, from [38].

However, with the transition to a larger and higher-capacity accumulator in the second vehicle, now featuring 144 cells in series, the team adopted a bigger Orion BMS to accommodate the increased number of cells, as illustrated in Figure 3.4.



Figure 3.4: Orion BMS used in the second-generation FSFEUP electric vehicle, from [38].

Although these systems provided an off-the-shelf solution, several significant limitations emerged during development, integration, and operation with this commercial BMS [39]. The following points summarize the most relevant technical and strategic issues that motivated the FSFEUP team to pursue a custom-designed BMS solution:

- **High Cost:** The commercial units represented a significant financial burden for the team, especially considering the need to change models between vehicle generations. Table 3.2 presents the approximate cost of each model.

Table 3.2: Estimated cost of Orion BMS units used, from [38].

Model	Estimated Cost (€)
Orion BMS 2 - 120cells (Gen1 Car)	1142
Orion BMS 2 - 144cells (Gen2 Car)	1244

- **Excessive Wiring and Mechanical Complexity:** Due to its centralized topology, all voltage taps and sensor lines had to be routed to a single unit. This drastically increased wiring density, leading to signal interference, mechanical failures, and complicated routing inside the accumulator container.
- **Large Physical Size:** The Orion BMS occupied considerable space inside the accumulator, limiting packaging flexibility and creating integration challenges as the battery design evolved. It also occupied a disproportionate amount of volume within the accumulator enclosure. In both the first and second vehicle designs, its size constrained the battery layout and forced compromises in terms of mechanical stability and thermal distribution.

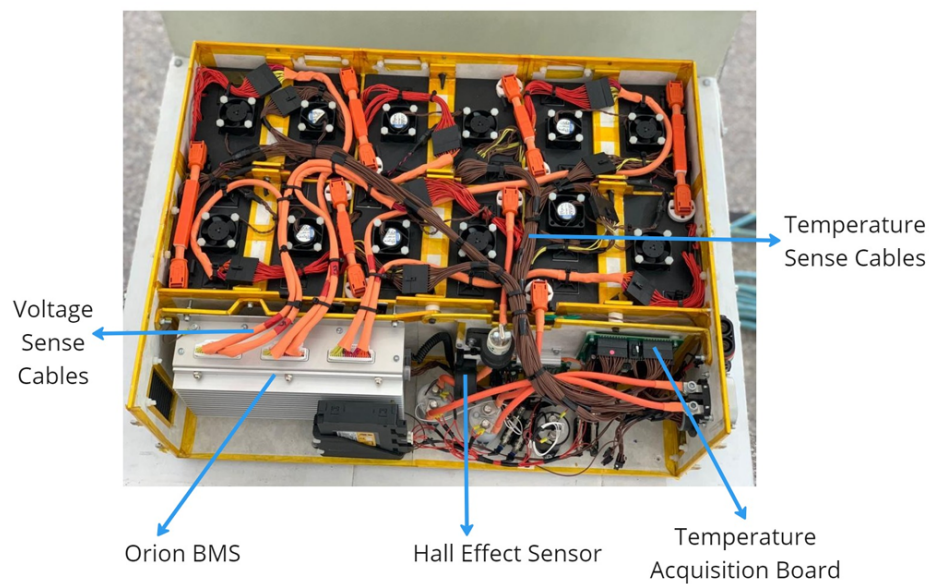


Figure 3.5: Accumulator configuration of the first FSFEUP car, showing the Orion BMS integration.

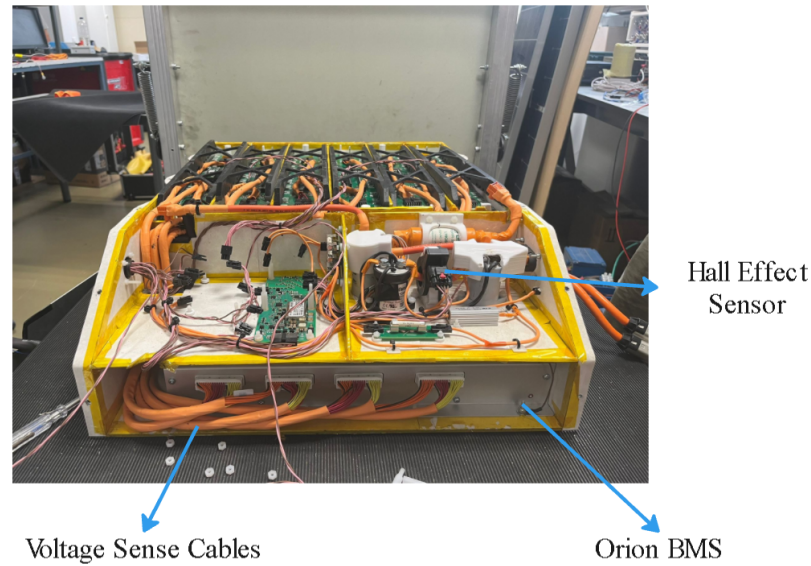


Figure 3.6: Accumulator configuration of the second FSFEUP car, with the updated Orion BMS.

Figures 3.5 and 3.6 show the packaging situation of the Orion BMS inside the battery containers used in the first and second car generations, respectively. As seen, the BMS's footprint significantly impacted available space.

- Limited Temperature Acquisition:** Both Orion BMS provided only 8 General-Purpose Input/Output (GPIO) inputs for temperature sensing. According to the FS regulations, the BMS must monitor the temperature of at least 30% of the battery cells, as mentioned in Table 3.1. For example, the second generation FSFeup car, the battery pack configuration consists of 144 cells in series and 5 in parallel, totaling 720 individual cells. This means the system must monitor at least 216 cells ($720 \times 0.30 = 216$) to comply with the rules. Even if each negative temperature coefficient thermistor (NTC) is allowed to monitor two adjacent cells, under the condition that it is installed within 10 mm of both, this would still require over 100 sensor channels, far exceeding the Orion's native capacity. As a result, the team was forced to design and integrate an entirely separate temperature acquisition system, which not only increased the complexity of the accumulator but also added weight, redundancy, and maintenance overhead. While this limitation is more difficult to visualize in the second generation setup because of the new design of the accumulator, it is clearly evident in the first generation accumulator design, where the extra wiring and sensor modules are visible (see Figure 3.5).
- Restricted Software Flexibility:** As a proprietary commercial product, the Orion system did not offer deep configurability or access to low-level control software. This severely limited algorithm customization, diagnostic expansion, and communication flexibility.

- **Lack of Integrated Diagnostics:** The Orion BMS required an additional, independent system for diagnostics and fault management. This added further complexity and demonstrated a lack of built-in support for robust fault detection and real-time safety response.

These technical and operational constraints made it clear that a new, tailored solution was urgently needed, one capable of reducing mechanical complexity, improving configurability, lowering cost, and aligning more directly with both FS competition requirements and FSFEUP team goals. Beyond addressing the limitations of the commercial Orion BMS, the new system also needed to ensure scalability across different battery configurations, enabling its adaptation to varying cell counts and layouts. Additionally, it had to be configurable for future use with different cell chemistries or battery technologies, in case of the FSFEUP team decide to change strategies in upcoming vehicle generations. The proposed BMS system described in the next subsection was designed precisely with these objectives in mind.

3.2.3 New BMS configuration

The proposed BMS architecture follows a master-slave topology, where a central master module manages multiple slave modules, each responsible for monitoring a subset of battery cells. This architecture was chosen due to its scalability, modularity, and efficiency in data acquisition, making it an industry-standard approach for high-voltage battery packs.

Each slave module measures the voltage of individual battery cells and acquires temperature readings using NTC thermistors placed at critical points to ensure reliable thermal monitoring. Additionally, it integrates a balancing circuit to regulate charge distribution between cells, preventing excessive differences in state of charge. The balancing method adopted in this architecture is passive switched-shunt balancing, a widely used technique that is integrated into most battery monitoring ICs due to its simplicity, reliability, and cost-effectiveness. Communication between the master and the slave modules is established through a daisy-chain communication protocol, enabling efficient data transfer between multiple slaves while maintaining a minimal wiring complexity.

The master module assumes a central role in system operation, being responsible for acquiring current measurements from the battery pack using a Hall-effect sensor. This method ensures accurate and non-intrusive current sensing, which is critical for the estimation of battery parameters such as the SOC and the SOH. In addition to acquisition and data processing, the master module also perform key computational tasks, including monitoring the data coming from the slaves, the control of the maximum battery current, the fault detection algorithms and control logic required for managing cell balancing and protection mechanisms. Furthermore, it serves as the communication hub between the BMS and external interfaces.

In a fully integrated system, the master module would be responsible for transmitting battery status information to the vehicle's Electronic Control Unit (ECU) via CAN bus. This communication would allow real-time monitoring and active control of the battery pack. Additionally, the BMS would interact with a latching printed circuit board (PCB) that manages a shutdown circuit and an AMS cockpit light, which activates in the event of critical failure, signaling the need for immediate vehicle shutdown. This PCB receives a digital signal from the BMS, indicating normal operation when the signal is 5V. If the signal drops to 0V, it signifies a critical fault, requiring the vehicle to enter a safe state. Beyond the vehicle-level integration, a secondary communication link with an external PC-based monitoring system would allow system diagnostics, configuration, and data logging, facilitating testing and development. However, this communication is only available when the vehicle is stationary and the BMS is operating in a test or monitoring phase.

The overall system architecture, depicted in Figure 3.7, illustrates the interaction between the master module, slave modules, and external systems, providing a comprehensive overview of how the BMS is structured and how the different components interact.

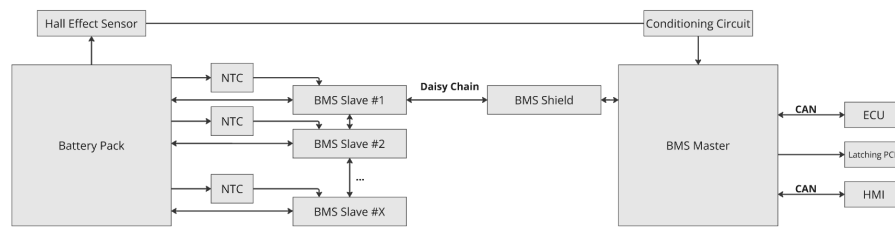


Figure 3.7: High-level BMS architecture.

3.2.4 Proposed solution description

While the full BMS configuration described in the previous section represents an optimal and scalable solution, the development of such a system requires extensive time and resources. Given the constraints of the present study, certain simplifications and modifications were made to ensure that a functional implementation could be achieved within the available timeframe. To maintain feasibility while focusing on core functionalities, the project scope was limited to the development and validation of a simplified proof-of-concept implementation.

One of the primary modifications was the number of slave modules. Instead of implementing multiple slave units, the present study focuses on integrating only a single slave module. This decision was made to reduce the complexity of system communication while ensuring that the core functionalities of cell monitoring, state estimations and balancing could be properly validated. The selected hardware for this implementation is the BQ79616EVM-021 evaluation board, which serves as the slave module responsible for voltage and temperature acquisition. Although the development of a dedicated PCB for the slave module would offer advantages in terms of size, cost, and optimization, the use of an evaluation board provides a faster and more flexible development process, allowing for rapid prototyping and validation.

Another key simplification in the proposed implementation concerns the communication protocols used in the system. While a fully integrated BMS would typically employ a CAN bus for communication with external systems, the present study adopts a different approach by exclusively using UART communication. The choice of UART over CAN was made to facilitate development, as the BQ79616EVM-021 evaluation board natively supports UART for master-slave communication, eliminating the need for additional protocol integration. Additionally, using UART to interface the master module with a PC-based monitoring system simplifies data acquisition and debugging, as it allows for direct serial communication using readily available tools. This adaptation does not affect the fundamental capabilities of the BMS, as the modular nature of the system design allows for the integration of CAN communication in future iterations if required.

As mentioned before, beyond the simplifications related to hardware and communication, certain functionalities that would typically be present in a production-grade BMS were excluded from this implementation. The most notable exclusion is the communication with the vehicle's ECU. In a full-scale system, the BMS would transmit critical battery status information to the ECU, enabling vehicle-level control and integration. However, due to the complexity of this task and the time constraints of the project, this communication was omitted. Instead, all system monitoring and configuration operations will be conducted through a PC-based interface, allowing for efficient validation of system performance without requiring full vehicle integration. Additionally, no interface with the vehicle's driver will be implemented, meaning that real-time feedback to the driver, such as dashboard indicators, will not be considered in this study.

Despite these simplifications, the proposed implementation retains the essential functionalities required for a fully operational BMS. Voltage and temperature monitoring, current sensing, state estimation, fault detection, diagnosis and cell balancing will all be implemented and validated, ensuring that the core objectives of the project are met. The resulting system architecture, illustrated in Figure 3.8, reflect these modifications while maintaining the fundamental structure required for future scalability.

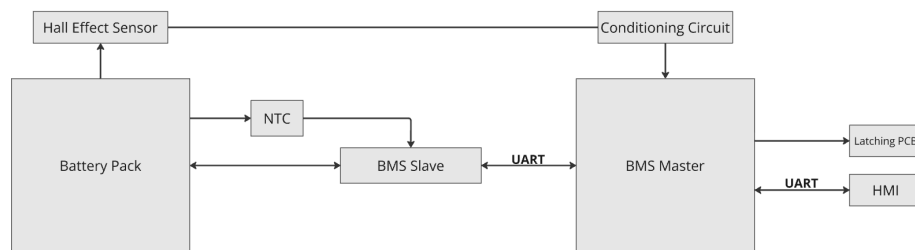


Figure 3.8: Final implemented BMS configuration.

3.3 System Structuring and Development Prioritization

The development of a Battery Management System (BMS) is a complex process that requires structured planning to ensure all functionalities are well-defined and properly implemented. Given the time and resource constraints, it was essential to establish a clear methodology for system structuring and development prioritization.

Throughout the early stages of the project, a wide range of diagrams and structural representations were created to support the planning process. These included the limit diagram, system identification diagram, use case diagram, black box diagram, activity diagram, system breakdown structure (SBS), work breakdown structure (WBS), work planning diagram, requirement tables, white box architecture, sequence diagram, high-level and low-level architecture diagrams, and module-level interaction diagrams. Although some of these elements are presented or mentioned in the main body of the report, selected examples have been included in appendix for reference.

These visual tools were directly inspired by the principles of MBSE, as introduced in Subsection 3.1.3. MBSE methodologies played a critical role in the early conceptualization and validation of the BMS. By representing the system structure graphically, it was possible to explore interfaces, dependencies, and design logic more effectively, and to support requirement decomposition in a structured and traceable manner.

These graphical and structural tools were fundamental to the system's conceptualization and played a key role in ensuring consistent logic, clear module definition, and effective scheduling across the project's phases. They helped define functional boundaries, clarify system interactions, and link each subsystem to its corresponding design constraints and verification strategy.

To structure the implementation effort and align with the project's goals, a Work Breakdown Structure (WBS) was defined based on the development priority ranking of the major subsystems established in the System Breakdown Structure (SBS), shown in Appendix B. This prioritization, presented in Table 3.3, ensured that critical functionalities were addressed early in the project timeline, while less critical modules were deferred to later stages.

Table 3.3: Development prioritization of system modules based on SBS.

Priority Level	System Module
7 (Highest)	Data Acquisition System
6	Communication System
5	State Estimation (SOC/SOH) System
4	Battery Protection System
3	Diagnostics System
2	Human-Machine Interface (HMI)
1 (Lowest)	Balancing System

This planning structure served as a guide throughout the project, helping maintain focus on the most critical areas while preserving flexibility for iterative improvement and validation.

3.4 Tools

Throughout the development of this BMS, a diverse set of digital tools was used to support planning, simulation, development, and analysis activities. These tools enabled effective organization, collaborative work, precise simulation of models, hardware programming, and structured data evaluation.

Table 3.4: Tools used in the BMS development.

Tool	Usage	Description
Notion	Project page	Used to centralize all project materials through a dedicated workspace, including links to OneDrive documents, project documentation, Google Teams meeting logs, weekly sprint and to the Miro platform for brainstorming
Draw.io	Diagram design	Used to design all architecture diagrams, block diagrams, system representations (white-box/black-box), and planning schemes, such as Use Case and Activity Diagrams
Microsoft Excel	Requirements tracking	Employed to create and manage the system requirement tables (SYS, HWE, SWE), enabling analysis and traceability of constraints and priorities
MATLAB and Simulink	Model simulation	Used extensively for the simulation of battery models, testing of SOC estimation methods, and validating theoretical results prior to real hardware implementation
STM32CubeMX	MCU configuration	Configured the master microcontroller peripherals, clocks, UART's, timers, and ADC's, and generated initialization code for the embedded environment
STM32CubeIDE	Firmware development	Served as the main development environment for writing, compiling, debugging, and flashing embedded C code into the master
Tera Term	UART interface	A serial terminal application used to interface with the STM32 via UART for real-time debugging, data visualization, and data logging during testing phases
KiCad	Circuit design	Used for the design and analysis of circuit schematics during early development phases and testing of hardware integration concepts
VSCoDe Python	Data analysis	Used to process and analyze UART-exported data from the BMS, enabling the creation of performance plots and side-by-side comparison with Simulink results

Chapter 4

Model-Based Simulations

This chapter presents the model-based simulations developed to analyze and validate the internal behavior of the BMS prior to integration with real acquisition hardware. The estimations and logic evaluated in this work are executed using software environments such as Simulink, and directly on the STM32F446RET6 microcontroller, representing the Master unit of the BMS [40].

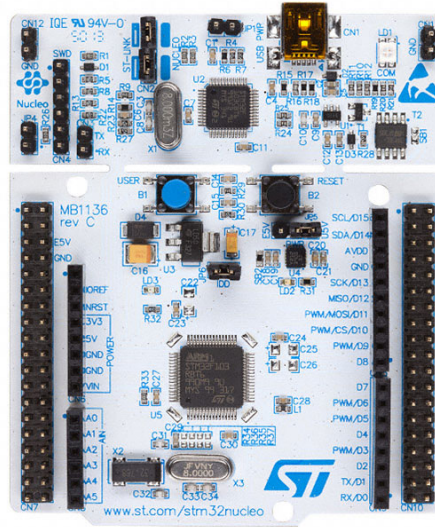


Figure 4.1: STMicroelectronics STM32F446 Nucleo-64 [41].

The goal of this stage is to confirm that all model parameters and estimation logic are defined and calibrated, forming a consistent base to proceed with real acquisition using the slave board. The data produced by the controller is exported via UART and processed using MATLAB, which serves for visualization and comparison purposes. By plotting the results alongside reference curves derived from literature or known behaviors, it becomes possible to evaluate the correctness and stability of the implemented models.

4.1 External Communication: BMS to PC Interface

To properly understand the communication structure that will be established between the Master microcontroller and the Slave module in the following sections, it is essential to clarify the external communication interface implemented in the first stages of the project. This interface continues to play a fundamental role in all subsequent exercises, as it supports configuration, control, and validation of all implemented algorithms during the testing and simulation processes. The communication link between the embedded system and the PC has served as the primary medium for data acquisition, data management and debugging.

As previously mentioned, the communication between the BMS and the PC is handled through a UART (Universal Asynchronous Receiver/Transmitter) protocol configured on the USART2 peripheral of the STM32F446RE microcontroller, illustrated in Figure 4.2. This interface enables real-time bidirectional communication and the baud rate of 1000000 bps (1Mbps) because that is the normal rate from the slaves normal operation. This permits transmitting all relevant information from the microcontroller to the external system for visualization and analysis.

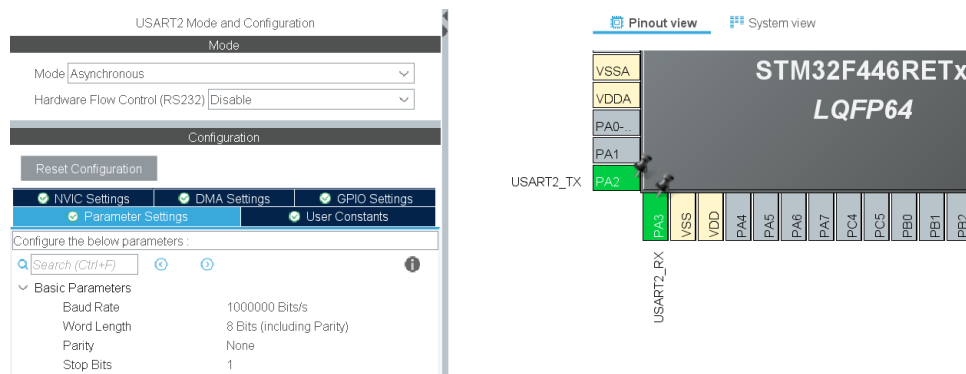


Figure 4.2: USART2 pin mapping for UART communication between the STM32 and the PC.

The terminal communication was initially managed using a serial console such as Tera Term to validate raw data transfer and string formatting. However, as the complexity of the system increased, a more structured HMI was developed in Python using the Visual Studio Code environment to offer more advanced visualization capabilities. Additionally, the HMI includes input fields where simulation parameters can be configured, such as initial SOC, battery capacity, discharge current and time steps, all of which are then sent to the microcontroller via the same UART interface.

Parallel to the HMI, the VSCode was also used extensively during the testing phase for result validation and comparative analysis, for printing graphics via Python language. Values transmitted from the STM32 were collected from Tera Term terminal to text files and using dedicated scripts it was possible to enable graphical visualization in plots and compare with simulation data produced in Simulink or datasheet values. The USART2 interface forms the backbone of external interaction with the embedded system and remains central to data validation, algorithm validation, and system debugging.

4.2 Initial Battery Modeling and Validation

The objective of the first test was to validate the embedded implementation of the battery model against a known reference provided by Simulink. This validation focused on confirming that the software developed for the STM32F446RE microcontroller could replicate the expected behavior of a standard battery model under controlled discharge conditions. The aim was not only to verify the numerical consistency but also to ensure that the implementation is robust and flexible enough to support different parametrizations.

To ensure proper validation, the results obtained were compared with both the datasheet values and with reference values presented in this simulation report [42], since both the modeling approach and the parameters used were derived from that same source. By achieving similar outcomes, it is possible to confirm that the simulation structure and the embedded logic are correctly implemented and ready for the next steps.

4.2.1 Battery Block Parametrization

To initiate the validation process, it was necessary to select a representative lithium-ion cell that could serve as the basis for the simulation. For this purpose, the INR-18650-P28A cell, currently used by the FSFEUP team, was chosen [43].

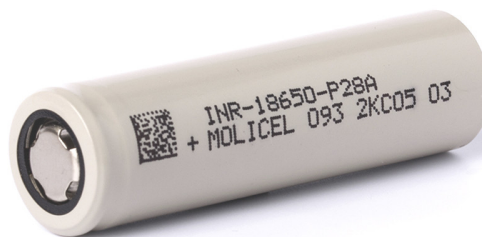


Figure 4.3: Molitel INR-18650-28P, from [44].

Its datasheet provides essential parameters that were used to configure the simulation environment, as shown in the Table 4.1.

Table 4.1: Molicel model INR-18650-P28A datasheet cell characteristics, from [43].

Parameter	Value
Capacity (Typical / Minimum)	2800 mAh / 2600 mAh
Energy (Typical / Minimum)	10.3 Wh / 9.56 Wh
Nominal Voltage	3.6 V
Charge Voltage	4.2 V
Cut-off (Discharge) Voltage	2.5 V
Standard / Max Charge Current	2.8 A / 6.0 A
Max Discharge Current	35 A
Charge Time (Standard)	1.5 hour
Internal Resistance (AC - 1 KHz)	20 mΩ (Max)
Internal Resistance (DC - 10A/1s)	20 mΩ (Max)
Ambient Temperature (Charge)	0°C to 60°C
Ambient Temperature (Discharge)	-40°C to 60°C
Energy Density (Volumetric)	610 Wh/l
Energy Density (Gravimetric)	220 Wh/kg

This block allows for the accurate representation of battery characteristics by utilizing manufacturer data.

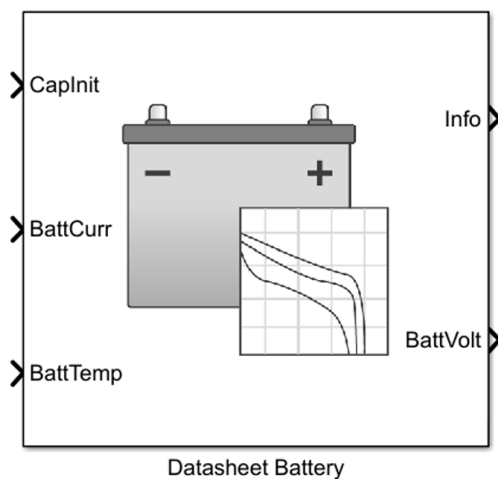


Figure 4.4: Datasheet battery block from Simulink, from [45].

This block computes the battery output voltage using internal lookup tables for the open-circuit voltage (V_{oc}) and internal resistance (R_{int}), which are expressed as functions of the state of charge (SOC) and temperature. The battery output voltage is then calculated using the following equation:

$$V_{out} = V_{oc} - (I_{batt} \cdot R_{int}) \quad (4.1)$$

and can be represented by this circuit:

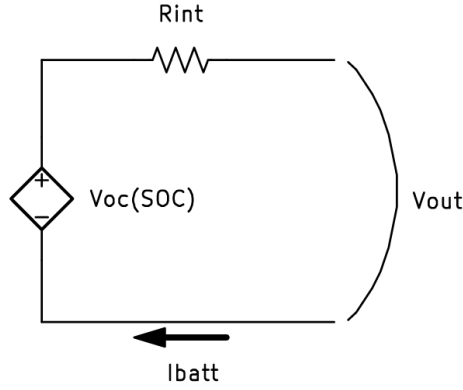


Figure 4.5: Battery model used by the Simulink battery block.

Internally, the SOC is tracked using the Coulomb counting method, according to:

$$SOC = SOC_0 - \frac{1}{Cap_{batt}} \int_0^t I_{batt} dt \quad (4.2)$$

where SOC_0 and Cap_{batt} represent the initial SOC and capacity of the battery respectively, and the variable I_{batt} represents the current flowing from the battery, where a positive current indicates battery discharge and a negative current indicates battery charge.

Based on the relationship between the Depth of Discharge (DoD) and the State of Charge (SOC), it is possible to determine the extracted charge from the cell (C_{ext}) using a known SOC value. The DoD is defined as:

$$DoD(\%) = \frac{C_{ext} \times 100}{C_{batt}} \quad (4.3)$$

and since:

$$SOC(\%) = 100 - DoD(\%) \quad (4.4)$$

the equations can be rearranged, resulting in:

$$C_{ext} = C_{batt} \cdot \left(1 - \frac{SOC}{100}\right) \quad (4.5)$$

This formula was used to convert the SOC values computed during the discharge test into accumulated capacity values (in mAh), and to fulfill the goal of validating the embedded implementation, it was necessary to simulate the behavior of the open-circuit voltage (V_{oc}) over time during a real discharge process. For this purpose, the function proposed in [46] was adopted due to its demonstrated accuracy when applied to Li-ion cell parametrization.

$$V_{OC}(SOC) = a \cdot \left(\exp^{(1-b \cdot SOC)} \right) + c \cdot SOC^3 + d \cdot SOC^2 + e \cdot SOC + f \quad (4.6)$$

The coefficients used in this function were determined through curve fitting in [42] and are listed in Table 4.2, ensuring that the curve closely matches the datasheet values.

Table 4.2: Coefficients used in the $V_{OC}(SOC)$ fitting, from [42].

Coefficient	Value
a	-0.3451
b	0.0992
c	6.3473×10^{-7}
d	-4.0329×10^{-5}
e	0.0044
f	3.4137

The results demonstrate a high level of accuracy, with minimal deviation between the fitted model and the actual datasheet data. The errors between the two curves are summarized in Table 4.3, confirming that the parametrization is suitable for reliable estimation.

Table 4.3: Error metrics between fitted model and datasheet curve.

Metric	Value
Max error	2.0690%
Mean error	0.1468%
Standard deviation	0.3355%

This polynomial was applied to fit the V_{OC} curves from the datasheet, enabling proper battery behavior approximation inside the simulation environment.

4.2.2 Battery Block validation

To verify the validity of the models implemented and the parameters defined in the previous section, it was necessary to conduct a direct comparison with the discharge behavior presented in the battery's official datasheet. The discharge curves from the cell datasheet provide open circuit voltage vs capacity data under different constant current scenarios, serving as a reliable benchmark for the simulation results.

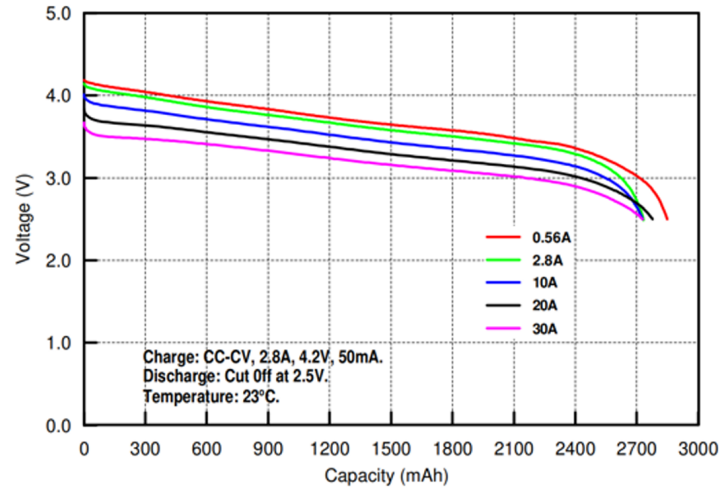


Figure 4.6: Discharge rate characteristics graph from cell datasheet [43].

As shown in Figure 4.6, the datasheet provides discharge characteristics for five different constant currents: 0.56 A, 2.8 A, 10 A, 20 A, and 30 A. To ensure a valid comparison, the embedded implementation used the following fixed parameters: an initial SOC of 100%, a total battery capacity of 2800 mAh, and a cut-off voltage of 2.5 V. The internal resistance R_{int} was set to 20 m Ω , as specified in Table 4.1. The open-circuit voltage values were computed using the previously fitted $V_{OC}(SOC)$ function from Table 4.2, and the output voltage was calculated in real time using the equation 4.15.

The results generated by the STM32F446RE microcontroller were logged and exported for comparison with the datasheet. The corresponding discharge curves obtained through simulation are shown in Figure 4.7, using 2800 mAh as the total cell capacity.

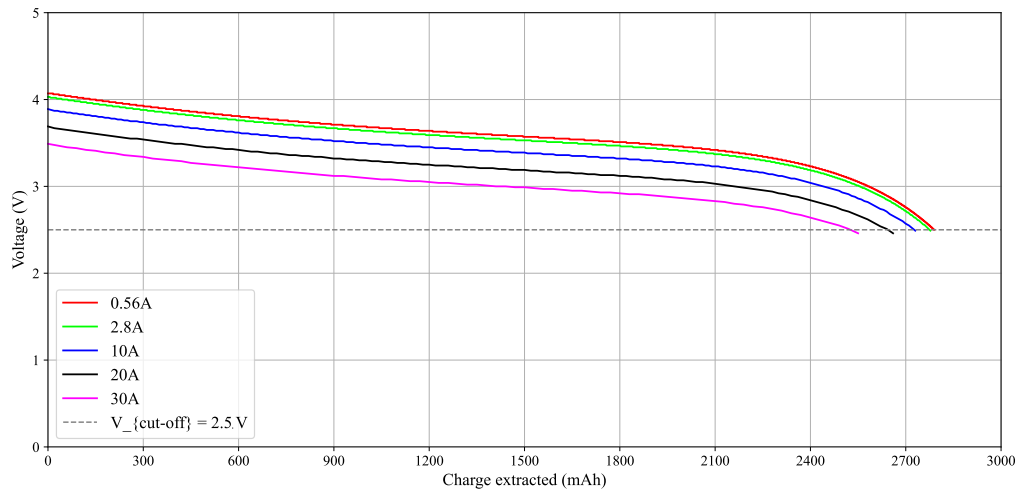


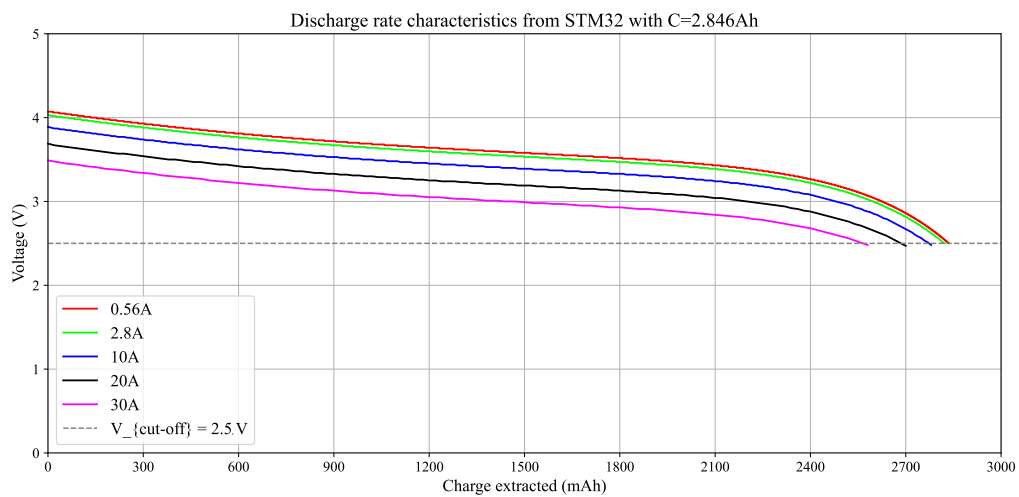
Figure 4.7: Discharge rate characteristics graph from Matlab with C=2800 mAh.

The comparison between the extracted charge from the datasheet and the simulated values is presented in Table 4.4.

Table 4.4: Comparison of extracted capacity using $C=2800$ mAh.

Current (A)	Datasheet charge extracted (mAh)	Simulated charge extracted (mAh)
0.56	2846.53	2789.79
2.8	2728.53	2777.57
10	2728.53	2729.99
20	2772.68	2660.00
30	2728.53	2550.00

It was observed that the simulation overestimated the extracted capacity at lower currents, like in the 0.56A current reaching 2846.53 mAh when in the the simulation the cell capacity was 2800 mAh. To address this discrepancy and more accurately reflect the real behavior observed in the datasheet, a second simulation was conducted using 2846 mAh as the battery capacity to match the real extracted charge from the 0.56 A curve. The updated discharge curves are presented in Figure 4.8.

Figure 4.8: Discharge rate characteristics graph from Matlab with $C=2846$ mAh.

And this is the comparison between the extracted charge from the datasheet and the simulated values is presented in Table 4.5.

Table 4.5: Comparison of extracted capacity using $C=2846$ mAh.

Current (A)	Datasheet charge extracted (mAh)	Simulated charge extracted (mAh)
0.56	2846.53	2835.78
2.8	2728.53	2822.38
10	2728.53	2780.01
20	2772.68	2700.00
30	2728.53	2580.00

The results obtained in this second simulation, using a total cell capacity of 2846 mAh, demonstrated a closer alignment with the reference curves from the datasheet. Despite this improvement, a perfect match was still not achieved across all currents. This discrepancy can be attributed to several factors inherent to the test conditions and the implementation method.

- Firstly, all discharge simulations were executed within the same embedded environment and exported in parallel, meaning that every current profile was sampled and recorded simultaneously. Although the sampling time used was small enough to capture rapid variations in voltage, the limitation of processing all test scenarios in parallel resulted in an unavoidable lack of precision in stopping the discharge exactly at the 2.5 V threshold. This effect was especially noticeable in the higher current scenarios, where voltage drops more abruptly. As a consequence, some of the discharge curves cross slightly below the expected cutoff value, introducing minor deviations in the final capacity measured.
- Secondly, to better understand the behavior of the curves, it is essential to analyze the Coulomb counting model in use. This method estimates the battery capacity by integrating the discharge current over time, linking charge depletion directly to the variation of cell voltage. The integration occurs from an initial fully charged state until the terminal voltage reaches a predefined lower limit, here set to 2.5 V. In this context, the minimum voltage reached (V_{min}) corresponds to the maximum extracted capacity (C_{max}), while the maximum initial voltage (V_{max}) represents the minimum capacity (C_{min}). This linear relationship between voltage and capacity, intrinsic to the Coulomb counting approach, creates a direct mapping of capacity over voltage. This behavior is illustrated in Figure 4.5, which represents the theoretical model of the coulomb counting method, a simple equivalent circuit model. The configuration models the battery using a voltage source dependent on $V_{OC}(SOC)$, in series with an internal resistance R_{int} , with the current I_{batt} driving a constant current sink. The integrator accumulates current over time to compute capacity.

What becomes evident through this representation is that, assuming a constant discharge current, the variation in capacity is linearly related to the time domain, and because $V_{OC}(SOC)$ is a function of SOC, the voltage also evolves linearly with extracted charge. Therefore, independently of the current value applied, the voltage drop over time maintains a consistent slope. This is clearly observable in the simulation results: despite each discharge current yielding slightly different total capacities, the slope of the voltage curves remains nearly unchanged. However, in reality, the battery behavior under different currents is not this linear, as shown in the discharge curves from the INR-18650-P28A datasheet (Figure 4.6).

In summary, while minor discrepancies exist between the simulation and datasheet result, the observed voltage behavior is fully consistent with the theoretical model. These findings validate the implementation and confirm that the embedded model reliably reproduces the dynamic behavior of the INR-18650-P28A cell. This solid foundation supports the integration of more advanced estimation algorithms and real-time acquisition systems described in the subsequent phases of this work.

4.2.3 HMI configuration

Knowing that there are an enormous amount of lithium-ion cells available in the market and the fact that a Battery Management System (BMS) must be adaptable to different chemistries and specifications, the embedded software was developed with flexibility in mind.

User Interface for Li-Ion Cell Battery Model
Parametrization (Exercise 1)

General Battery Parameters

Internal Resistance (Ohms)

0.02

Max. Cell Capacity (mAh)

2846

Discharge Current (A)

2.8

Initial SOC (%)

100

Time Step (s)

1.0

Cut-off Voltage (Vout_min)

2.5

Electrical Model Coefficients

a

-0.3451

b

0.0992

c

6.3473e-07

d

-4.0329e-5

e

0.0044

f

3.4137

Parametrize

Figure 4.9: HMI interface used to configure cell parameters.

As shown in Figure 4.9, the HMI interface allows the user to input key parameters directly before starting the simulation or real-time operation. This configurability enables the system to adapt to the specific characteristics of the cells under management. While the INR-18650-P28A cell was used as a reference in this test, the structure of the software ensures that the same logic can be applied to any compatible cell model by updating the initial configuration via the HMI. For validation purposes, the parameters used in the Simulink simulation were manually replicated in the firmware running on the STM32F446RE microcontroller, ensuring that both environments shared identical conditions.

It is also important to note that different SOC estimation models require their own specific parameters and internal logic. While the current implementation has been validated for the Coulomb Counting method, future extensions using other estimation strategies will demand additional configuration steps, and software modifications.

4.3 Hybrid Estimation Model

As previously discussed, no two cells behave identically. Each lithium-ion cell follows its own discharge pattern, influenced by internal chemistry, manufacturing variations, and operating conditions. Due to this inherent variability, it becomes impossible to compute exact values for SOC or other internal states using purely deterministic methods. As a result, estimation algorithms are essential, serving to approximate the battery’s internal behavior as closely as possible. Some estimation strategies are more sophisticated and accurate than others, and the choice of model can significantly affect the reliability and performance of a BMS.

The estimation approach adopted in this work is based on a hybrid battery model that combines two elements: a KiBaM and a simplified electric circuit model. This hybrid configuration offers improved adaptability and accuracy under different operating conditions. It leverages the high precision of VOC-based estimation during rest periods and the responsiveness of the KiBaM model during active current flow, while the electric model provides the interface to compute the output voltage considering internal resistance effects.

This model was selected for its advantages discussed in Section 2.4.3.7, including a good trade-off between computational simplicity and physical realism. Figure 2.17, already introduced in the state of the art, illustrates the KiBaM principle, where the battery is represented as two interconnected tanks symbolizing the available and bound charges. This physical analogy can be translated into an equivalent electrical representation, as shown in Figure 4.10.

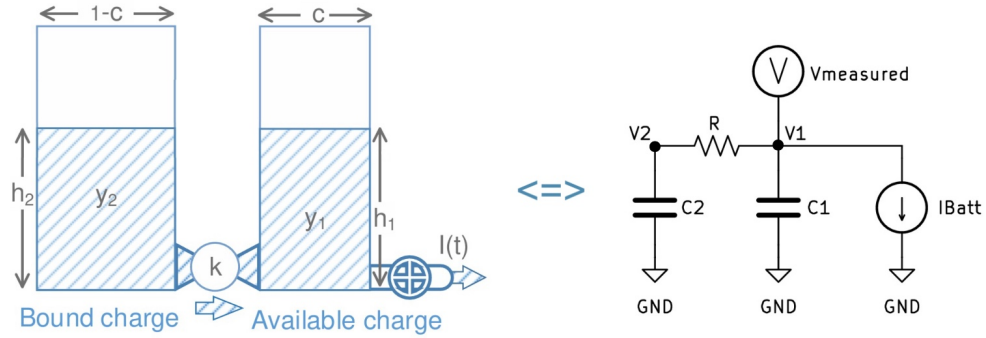


Figure 4.10: Equivalent electrical representation of the KiBaM system, adapted from [20].

As explained in [20], Table 4.6 summarizes the correspondence between the physical variables of the hydraulic model and their electrical counterparts.

Table 4.6: Equivalence between KiBaM and electrical circuit parameters, adapted from [20].

KiBaM Representation	Symbolic Variable	Symbolic Variable	Electrical Representation
Available charge	y_1	Q_1	Available charge
Bound charge	y_2	Q_2	Bound charge
Tank height 1	h_1	V_1	Capacitor voltage 1
Tank height 2	h_2	V_2	Capacitor voltage 2
Valve conductivity	k	$1/R$	Conductance
Tank capacity 1	c_1	c_1	Capacitor 1 capacity
Tank capacity 2	c_2	c_2	Capacitor 2 capacity
Load flow	I_{Batt}	I_{Batt}	Load current

Where the charge reservoirs are modeled as capacitors, the transfer valve k as a resistor, and the load current as a sink. This transformation allows the dynamic behaviour of the tanks to be described through differential equations that are formally equivalent to those of an RC electrical circuit.

Building on this analogy, the complete hybrid model integrates the KiBaM-based SOC estimation with the open-circuit voltage function and an output resistance, leading to Figure 4.11. This representation highlights how the dynamic SOC estimation is coupled with the electrical output stage to reconstruct the terminal voltage of the battery.

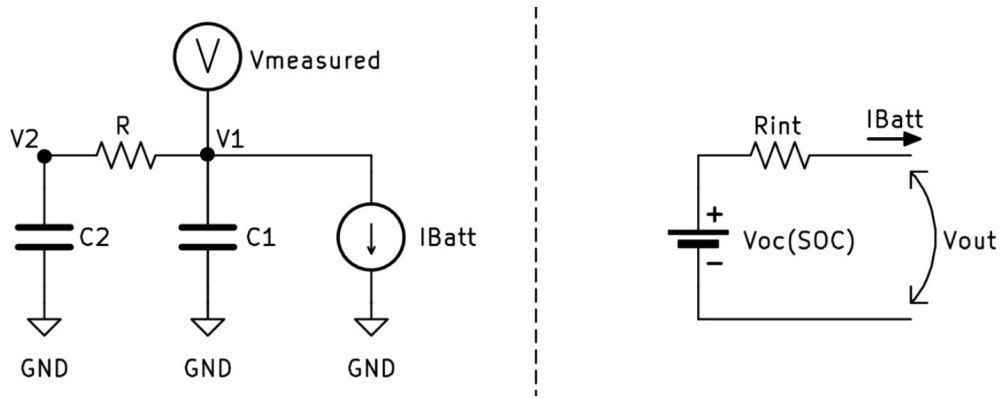


Figure 4.11: Hybrid battery model representation, adapted from [47].

The following subsections present the development of this model, beginning with a description and initial validation of the KiBaM module. This is followed by the formulation of the VOC estimation approach and the switching logic between both models, based on the analysis of voltage variation over time. Finally, the electrical model is discussed in detail, including the calibration of electrical parameters and a reflection on the hybrid model's performance in simulation.

4.3.1 KiBaM Model Estimation

The KiBaM is a widely used approach for estimating the SOC in lithium-ion batteries due to its ability to capture the dynamic behavior of charge availability within a cell. This model is based on the analogy of two interconnected charge tanks: one representing the readily available charge and the other representing bound charge, which becomes accessible over time, as Figure 4.12 shows. The distribution of total battery capacity across these two reservoirs is governed by two capacity fractions, commonly denoted as c_1 and c_2 , where $c_1 + c_2 = 1$. These coefficients represent the proportion of charge stored in each tank, with c_1 associated with the available charge tank and c_2 with the bound charge tank.

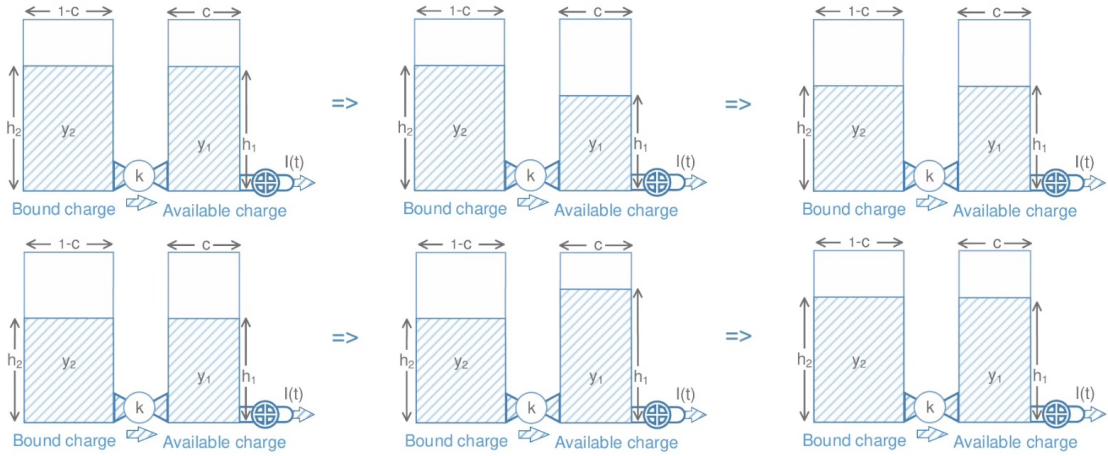


Figure 4.12: Graphical representation of the KiBaM analogy under discharge and recharge processes.

As shown in Figure 4.12, when a discharge current is applied, the level of the available tank decreases more rapidly than the bound tank, because the transfer of charge through the valve is slower than the rate at which current is extracted. Conversely, when the current is removed, the difference in fluid heights drives a redistribution between the two tanks until equilibrium is restored. This dynamic explains why the KiBaM can accurately capture both the immediate response of the cell and the delayed release of charge from diffusion-like processes.

The total capacity of the battery, C , is split accordingly such that the initial charge in each tank is given by $y_1(0) = c_1 \cdot C \cdot SOC_0$ and $y_2(0) = c_2 \cdot C \cdot SOC_0$, where SOC_0 is the initial state of charge. As the battery discharges, the current I is drawn from the available charge tank, and charge flows between the two tanks at a rate determined by the kinetic constant k , which controls the transfer speed of energy from the bound to the available reservoir.

To describe the time evolution of the system, the KiBaM model defines the analytical expressions for the state of each tank as functions of time t under constant current discharge. These expressions are given by:

$$y_1(t) = y_{1,0}e^{-k't} + \frac{y_0k'(c_1 - 1)(1 - e^{-k't})}{k'} - \frac{Ic_1(k't - 1 + e^{-k't})}{k'} \quad (4.7)$$

$$y_2(t) = y_{2,0}e^{-k't} + y_0c_2(1 - e^{-k't}) - \frac{Ic_2(k't - 1 + e^{-k't})}{k'} \quad (4.8)$$

where $y_1(t)$ and $y_2(t)$ represent the instantaneous charge in the available and bound tanks, respectively, and y_0 is the total initial charge. The parameter k' is a derived constant defined as:

$$k' = \frac{k}{c_1 \cdot c_2} \quad (4.9)$$

Incorporating both the kinetic constant and the charge partitioning. These expressions allow for the continuous tracking of energy flow and the redistribution of charge between the two reservoirs throughout the discharge process.

In order to compute the SOC at a given time, the contributions of both tanks are combined. The total SOC is derived from the sum of the charge in each tank normalized by the total battery capacity:

$$SOC(t) = \frac{y_1(t) + y_2(t)}{C} \cdot 100 \quad (4.10)$$

Additionally, the individual states of each tank can be expressed in terms of their respective SOC contributions:

$$SOC_{tank1}(t) = \frac{y_1(t)}{c_1 \cdot C} \cdot 100 \quad \text{and} \quad SOC_{tank2}(t) = \frac{y_2(t)}{c_2 \cdot C} \cdot 100 \quad (4.11)$$

Another important concept within the KiBaM framework is the notion of “height” in each tank and these are computed as:

$$h_1(t) = \frac{y_1(t)}{c_1} \quad \text{and} \quad h_2(t) = \frac{y_2(t)}{c_2} \quad (4.12)$$

These height variables are not used directly in the SOC estimation but play a crucial role in understanding how charge dynamically moves between the two tanks, since the rate of internal transfer is driven by the height difference $h_1(t) - h_2(t)$.

In the embedded implementation, this model is updated periodically based on the measured discharge current and sampling time. The previously estimated charge in each tank ($y_{1,prev}$ and $y_{2,prev}$) is stored and used in each iteration to compute the next state. The use of precomputed exponentials and differential equations ensures efficient execution even on resource-limited microcontrollers like the STM32F446RE. This implementation allows for accurate tracking of the state of charge under variable load conditions, accounting for both immediate and delayed energy availability.

4.3.2 Open-Circuit Voltage (Voc) Estimation

Another widely adopted method for estimating the SOC in lithium-ion cells is based on the OCV. This approach leverages the fact that, after a sufficient rest period, the terminal voltage of the battery converges to a value uniquely correlated with its internal SOC. One of the key advantages of this method is its independence from the current flowing through the cell. In contrast to dynamic models like KiBaM, which rely heavily on current integration and are sensitive to measurement noise and transient behaviors, the OCV method provides a highly accurate estimate unaffected by external influences.

However, this precision comes with a limitation. Since the OCV only stabilizes after a period of rest (when $\frac{dV}{dt} \approx 0$) it cannot be used continuously during active operation. Consequently, this estimation method is best suited for scenarios where the battery has been idle long enough to reach voltage equilibrium. Despite this constraint, its accuracy makes it especially useful in two important moments within the hybrid estimation framework implemented in this work.

First, the OCV-based estimation is used to provide the initial SOC required to initialize the KiBaM model. This ensures that the model starts from a trustworthy and physically consistent state. Second, and more importantly, whenever the system detects that the battery has reached a stable condition, the estimation switches from the KiBaM method to the OCV approach, which means that the switching logic will always prioritize the OCV method over KiBaM. This ensures that any accumulated drift in the dynamic model can be corrected in real time, improving the long-term reliability of the SOC tracking.

The OCV-based SOC estimation is computed from the open-circuit voltage using a fitted logistic function, given by:

$$SOC(V_{OC}) = \frac{A}{1 + \exp(-B \cdot (V_{OC} - C))} \quad (4.13)$$

Witch is a adapted function from the original Equation 4.6.

The coefficients used in this equation were obtained through curve fitting to the discharge voltage curve of the INR-18650-P28A cell and are listed in Table 4.7.

Table 4.7: Coefficients used in the $SOC(V_{OC})$ fitting function.

Coefficient	Value
A	108.7827
B	4.50
C	3.6491

This hybrid approach ensures that the battery management logic benefits from the responsiveness of the KiBaM model during dynamic operation while maintaining high accuracy and correction capability through the application of the OCV-based estimation.

4.3.3 Eletric Model Estimation

In order to simulate the behavior of a lithium-ion cell without relying on real-time hardware measurements, it is necessary the estimation of two key quantities: the open-circuit voltage (V_{OC}) and the output voltage (V_{out}), based on the estimated SOC and the applied current. Both of them were already used within the context of Coulomb Counting, and the equations are:

$$V_{oc}(SOC) = a \cdot \left(e^{(1-b \cdot SOC)} \right) + c \cdot SOC^3 + d \cdot SOC^2 + e \cdot SOC + f \quad (4.14)$$

$$V_{out}(t) = V_{oc}(SOC) - I_{batt}(t) \cdot R_{int} \quad (4.15)$$

During the purely simulated phase of this project (where no physical cells were involved), this electric model was used to emulate the dynamic voltage behavior of the cell. The output voltage, V_{out} , served as a virtual observation point, reflecting the behavior expected at the terminals of a real battery, assuming ideal operating conditions. This enabled the visualization and analysis of model responses across varying current profiles, as if the battery were being physically exercised.

However, the real importance of this model emerges in later stages of development, when the BMS is interfaced with real battery cells. In this context, the estimated V_{out} becomes a critical benchmark against which actual cell voltages can be compared. The following expression becomes particularly relevant:

$$R'_{int}(t) = \frac{V_{oc}(SOC(t)) - V_{batt}(t)}{I_{batt}(t)} \quad (4.16)$$

Where $R'_{int}(t)$ represents the internal resistance of the battery cell estimated at a given instant t . This value is inferred by comparing the model estimation of the open-circuit voltage, $V_{oc}(SOC(t))$, with the actual voltage measured directly from the cell under load, denoted as $V_{batt}(t)$. The denominator, $I_{batt}(t)$, corresponds to the current flowing through the cell at that same moment.

This comparison between V_{oc} and V_{batt} enables effective diagnosis. If the cells were ideal and the model perfectly accurate, both values would converge. Any deviation, however, is indicative of real-world cell behavior diverging from the model's assumptions. These deviations can then be interpreted to estimate the cell's aging profile, health status, and possible abnormal operation. Consequently, the electric model not only supports simulation, but will also serve as a key tool for real-time monitoring and adaptation in the complete BMS implementation. In contrast to simulation scenarios where R_{int} is kept constant, this adaptive evaluation becomes essential when the BMS operates with physical battery packs under real conditions.

4.4 Hybrid Model Validation

To validate the hybrid battery model implemented in this work, a test was performed using the predefined parameters that were used in the [42]. The primary objective of this validation is to ensure that both the Simulink reference implementation and the STM32 embedded model accurately reflect the real-world behavior of the cell, especially when compared to the datasheet values.

Table 4.8 summarizes the complete set of parameters used in the estimation process, encompassing the KiBaM model and the electric model required to simulate output voltage.

Table 4.8: Parameters used in the hybrid model validation.

Parameter	Value / Expression
Current (I_{batt})	0.56 A
Internal resistance (R_{int})	0.02 Ω
Capacity (C_{total})	2846 mAh
KiBaM model	
k	0.00015
c_1	0.9586
c_2	$1 - c_1$
k'	$k / (c_1 \cdot c_2)$
Voc estimation (logistic)	
A_{SOC}	108.7827
B_{SOC}	4.50
C_{SOC}	3.6491
Electric model (polynomial)	
a_{elet}	-0.2655
b_{elet}	0.1711
c_{elet}	2.403×10^{-7}
d_{elet}	-2.16×10^{-14}
e_{elet}	0.006642
f_{elet}	3.239

To validate the implementation, simulations were run both in Simulink and on the STM32 microcontroller using these parameters. The output voltage (V_{out}) was recorded throughout the discharge process and compared at key SOC intervals, corresponding to fixed capacity values. These reference points were obtained from the datasheet and entered via the HMI, as summarized in Table 4.9.

Table 4.9: Reference voltage values from datasheet at specific SOC levels.

SOC (%)	Charge extracted (mAh)	Datasheet Voltage (V)
100	0	4.198
80	570	3.863
60	1139	3.689
40	1708	3.520
20	2276	3.320
0	2846	2.503

Figure 4.13 and Figure 4.14 illustrate the discharge behavior of the model under the same test conditions for both Simulink and STM32 implementations. The curves show an almost identical response, validating the correctness of the hybrid model implementation on the microcontroller.

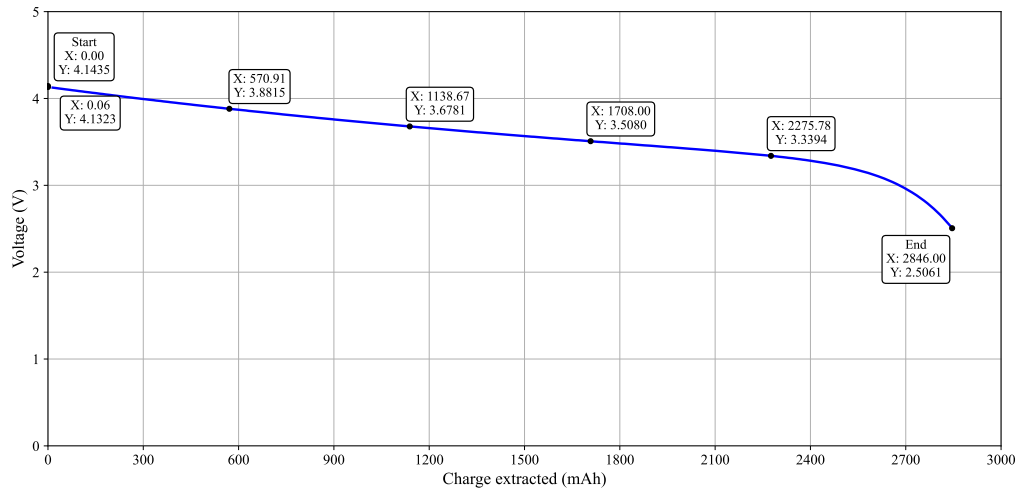


Figure 4.13: Discharge curve of the hybrid model in Simulink.

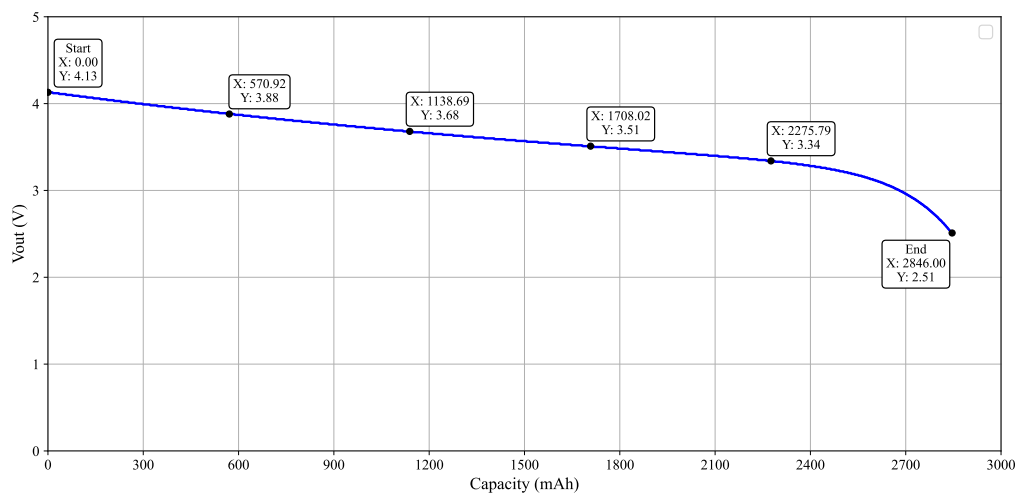


Figure 4.14: Discharge curve of the hybrid model on STM32.

To quantify the fidelity of the STM32 implementation, Table 4.10 presents the average relative error and standard deviation between the STM32 results and the Simulink reference values. These metrics confirm that the embedded implementation was well done.

Table 4.10: Error metrics between STM32 implementation and Simulink reference model.

Metric	Value
Average relative error	0.0932%
Voltage average standard deviation	0.0716%

In addition, a validation was conducted against the actual manufacturer datasheet. By comparing the simulated voltages at SOC intervals with the corresponding datasheet values, an accuracy benchmark was established. The comparison, shown in Table 4.11, demonstrates that the hybrid model reproduces the real discharge curve with minimal error.

Table 4.11: Voltage error between STM32 simulation and cell datasheet.

SOC (%)	Datasheet Voltage (V)	STM32 Voltage (V)	Absolute Error (V)
100	4.198	4.13	0.068
80	3.863	3.88	0.017
60	3.689	3.68	0.009
40	3.520	3.51	0.010
20	3.320	3.34	0.020
0	2.503	2.51	0.007

The small magnitude of the observed errors confirms the high accuracy of the hybrid estimation model, reinforcing its suitability for future tests and system integration.

Chapter 5

Master–Slave Integration and Communication

Following the implementation and validation of all internal estimation algorithms and software routines on the STM32F446RE Master microcontroller, the second development stage of this project focused on extending the system by establishing communication with an external analog front-end: the BQ79616EVM-021 Slave board [48].

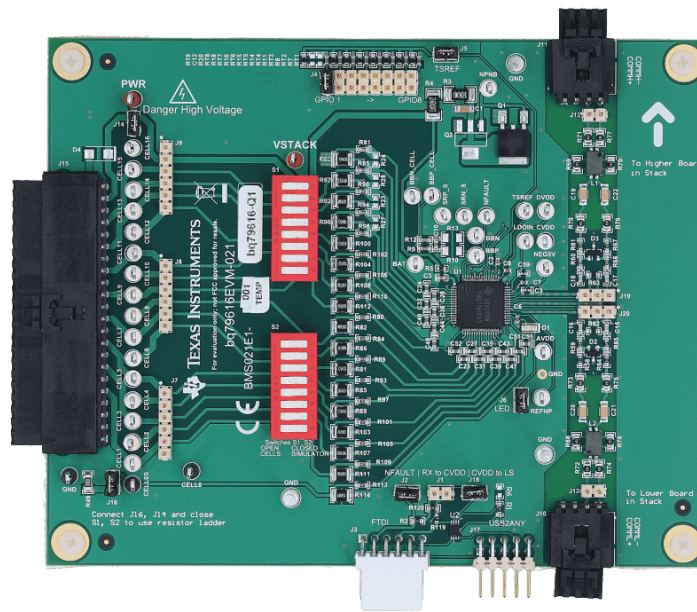


Figure 5.1: BQ79616 evaluation module (EVM), from [49].

This stage served not only to validate the physical and logical integration between the Master and Slave but also to execute and test the entire software stack previously developed, now using real data acquired through the slave’s embedded hardware components. It is important to clarify that, in this stage, no physical battery cells or external temperature sensors were connected to the system. Instead, the BQ79616EVM-021’s onboard resistor network and switching elements, were

used to simulate cell voltages. This approach enabled functional testing of the system’s acquisition chain without introducing the additional risks and variability associated with early interaction with real lithium-ion cells.

The main objective was to ensure that the estimation routines, acquisition mechanisms, and communication logic operate correctly with physical hardware, even in the absence of real battery cells. To this end, the Slave’s internal resistor ladder and auxiliary circuit elements were used to simulate a 16-cell stack, enabling comprehensive testing of the full communication and acquisition pipeline.

The chapter details each step of this integration process, beginning with physical and electrical adaptations necessary to safely connect both devices, followed by protocol configuration, command framing, and real-time data acquisition. Practical validation is also demonstrated, including the reconstruction of cell voltages from the acquired frames and their display on an external HMI. This stage represents a critical milestone in confirming the reliability and performance of the embedded communication system prior to full hardware integration with actual lithium-ion cells.

5.1 Communication Protocol Architecture

In this system, the internal communication between the STM32F446RE and the BQ79616EVM-021 slave device is established via USART1, which is configured to operate in asynchronous mode with a baud rate of 1 Mbps. This value was specifically chosen to comply with the normal operating specifications of the BQ79616, which requires a UART communication speed of 1 megabit per second for optimal performance and reliable frame transmission. To ensure full system consistency and simplify firmware integration, the same baud rate configuration was adopted for USART2. The connection between the master and slave is implemented through the STM32’s PA9 (TX) and PA10 (RX) pins for USART1, while the external communication maintains the PA2 (TX) and PA3 (RX) pins for USART2. Figure 5.2 shows the configuration panels for both USART interfaces within STM32CubeMX, confirming their respective roles.

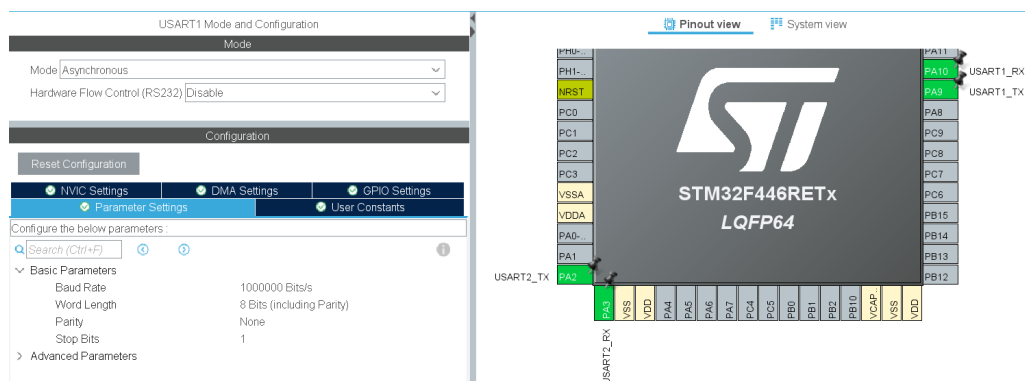


Figure 5.2: STM32F446RE UART configuration for USART1 (Master-Slave) and USART2 (PC communication).

5.2 Physical Integration and Electrical Adaptations

Before detailing the communication protocol and frame structure between the Master and the Slave, it is crucial to describe the physical integration process, as this was the foundational step in enabling reliable bidirectional communication. This integration involved addressing key electrical incompatibilities, validating the first UART frames, and ensuring the Slave module's correct initialization sequence. These steps were essential to guarantee that subsequent logic layers could be implemented successfully.

5.2.1 Slave Power-Up and Resistor Ladder Configuration

The first requirement for activating the BQ79616EVM-021 board was providing an appropriate external power source. According to the board documentation, the operating voltage for the internal circuitry and resistor ladder simulation must be applied between the “VSTACK” and “GND” test points. This configuration is intended specifically for situations where no real battery cells are present and the resistor ladder is used to simulate a 16-cell battery stack.

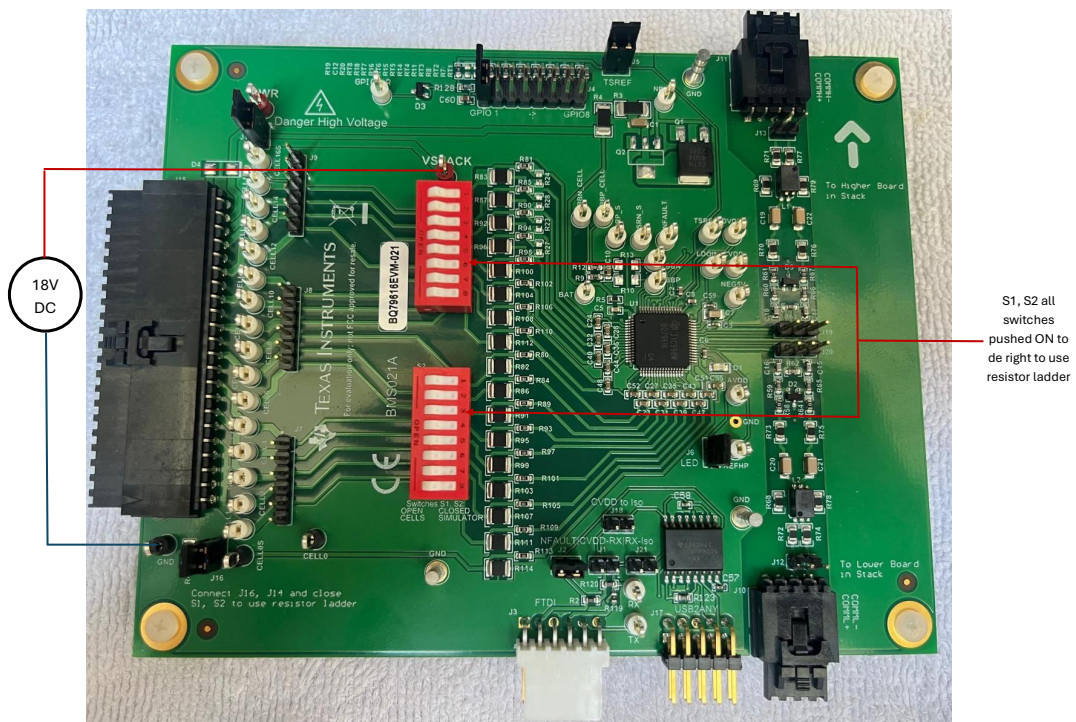


Figure 5.3: Slave configuration using the internal resistor ladder.

To ensure proper power routing from the DC source through the resistor ladder, both jumpers J14 and J16 must be closed, as they enable internal power and ground pathways. Additionally, all actuator switches on S1 and S2 must be pushed to the right (away from the IC and towards J15), placing them in the "ON" position. This action connects the on-board resistor network to the

analog front-end measurement circuitry, allowing cell voltages to be simulated through a divided DC input.

A schematic of the configured board with these key adjustments is shown in Figure 5.3, illustrating the external 18V power supply, jumper settings, and switch orientation for resistor ladder activation.

5.2.2 Voltage Compatibility and Signal Conditioning

The STM32F446RE microcontroller operates with a 3.3V logic level, whereas the BQ79616EVM-021 board was designed to communicate using 5V logic. This voltage discrepancy presented a risk of damaging the STM32's RX pin, which tolerates a maximum input of 3.6V. Consequently, a direct connection between the two devices, especially from the Slave's TX (5V) to the Master's RX (3.3V), could not be assumed safe without proper signal conditioning.

However, to safely allow reception of UART messages from the Slave to the Master, a voltage level reduction mechanism was required. This was achieved by inserting a voltage divider between the TX output of the Slave and the RX input of the Master. The chosen configuration used two resistors, one of 10k Ω and one of 4.7k Ω , effectively reducing the 5V signal to approximately 3.4V, well within the safe operating threshold of the STM32's input.

The theoretical output voltage of the divider is calculated as:

$$V_{out} = V_{in} \cdot \frac{R2}{R1 + R2} = 5V \cdot \frac{10k\Omega}{4.7k\Omega + 10k\Omega} \approx 3.4V \quad (5.1)$$

This solution enabled safe and continuous bidirectional communication and was a critical part of the physical integration strategy.

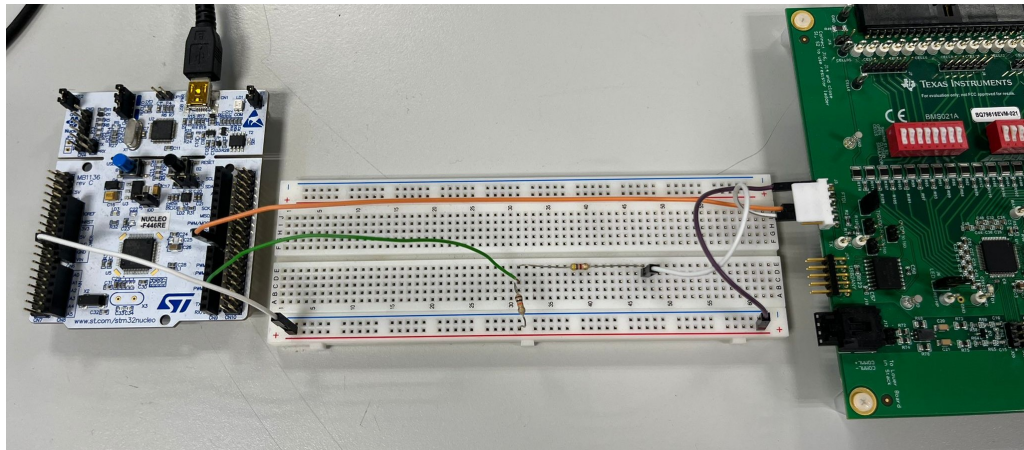


Figure 5.4: Physical implementation of the UART voltage divider between Master and Slave.

As observed in the Figure 5.4, the connection between the STM32's TX and the Slave's RX is direct, without level shifting. Despite the 3.3 V logic level of the Master, subsequent tests confirmed that the Slave reliably interpreted the logic highs and lows, indicating robust communication performance in this direction without the need for further adaptation.

5.2.3 Wake-Up Signal and Jumper Interference

Once physical wiring was complete, the Slave must first be activated from the Master through a wake-up sequence. This is achieved by transmitting a specific wake-up frame via the UART TX line, which forces the RX line of the Slave to transition from a HIGH to a LOW state for at least 2.5 milliseconds. Upon detecting this transition, the Slave must lit up the D1 light-emitting diode (LED) on the BQ79616 board. This LED serves as a diagnostic indicator confirming that the wake-up command has been successfully received and that the board is now active and prepared for further communication.

However, in early tests, this expected behavior was not observed while using the default jumper configuration on the evaluation board. Specifically, both Jumper J18 (CVDD-ISO) and Jumper J21 (RX-ISO) were originally connected, routing the RX signal through an onboard digital isolator before reaching the internal logic of the BQ79616. Although this configuration is intended to provide galvanic isolation in high-voltage systems, it proved incompatible with the UART wake-up signaling used in this project. Oscilloscope measurements revealed that even when the Master pulled the TX line fully LOW, the isolator output did not drop below approximately 0.45 volts. Since the BQ79616 only recognizes a valid LOW state when the RX voltage falls below 0.3 volts, the wake-up signal was effectively blocked by the isolator. This prevented the board from registering the condition necessary to begin communication.

To resolve this issue, both jumpers were removed to bypass the isolation stage and establish a direct electrical connection between the STM32's TX output and the Slave's RX input. Once this adjustment was made, the voltage could drop low enough to satisfy the wake-up requirement, and the D1 LED successfully turned on in response to the wake-up pulse. This confirmed that the BQ79616 had exited its standby state and was now fully responsive to UART communication. This diagnostic step revealed the importance of understanding not only communication protocols but also the default hardware protections and their impact on low-level signal behavior.

5.2.4 Initial Frame Handling and Addressing Strategy

Although wake-up was successful, as indicated by LED D1, the Slave didn't initially respond to any data frames sent by the Master. Further analysis revealed that the default configuration of the BQ79616EVM-021 placed the device in stack mode, configured to expect daisy-chained communication and not respond in standalone operation. In this default mode, the device had no assigned address and could not be addressed directly through the standard UART protocol.

To overcome this limitation, a strategy using broadcast commands was implemented. By transmitting commands using frame headers 0xC0 for reads and 0xD0 for writes to the register address 0x01 0x00, which is a read-only status register defined by the BQ79616 protocol that stores internal fault flags and diagnostic error codes, the Slave accepted commands without requiring a specific node ID. This technique enabled communication with the board even in the absence of standalone configuration and made it possible to confirm that the Slave was correctly receiving, processing, and responding to the UART frames sent by the Master.

This test sequence marks the first successful step toward full integration of the communication system in the BMS. It not only validated the electrical compatibility between the Master and the Slave but also demonstrated that, with the appropriate initialization sequence and configuration adjustments, functional communication could be established in real hardware. From this foundation, more advanced protocol handling and data acquisition logic could be reliably developed, knowing that the underlying communication link was physically sound and protocol-compliant.

5.3 Command Structure and Frame Format

Once the wake-up process is completed, data transmission follows this structured frame format, ensuring reliable and error-free communication.

- **Frame Initialization Byte (1 byte):** Indicates the type of command, either write (0xD0) or read (0xC0) and the size of it, being D3 a write comand of 4 bytes.
- **Device Address (1 byte):** Not used in broadcast communication, where the command is accepted by all Slaves regardless of address.
- **Register Address (2 bytes):** Specifies the target register in the BQ79616 memory map for the read or write operation.
- **Data Field (variable length):** Contains either the data to be written to the register (write command) or the expected length of the return message (read command).
- **CRC (2 bytes):** A 16-bit cyclic redundancy check value, calculated using a standard polynomial and appended to the end of the frame for validation. This ensures the integrity of the entire message.

Figure 5.5 outlines the generic frame format, which consists of five main fields:

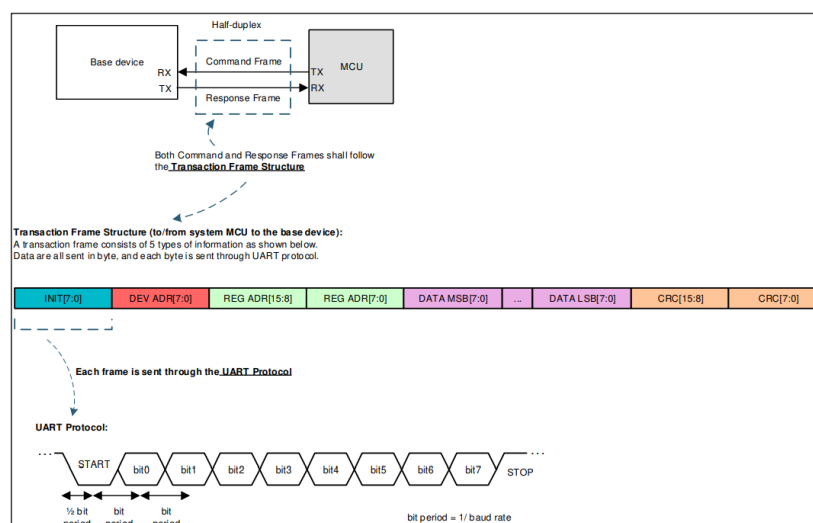


Figure 5.5: Structure of the UART command frame used for Master–Slave communication, from [50].

Once the Slave receives a valid write command from the Master, it stores the data and executes the corresponding action. In contrast, when the Slave receives a valid read command, it responds with its own frame. This return frame includes the requested data and is automatically structured by the Slave's internal UART logic. The integrity of the response is also ensured by a CRC field. It is essential to implement a short delay (hold-off time) between the command and the response to ensure that the Slave has sufficient time to interpret the request and prepare its reply.

The structure of the response frame returned by the Slave follows a similar logic, but with slightly different fields, as described below:

- **Data Length (1 byte):** Specifies the length of the data returned from the read request.
- **Device Address (1 byte):** Identifies the responding device. In this project, only a single slave is used, so the address is always the same.
- **Register Address (2 bytes):** Indicates the address of the register from which the data was retrieved.
- **Data Field (variable length):** Contains the contents of the addressed registers, returned in little-endian format.
- **CRC (2 bytes):** A 16-bit cyclic redundancy check computed over all previous bytes in the response, used to validate message integrity on reception.

This response format is used consistently across all read operations performed by the Master and plays a crucial role in validating both communication integrity and the correctness of the retrieved data. A CRC mismatch, incorrect length, or unexpected register content are all indicators of communication errors or protocol misconfiguration, which are monitored during all validation tests described in the following sections.

5.4 Communication Initialization and Slave Configuration

To ensure accurate and efficient data collection, the BQ79616EVM-021 Slave module integrates two Analog-to-Digital Converter (ADC). Each ADC has its own independent control and can be enabled or disabled separately. The main ADC is responsible for acquiring voltage and temperature measurements and the AUX ADC is mainly used during diagnostic procedures and serves as a redundancy measurement for cell voltage inputs and thermistor temperature input through the GPIOs. The Master microcontroller (STM32F446) communicates with the Slave through predefined UART commands to configure and operate the ADC, retrieve measurement data, and maintain system synchronization.

Before the data acquisition process begins, the Master must initialize and configure the ADC's within the Slave module. This is achieved by writing specific values into control registers through UART commands, defining the ADC's operational mode, input channel selection, and filtering settings. For this project, only the main ADC is going to be used because it provides all the

necessary voltage and temperature measurements required for battery management while reducing system complexity. The auxiliary ADC, which primarily serves as a redundancy mechanism for diagnostics and additional measurements, is not essential for this implementation.

The main ADC architecture of the Slave module incorporates multiple stages of signal processing before digital conversion. As shown in Figure 5.6, the cell voltage inputs (VC0 to VC16) and busbar voltage (BBP and BBN) pass through a series of conditioning circuits, including a BCI filter, a level shifter, and an anti-aliasing filter (AAF) before reaching the multiplexer (MUX). Additionally, the GPIO multiplexer (GPIO MUX) handles external sensor inputs, such as temperature readings from thermistors. The selected signals are then passed to the Successive Approximation Register (SAR) ADC, where they undergo digital conversion. After conversion, the data is routed through a de-multiplexer (De-MUX), which directs the digitized values to their respective registers for voltage, busbar, and temperature readings. A digital low-pass filter (LPF) is also available to further enhance measurement stability by reducing signal noise. This entire pipeline ensures that high-fidelity voltage and temperature data is captured and stored for subsequent processing.

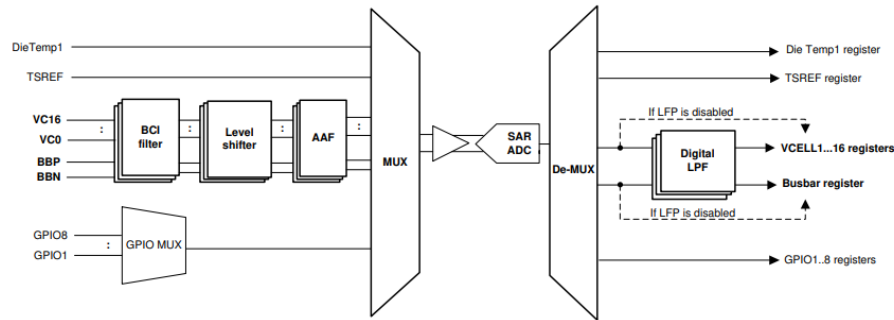


Figure 5.6: BQ79616EVM-021 main ADC measurements path, from [50].

5.4.1 ADC Configuration and Initialization

To achieve reliable and optimized performance, the main ADC must be configured according to system requirements. The configuration process is managed through two key control registers: ADC_CONF1 and ADC_CTRL1.

The ADC_CONF1 register, shown in Figure 5.7, is responsible for defining filtering configurations and setting the cutoff frequency of the low-pass filters applied to both cell voltage (VCELL) and busbar (BBP/BBN) measurements. The cutoff frequency is adjustable through the LPF_VCELL[2:0] and LPF_BB[2:0] fields, allowing different filtering options ranging from 6.5 Hz to 240 Hz, with lower frequencies providing greater noise suppression at the expense of response time. Additionally, the AUX_SETTLE[1:0] field controls the settling time for auxiliary ADC measurements, determining how long the system waits before sampling data after enabling the anti-aliasing filter (AAF). These configurations ensure that signal integrity is maintained while optimizing the trade-off between noise reduction and system response.

Address	0x0007							
NVM	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	AUX_SETTLE[1:0]		LPF_BB[2:0]				LPF_VCELL[2:0]	
Reset	0	0	0	0	0	0	0	0
AUX_SETTLE[1:0] = The AUXCELL configures the AUX CELL settling time. Each AUXCELL has to wait for the anti-aliasing filter (AAF) settling time in order to consider as a valid measurement. These bits provide the option to use different AAF or bypass an AAF to trade for a fast measurement. 00 = 4.3 ms 01 = 2.3 ms 10 = 1.3 ms 11 = Reserved								
LPF_BB[2:0] = Configures the post main SAR ADC low-pass filter cut-off frequency for BBP/N measurement. Same options as the LPF_VCELL[2:0].								
LPF_VCELL[2:0] = Configures the post ADC low-pass filter cut-off frequency for VCELL measurement. 0x0 = 6.5 Hz (154 ms average) 0x1 = 13 Hz (77 ms average) 0x2 = 26 Hz (38 ms average) 0x3 = 53 Hz (19 ms average) 0x4 = 111 Hz (9 ms average) 0x5 = 240 Hz (4 ms average) 0x6 = 600 Hz (1.6 ms average) 0x7 = 240 Hz								

Figure 5.7: ADC_CONF1 register configuration, from [50].

For the ADC_CONF1 register, the default configuration was used throughout the entire project, which sets the low-pass filter cutoff and settling parameters to their factory values. The corresponding command sent by the Master is:

0xD0 0x00 0x07 0x00 CRC1 CRC2

The ADC_CTRL1 register, depicted in Figure 5.8, is used to define the main ADC’s operational mode and execution flow. The MAIN_MODE[1:0] field allows selection between different operating modes, including single-run mode, round-robin sampling, and continuous acquisition. For this project, continuous mode is selected, ensuring uninterrupted monitoring of battery parameters. This mode enables the ADC to perform successive conversions without requiring a restart, allowing real-time tracking of voltage and temperature fluctuations. Additionally, the LPF_VCELL_EN and LPF_BB_EN fields activate post-ADC digital low-pass filtering, further improving measurement stability by reducing signal noise. The ADC is activated by setting the MAIN_GO bit to 1, which initiates the sampling process. Once started, all configured ADC inputs are sampled in sequence, and the data is stored in dedicated memory registers until requested by the Master microcontroller.

Address	0x030D							
RW	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	RSVD	RSVD		LPF_BB_EN	LPF_VCELL_EN	MAIN_GO	MAIN_MODE[1:0]	
Reset	0	0	0	0	0	0	0	0
RSVD = Reserved								
LPF_BB_EN = Enables digital low-pass filter post-ADC conversion. LPF applies to BBP/N measurements only. The cut-off frequency is configured by ADC_CONFIG1[LPF_BB[2:0]].								
LPF_VCELL_EN = Enables digital low-pass filter post-ADC conversion. LPF applies to VCELL measurements only. The cut-off frequency is configured by ADC_CONFIG1[LPF_VCELL[2:0]].								
MAIN_GO = Starts main ADC conversion. When this bit is written to 1, all Main ADC inputs are sampled. Once the Main ADC is started, any change to the Main ADC control setting has no effect until this bit is written to 1 again. This bit is cleared to 0 in read. 0 = Ready. Writing 0 has no effect 1 = Start Main ADC								
MAIN_MODE[1:0] = Sets the Main ADC run mode. In continuous run, if user would like to stop ADC, user must read all the ADC conversion results, then stop it. ADC results are not valid before ADC is reenabled next time. 00 = Main ADC not running 01 = Single run. Run the main ADC round robin 8 times and then stop 10 = Continuous run. Continuous running the Main ADC round robin until host sends command to stop 11 = Reserved								

Figure 5.8: ADC_CTRL1 register configuration, from [50].

The corresponding command transmitted to the Slave was:

0xD0 0x03 0x0D 0x06 CRC1 CRC2

Once configured, the ADC enters its operational state, ready to acquire and store measurement values in designated memory registers. Both configuration commands were maintained unchanged throughout the entire project. They were applied during system initialization and remained active during all voltage and temperature acquisition routines, ensuring stable and consistent ADC behavior across every test stage.

5.4.2 ADC Operation and Data Processing

The data acquisition process in the BQ79616EVM-021 follows a round-robin sampling approach, ensuring a sequential and systematic measurement of multiple input channels. The integrated multiplexer (MUX) dynamically selects the appropriate input signals, directing them to the Successive Approximation Register (SAR) ADC for digital conversion.

As illustrated in Figure 5.9, each complete ADC sampling cycle consists of eight sequential conversions, with each conversion taking approximately 8 microseconds, leading to a total round-robin cycle duration of 192 microseconds. During each cycle, the ADC samples DieTemp1 (internal temperature sensor), TSREF (reference voltage), all active cell voltages (Cell1 to Cell16), busbar voltage, and GPIO-based temperature sensors (GPIO1 and GPIO8). Spare slots are reserved for future expansion or diagnostic readings, ensuring flexibility in data acquisition.

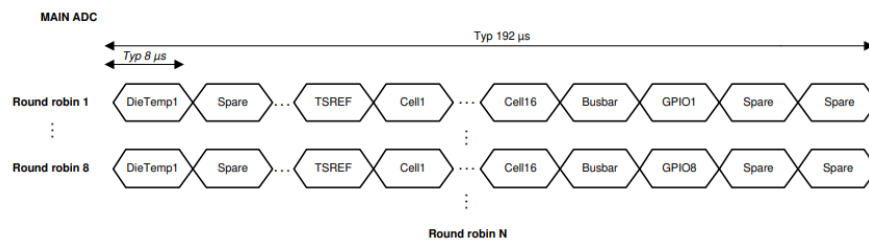


Figure 5.9: Round-Robin ADC sampling process, from [50].

Following each round-robin cycle, the digitized values are stored in dedicated memory registers within the Slave module. These registers maintain the most recent measurement values until the next acquisition cycle, ensuring that the Master microcontroller (STM32F446) always retrieves the latest data.

To guarantee real-time monitoring and data integrity, the ADC implements an automatic status flag, which signals when all active channels have been successfully sampled. Before retrieving measurement values, the Master reads this flag to verify that the data is ready, preventing unnecessary communication delays and redundant requests. This approach ensures synchronization between the Master and Slave, optimizing the efficiency of the data transfer process and maintaining high accuracy in battery monitoring operations.

5.4.3 Configuration of the Number of Active Cells

As illustrated in Figure 5.9, the ADC follows a 192-microsecond round-robin cycle, where it sequentially samples each voltage input, ensuring all active cells and busbar values are updated in real-time.

Before the ADC starts measuring, the ACTIVE_CELL register is used to define which battery cells are included in the acquisition process. This configuration allows flexibility in system setup by enabling only the required number of cell channels while ignoring inactive ones. If a cell is not active or its channel is disabled, the corresponding ADC register is automatically set to 0x8000, ensuring that the data remains structured and the sampling process continues uninterrupted. This configuration is also essential for the balancing module, as it simultaneously enables the cell balancing MOSFET (CBFET) (cell balancing (CB) channels) corresponding to the active cells, a mechanism that will be detailed further in the subsection 7.5.2.

Address	0x0003							
NVM	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	SPARE[3:0]				NUM_CELL[3:0]			
Reset	0	0	0	0	0	0	0	0
Factory OTP Reset	0	0	0	0	1	0	1	0
SPARE[3:0] = Spare								
NUM_CELL[3:0] = Configures the number of cells in series. 0x0 = 6S 0x1 = 7S 0x2 = 8S : 0xA = 16S Unused code defaults to <i>CHIP_TYPE</i> [MAX_CH1:0] setting (in factory trim). If the NUM_CELL setting has more channels than the device offers, it would be capped to highest number of channel the device offers.								

Figure 5.10: ACTIVE_CELL command and configuration, from [50].

In this project, two different ACTIVE_CELL configurations are used:

- For the simulation phase using the internal resistor ladder of the slave board (16 simulated cells), the following command is sent:

```
0xD0 0x00 0x03 0x0A CRC1 CRC2
```

- For real tests using the actual physical battery setup (6 cells), a reduced configuration is applied with the following command:

```
0xD0 0x00 0x03 0x00 CRC1 CRC2
```

These configurations ensure that the system always acquires data from the correct number of cells, avoiding unnecessary sampling of unused channels and preparing the system for accurate measurement and safe balancing control.

5.5 Voltage Acquisition and Validation

Once the ADC completes the conversion, the digitized values are stored in dedicated registers. Each voltage measurement is represented by two separate registers: VCELL*_HI and VCELL*_LO, which store the high and low bytes of the 16-bit ADC output, respectively.

Finally, when the Master microcontroller (STM32F446) requests a voltage measurement, it reads the corresponding VCELL registers via UART communication. The conversion process follows a standard 16-bit reconstruction, where the high byte and low byte values are combined using a bitwise shift operation and then multiplied by the ADC resolution factor of $190.73 \mu\text{V/LSB}$ to obtain the final voltage value. This allows the Master to interpret the data accurately and utilize it for battery management and safety monitoring.

To validate the complete communication and acquisition pipeline, a series of practical tests were executed using the internal resistor ladder of the BQ79616EVM-021. These tests aimed to verify the integrity of data transfer between the Master and Slave, confirm the correct configuration and operation of the ADC, and assess the accuracy of the voltage values acquired.

To verify the integrity of the communication between the Master and the Slave, a read command was issued to the VCELL registers, which hold the latest voltage measurements acquired by the Slave's main ADC. This read command follows the frame structure described previously and targets the BQ79616 register address responsible for storing the converted voltage values of the simulated battery cells.

The complete command frame transmitted by the Master is shown below:

0xC0 0x05 0x68 0x1F CRC1 CRC2

Where:

- **0xC0** – Frame initialization byte indicating a read command.
- **0x05 0x68** – Register address corresponding to VCELL, where cell voltage data is stored.
- **0x1F** – Number of bytes to be read (31 bytes = 16 cells × 2 bytes).
- **CRC1 CRC2** – Two CRC bytes automatically computed in the STM32 firmware for message integrity.

Following this request, the Slave responded with a frame structured according to the BQ79616 UART protocol. The response contained not only the raw voltage values, but also confirmation of the addressed register and device ID. Each voltage value is encoded in little-endian format, consistent with the device's memory layout, which means that the least significant byte (low byte) is transmitted first, followed by the most significant byte. Furthermore, the voltages are returned in descending cell order, starting from VCELL16 and ending with VCELL1, which means the first voltage pair in the frame corresponds to the highest-indexed cell. The full response frame is as follows:


```

0x1F    0x01    0x05    0x68    VCELL16_HI    VCELL16_LO    ...    VCELL1_HI
                                VCELL1_LO    CRC1    CRC2

```

Where:

- **0x1F**: Total number of bytes returned in the response (as requested).
- **0x01**: Device address of the responding Slave.
- **0x05 0x68**: Register address from which the data was read (VCELL).
- **[0x17, 0x77] to [0x17, 0x6E]**: 16 cell voltage measurements, with each voltage encoded as a 16-bit word (high byte first).
- **0x50 0x59**: 16-bit CRC checksum, verifying that the frame content is valid.

A consolidated visual summary of the frame sent and the frame received is provided in Figure 5.11.

```

[Sent]    0x00 0x05 0x68 0x1F 0x42 0x20
[Received] 0x1F 0x01 0x05 0x68 0x17 0x77 0x17 0x76 0x17 0x77 0x17 0x74 0x17 0x74 0x17 0x70 0x17 0x60 0x17 0x71 0x17 0x69 0x17 0x75 0x17 0x69 0x17 0x68 0x17 0x6C 0x17 0x64 0x17 0x68 0x17 0x6E 0x50 0x59

[Sent]    0x00 0x05 0x68 0x1F 0x42 0x20
[Received] 0x1F 0x01 0x05 0x68 0x17 0x77 0x17 0x76 0x17 0x77 0x17 0x74 0x17 0x74 0x17 0x70 0x17 0x60 0x17 0x71 0x17 0x69 0x17 0x75 0x17 0x69 0x17 0x68 0x17 0x6C 0x17 0x64 0x17 0x68 0x17 0x6E 0x50 0x59

[Sent]    0x00 0x05 0x68 0x1F 0x42 0x20
[Received] 0x1F 0x01 0x05 0x68 0x17 0x77 0x17 0x76 0x17 0x77 0x17 0x74 0x17 0x74 0x17 0x70 0x17 0x60 0x17 0x71 0x17 0x69 0x17 0x75 0x17 0x69 0x17 0x68 0x17 0x6C 0x17 0x64 0x17 0x68 0x17 0x6E 0x50 0x59

[Sent]    0x00 0x05 0x68 0x1F 0x42 0x20
[Received] 0x1F 0x01 0x05 0x68 0x17 0x77 0x17 0x76 0x17 0x77 0x17 0x74 0x17 0x74 0x17 0x70 0x17 0x60 0x17 0x71 0x17 0x69 0x17 0x75 0x17 0x69 0x17 0x68 0x17 0x6C 0x17 0x64 0x17 0x68 0x17 0x6E 0x50 0x59

[Sent]    0x00 0x05 0x68 0x1F 0x42 0x20
[Received] 0x1F 0x01 0x05 0x68 0x17 0x77 0x17 0x76 0x17 0x77 0x17 0x74 0x17 0x74 0x17 0x70 0x17 0x60 0x17 0x71 0x17 0x69 0x17 0x75 0x17 0x69 0x17 0x68 0x17 0x6C 0x17 0x64 0x17 0x68 0x17 0x6E 0x50 0x59

[Sent]    0x00 0x05 0x68 0x1F 0x42 0x20
[Received] 0x1F 0x01 0x05 0x68 0x17 0x77 0x17 0x76 0x17 0x77 0x17 0x74 0x17 0x74 0x17 0x70 0x17 0x60 0x17 0x71 0x17 0x69 0x17 0x75 0x17 0x69 0x17 0x68 0x17 0x6C 0x17 0x64 0x17 0x68 0x17 0x6E 0x50 0x59

```

Figure 5.11: Frame sent and received for VCELL voltage read command.

Internally, the STM32 firmware reconstructs each cell voltage using a bitwise operation:

$$V_{cell} = (HighByte \ll 8) + LowByte \quad (5.2)$$

Each result is then multiplied by the ADC resolution factor ($190.73 \mu\text{V}/\text{LSB}$) to convert from raw ADC units to volts, and then compared against theoretical expectations based on the 18V supply distributed across the resistor ladder. The expected voltage values follow an approximate linear decrement of $18\text{V}/16\text{cells} = 1.13 \text{ V}$ per cell, starting from VCELL1 down to VCELL16. The data acquired was displayed in real-time using the external HMI, confirming that all active cells reported consistent, non-zero voltages, with minimal variation across repeated measurements. Figure 5.12 presents the HMI display captured during this validation test.

User Interface for Li-Ion Cell Battery Model
Master-Slave Integration (Exercise 2)

General Battery Parameters	Electrical Model Coefficients
Number of Cells 16	a -0.3451
Max. Cell Capacity (mAh) 2846	b 0.0992
Discharge Current (A) 0	c 6.3473e-07
Time Step (s) 1.0	d -4.0329e-5
Cut-off Voltage (Vout_min) 2.5	e 0.0044
	f 3.4137

Cell Voltages

C1: 1.146 V	C2: 1.146 V	C3: 1.146 V	C4: 1.146 V	C5: 1.146 V	C6: 1.145 V	C7: 1.144 V	C8: 1.145 V
C9: 1.143 V	C10: 1.145 V	C11: 1.143 V	C12: 1.143 V	C13: 1.144 V	C14: 1.142 V	C15: 1.143 V	C16: 1.144 V

Figure 5.12: HMI interface displaying acquired VCELL voltages from the BQ79616EVM-021 slave.

The successful acquisition of cell voltages using the internal resistor ladder of the BQ79616 EVM-021 represents a critical validation step in the system's development. This stage confirmed that the system is fully capable of managing communication, configuration, and data acquisition from the analog front-end under real hardware conditions. With all communication protocols functioning correctly, data being accurately retrieved, and processing pipelines verified through the external HMI, the system is now prepared for its final integration stage. In the next chapter, the focus shifts to incorporating real battery cells and temperature sensors, enabling the prototype to operate under full physical load conditions and validate the performance of the estimation algorithms in a realistic environment.

Chapter 6

Full Prototype Integration and Acquisition System Validation

With the communication pipeline between the Master and Slave validated in the previous stage, this chapter focuses on completing the hardware integration and implementing all sensor acquisition routines required for a fully functional BMS prototype. The goal of this phase was to move beyond simulated data and operate the system using real-world inputs: actual battery cells, thermistors, and a current sensor for real time data monitoring.

This process involved two key dimensions. From a hardware perspective, it included the physical assembly of a six-cell lithium-ion battery pack, the selection and connection of temperature and current sensors, and the design of suitable acquisition circuits for accurate signal conditioning and protection. From a software perspective, the embedded code on the STM32 Master was extended to support real-time acquisition of voltage, temperature, and current measurements. This included configuring ADC channels, implementing acquisition logic for the GPIO and analog inputs, and applying conversion equations based on sensor characteristics.

This chapter begins by describing the integration of the real battery pack and the implementation of the voltage, temperature and current acquisition system, including the circuit principle and firmware processing. Afterwards, the timer-driven buffering method is introduced as a means to stabilize all sensor readings and ensure consistent operation. Finally, the accuracy of the complete acquisition system is assessed through dedicated validation tests, confirming that voltage, temperature, and current values are reliably captured within small error margins, thus providing a solid foundation for the functional testing presented in Chapter 7.

6.1 Battery Pack Assembly and Real Voltage Acquisition

As previously introduced, the cell model selected for this project is the INR-18650-P28A, a lithium-ion cell currently in use by the FSFEUP team. All discharge curves, estimation models, and parametrizations validated in earlier stages of the work were based on this specific cell, presented in Figure 4.3.

6.1.1 Battery Pack Integration

While the initial intention was to validate the voltage acquisition system using just two or three cells that are sufficient to test estimation models across their voltage range, the hardware constraints of the BQ79616EVM-021 Slave board made it necessary to revise this plan. According to the slave datasheet, the minimum operating voltage required for standard functionality is 12V. Since each INR-18650-P28A cell has a minimum operating voltage of approximately 2.5V, a series connection of at least six cells was required to meet this threshold. Therefore, a 6-cell pack was assembled for this integration stage.



Figure 6.1: Assembled 6-cell battery pack.

The battery pack was physically mounted on a custom-designed support that safely housed the cells in series. To ensure full system integration and flexibility for both monitoring and external energy flow, two distinct types of connections were required: one for voltage sensing and another for power delivery. Moreover, each of these two connection types had to be implemented twice, once for the slave board, to enable monitoring, measurement and power supply, and once for the external systems, namely the charger and discharge circuit, which will be discussed in later sections.

The voltage sensing connections were made using a set of thin signal wires. These wires were soldered to each cell's positive terminal, as well as to the overall negative terminal at the bottom of Cell 1. This same set of wires also served to power the Slave board, with the total battery voltage (from Cell 1 to Cell 6) being applied across the board's power input through the connector J15. These connections enabled the analog front-end (slave) to both measure individual cell voltages and receive its required operating voltage. Table 6.1 provides a simplified view of the essential pins used in this 6-cell configuration:

Table 6.1: Relevant pin assignments for 6-Cell connection on J15, adapted from [48].

Pin	Label	Description
1	CELL0/GND	Negative terminal of Cell 1 (GND)
2	CELL16	Positive terminal of Cell 6 (Pack +)
4	CELL0S	Sense line for Cell 1 negative
5–10	CELL1–CELL6	Positive terminals of Cells 1 to 6

Figure 6.2 shows the final battery pack mounted and wired to the slave board connector.

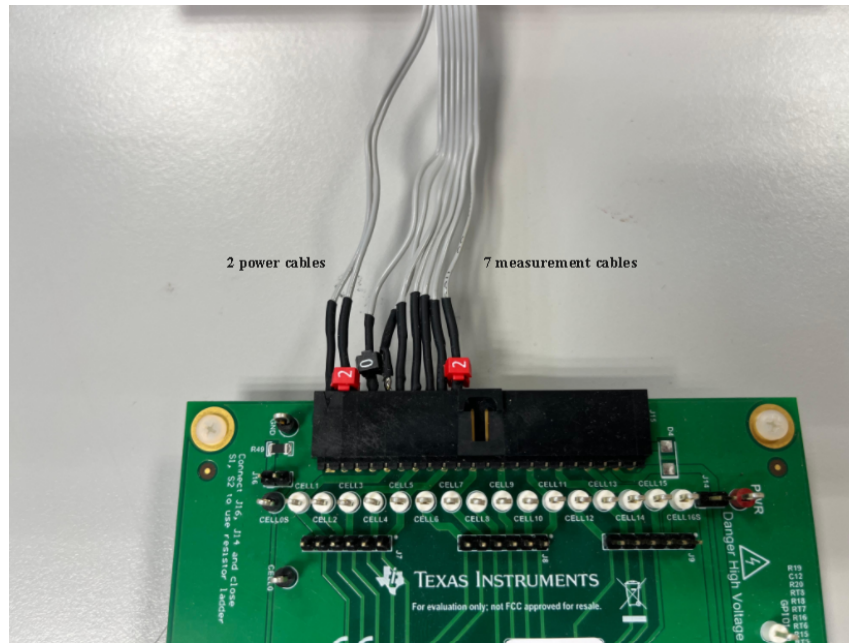


Figure 6.2: Assembled 6-cell battery pack connected to the slave board.

Separately, thicker power wires were added between the most positive terminal (top of Cell 6) and the most negative terminal (bottom of Cell 1) to handle the higher current flow associated with external components. These high-current paths were created exclusively for use with the external charger and discharge modules, which will be discussed in detail in later sections.

Once the physical wiring was completed, the system was powered exclusively through the battery pack. The external DC power supply used in previous stages to energize the resistor ladder and board components was no longer needed and was turned off. This configuration mimics the final use-case scenario of the BMS, where the system is powered by the battery under management.

6.1.2 Internal Voltage Acquisition Circuit

Internally, the voltage acquisition process of the BQ79616 slave is based on differential measurements between adjacent nodes. Specifically, the voltage of cell n is obtained by subtracting the potential at node V_{Cn-1} (negative terminal) from that at V_{Cn} (positive terminal). This approach

enables accurate measurement of each individual cell voltage in a stacked series configuration. Figure 6.3 illustrates the internal signal path used for voltage sensing.

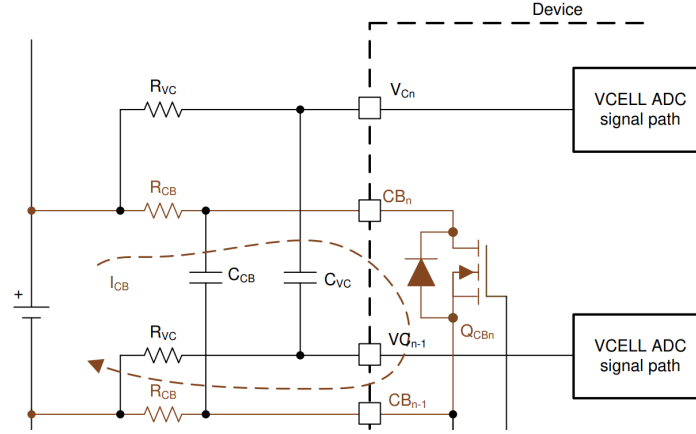


Figure 6.3: Internal voltage acquisition circuit of the BQ79616, from [50].

To ensure reliable readings, the slave board incorporates internal filtering elements, namely the resistor R_{VC} and capacitor C_{VC} , which act as a low-pass filter to suppress high-frequency noise on the voltage sense lines. Additionally, each node includes components related to the passive balancing circuit, such as R_{CB} , C_{CB} , and the control switch Q_{CBn} . These elements are part of the cell balancing system and will be addressed in detail in later chapters.

6.1.3 Firmware Adaptation and Data Processing

On the firmware side, a key adaptation was required to accommodate the reduced number of active cells in this test. Although the slave always transmits a full frame with 16 voltage values, only the first six channels contained valid measurements. The remaining channels default to the value 0x8000, a reserved code used by the BQ79616 to indicate an inactive cell. During parsing, these values were automatically filtered, and only the valid data was processed. Each 16-bit voltage reading was then converted using the ADC resolution factor of $190.73 \mu\text{V/LSB}$, as established in previous tests.

All six voltage values were displayed in real time via the same HMI developed earlier in this project. Figure 5.12 presents the live voltage acquisition screen. The system's accuracy was gradually verified throughout the tests: voltage measurements obtained from the BMS were periodically compared with reference readings taken using a calibrated multimeter directly at the battery terminals. These manual measurements consistently matched the HMI output within a small tolerance margin, confirming the reliability of the voltage acquisition path and the correctness of the overall integration.

6.2 Temperature Acquisition System

The temperature acquisition subsystem plays a crucial role in monitoring the thermal condition of the battery pack. It was developed based on the embedded features of the BQ79616EVM-021 board, which allows the use of GPIOs as analog inputs to measure the voltage drop across NTC thermistors. This section details the physical circuit setup, the measurement principle, the commands used for ADC configuration and data reading, and the rationale behind the selection of the final NTC component used in the project.

6.2.1 Circuit Setup and Board Configuration

The BQ79616EVM-021 evaluation board enables temperature measurement using NTC thermistors connected to its GPIO pins. The circuit used for this purpose is shown in Figure 6.4. The measurement relies on a voltage divider configuration that converts changes in thermistor resistance, which varies inversely with temperature, into a measurable analog voltage.

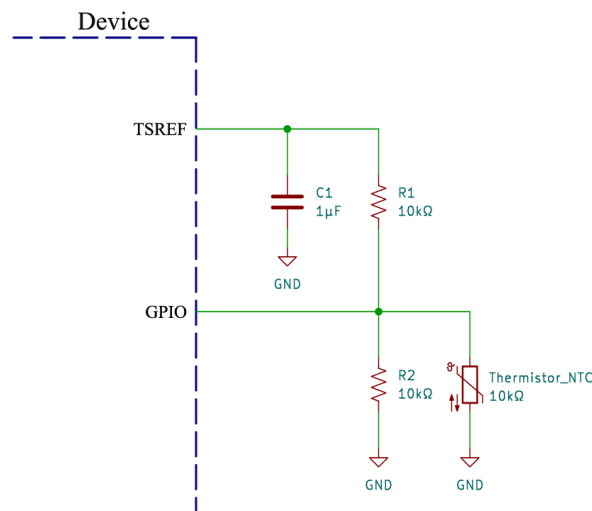


Figure 6.4: GPIO circuit for NTC measurements.

In this circuit, the reference voltage (TSREF) is internally generated and kept constant. It acts as the top rail of the voltage divider. The pull-up resistor R_1 is connected between TSREF and the node shared with the NTC thermistor. On the other side, the NTC is connected in parallel with a resistor R_2 , from this node to ground. The GPIO pin reads the voltage at this shared node.

As the temperature increases, the resistance of the NTC decreases, resulting in a lower voltage at the GPIO pin. Conversely, a lower temperature leads to higher resistance and a higher GPIO voltage. This analog voltage is digitized by the internal ADC of the BQ79616, allowing the Master microcontroller to compute the corresponding temperature based on the known thermistor characteristics.

This setup allows the BMS to monitor temperature at multiple points (one per GPIO channel), supporting thermal protection mechanisms such as overtemperature warnings, thermal balancing restrictions, or system shutdown in critical conditions.

6.2.2 Measurement Principle and Software Implementation

To comply with the FSFEUP competition regulations, which require temperature monitoring on at least 30% of the cells, rule that is imposed by the FS competition and is presented in 3.1, three NTC thermistors were integrated into the system. Given that the prototype battery pack includes six cells, this configuration satisfies the minimum requirement. The rule additionally states that temperature must be measured at the negative terminal of each monitored cell, with the sensor placed directly on the terminal or up to 10 mm along the high current path.

While in some designs this allows a single NTC to monitor two adjacent cells, particularly in configurations where negative terminals are physically close, such as in parallel cell arrangements, this approach was not viable in this prototype. The cells were connected in series, resulting in physically distant negative terminals. As a result, it was only possible to allocate one NTC per monitored cell, which led to 50% of the cells being measured, which is more than the required.

Figure 6.5 shows how the NTC's were placed in the battery prototype throughout all the following tests.

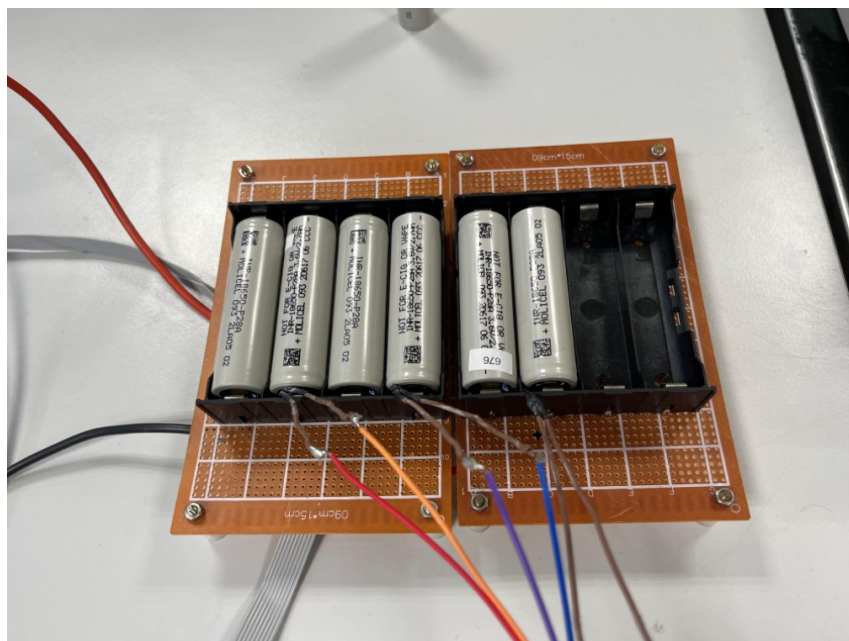


Figure 6.5: NTC's placement during the test of the BMS prototype.

Although GPIO1 could also be used for temperature acquisition, its configuration logic differs slightly from other GPIOs, as shown in Figure 6.6. To streamline the software implementation and avoid additional configuration complexity, only GPIO2, GPIO3, and GPIO4 were selected for analog temperature acquisition.

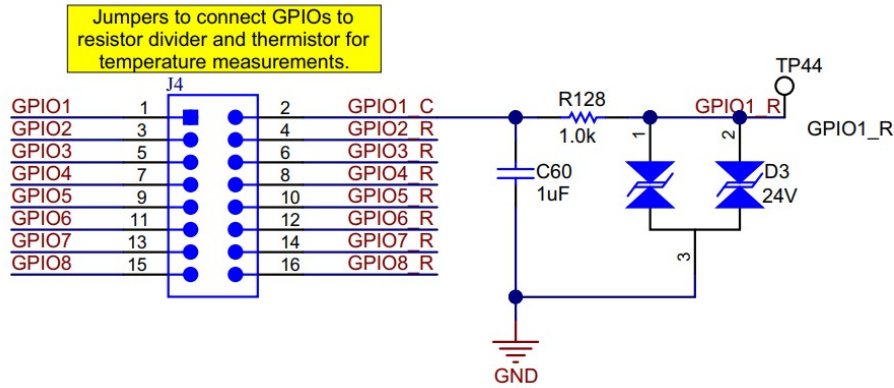


Figure 6.6: Slave board GPIO configuration, from [48].

The configuration is performed using two registers: GPIO_CONF1 and GPIO_CONF2. The GPIO_CONF1 register, shown in Figure 6.7, controls the operating mode of GPIO1 and GPIO2.

9.5.4.9.1 GPIO_CONF1

Address	0x000E							
NVM	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	FAULT_IN_EN	SPI_EN	GPIO2[2:0]			GPIO1[2:0]		
Reset	0	0	0	0	0	0	0	0
FAULT_IN_EN = Enables GPIO8 as an active-low input to trigger the NFAULT pin when the input signal is low. 0 = No fault input function. GPIO8 is configured based on [GPIO8_CONF2:0] setting. 1 = GPIO8 is set as active-low input to trigger NFAULT pin, [GPIO8_CONF2:0] setting is ignored.								
SPI_EN = Enables SPI master on GPIO4, GPIO5 and GPIO6, GPIO7. 0 = SPI master disabled. 1 = SPI master enabled. Overwrite the [GPIO4_CONF2:0], [GPIO5_CONF2:0], [GPIO6_CONF2:0], and [GPIO7_CONF2:0] settings.								
GPIO2[2:0] = Configures GPIO2. 000 = As disabled, high-Z 001 = As ADC and OTUT inputs 010 = As ADC only input 011 = As digital input 100 = As output high 101 = As output low 110 = As ADC input and weak pull-up enabled 111 = As ADC input and weak pull-down enabled								
GPIO1[2:0] = Configures GPIO1. 000 = As disabled, high-Z 001 = As ADC and OTUT inputs 010 = As ADC only input 011 = As digital input 100 = As output high 101 = As output low 110 = As ADC input and weak pull-up enabled 111 = As ADC input and weak pull-down enabled								

Figure 6.7: GPIO_CONF1 register configuration, from [50].

By setting the corresponding mode bits, GPIO2 is placed into ADC mode, enabling it to read the voltage drop across the connected NTC. The configuration command sent by the Master was:

0xD0 0x00 0x0E 0x09 CRC1 CRC2

Although the value 0x09 was written to the GPIO_CONF1 register, the only intended measurement in this configuration was from GPIO2. Consequently, the data corresponding to GPIO1 consistently returned the inactive default code 0x8000, and was therefore excluded from the acquisition dataset. The GPIO_CONF2 register, depicted in Figure 6.8, controls GPIO3 and GPIO4.

9.5.4.9.2 GPIO_CONF2

Address	0x000F							
NVM	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	SPARE	RSVD	GPIO4[2:0]			GPIO3[2:0]		
Reset	0	0	0	0	0	0	0	0
SPARE = Spare								
GPIO4[2:0] = Configures GPIO4. If <i>[SPI_EN]</i> = 1, these configuration bits are ignored and the pin is used as SS for SPI master. See Section 9.3.6.1.7 for details. 000 = As disabled, high-Z 001 = As ADC and OTUT inputs 010 = As ADC only input 011 = As digital input 100 = As output high 101 = As output low 110 = As ADC input and weak pull-up enabled 111 = As ADC input and weak pull-down enabled								
GPIO3[2:0] = Configures GPIO3. If <i>MB_TIMER_CTRL</i> is not 0x00, this configuration is ignored and the pin is configured for module balancing. 000 = As disabled, high-Z 001 = As ADC and OTUT inputs 010 = As ADC only input 011 = As digital input 100 = As output high 101 = As output low 110 = As ADC input and weak pull-up enabled 111 = As ADC input and weak pull-down enabled								

Figure 6.8: GPIO_CONF2 register configuration, from [50].

In this implementation, GPIO3 and GPIO4 were both set to ADC mode, allowing the acquisition of voltages corresponding to their respective NTC sensors. The configuration command transmitted from the Master was:

0xD0 0x00 0x0F 0x09 CRC1 CRC2

Once GPIO configuration is complete, the main ADC is set to continuous conversion mode using a write command to the ADC_CTRL1 register (address 0x030D), identical to the configuration used for voltage acquisition 5.8. This allows uninterrupted sampling of all selected analog inputs, including VCELL and GPIOs.

To extract temperature measurements, two types of ADC data must be retrieved. A reference internal voltage necessary to calibrate GPIO readings, the TSREF reading and the voltage values corresponding to each NTC connected to GPIO2, GPIO3, and GPIO4, the GPIO readings.

Similar to the procedure described in Section 5.5 for cell voltage acquisition, temperature measurements also require reading both the high byte and low byte of each register. In this case, two types of ADC data must be retrieved: the internal reference voltage (TSREF), used to calibrate the GPIO readings, and the voltages from each NTC, connected GPIO input. The TSREF values are stored in the consecutive registers 0x058C (high byte) and 0x058D (low byte), while the GPIO readings start at address 0x058E and span up to 0x059D, with two bytes per channel.

The Master requests the TSREF value by sending the following command:

0xC0 0x05 0x8C 0x01 CRC1 CRC2

The GPIO voltages for channels 1 to 8 are read using:

0xC0 0x05 0x8E 0x0F CRC1 CRC2

Although all eight GPIO channels are retrieved in this process, only 2, 3, and 4 were used in this implementation, with the remaining channels being discarded during data processing. The structure of these registers mirrors that of the voltage acquisition process, with HI and LO bytes combined to reconstruct the raw ADC values for each channel before conversion into physical units.

The TSREF value is then converted to volts using the ADC gain:

$$V_{TSREF} = TSREF_{raw} \times 169.54 \mu V / LSB \quad (6.1)$$

The voltages from the NTC's are obtained from the relevant GPIO registers and converted using:

$$V_{GPIO*} = GPIO*_{raw} \times 152.59 \mu V / LSB \quad (6.2)$$

Having the reference and GPIO voltages, the equivalent resistance of the parallel connection between the NTC and the internal resistor R_2 ($10k\Omega$) shown in Figure 6.4 can be calculated using the voltage divider equation:

$$V_{GPIO} = V_{TSREF} \cdot \frac{R_{eq}}{R_{eq} + R_{pullup}} \Rightarrow R_{eq} = R_{pullup} \cdot \frac{V_{GPIO}}{V_{TSREF} - V_{GPIO}} \quad (6.3)$$

In this configuration, $R_{pullup} = R_1 = 10k\Omega$ is the internal pull-up resistor connected to the TSREF reference voltage, while $R_{eq} = R_{NTC} // R_2 = R_{NTC} // 10k\Omega$.

Once the NTC resistance is calculated, the corresponding temperature is computed using the Beta parameter model of thermistors:

$$T = \left[\frac{1}{T_0} + \frac{1}{\beta} \cdot \ln \left(\frac{R_{NTC}}{R_0} \right) \right]^{-1} - 273.15 \quad (6.4)$$

Where $T_0 = 298.15, K$ ($25^\circ C$), $R_0 = 10, k\Omega$, and $\beta = 3380$. The logarithmic term accounts for the exponential behavior of thermistors.

This software implementation was validated by reading raw GPIO data from the Slave's response frame and confirming successful CRC checks and expected voltages at room temperature.

6.2.3 Component Selection and Final NTC Integration

Initially, the temperature acquisition tests used standard $10k\Omega$ THT NTC thermistors with a B-value of $4000K$ and a rated dissipation of $500mW$, as shown in Table 6.2. These components were selected due to their commercial availability and apparent suitability for prototype-level testing.

Table 6.2: Characteristics of initial NTC thermistors, NTCC-10K from SR PASSIVES, from [51].

Nominal Resistance	10k Ω at 25°C
Tolerance	$\pm 5\%$
B-Constant (25–50°C)	4050K
Power Rating	500 mW
Temperature Range	–30°C to 125°C
Package	THT, $\varnothing 6.5$ mm epoxy bead

Despite the theoretical adequacy of these components, empirical tests showed significant inconsistencies between thermistors. Although all sensors were exposed to the same ambient temperature, the measured resistance values differed considerably due to the loose tolerance of $\pm 5\%$, making calibration unreliable.

Due to these limitations, the final NTC chosen for the project was the Murata NXFT15XH103FEAB0 thermistor [51], which are the ones that FSFEUP team adopted for use in the final FS race car this year.



Figure 6.9: NTC Thermistor Murata Electronics NXFT15XH103FEAB035, from [52].

These components offer the same nominal resistance (10k Ω at 25°C) but with significantly improved manufacturing tolerance ($\pm 1\%$) and tighter B-constant variation, ensuring greater uniformity across sensors and more dependable calibration.

Figure 6.10 shows the final physical layout of the temperature acquisition circuit, with NTCs connected to GPIO2, GPIO23 and GPIO4 via J4 jumpers.

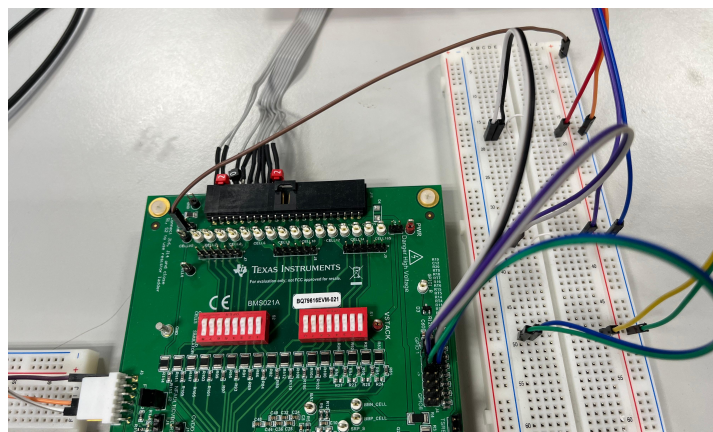


Figure 6.10: Final implementation of the temperature acquisition circuit with Murata NTCs.

This complete temperature acquisition system was fully integrated into the existing software framework and validated using repeated temperature measurements in ambient conditions and under controlled heating.

Despite the improved accuracy achieved with the Murata sensors, temperature readings remained more sensitive and susceptible to noise than voltage measurements. While the variance was significantly reduced compared to the initial thermistors, it highlighted the inherent difficulty of acquiring low-noise analog measurements in high-impedance thermal circuits.

6.3 Current Acquisition System

To complement the battery monitoring capabilities of the BMS prototype, a current acquisition system was developed to measure the real time currents of the battery and enabling the continuous conversion mode. As mentioned before, unlike cell voltages and temperatures, which are acquired via the BQ79616EVM-021 Slave board, the current signal is measured directly by the Master (STM32F446RE) using its internal ADC peripheral.

6.3.1 Hardware Configuration and Sensor Selection

The first step was enabling one of the STM32's ADC channels to capture analog voltage from the current sensor. Pin PC0 was configured as ADC1_IN10 in STM32CubeMX, ensuring it would not interfere with ongoing UART communication lines (Figure 6.11).

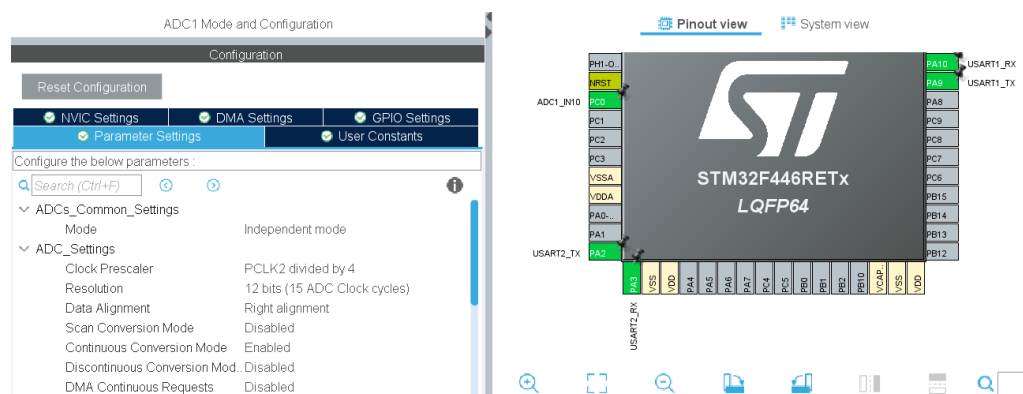


Figure 6.11: STM32 CubeMX configuration for ADC1_IN10 (PC0).

For current sensing, the chosen device was the ACS759LCB-050B-PFF-T [53], a Hall-effect-based linear current sensor from Allegro Microsystems.



Figure 6.12: Allegro current sensor ACS759LCB-050B-PFF-T.

This model offers high accuracy, a bidirectional current measurement range of $\pm 50\text{A}$, and an analog voltage output linearly proportional to the current. Crucially, it operates with a 3.3 V power supply, fully compatible with the STM32F446RE microcontroller, and produces a ratiometric output centered at $V_{CC}/2$ when no current flows. Therefore, at 0 A, the output is approximately 1.65 V, and deviations from this value represent positive (discharging) or negative (charging) currents.

This sensor was selected not only for its precision (26.4 mV/A sensitivity at 3.3 V) but also due to its robust noise immunity, low offset drift, galvanic isolation, and proven stability under high-side current sensing applications, making it ideal for the BMS prototype.

6.3.2 Physical Integration

Based on the datasheet’s application schematic, the ACS759 was connected to the STM32 as follows:

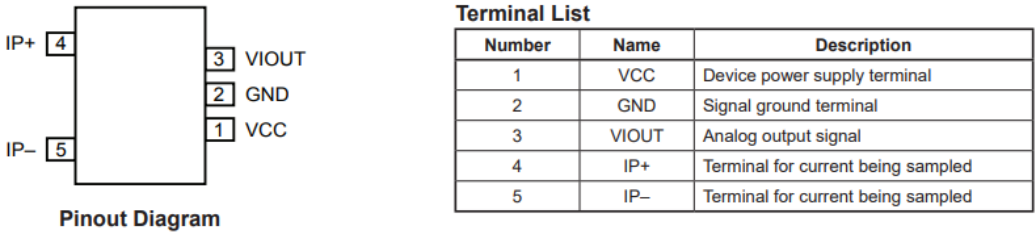


Figure 6.13: Current sensor pinout diagram, from [53].

This setup ensures that the voltage output from the sensor reflects the instantaneous current flowing into or out of the battery pack. At 0 A, the sensor outputs 1.65 V, and this value linearly increases or decreases depending on the current polarity.

6.3.3 Software Implementation and Measurement Routine

Since the ADC is in continuous conversion mode, the STM32 firmware periodically triggers ADC conversions on PC0 to acquire the voltage corresponding to the sensed current. The acquisition routine is structured to periodically start a new ADC conversion, wait for the conversion to complete, and then retrieve the digital result. This process uses polling-based synchronization, ensuring that the system only proceeds once the conversion is finalized.

Once the ADC completes its conversion, it returns a 12-bit digital value ranging from 0 to 4095, which corresponds linearly to an input voltage between 0 and 3.3 V. The measured voltage at the sensor output is then calculated by scaling the raw digital value according to the ADC resolution. The voltage is obtained by multiplying the digital value by the voltage per bit, defined as 3.3 V divided by 4095.

$$V_{adc} = \frac{ADC_{raw}}{4095} \times 3.3V \quad (6.5)$$

After determining the voltage value, the actual current is calculated by subtracting the quiescent voltage of 1.65 V, which represents zero current flow, and dividing the result by the sensor's sensitivity, which is 26.4mV/A for the ACS759LCB-050B-PFF-T.

$$I = \frac{V_{adc} - 1.65}{0.0264} \quad (6.6)$$

Initial validation involved powering the ACS759 from the STM32's 3.3V line and verifying that with no current flow, the output voltage remained close to 1.65 V. The sensor response was shown to be linear and in close agreement with theoretical expectations. When a constant external current was applied, the voltage readings obtained via the STM32's ADC were accurately converted to current values that closely matched the expected nominal value. However, despite this accuracy, the measured current exhibited significant fluctuations between consecutive samples, a behavior similar to that observed in the temperature acquisition system.

6.4 Sensor Reading Stabilization via Buffered Sampling

While the voltage acquisition system proved to be inherently stable, both current and temperature readings exhibited significant fluctuations due to sensor tolerances and analog-to-digital conversion noise. This instability motivated the implementation of a software-based filtering strategy using time-triggered sampling and First-In-First-Out (FIFO) style buffers.

To comply with the FS regulation that mandates a maximum sampling interval of 200 ms for voltage, current, and temperature monitoring, a timer-based acquisition routine was developed. The STM32F446RE microcontroller, operating at an internal clock frequency of 180 MHz, allows precise scheduling of periodic tasks through hardware timers. Using this capability, a timer interrupt was configured to trigger every 10 ms. This interval was selected to allow for 20 independent measurements to be collected over 200 ms, after which an average is calculated and the result transmitted.

6.4.1 Timer Configuration and Sampling Strategy

The timer TIM2 was configured via STM32CubeMX with the following settings:

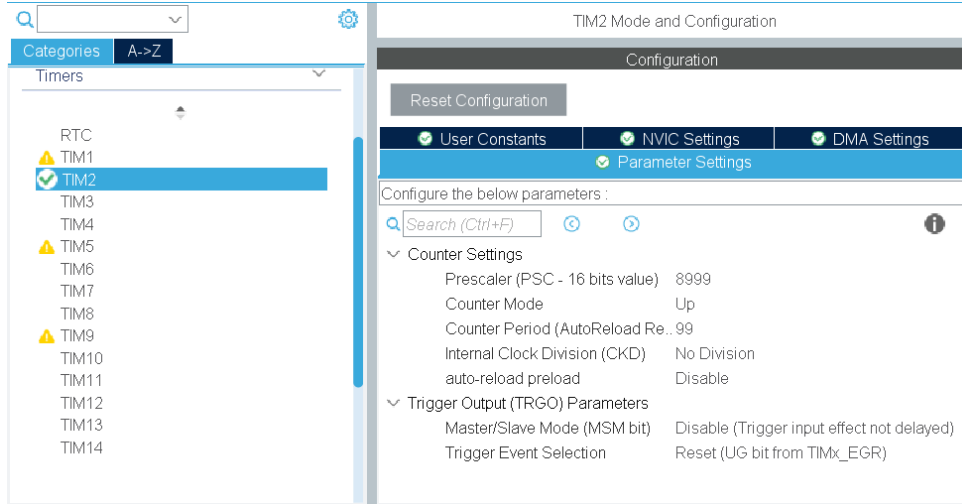


Figure 6.14: TIM2 configuration in STM32CubeMX for 10 ms sampling interval.

Given that the TIM2 peripheral receives its clock from the APB1 timer bus (which runs at 90 MHz due to the STM32's default clock tree configuration) the timer's effective frequency is reduced accordingly, despite the STM32 core itself operating at 180 MHz. Nevertheless, this 90 MHz timer clock is still extremely fast for the intended purpose. Considering the system only requires triggering an interrupt every 10 ms, the reduced clock remains more than sufficient to ensure precise timing and high-resolution sampling. The STM32's high-speed architecture guarantees that this reduction has no negative impact on real-time performance or system responsiveness. The timer period calculation is as follows:

$$f_{TIM2} = \frac{90MHz}{(Prescaler + 1)} = \frac{90MHz}{9000} = 10,000Hz \quad (6.7)$$

$$Time = \frac{1}{f_{TIM2}} \times (ARR + 1) = \frac{1}{10,000} \times 100 = 10ms \quad (6.8)$$

Thus, an interrupt is generated every 10 ms. At each interrupt, new readings for voltage, current, and temperature are acquired and inserted into their respective FIFO buffers, each configured with a length of 20 samples. Each 200 ms the average is calculated, and the resulting values are sent over UART for monitoring.

6.4.2 Buffering Mechanism and Validation

To validate the correct behavior of the timer-based acquisition and FIFO buffering system, a dedicated test was executed. In this test, the STM32 was configured to perform one measurement cycle every 10 ms, using the timer interrupt. Each cycle included the acquisition of all system parameters: cell voltages, temperatures, and current.

Each time a measurement was performed, a counter variable was incremented and printed via UART to track the number of readings accumulated. Only when the buffer for all parameters was completely filled with 20 fresh values, corresponding to a 200 ms window, was the average calculated and transmitted over UART. If, for any reason, fewer than 20 measurements were collected, no average was printed, ensuring robustness in the timing logic.

The printed message structure includes:

Table 6.3: Example of UART output message content.

Field	Description
Time	Real Running Time (ms)
Count_Timer = 20	Number of new measurements stored in FIFO
VCell1–VCell6	Averaged voltages of each cell (V)
NTC1–NTC3	Averaged temperatures (°C)
Current	Averaged current (A)

```

Count_Timer = 12 * 10ms
Count_Timer = 13 * 10ms
Count_Timer = 14 * 10ms
Count_Timer = 15 * 10ms
Count_Timer = 16 * 10ms
Count_Timer = 17 * 10ms
Count_Timer = 18 * 10ms
Count_Timer = 19 * 10ms
Count_Timer = 20 * 10ms
200;20;3.487;3.497;3.446;3.493;3.492;3.472;23.43;23.34;23.34;0.03;
Count_Timer = 1 * 10ms
Count_Timer = 2 * 10ms
Count_Timer = 3 * 10ms
Count_Timer = 4 * 10ms
Count_Timer = 5 * 10ms
Count_Timer = 6 * 10ms
Count_Timer = 7 * 10ms
Count_Timer = 8 * 10ms
Count_Timer = 9 * 10ms
Count_Timer = 10 * 10ms
Count_Timer = 11 * 10ms
Count_Timer = 12 * 10ms
Count_Timer = 13 * 10ms
Count_Timer = 14 * 10ms
Count_Timer = 15 * 10ms
Count_Timer = 16 * 10ms
Count_Timer = 17 * 10ms
Count_Timer = 18 * 10ms
Count_Timer = 19 * 10ms
Count_Timer = 20 * 10ms
400;40;3.487;3.497;3.446;3.493;3.492;3.472;23.27;23.45;23.54;0.04;
Count_Timer = 1 * 10ms
Count_Timer = 2 * 10ms
Count_Timer = 3 * 10ms
Count_Timer = 4 * 10ms
Count_Timer = 5 * 10ms
Count_Timer = 6 * 10ms
Count_Timer = 7 * 10ms

```

Figure 6.15: Terminal output showing structured data logging every 200 ms after 20 measurements.

To further validate this mechanism, a graphical analysis was performed. The cells were pre-balanced to eliminate significant voltage variation, ensuring the plotted signal remained flat and suitable for time-domain verification. No current was injected or drawn from the pack during this test, making the electrical environment stable.

A plot was created with time (ms) on the x-axis and the six cell voltages on the y-axis, using a scatter plot format. As shown in Figure 6.16.

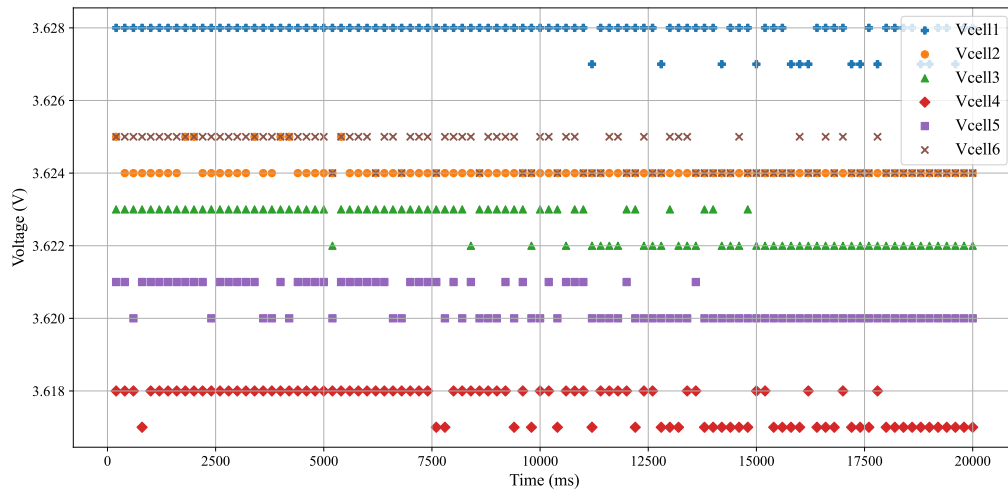


Figure 6.16: Terminal output showing 200 ms VCell buffer print intervals.

All points are uniformly spaced in 200 ms increments, precisely aligning with the expected behavior. For example, over a 2000 ms span, 10 dots are observed per VCell line, as expected:

$$\frac{2000 \text{ ms}}{200 \text{ ms}} = 10 \text{ points} \quad (6.9)$$

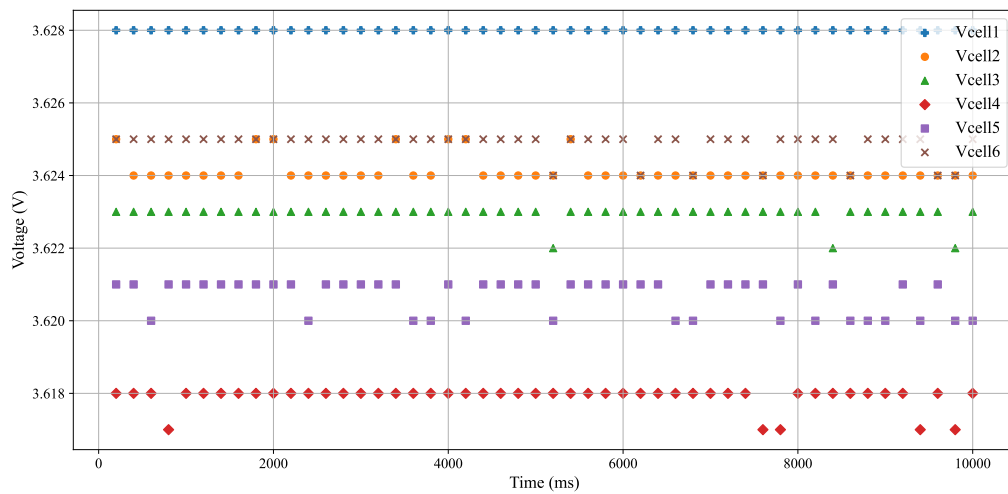


Figure 6.17: Closer look at VCell readings.

These results confirm that the STM32 system, is fully capable of adhering to the FS data acquisition requirements with 0% timing error. The FIFO buffer strategy, combined with periodic sampling and printout logic, ensures both measurement reliability and data clarity, as it is going to be demonstrated in the following chapter.

6.4.3 System Validation

With the timer-based acquisition stabilized and validated in terms of timing accuracy, it was necessary to evaluate the correctness of the measured values. For this purpose, specific experimental tests were performed, as will be detailed in Chapter 7. These tests compared the BMS readings against external references for voltage, temperature, and current, allowing the quantification of the error introduced by the acquisition system.

For the voltage acquisition, the reference values were obtained by performing a complete discharge cycle of approximately 250 minutes at a constant current of 0.56 A. Throughout the process, the BMS readings were periodically verified every 15 minutes against calibrated multi-meter measurements. The results consistently showed a nearly constant offset, confirming that the voltage path was highly accurate. The error analysis for all six monitored cells is summarized in Table 6.4.

Table 6.4: Voltage acquisition error validation results (all 6 cells)

Mean Error (%)	Max Error (%)	Standard Deviation (%)	Relative Error (%)
0.35	0.80	0.15	0.40

For the temperature acquisition, no external reference thermometer was available, nor any dedicated component that could provide a controlled, variable, and stable temperature for calibration. Instead, validation was performed by comparing BMS readings against the ambient temperature values provided by the internal room sensor. Tests were conducted across different days and times, using air conditioning to vary the reference conditions. As expected, the observed errors were slightly higher than for voltage, but still within acceptable margins, as shown in Table 6.5.

Table 6.5: Temperature acquisition error validation results (all 3 sensors)

Mean Error (%)	Max Error (%)	Standard Deviation (%)	Relative Error (%)
1.70	2.50	1.90	1.50

For the current acquisition, the validation reference was the programmable load used to discharge the battery cells at fixed currents, complemented by controlled charging tests. The BMS readings were directly compared to the set current values, proving that the Hall-effect sensor could provide reliable results across the tested range. The error analysis is summarized in Table 6.6.

Table 6.6: Current acquisition error validation results

Mean Error (%)	Max Error (%)	Standard Deviation (%)	Relative Error (%)
1.30	2.20	1.10	1.50

These results demonstrate that the complete acquisition system, covering voltage, temperature, and current, operated within very small error margins, confirming its robustness and accuracy. This validation provided confidence to proceed with the next stage of testing, where the BMS performance under real operational scenarios is further evaluated.

Chapter 7

Model-Based Testing and System Validation

This chapter presents the complete validation process of the BMS prototype, combining both the methodology adopted for testing and the analysis of results obtained under real-world operational conditions. Building upon the hardware and communication system integration discussed in Chapter 6, the focus here is on evaluating the firmware strategies implemented throughout this dissertation, including estimation algorithms, control logic, and system-level decision-making.

All tests were executed on the fully integrated system, with physical components and communication interfaces already verified to be stable. The firmware operated continuously within the timer-driven control loop configured to run every 200 ms, as previously described. At each cycle, relevant system variables were acquired, processed, and printed via UART, allowing data to be logged in real time using Tera Term and analyzed offline with Python scripts developed in Visual Studio Code.

The tests cover several critical functionalities of the BMS: SOC estimation using a hybrid model, state machine transitions, safety shutdowns, internal resistance reestimation, and cell balancing logic. Each test was designed to target specific elements of the system's logic and assess their behavior under long-duration discharges, rest periods, and current flow dynamics.

The internal consistency and correctness of the embedded algorithms were validated through synchronized plots of measured and estimated variables, error analysis, and cross-validation against theoretical expectations or datasheet values. While each test is presented with its own objectives and observations, it is important to note that all functionalities were implemented within a unified firmware architecture. The BMS operates as a fully integrated system, with every module running concurrently in the same control loop and sharing real-time data. Thus, the performance of each module is inherently coupled to the others.

The model parameters used for estimation and electrical modeling were kept constant throughout all tests and correspond to the reference values provided in Table 4.8. Where necessary, the results also include a discussion on parameter sensitivity and opportunities for refinement.

Through this chapter, the effectiveness, robustness, and limitations of the BMS firmware are critically assessed. The goal is to demonstrate whether the embedded control system is capable of maintaining accurate monitoring, adaptive behavior, and safe operation under realistic conditions, ultimately validating the project's proposed architecture and design methodology.

7.1 Charging and Discharging Infrastructure

To perform real-world testing of the BMS prototype under realistic dynamic conditions, it was first necessary to evaluate and select external equipment capable of accurately charging and discharging the battery pack. Since the objective of this phase was not only to execute tests but also to validate core BMS functionalities, the choice of hardware had to be made with careful consideration, and for that a technical review was performed to understand the specifications, operating principles, and compatibility of each available device with the test goals.

This section introduces the key infrastructure chosen for the experimental setup and justifies its relevance to the project's requirements.

7.1.1 ISDT T8 BattGO Smart Charger

The charger selected for this work was the ISDT T8 BattGO Smart Charger, a commercially available device widely used by the FSFEUP team to charge the low-voltage battery subsystem. Its selection was based on a prior technical analysis of its features, functionalities, and applicability to the specific requirements of this BMS validation phase [54, 55].

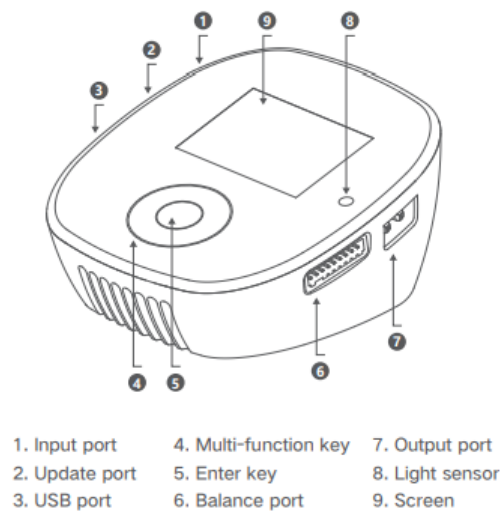


Figure 7.1: ISDT T8 BattGO Smart Charger, physical layout and ports, from [55].

As illustrated in Figure 7.1, the charger includes dedicated ports for input power, voltage readings (cell balancing), and power output. This structure enables the connection of both power

and balance leads simultaneously, which is essential for full use of the charger’s balancing and monitoring features.

Among its most useful practical features are the following:

- **Real-time cell voltage display**, enabling external verification of the BMS’s cell voltage acquisition.
- **User-configurable cutoff parameters**, such as charge current, target voltage, and charge timeout settings, which help standardize test conditions.
- **Internal protection logic** for overvoltage, overcurrent, and overheating.

One of the most notable characteristics of the ISDT T8 is its ability to perform passive balancing between cells. When the balance cable is connected, the charger monitors individual cell voltages and applies balancing current to higher-voltage cells to reduce the spread. This occurs automatically during the charging process and aims to reduce the voltage dispersion across the pack. However, the logic for when balancing is initiated or terminated is not transparent, and cannot be modified or externally accessed. Even though this feature is not the best because of this reason it is still very helpful for testing setups.

Another core feature is its ability to operate in automated CC–CV charging mode. This is a fundamental requirement in lithium-ion battery systems, where charging begins with a constant current phase and then transitions to a constant voltage phase as the cells approach full charge. In practice, the ISDT maintains the configured current until a target voltage is reached, after which it gradually reduces the current while holding the voltage steady. Unfortunately, the same thing happens with this feature and the device does not expose its internal decision-making thresholds or control logic, thus functioning as a black box, its behavior in test conditions confirms compliance with standard CC–CV protocols.

In addition to these functional features, a selection of key specifications from the charger are summarized in Table 7.1. Only the parameters relevant to this project have been included.

Table 7.1: Relevant ISDT T8 charger specifications, from [54, 55].

Parameter	Value
Input Voltage	DC 12–40V
Maximum Input Current	35A
Input/Output Support	Yes (Smart Battery)
Max Charge Capacity	1000W
Max Discharge Capacity	20W
Balance Cells Supported	2–8S
Supported Battery Types	LiFe, LiIon, LiPo, LiHv (1–8S)
Operating Temperature	0–40°C

The ISDT charger was powered using a Mean Well HRPG-1000 Series power supply, capable of delivering up to 1000 W with Power Factor Correction [56]. This ensured a clean and stable voltage source for all charging cycles, preventing unexpected drops or spikes that could compromise test repeatability.



Figure 7.2: Mean Well HRPG-1000-24 power supply used to power the ISDT T8 charger, from [56].

In summary, despite its black-box nature, the ISDT T8 provided all the necessary features to safely and consistently prepare the battery pack for validation tests. It also served as a useful reference system against which the BMS's voltage readings and future CC–CV implementation and balancing logic could be compared.

7.1.2 BK Precision 8510 DC Electronic Load

For discharge testing, the BK Precision 8510 programmable DC load was used. It is capable of operating in constant current, constant power, or constant resistance modes, with a maximum rating of 600 W [57, 58]. In this project, the device was operated in CC mode to ensure a consistent discharge profile across the battery pack.



Figure 7.3: BK Precision 8510 DC Electronic Load used for programmable discharge testing, from [57].

Although outdated, the 8510 remains a programmable device, which means custom discharge profiles can be defined by the user. This includes fixed current settings or dynamic profiles where

the current can vary over time according to predefined values [58]. This feature was particularly useful in the development and testing of the BMS, as it allowed a controlled and repeatable discharge to be executed without relying on manual load switching. Moreover, unlike the ISDT charger, the BK Precision 8510 allows data to be exported, making it possible to retrieve logs of applied current, total pack voltage, and discharged capacity. These data were later used to build comparison graphs between the behavior observed by the BMS and that recorded externally by the discharger.

However, the device also presents several limitations that made its integration with the BMS more complex. One of the most critical is that it only measures and applies thresholds based on the total pack voltage, and does not have any capability to monitor or act on individual cell voltages. This is problematic when working with lithium-ion cells, where safety is highly dependent on cell-level management. For example, if one wishes to stop discharging when a specific cell reaches a threshold, like 2.5 V, this is not possible to enforce directly through the device. Even if the battery is made of six cells and the theoretical cutoff is set at $6 \times 2.5 = 15$ V, degradation among the cells means that one of them could reach 2.50 V while others remain at 3.00 V, and the discharger would continue operating until the total pack voltage drops to 15 V. By this point, the most degraded cell could already be pushed below its safe minimum voltage [58]. This behavior would potentially damage the cells and distort SOC estimation, so it required constant manual supervision during testing.

Another limitation is that, unlike the charger, the BK 8510 does not implement CC–CV discharge control. The current remains fixed throughout the discharge, regardless of the pack voltage. This is a critical distinction because, in a complete system, it is the role of the BMS to actively manage the maximum charge and discharge current based on parameters such as SOC, cell voltage, and temperature. However, in the context of these tests, even though the BMS was designed to include logic capable of reducing discharge current, this functionality could not be practically validated. This is because the BK 8510 cannot be externally controlled by the BMS, the current must be pre-programmed manually and remains constant during operation.

Because of this limitation, all the discharge tests required manual supervision throughout the discharge process. The user had to monitor the individual cell voltages through the BMS and stop the discharger manually once the most degraded cell reached the lower voltage limit. Although suboptimal in terms of automation and in terms of full predictable results, this approach ensured test safety and allowed full control over the stopping condition based on cell-level behavior.

7.2 Validation of SOC Estimation Under Full Discharge Test

As previously mentioned, one of the most fundamental responsibilities of a BMS is the estimation of the SOC, as it directly influences energy availability prediction, control strategies, and safety mechanisms. Inaccurate SOC values can lead to premature cutoffs, over-discharge, underutilization of capacity, or failure to trigger essential protection mechanisms. As such, validating the

accuracy and robustness of the SOC estimation model is a critical step in confirming the reliability of the BMS prototype.

This section presents the experimental results obtained from complete discharge cycles of the battery pack, conducted under controlled current conditions. The test was designed to assess how accurately the embedded hybrid SOC estimation algorithm, could track the actual battery depletion process in a real environment.

7.2.1 Test Motivation and Objectives

The main motivation behind this test was to evaluate whether the hybrid SOC estimation algorithm, which had been previously validated through simulation, could produce consistent and accurate results when deployed on real hardware. Given the central role of SOC in nearly every control decision made by the BMS, this test served as a first critical step in determining the reliability of the embedded estimation logic.

Additionally, this test was used to identify practical issues that could affect SOC estimation, such as switching artifacts, rest-phase behavior, voltage relaxation, and model drift over time. The outcomes of this experiment would define the starting point for further refinement of the firmware, based on observed errors or inconsistencies.

While SOC estimation is relevant for both charging and discharging phases, the validation presented here was conducted exclusively during discharge. This decision was based on limitations of the external charging equipment. The ISDT T8 smart charger, although used to safely prepare the battery, includes internal logic that cannot be externally configured or monitored. Key features such as its transition between constant current and constant voltage modes, and its passive balancing algorithm, operate as a black box. As such, it was not suitable for SOC estimation validation, since no precise control or observation of charging current or voltage behavior could be ensured.

Instead, the test focused on the discharge process using the BK Precision 8510 programmable DC load, which supports fixed-current operation. The selected discharge current was 0.56 A, equivalent to 1/5C of the Molicel INR-18650-P28A nominal capacity, aligning with the conditions previously used in simulation and datasheet reference tests (as described in Chapter 4). The battery pack was fully charged to approximately 4.15 V per cell, which corresponds to the maximum charging voltage allowed by the ISDT smart charger for Li-ion cells. Balancing was also performed automatically during this process, ensuring a uniform starting condition across all cells before initiating the test.

Because the BK load measures only total pack voltage, it could not automatically terminate the test based on individual cell limits. Therefore, the stopping condition was monitored manually using real-time warnings generated by the BMS. The test was halted when the most discharged cell reached the minimum safe voltage of 2.5 V. Although this approach lacked full automation, it ensured safe operation and provided a valid framework for observing the real-time behavior of the embedded SOC estimation algorithm during the entire discharge cycle.

7.2.2 Results from Initial Estimation Strategy

The first round of SOC validation was conducted using the initial version of the hybrid estimation firmware, described in Chapter 7. This strategy relied on direct switching between the open-circuit voltage (Voc) method and the KiBaM model based on the presence or absence of current flow. The goal was to assess whether this logic, when implemented in firmware, could accurately track the SOC of the battery over a full discharge process with constant current.

As described in the test procedure, the cells were fully charged using the ISDT smart charger to approximately 4.15 V, that is the maximum limit supported by the device for lithium-ion cells. After balancing was confirmed, the battery was discharged using the load in constant current mode at 0.56 A, equivalent to 1/5 C. The discharge continued until one of the cells reached 2.5 V, at which point the test was manually terminated based on BMS warnings, since the discharger does not monitor individual cell voltages.

The software provided with the BK Precision 8510 (PV8500) generated a discharge graph of total battery voltage versus extracted capacity, which was exported in both graphical and numerical format. These values were always used as a ground truth reference, against which the SOC-based estimation of extracted capacity could be compared.

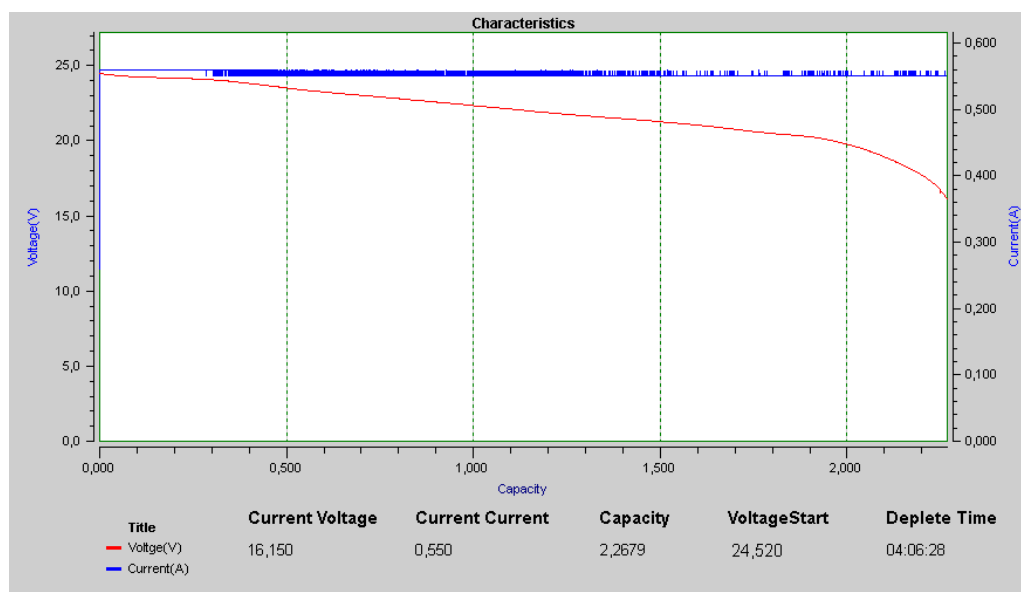


Figure 7.4: Battery voltage vs. accumulated capacity from PV8500 software (reference curve).

A custom Python script was developed to reconstruct and improve the original load graph (Figure 7.5) using the downloaded data.

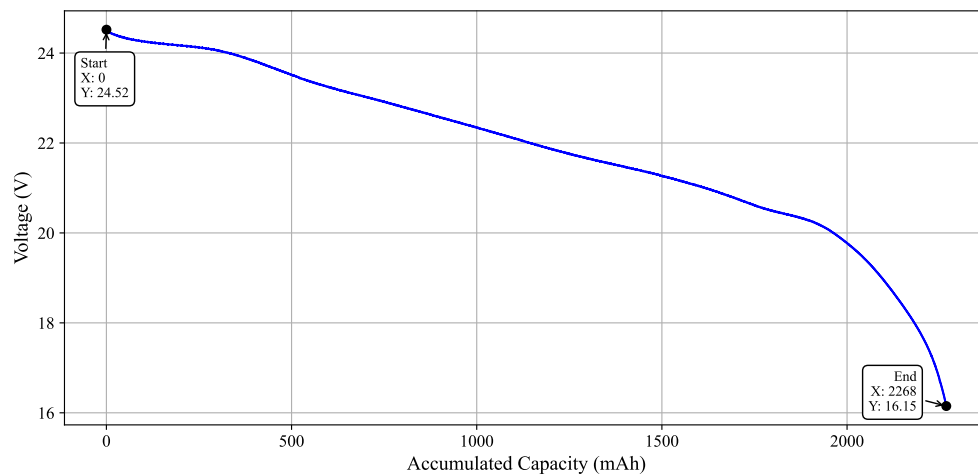


Figure 7.5: Battery voltage vs. accumulated capacity from VSCode (reference curve).

The overall battery voltage evolution, computed as the sum of the six individual cell voltages acquired by the BMS firmware, was reconstructed and is shown in Figure 7.6. When compared to the reference curve, the voltage profile obtained from the BMS matches the expected discharge behavior. This confirms that the acquisition module was functioning correctly, with reliable and consistent voltage measurements throughout the entire test, as previously mentioned in Chapter 6. Therefore, any discrepancies observed must originate from limitations in the control logic or switching strategy between estimation models, rather than from measurement or timing errors.

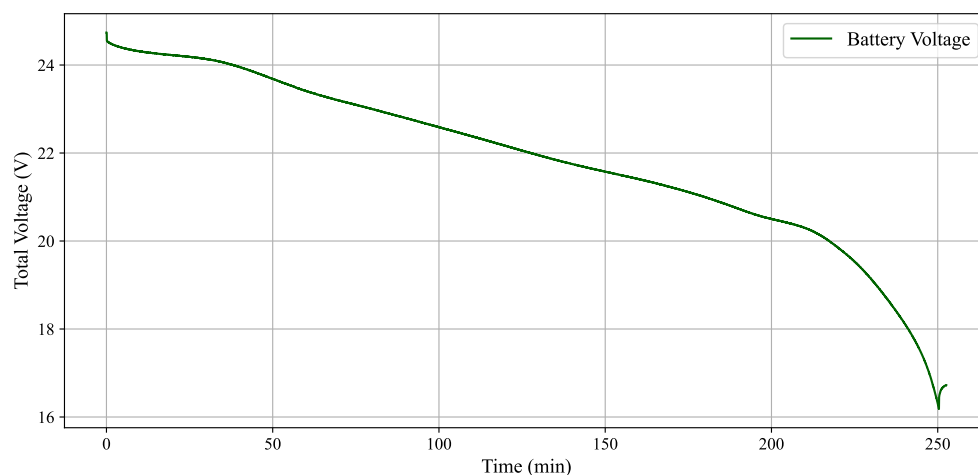


Figure 7.6: Total battery voltage over time during discharge, acquired by the BMS.

With voltage acquisition validated, the next step was to assess the accuracy of the SOC estimation itself. The key validation metric for the SOC validation is the similarity between the real amount of charge extracted from the battery with the value estimated by the BMS firmware, computed from the reduction in SOC relative to its initial value.

The real extracted capacity corresponds to the value measured by the BK Precision 8510 electronic load, represented graphically in Figure 7.5, and is computed according to the following expression:

$$Extracted_Charge_{real} = \sum \left(I \cdot \frac{\Delta t}{3600} \right) \times 1000 \quad [mAh] \quad (7.1)$$

The SOC-based capacity estimation, used to indirectly validate SOC behavior, was given by:

$$Extracted_Charge_{estimated} = Capacity_{total} \cdot \left(\frac{SOC_{init} - SOC(t)}{100} \right) \quad [mAh] \quad (7.2)$$

Figure 7.7 presents the battery voltage curve plotted against the internally computed extracted capacity from the BMS firmware. This capacity axis was derived in real time during the test using the estimation logic embedded in the BMS firmware.

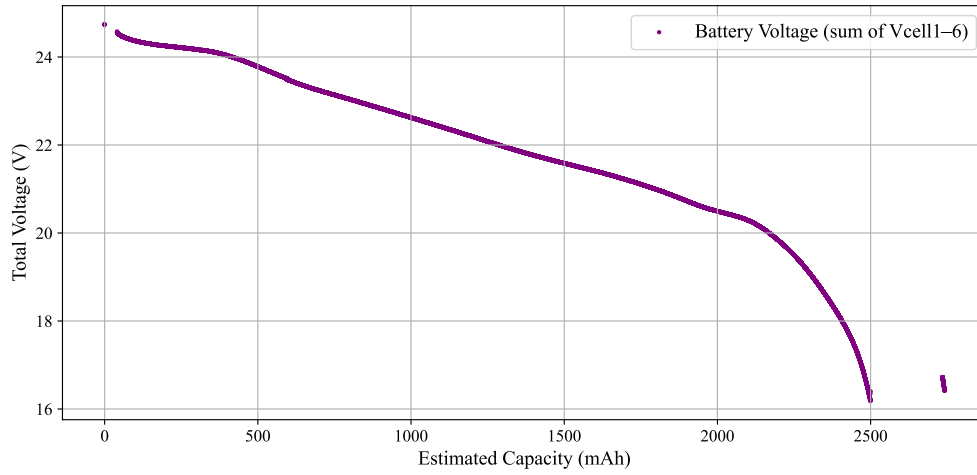


Figure 7.7: Estimated capacity vs. battery voltage (from BMS firmware).

When comparing this result to the reference discharge curve in Figure 7.5, it becomes evident that a noticeable mismatch occurs, particularly in the beginning and final stages of the discharge. The capacity estimated from SOC values diverges from the true extracted capacity, indicating that the initial estimation logic was not yet sufficiently accurate and required refinement.

To better understand the source of this discrepancy, the time-domain evolution of each cell voltage was plotted alongside the SOC minimum and maximum values across all cells.

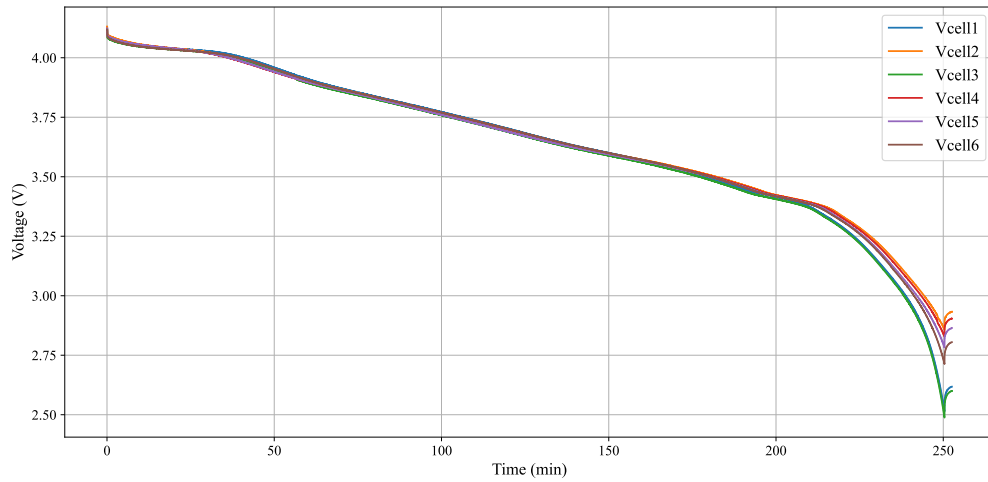


Figure 7.8: Voltage evolution of each cell over time during the discharge.

As shown in Figure 7.8, as expected, the cells begin the discharge process with nearly identical voltages but diverge over time due to internal differences and slight degradation mismatches. This variation in cell behavior directly affects SOC estimation. Figure 7.9 illustrates the evolution of the maximum and minimum SOC values across all cells.

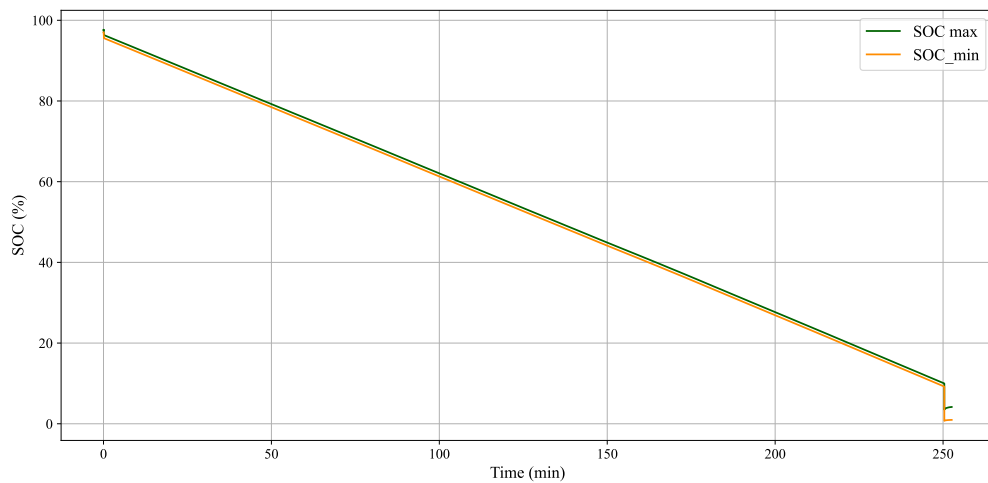


Figure 7.9: SOC minimum and maximum evolution during discharge (initial estimation strategy).

Before the discharge process starts, the SOC values remain nearly identical since they are calculated using the Voc model and the cells are all with approximate voltage. Throughout the discharge, the SOC was estimated with the KiBaM model, which is driven by current integration and does not account for voltage differences. However, when the current drops and the system switches back to the Voc-based estimation, the SOC becomes directly dependent on each cell's voltage. Since minor degradation effects lead to slightly different voltages among the cells, this switch causes visible divergence in the estimated SOC values.

Although not immediately evident in the plots, a closer inspection reveals that after the transition to Voc estimation, the SOC values continue to drift slightly over time instead of stabilizing. This behavior occurs because the system begins using voltage-based estimation before the battery has fully relaxed, leading to inaccuracies as the cell voltages are still recovering. This reinforces the importance of introducing proper relaxation detection logic, rather than switching estimation methods immediately when the current reaches zero. Without this, the system risks misinterpreting transient voltage fluctuations as meaningful state changes, which undermines the accuracy of the SOC estimation.

These results demonstrated that, although the initial hybrid estimation logic was correctly implemented in firmware, it was not sufficient to reproduce the expected discharge profile. The divergence between real and estimated capacity, along with the SOC inconsistencies during transitions between estimation methods, highlighted the need for improvements in model switching criteria and rest-phase handling. These adjustments are addressed in the following subsection.

7.2.3 Relaxation Triggering and One-Time Voc-Based Estimation

Following the observations from the initial estimation test, a major source of inaccuracy was traced to the premature use of the Voc method. This led to drifting SOC values and inconsistent transitions between the KiBaM and Voc estimation strategies. To address these issues, a new firmware mechanism was introduced, consisting of two levels of relaxation flags: an individual flag for each cell, referred to as `Relax_cell*`, which detects when a specific cell is fully relaxed, and a global relaxation flag, `flag_relax`, which is activated only when all six `Relax_cell*` flags are set.

The `Relax_cell*` flags system was designed around the principle that a cell can only be considered at rest when its terminal voltage remains unchanged over a period of time. To capture this, a FIFO-style buffer was introduced in the firmware, storing the four most recent voltage readings for each cell. The flag was set only when all four values in the buffer were numerically identical, indicating voltage stabilization. This strategy ensured that the transition to Voc estimation would only occur when the electrochemical equilibrium of the cell was reliably reached.

This strategy was designed to correct two major issues previously identified: the incorrect SOC initialization at the start of the discharge, and the premature and repeated use of Voc method for SOC estimation at the end of the discharge. These two adjustments are discussed separately in the following subsections.

7.2.3.1 Transition from Relaxed State to Discharging Process

As previously mentioned, the Voc-based SOC estimation is theoretically the most accurate method when the cell is truly at rest, since it directly relates the terminal voltage to the internal state-of-charge through a fitted model. However, this relationship only holds if the battery behaves as an open circuit, which means that there is no current flow and the voltage has stabilized. If this condition is not met, using Voc leads to an overreaction to transient voltage variations, and thus inaccurate SOC values.

To better understand the issue addressed in this test, Figures 7.10 and 7.11 are revisited from the results of the previous test. These plots illustrate the core problem: the incorrect usage of the Voc estimation method beyond the relaxation period (after starting the discharge process).

Figure 7.10 shows the behavior of a cell, in this case the cell with the highest voltage, during the initial phase of the discharge. In the previous firmware version, the system continued using the Voc model even after current had started flowing. As the voltage began to drop due to the onset of discharge, the SOC, shown by the dashed red line, dropped as well. This happened because the system was still using voltage-dependent estimation while the battery was no longer in open-circuit conditions. The green vertical line marks the moment where the system eventually transitioned to the KiBaM model, though this occurred too late.

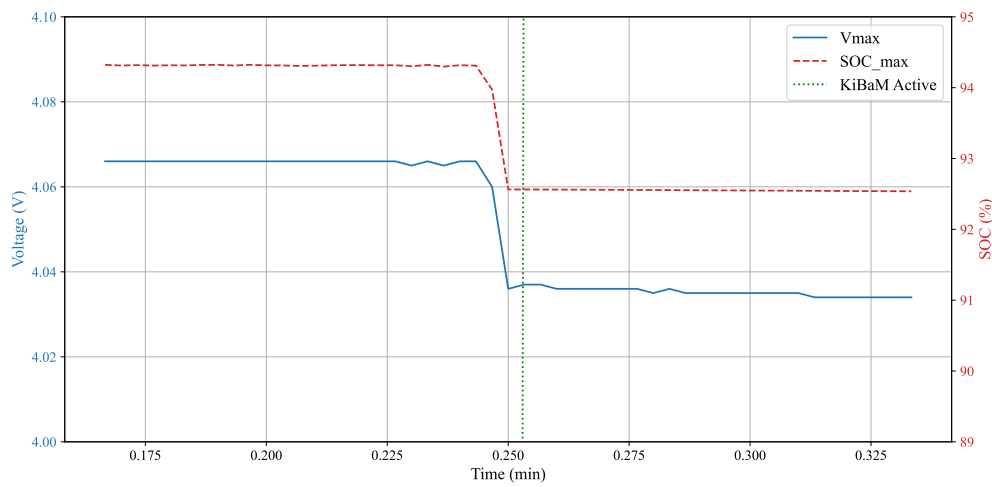


Figure 7.10: SOC and voltage evolution of the highest-voltage cell during the beginning of the discharge process.

This behavior is further clarified in Figure 7.11, which plots the SOC and current over time. It can be observed that the SOC begins to drop slightly before the current flow is detected, and the transition to the KiBaM model occurs only after this drop is already done. This mismatch arises from the delay introduced by real current acquisition. As expected from a physical current sensor, the measured values are not perfectly clean nor instantaneous, they contain small fluctuations and noise, making it difficult to distinguish between true rest and the initial stages of charge or discharge.

To address this, a corrected current signal was introduced in the firmware. This signal applies a deadband threshold such that any real current measurement within the interval $[-0.2, +0.2]$ A is considered effectively zero. All other values are passed through as-is. This adjustment helps to filter out noise and spurious current readings during rest, but it introduces a limitation: since current fluctuations are natural in real acquisition, and the firmware cannot predict whether a discharge or charge process is about to begin, the system is unable to anticipate the start of dynamic behavior before the actual current exceeds the deadband.

As a result, KiBaM estimation can only begin after a valid current reading is observed, by which time, the voltage have already dropped. This is clearly visible in Figure 7.11, where the system switches to KiBaM just after the current rises above the 0.2 A threshold, but too late to prevent the SOC from reacting to the voltage drop. As a result, the SOC estimation is still being performed using Voc, even though the cell is no longer in true rest, which undermines the accuracy of the SOC tracking.

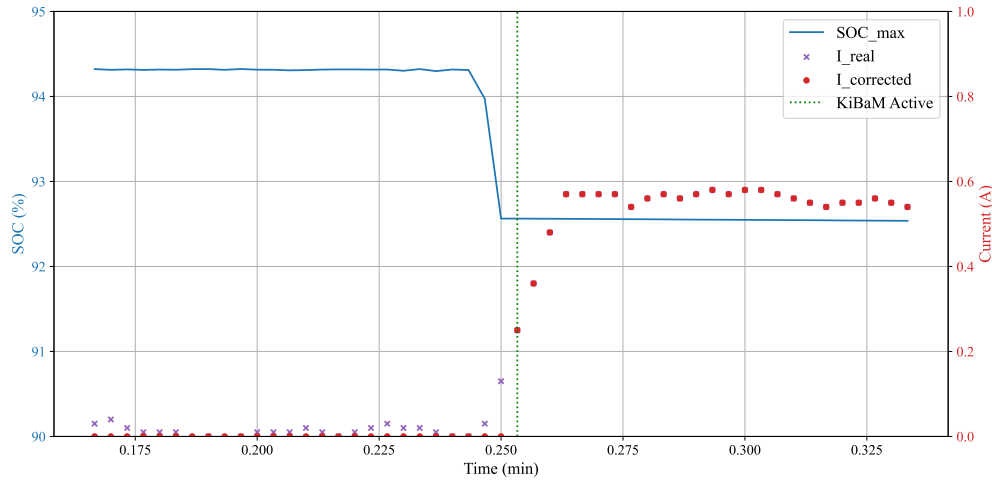


Figure 7.11: Voltage and current during initial discharge.

Then, the objective here was to implement a robust initialization strategy for SOC estimation by ensuring that the transition from the Voc method to the KiBaM method was made at the appropriate time. Rather than focusing yet on validating whether the relaxation detection was perfectly accurate, this step aimed to resolve the critical issue identified above.

The central idea behind this test relies on the assumption that the relaxation flags are correctly triggered, meaning that the cells are genuinely in an open-circuit and thermodynamically stable state. Under such conditions, the terminal voltage of a cell will always remain constant over time as long as no current flows through it. Consequently, the SOC can be estimated once using the Voc model at that precise moment that the cells reach this relaxation state and the SOC will also remain unchanged. This justifies the decision to apply the Voc estimation only once per rest event, rather than continuously.

With the new one-time estimation logic implemented, the results improved substantially. As shown in Figure 7.12, the SOC remains flat before the start of discharge, indicating that it was only estimated once using Voc and not updated during the rest period. Once current begins to flow, the SOC starts decreasing smoothly and consistently according to KiBaM dynamics.

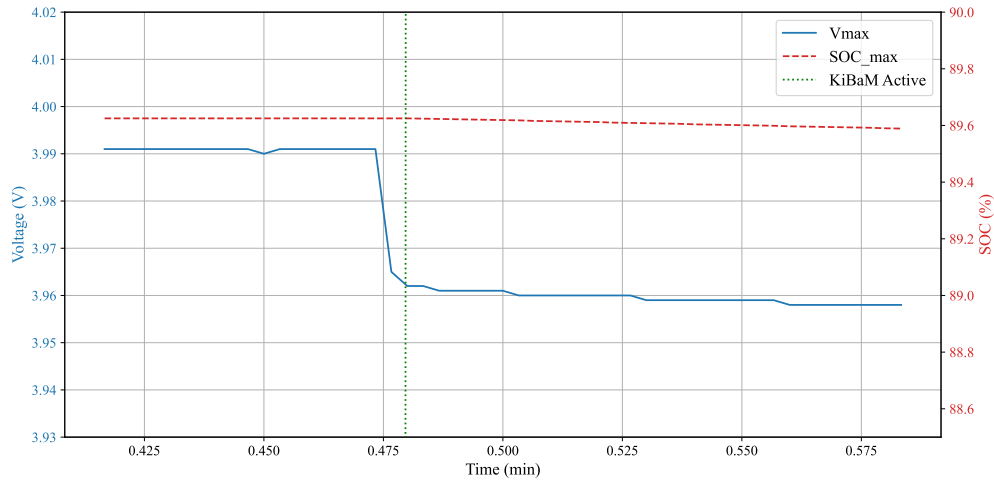


Figure 7.12: Corrected SOC estimation in the beginning of the discharge process.

The current profile in Figure 7.13 confirms that the transition to KiBaM occurred exactly at the beginning of current flow, as intended. This validated the logic that SOC should only be initialized using Voc once, under well-defined relaxation conditions.

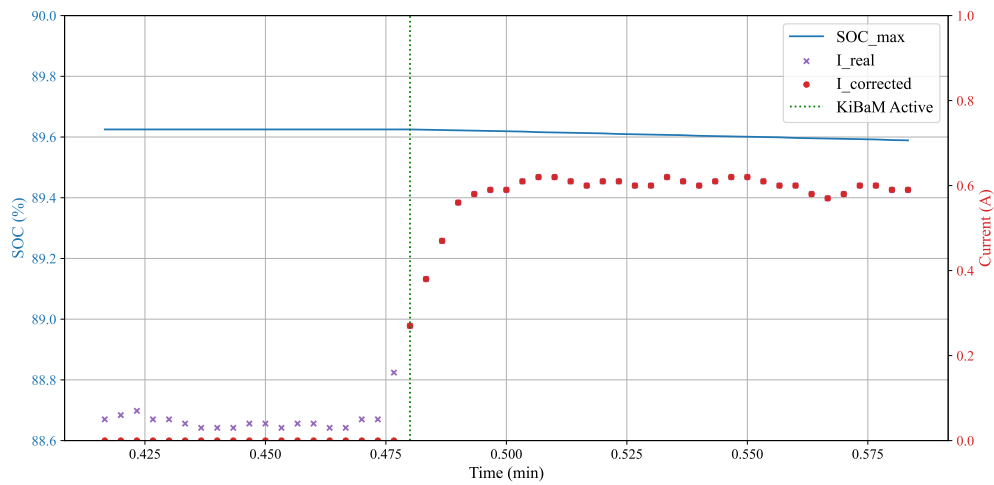


Figure 7.13: Current and voltage evolution after logic correction.

As shown in the results, this adjustment produced the intended effect. Assuming that the relaxation flags are correctly triggered and the cells are indeed in open-circuit conditions, estimating the SOC only once using the Voc method proved to be both sufficient and effective. Although it remains impossible to predict precisely when a discharge (or charge) process will begin, this strategy eliminates the transition error observed previously, where the system continued using Voc after the discharge had already started. As a result, the system now switches to KiBaM with a slight delay of one or two iterations (maximum of 400 ms), which introduces a minimal and acceptable error. This residual offset is a natural trade-off when using real acquisition hardware, but

it does not compromise the reliability or consistency of the SOC estimation process. Therefore, the system behavior in the beginning of the discharge (or charge) process is now stable, predictable, and suitable for dynamic operation.

7.2.3.2 Final SOC Control After Discharge Stabilization

Following the successful implementation of the one-time SOC initialization using the Voc method, the next challenge was to define an accurate moment to apply Voc estimation at the end of the discharge process. The goal was to ensure that SOC estimation would only be updated once the cells were truly in a relaxed state, that is, when no current was flowing and the terminal voltage had stabilized. Any premature use of Voc during transient voltage recovery could reintroduce the same inaccuracies previously addressed at the beginning of the discharge.

To achieve this, the control logic was updated to move away from relying solely on current measurements, which proved to be insufficient due to real-world current sensor fluctuations. Instead, a new relaxation detection mechanism based on voltage stability was introduced. Specifically, a FIFO buffer was implemented for each cell to store the four most recent voltage readings. The relaxation flag `Relax_cell*` would only be set when all four values in the buffer were numerically identical, indicating that the voltage had stabilized.

This adjustment was immediately revised, as it quickly proved inadequate. Although it avoided triggering the relaxed state at the exact moment the discharge stopped, it still activated the relaxation flag almost instantly after the current dropped to zero. This was due to the high frequency of voltage acquisition. With multiple readings occurring within a short time span, the buffer could quickly fill with identical values before the cell had genuinely relaxed. As a result, the system risked calculating the SOC using the Voc method while the terminal voltage was still undergoing electrochemical recovery, which reintroduced estimation inaccuracies.

Figure 7.14 illustrates the behavior of the system using this new logic.

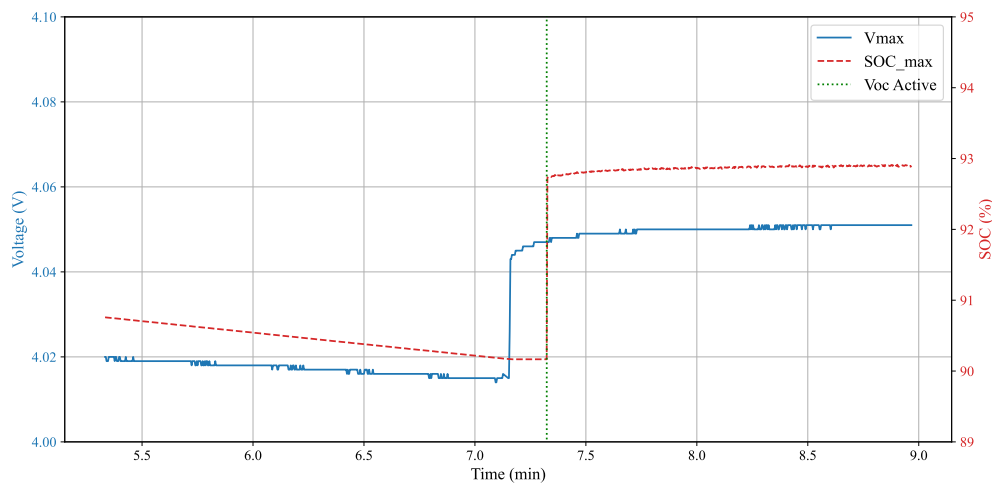


Figure 7.14: Premature Voc estimation using basic FIFO buffer.

As it is possible to observe, as expected the SOC decreases along with the voltage during the discharge. Once the current stops, meaning the discharge process ends, the SOC remains constant, as the KiBaM model depends on the presence of current to update the internal state. However, when the FIFO buffer detects four equal voltage readings, the system assumes relaxation and transitions to Voc-based estimation (indicated by the green dotted line). At this moment, the voltage is still rising as part of the natural electrochemical recovery, meaning that the SOC calculated from Voc would also continue to increase slightly. The results revealed that it was still not sufficient to guarantee that the cells had reached steady-state conditions, and highlights the need for a more robust detection strategy, which is introduced in the following subsection.

This test therefore marked a major update in the control strategy, it introduced the concept of estimating relaxation based on Δ Voltage, rather than current magnitude.

7.2.4 Time-Based Validation Using Delayed Comparison

In order to overcome the premature triggering of the relaxed state observed in the previous test, a new control strategy was developed. This version maintained the same initial FIFO logic, detecting four equal consecutive voltage samples, introducing a delayed verification mechanism to ensure that voltage stability persisted over time. Once four identical readings were detected, they were stored in a secondary buffer and compared again after a fixed delay of 5 seconds. If the values still matched, the system considered the cell to be in a relaxed state. Otherwise, the process would restart, continuously replacing the reference buffer with new candidates until stability was confirmed after the delay.

At first glance, this strategy appeared highly promising. The transition from the KiBaM model to the Voc-based estimation was now occurring further away from the end of the discharge process, as expected. This behavior is illustrated in Figure 7.15, which shows a close-up of a single cell where the transition takes place precisely when the cell voltage had already stabilized. It is also possible to observe that the SOC remains constant after the current drops to zero and only updates again once the system detects relaxation and performs a new estimation using the Voc model.

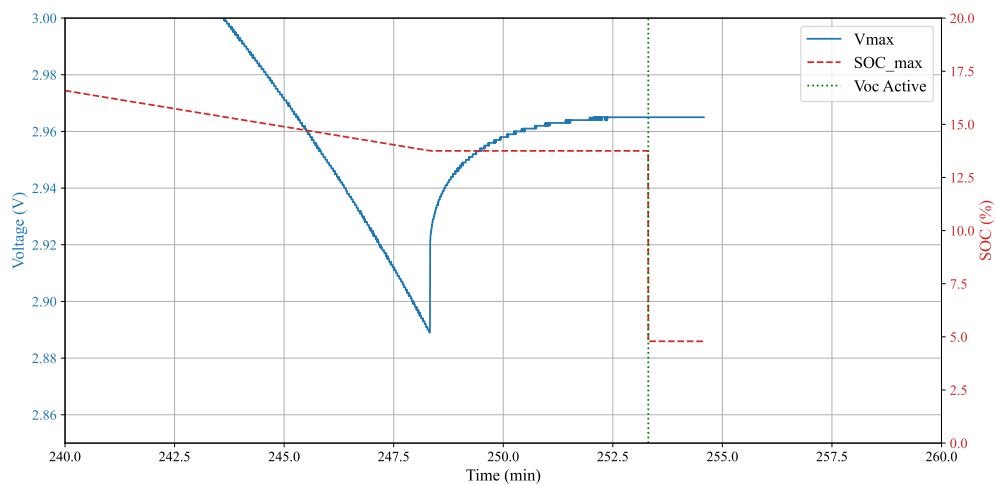


Figure 7.15: Transition from KiBaM to Voc estimation using time-delayed validation logic.

However, when the same logic was applied to all the cells simultaneously, a critical issue became evident: the relaxation flag was being triggered at different and unpredictable moments for each cell. This inconsistent behavior is shown in Figure 7.16, where all the individual SOC's are compared, and the moment of relaxation is indicated by the appearance of its new SOC value. As seen, some cells entered the relaxed state seconds earlier than others, even though the discharge process had stopped for all cells at the same time.

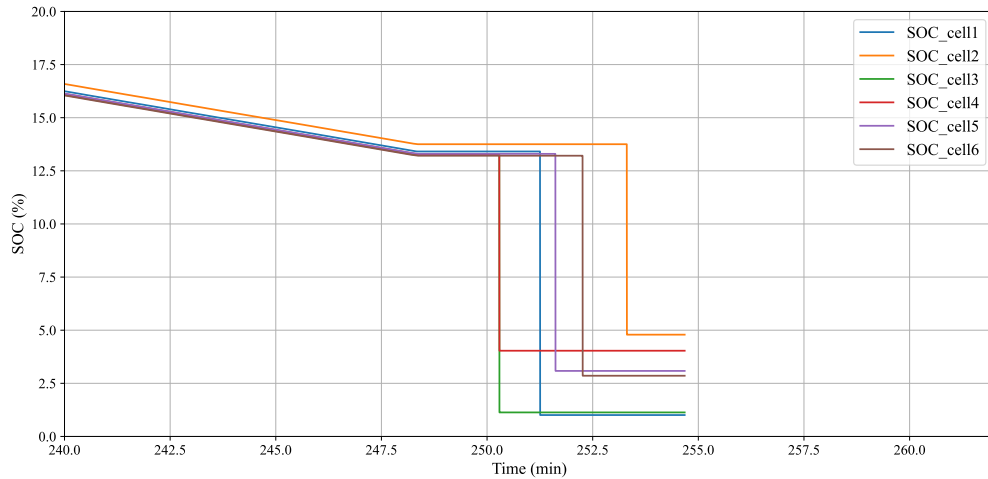


Figure 7.16: Inconsistent activation of the relaxation flag across multiple cells using the time-based delayed comparison logic.

This unpredictability arose from subtle voltage fluctuations in the measurement process. While the voltage acquisition system was very accurate, real-world components are inherently affected by small noise or drift. A particular issue emerged when a cell's terminal voltage hovered between two close values, switching back and forth between them. This made it extremely difficult for the system to validate four equal values in both the initial and delayed buffer, even if the cell was effectively relaxed. For example, the system could initially capture a buffer of [3.781, 3.781, 3.781, 3.781], but after three seconds, the buffer could become [3.781, 3.781, 3.780, 3.781], thus invalidating the condition.

This behavior introduced significant errors when analyzing the system from a full battery perspective. Since the SOC estimation from V_{oc} was applied at different times for different cells, the calculated values of SOC_{max} and SOC_{min} no longer reflected a coherent snapshot of the pack's state. For instance, a cell with a higher voltage might trigger the relaxed state and update its SOC early, while another cell with a lower voltage, potentially the limiting cell, remained locked in KiBaM mode for several seconds longer. This mismatch resulted in misleading indicators of extreme SOC estimations, as it could cause the system to underestimate the battery's remaining capacity or prematurely declare a fault.

To better investigate this systemic discrepancy, a new test was designed using pulsed current to emulate intermittent load conditions and observe how these timing mismatches affected SOC estimation in a dynamic scenario. This pulsed discharge test was specifically crafted to simulate real-world usage patterns, where load is not constant but rather applied intermittently. The goal was to evaluate whether the delayed relaxation detection could reliably trigger across all cells during short idle periods between current pulses. The current profile consisted of alternating discharge intervals and rest periods, during which no current flowed. Ideally, the system should identify these rest phases, detect stable voltages, and perform Voc-based SOC corrections synchronously across all cells.

Figure 7.17 illustrates a short segment of this pulsed discharge experiment, focusing on the voltage behavior of all cells. This specific window highlights that the battery was discharged for five minutes and then allowed to rest for ten minutes. The purpose of this zoomed view is to allow a clear time reference for the analysis of SOC estimation behavior in the subsequent figures.

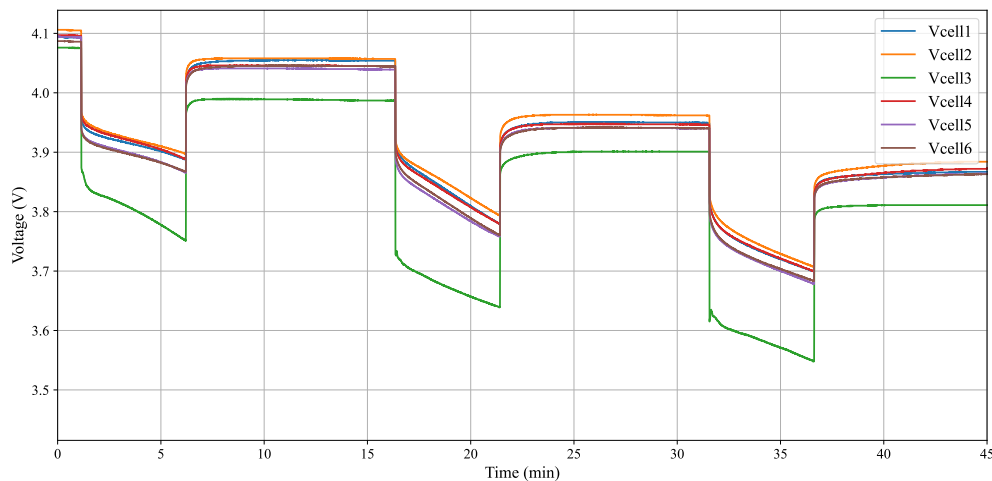


Figure 7.17: Zoomed view of the pulsed discharge test, showing the voltage evolution of all cells during a 5 minute discharge and 10 minute relaxation cycle.

The system-wide effect of these mismatches is illustrated in Figure 7.18, where the SOC each cell is represented during the same pulsed test. It becomes evident that some cells successfully entered the relaxed state and updated their SOC during the rest interval, while others remained in KiBaM mode and continued to accumulate SOC drift, although the cell voltages were already stabilized as shown in figure 7.17.

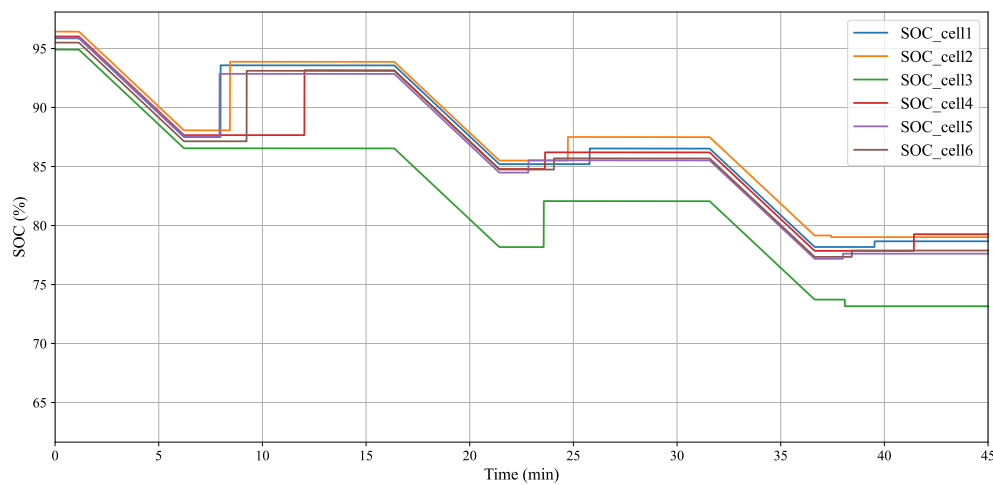


Figure 7.18: Divergent relaxation flag activation across multiple cells during a pulsed discharge scenario.

The asynchronous triggering of the relaxation flag again caused distorted readings of SOC_min and SOC_max, which in turn compromised the reliability of the BMS control logic. Figure 7.19 reinforces this observation by presenting the evolution of the SOC_max and SOC_min values during the same pulsed discharge scenario. It is clearly visible that these two estimations lose coherence during the rest intervals, precisely because the Relax_cell flags are being activated at different and unpredictable moments for each cell. This misalignment leads to an inconsistent snapshot of the battery's true state, with SOC_max and SOC_min reflecting different internal update timings rather than synchronized pack behavior. As a result, this behavior risks misleading the state machine, potentially prompting premature warnings, incorrect balancing actions, or erroneous capacity calculations.

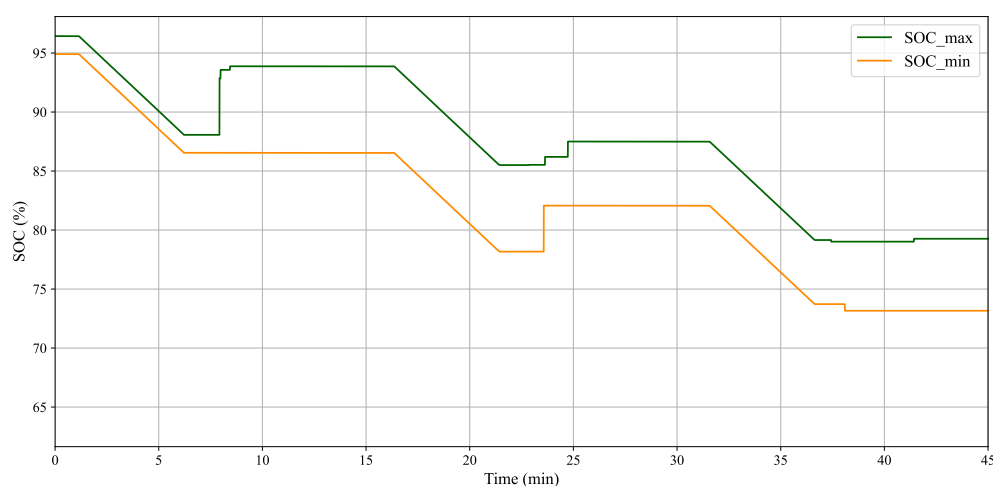


Figure 7.19: Evolution of SOC_max and SOC_min during the pulsed discharge test.

Therefore, the need for a new approach became evident, one that ensures synchronization of the relaxation state across all cells and mitigates the impact of minor voltage jitter. This challenge and its resolution are addressed in the following subsection.

7.2.5 Robust Relaxation Detection with Timed FIFO Sampling

The objective of this test was to develop a final solution that would eliminate the relaxation detection issues encountered in the previous strategies. Rather than continuously updating the FIFO buffer with new voltage samples, a new method based on fixed-interval sampling was implemented. In this approach, voltage measurements were saved only once per minute, and only during periods in which the current remained strictly equal to zero. This ensured that the buffer contained truly idle-state values and avoided transient fluctuations associated with switching phases.

Once four values were collected in this one-minute interval FIFO buffer, the system evaluated relaxation by computing the difference between the maximum and minimum voltages in the buffer. If the voltage spread across the four samples was less than or equal to 0.01 V, the cell was considered to be in a relaxed state, and the SOC was re-estimated using the Voc model. Otherwise, the system would continue waiting, collecting the next valid sample after another minute had passed.

To validate this approach, the same pulsed discharge conditions were applied again, with five minutes of discharge followed by ten minutes of rest to ensured direct comparability with previous tests.

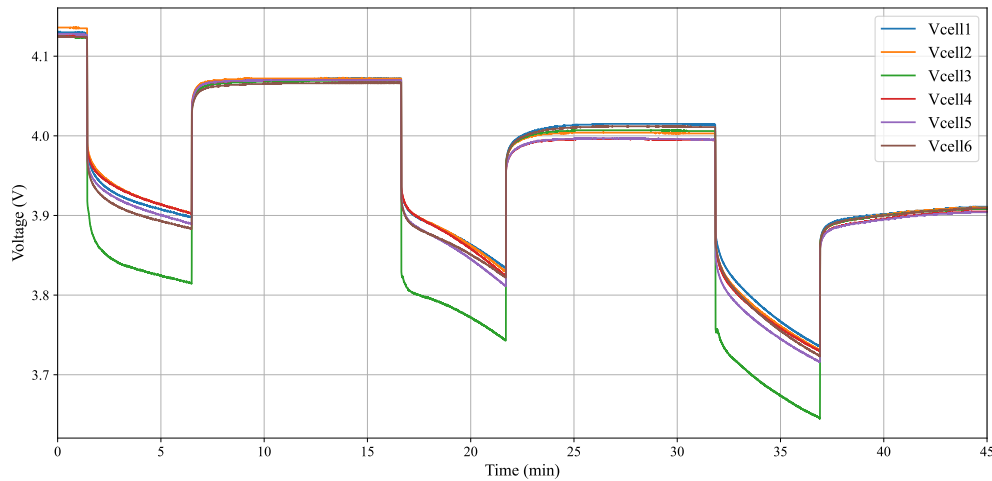


Figure 7.20: Vcells evolution of all cells using the robust relaxation detection strategy.

Figure 7.20 shows the voltage evolution of all cells over the time, and Figure 7.21 shows the SOC evolution, clearly demonstrating that all transitions to Voc-based estimation now occurred simultaneously across the entire pack. The synchronous activation of the Relax_cell flags eliminated the inconsistencies observed earlier, aligning the update timing for every cell.

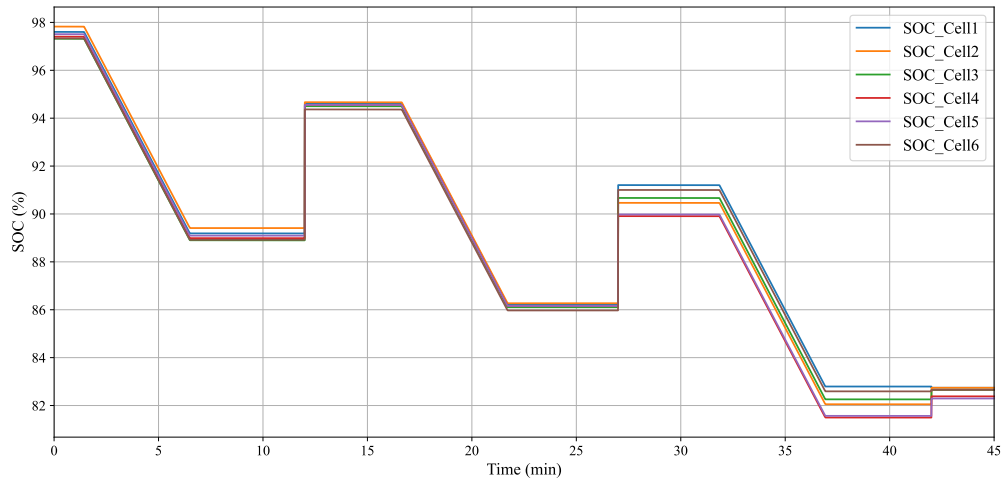


Figure 7.21: SOC evolution of all cells using the robust relaxation detection strategy.

The corresponding system-level effect is shown in Figure 7.22, which plots the SOC_max and SOC_min values during the same test. Unlike the erratic behavior seen in the previous subsection, both indicators now evolve in a consistent and coherent manner, accurately reflecting the battery's actual limits. The maximum and minimum SOC estimates now shift together in well-defined steps, directly following the synchronized detection of relaxed states.

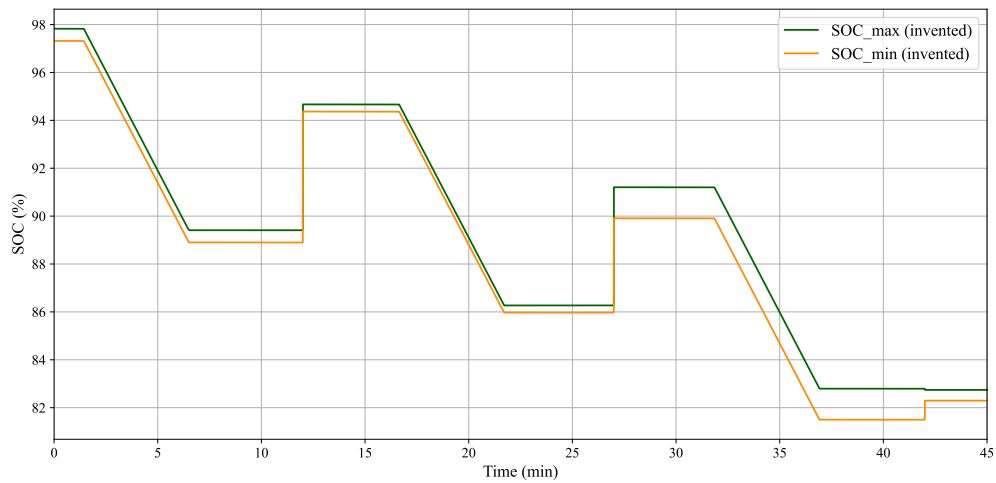


Figure 7.22: SOC_max and SOC_min evolution during the robust relaxation detection test.

This result confirms that the new timed sampling approach was not only resistant to small voltage fluctuations but also capable of enforcing strict synchronization across all cells. For the first time, the SOC transition from KiBaM to Voc was achieved simultaneously under real operating conditions, removing timing ambiguity and significantly improving the interpretability of pack-level metrics. However, the reliability of the newly estimated SOC values still remained to be demonstrated, which is the focus of the following subsection.

7.2.6 Impact of Voc Estimation Errors on Capacity Calculation

After achieving synchronized relaxation detection through the timed FIFO approach, a new issue emerged when comparing the estimated capacity (based on SOC) with the real capacity extracted from the cells. This analysis aimed to verify the accuracy of the SOC values by validating them against a physical reference: the total extracted charge measured during discharge.

The comparison was performed using the following two equations. The real extracted charge, calculated by integrating the current over time, was given by the equation 7.1. On the other hand, the estimated extracted charge was derived from the evolution of the SOC values and used as an indirect validation metric for SOC accuracy, the equation is 7.2. In the firmware implementation, both quantities were continuously monitored and stored. Figure 7.23 shows the results of this comparison, plotting both the real extracted charge and the estimated extracted charge based on SOC of each cell, using the same frame window as used above.

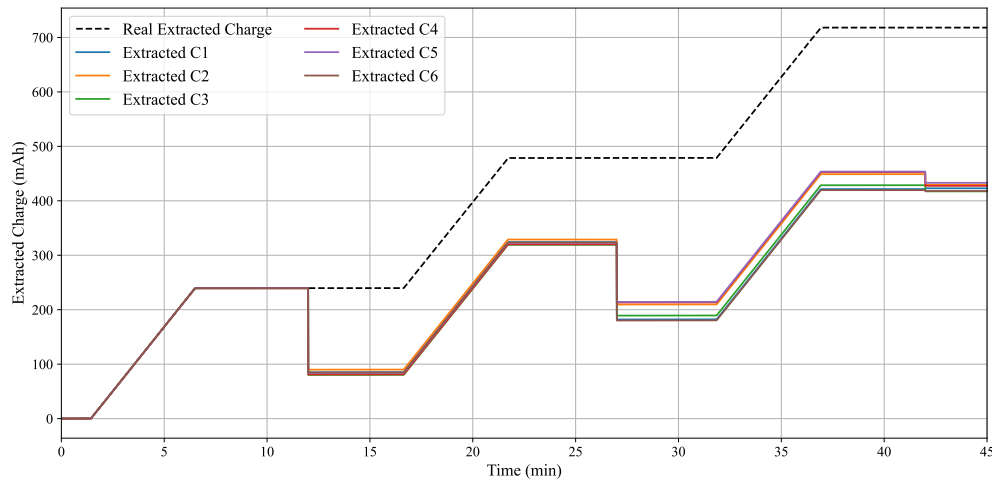


Figure 7.23: Comparison between real extracted capacity and estimated capacity.

To further quantify the deviation between the two, the error between real and estimated capacity was computed and is shown in Figure 7.24. The error is expressed as a percentage and highlights the discrepancy introduced during the SOC estimation process.

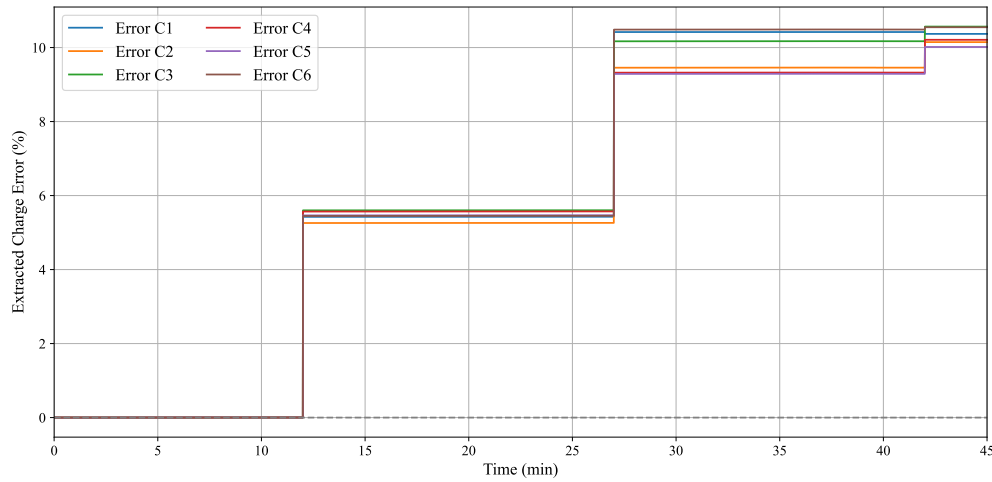


Figure 7.24: Estimation error between real and SOC-based capacity values during the test.

As observed, the estimation error increases notably at specific points in time, precisely when the SOC estimation method transitions from the KiBaM model to the Voc-based approach. Although Voc was originally considered the most reliable estimation method due to its equilibrium-based nature, it became clear that significant deviations were being introduced at the moment this estimation method was activated.

At this point, the system logic was expanded to include a condition that had been originally conceived during the planning phase of the SOC estimation tests but had not yet been fully integrated. The condition was based on the idea of evaluating the accuracy of the internal estimation performed by the KiBaM model, by comparing the estimated terminal voltage of the cells V_{out} , estimated by the equation 4.15, to the actual measured voltage.

The original objective of this method was to maintain continuous operation of the KiBaM model, even during short resting phases, under the assumption that the electrical model could deliver valid SOC estimations in the absence of current flow, and could continuously estimate an accurate value of the V_{oc} and V_{out} , based on SOC values, throughout all the discharge process. The key insight was to monitor the estimation error itself: if the KiBaM model began to diverge from physical behavior, the deviation could be captured by the growing difference between the estimated terminal voltage (V_{out}) and the measured cell voltage.

To implement this strategy, the system continuously computed V_{out} using the electrical model. If the absolute difference between V_{out} and the measured voltage exceeded 5%, it was interpreted as an indication that the SOC estimation was no longer reliable by the KiBaM method. Consequently, the system would then wait for the next opportunity, a relaxed state with no current flow, to correct the SOC using the Voc-based estimation. Allowing KiBaM to remain the default estimator for as long as its predictions were consistent with the measured voltages.

Figure 7.25 shows the terminal voltage behavior of all cells during a new pulsed discharge process with the same characteristics as the last ones.

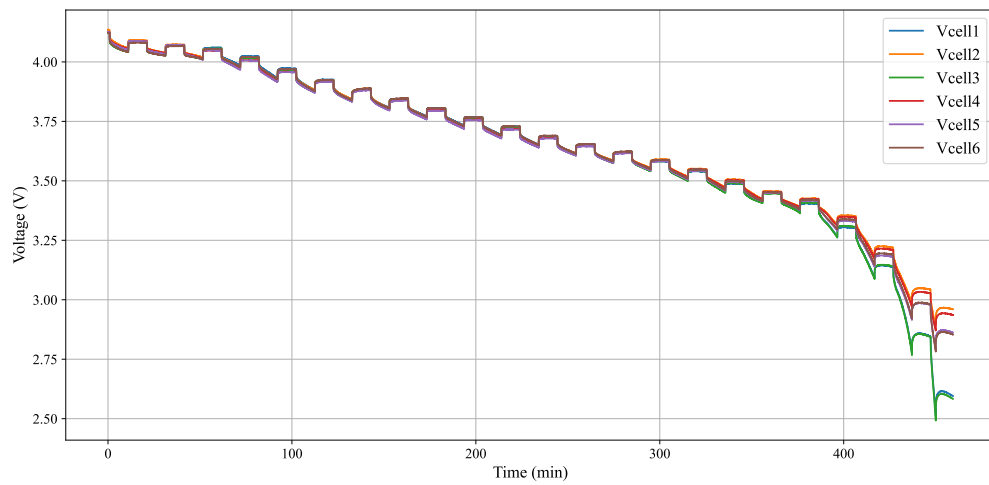


Figure 7.25: Terminal voltage evolution of all cells during the pulsed discharge in Test 5.

Figure 7.26 shows that throughout the entire test it is possible to see that in every rest period the cells would eventually enter in the relaxing state.

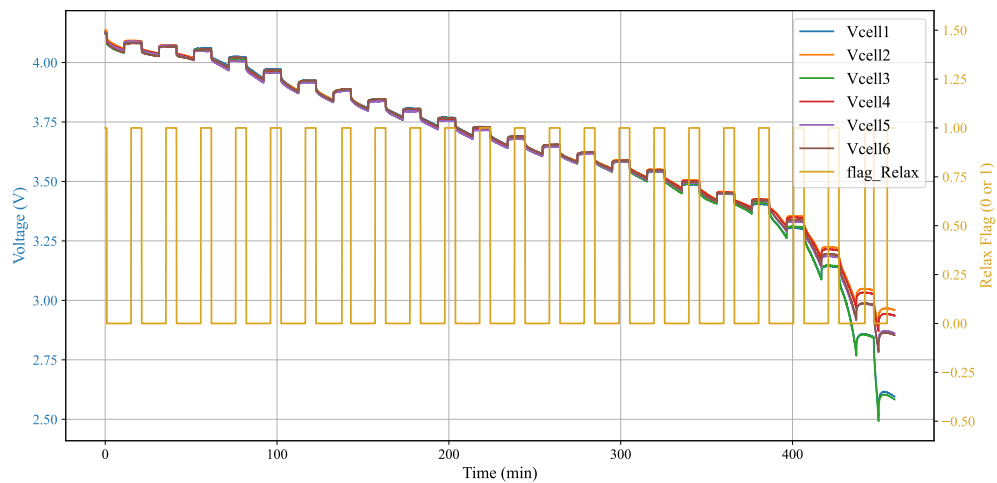


Figure 7.26: Relaxation state detection during Test 5.

Figure 7.27 presents the behavior of the measured terminal voltage, the estimated open-circuit voltage (V_{oc}), and the estimated terminal voltage (V_{out}) for a single representative cell, in this case cell 1. This visualization was limited to one cell to ensure clarity, as displaying all would hinder interpretation.

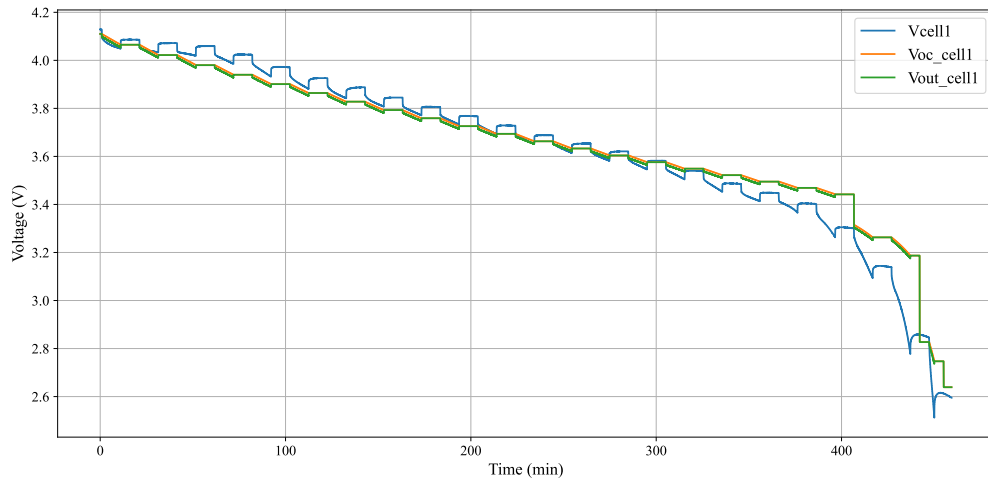


Figure 7.27: Comparison of measured voltage (V_{cell1}), estimated open-circuit voltage (V_{oc}), and estimated terminal voltage (V_{out}) of Cell 1.

Notably, the behavior of V_{oc} and V_{out} changes significantly towards the end of the discharge process. As shown in Figure 7.28, this change occurs precisely when the error between the measured voltage and the estimated V_{out} surpasses the 5% threshold. This confirms that the logic defined during the test planning phase was successfully implemented in practice: when the electrical model begins to deviate, the system reacts by activating the V_{oc} -based estimation to correct the SOC.

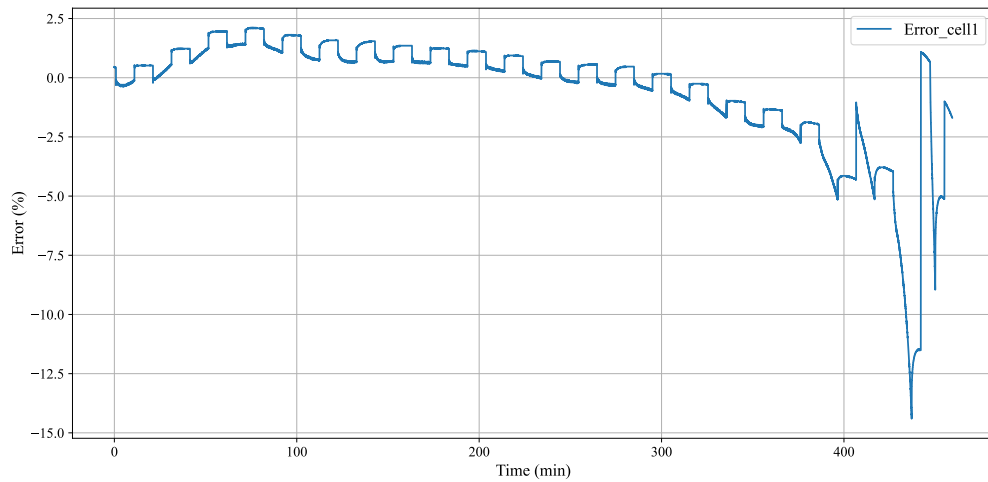


Figure 7.28: Percentage error between measured voltage and estimated V_{out} of Cell 1.

After switching to V_{oc} , the estimated V_{out} aligns more closely with the measured terminal voltage, demonstrating the system's ability to adapt and restore voltage accuracy through the hybrid approach. These results corroborate the initial assumptions made during the planning phase, indicating that the V_{oc} -based correction logic is functional and effective in restoring consistency to the electrical model.

However, while the V_{out} estimation visibly improved after the transition, the SOC values themselves did not benefit in the same way. As illustrated in Figure 7.29, the evolution of the SOC for Cell 1 remains consistent and accurate while the KiBaM model is active, but exhibits a sudden and significant deviation when the V_{oc} -based method is introduced.

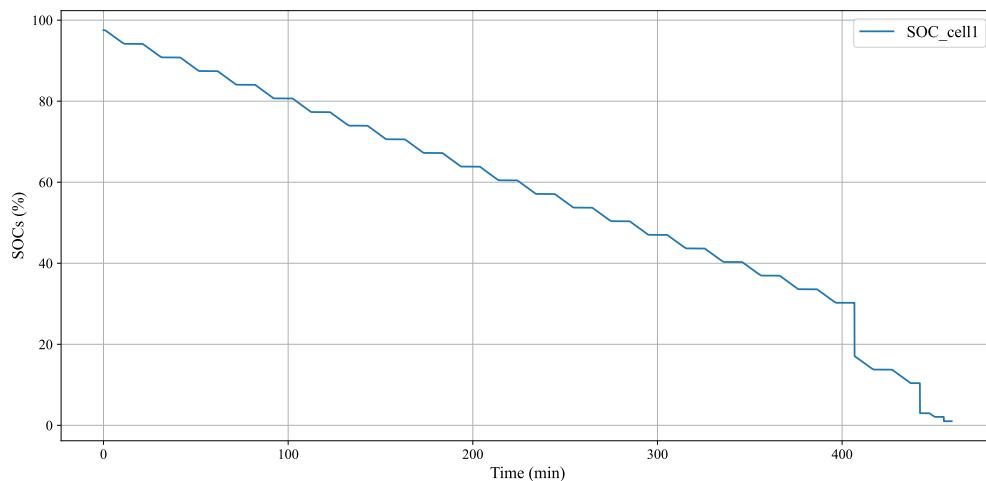


Figure 7.29: SOC evolution over time of Cell 1.

This becomes more evident in Figure 7.30, which compares the extracted charge computed from the SOC evolution to the actual extracted charge. While the KiBaM estimation maintains a near-perfect match with reality, the introduction of V_{oc} results in an abrupt increase in error.

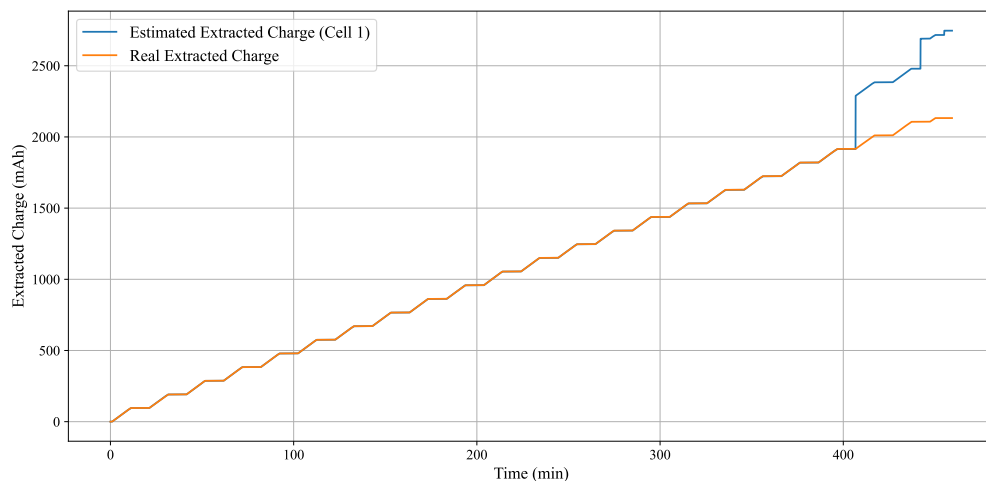


Figure 7.30: Comparison between extracted charge computed from SOC and actual extracted charge.

This outcome highlights a critical limitation of the current hybrid model. Although the transition to the estimation of the SOC by the V_{oc} method successfully improved the estimated terminal voltage alignment, it failed to preserve SOC consistency, thereby compromising energy estimation.

This discrepancy suggests that the root cause of the SOC estimation error lies in the parameterization of the hybrid model's electrical functions.

7.2.7 Final Estimation Strategy: KiBaM-Only with Voc Initialization

Despite the new condition explained above, the analysis revealed that most errors came from the Voc parametrization itself. Contrary to the initial assumption that Voc was more reliable than KiBaM, it was found to introduce greater SOC errors, particularly in the later stages of the discharge process, as expected and explained in [42].

To address this issue, the final test of the validation campaign was conducted with a revised objective: assess how accurately the KiBaM model alone could estimate the SOC. The only use of V_{oc} in this configuration was at the very beginning of the test to provide a reasonable initial SOC value, seeding the KiBaM tanks accordingly. From that point onward, the system operated exclusively with current-based integration and KiBaM dynamics. This approach aimed to eliminate the impact of electrical model parametrization errors during operation, while still ensuring realistic starting conditions.

Figure 7.31 shows the SOC evolution of a cell, in this case Cell 1, over time, highlighting the consistent and smooth behavior of the estimation without any abrupt transitions or artifacts.

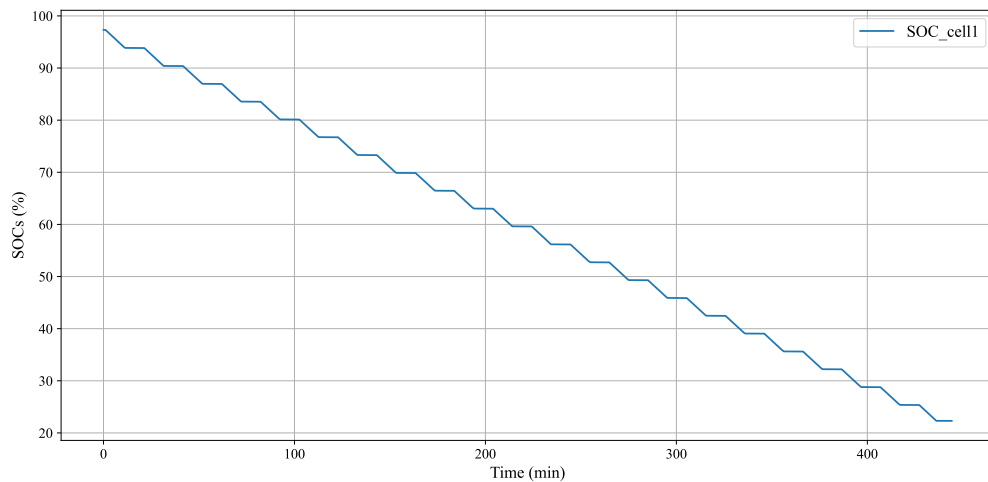


Figure 7.31: SOC estimation of Cell 1 using KiBaM-only strategy.

However, SOC alone is not sufficient to quantify estimation quality. To verify its accuracy, Figure 7.32 compares the total extracted charge derived from SOC values with the real measured charge extracted during the discharge. The close alignment of these two curves confirms that the KiBaM estimation produced minimal error and effectively captured the real dynamics of the system.

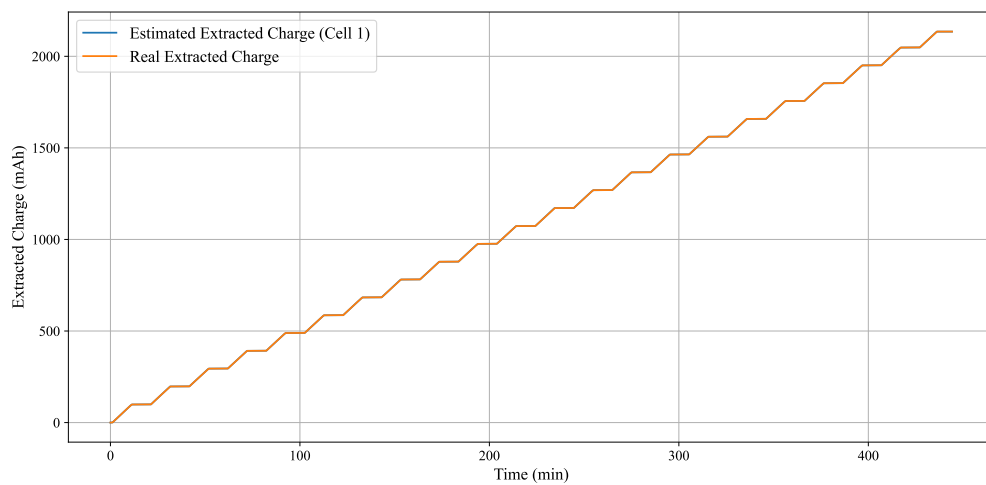


Figure 7.32: Comparison between extracted charge from SOC estimation and actual charge extracted.

The minimal discrepancy observed in this figure is quantitatively supported by the direct SOC error curve shown in Figure 7.33. This plot represents the absolute error between the extracted charge estimated by the KiBaM model and the real charge measured during the discharge.

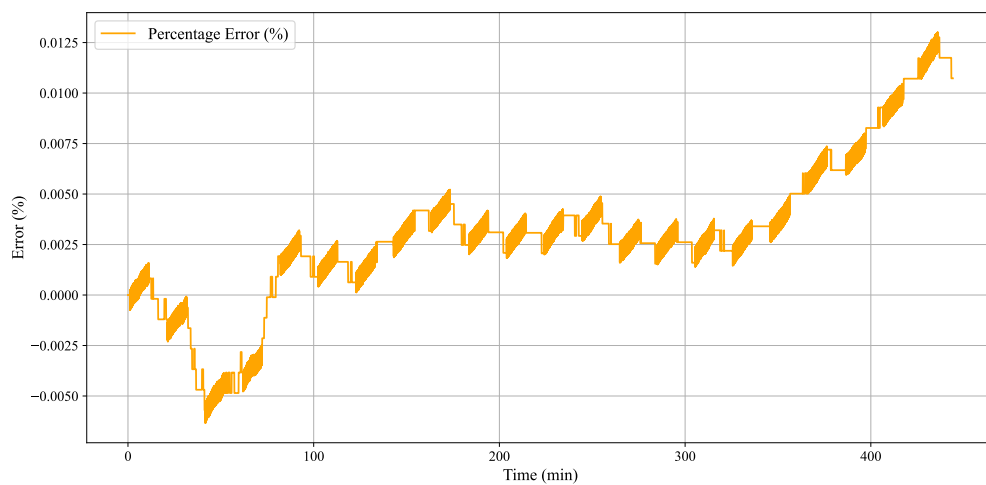


Figure 7.33: Percentage SOC error between extracted charge from KiBaM estimation and actual extracted charge.

The magnitude of this error remained consistently low throughout the test, confirming that the KiBaM-based SOC estimation produced highly accurate results.

In contrast, the voltage estimation performance was notably impacted. Figure 7.34 illustrates the V_{cell1} , the estimated V_{oc} , and the estimated V_{out} for Cell 1 throughout the test. While V_{out} generally followed the trend of the measured voltage, the mismatch remained significant due to the imperfect electrical model parameters.

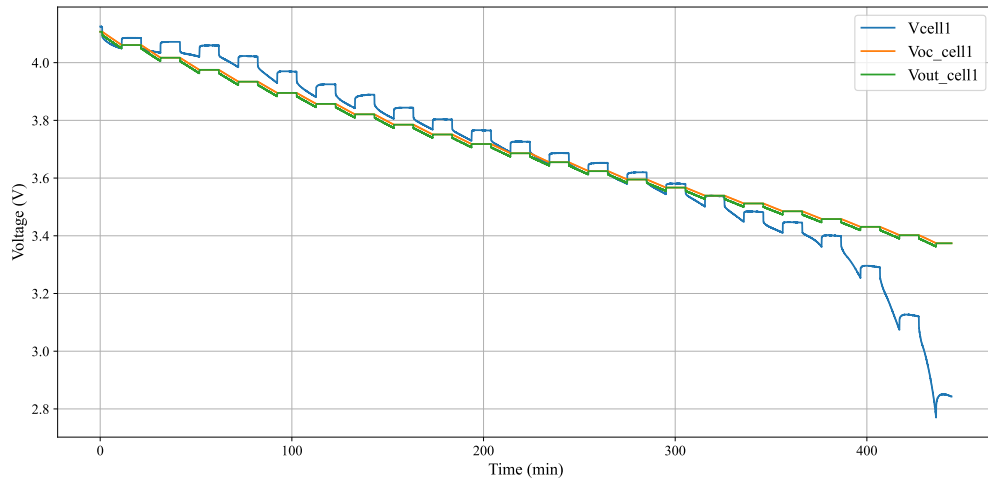


Figure 7.34: Comparison of V_{cell1} , V_{oc} , and V_{out} of Cell 1, with the final strategy.

This is further emphasized in Figure 7.35, which plots the percentage error between V_{out} and V_{cell1} , revealing persistent deviations.

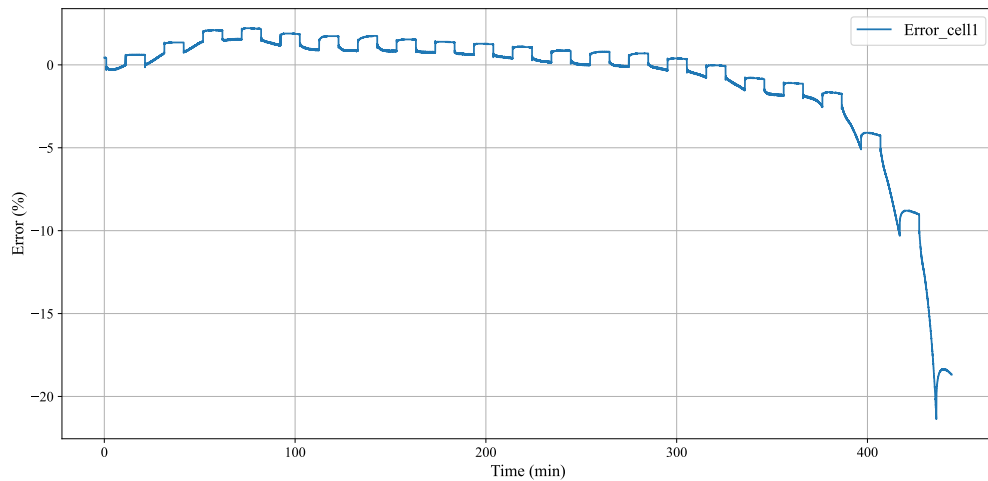


Figure 7.35: Percentage error between V_{cell1} and estimated V_{out} of Cell 1.

These results confirm that the problem lies not in the KiBaM SOC estimation, which proved to be highly reliable. Ironically, the use of V_{oc} estimation, which was initially meant to correct possible SOC errors from KiBaM, ended up amplifying the error when the electrical model was poorly tuned. Thus, the decision to remove V_{oc} from the final implementation was not only justified but necessary to improve the overall consistency of the system.

With this objective accomplished, and the root cause of the V_{out} estimation errors clearly attributed to the high error parameterization of the electrical model, the next logical step was to investigate this with more detail. Section 7.4 will address a dedicated strategy to correct this problem, introducing an automatic internal resistance reestimation algorithm designed to dynamically

update the parameters used in voltage reconstruction, ultimately improving the fidelity of V_{out} and improving the performance of future tests.

7.2.8 Evaluation of Electrical Model Parametrization

To better understand the origin of these deviations, it became necessary to revisit the electrical model parametrization. The parameterization of the functions responsible for relating SOC and open-circuit voltage were previously presented in Equations 4.6 and 4.13 in Section 4.3.

The parameter set applied in this work corresponds exactly to the values originally defined by [42] are:

Table 7.2: Electrical model and logistic SOC inversion parameters used.

Parameter	Value / Expression
Voc estimation (logistic)	
A_{SOC}	108.7827
B_{SOC}	4.50
C_{SOC}	3.6491
Electric model (polynomial)	
a_{elet}	-0.2655
b_{elet}	0.1711
c_{elet}	2.403×10^{-7}
d_{elet}	-2.16×10^{-14}
e_{elet}	0.006642
f_{elet}	3.239

As discussed previously in Section 4.3, the second equation is a simplified inversion of the first one, adjusted to directly map V_{OC} to SOC. This inverted function was implemented to enable SOC estimation during rest periods when only voltage is available. However, since it is not an exact analytical inversion of Equation 4.6, but rather a fitted approximation, discrepancies can arise, especially at the extremities of the SOC curve.

Although in the information based report had already noted these limitations in his dissertation, the goal here was to validate the real-world behavior of the estimation system using the exact parameters he proposed. The next part of this subsection will present the parameter values used and further analyze their impact on estimation accuracy.

To quantify these discrepancies, it is possible to take the $V_{OC}(SOC)$ equation and numerically invert it to obtain $SOC(V_{OC})$ directly.

$$SOC(V_{OC}) = \frac{A}{1 + \exp(-B \cdot (V_{OC} - C))} \Leftrightarrow V_{OC}(SOC) = C - \frac{1}{B} \cdot \ln\left(\frac{A}{SOC} - 1\right) \quad (7.3)$$

Then is possible to compare this numerically derived function against the fitted logistic form from Equation 4.6. This approach allows both SOC estimation curves to be represented on the same graph, making it possible to visually assess the divergence between the exact inversion and the fitted approximation.

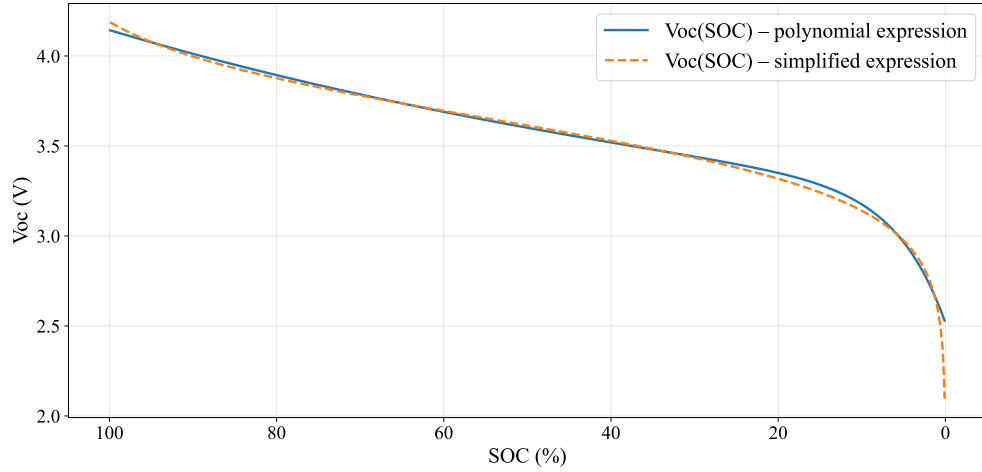


Figure 7.36: Comparison between the SOC function obtained by inverting $V_{OC}(SOC)$ and the fitted logistic $SOC(V_{OC})$ function.

The voltage discrepancy $\Delta V = V_{OC}^{poly}(SOC) - V_{OC}^{log-inv}(SOC)$ is then plotted in Figure 7.37.

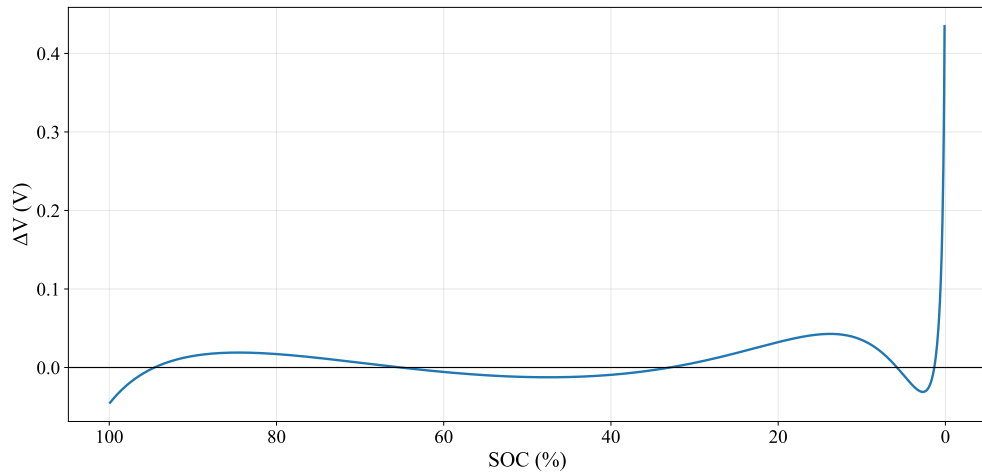


Figure 7.37: Discrepancy between both functions.

Finally, the translation of that discrepancy to a percentage of the full OCV swing, from 4.2 V to 2.5 V, is shown in Figure 7.38.

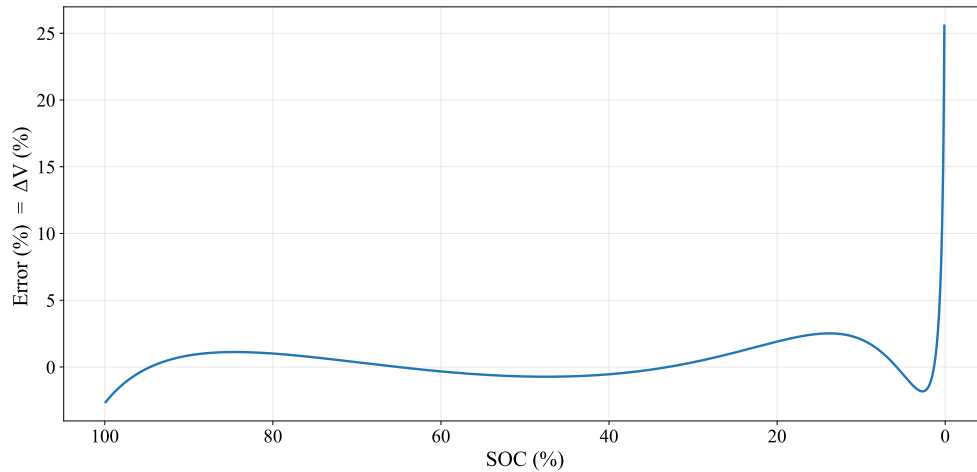


Figure 7.38: Relative error of the comparison between the 2 electrical model functions.

The figures above shows that in the two equations already exhibits discrepancy relative to the full parametrization of both. The differences remain negligible throughout the mid-SOC region and becomes more significant at the end of the discharge process, when SOC approaches 0%.

Now that it is clear that these functions are not fully compatible, it becomes essential to assess how they behave under real operating conditions. To do this, the same experimental test described in the previous subsection was used, in which the cells were discharged from approximately 4.15V until one of them reached the lower cutoff voltage of 2.50V. This allows the model outputs to be compared directly against measured values throughout the entire usable SOC range, providing insight into how the $V_{OC}(SOC)$ and $SOC(V_{OC})$ parameterizations perform in practice and how they influence the final V_{out} estimation.

The next step is to investigate whether the dominant source of error arises from the $SOC(V_{OC})$ inversion or from the $V_{OC}(SOC)$ function. This was carried out in two stages:

7.2.8.1 Evaluation of $SOC(V_{OC})$

It is important to note that the evaluation presented here is not 100% accurate, since the tests did not cover the full SOC range from 4.20, V ($SOC = 100\%$) down to 2.50, V ($SOC = 0\%$). The charger characteristics prevented charging the cells to exactly 4.20 V before the discharge tests, which means that the SOC–Voc relationship could not be perfectly mapped over the entire range. Nevertheless, the available data is sufficient to provide a solid basis for identifying trends and guiding future reparameterization.

Since the analysis is based on the previous discharge test, the first and most straightforward verification consisted of examining the behaviour of the $SOC(V_{OC})$ estimation at the very beginning of the test. This initial instant is the unique moment in which $SOC(V_{OC})$ is actually applied to estimate SOC, as once the discharge current becomes non-zero the estimation is handed over to the KiBaM model. In this condition, the cells were fully relaxed and the SOC could be derived

directly from the open-circuit voltage, making it possible to isolate and evaluate the accuracy of the $SOC(V_{OC})$ function.

The only available reference for this comparison is the manufacturer's datasheet discharge curve, which provides the nominal relation between SOC and terminal voltage. It must be noted, however, that the datasheet values reflect operation under load, and therefore include the effect of discharge current and extracted charge, whereas the measured value here corresponds to a true open-circuit condition. Knowing this, the comparison performed here cannot be considered 100% accurate, but it remains helpful because the selected values correspond to the exact voltage measured immediately after the current drop and the last SOC estimated by the $SOC(V_{OC})$ function, which would not vary significantly in this short interval.

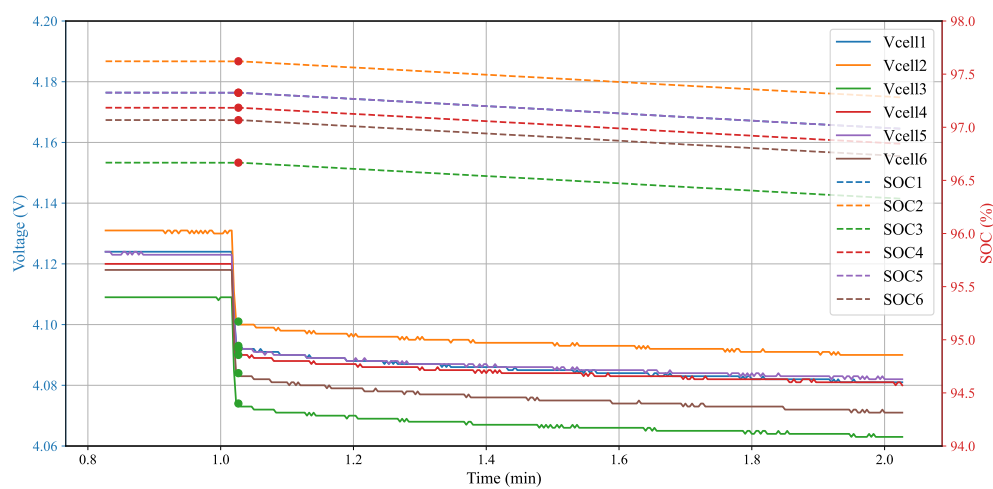


Figure 7.39: Extracted SOC–voltage points at the beginning of the discharge, used for comparison against the datasheet curve.

The SOC and measured voltage values obtained for all cells at this initial instant were then transformed into extracted charge, assuming that each cell had the nominal total capacity of 2800 mAh defined in the datasheet. The Table 7.3 provides the first set of points that could be placed against the manufacturer's discharge profile.

Table 7.3: Initial point values.

Cell	V_{cell} (V)	SOC (%)	Extracted Charge (mAh)
1	4.093	97.325	74.900
2	4.101	97.621	66.612
3	4.074	96.667	93.324
4	4.090	97.184	78.854
5	4.092	97.325	74.900
6	4.084	97.067	82.124

As can be seen from the datasheet curve, the voltage values at this early stage are close to the measured ones, suggesting that the $SOC(V_{OC})$ parameterization behaves reasonably well under near-fully charged conditions. Nevertheless, it must be emphasized that this agreement is not a definitive validation of the function, as the largest deviations are expected to appear at the end of the discharge process.

To gain a more complete view of the function's accuracy, the analysis was extended beyond this initial instant. With the objective of evaluating the behaviour of the $SOC(V_{OC})$ function throughout the discharge, specific instants were selected where the current dropped to zero and the cells reached a relaxed state. The selected instants and their corresponding relaxed-state variable are illustrated in Figure 7.40.

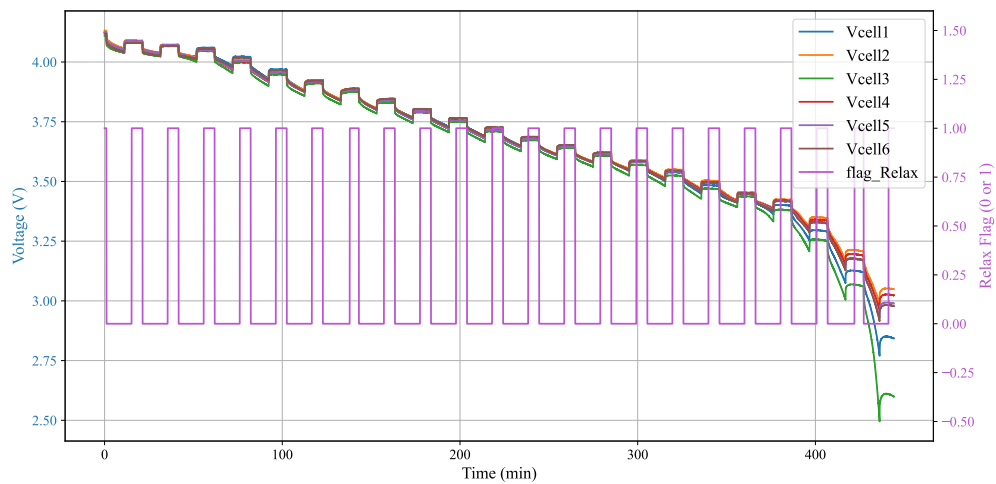


Figure 7.40: Voltage profile during the reference discharge test and the relaxed state variable.

These conditions provide the closest approximation to true open-circuit voltage, making them particularly valuable for validating the function parameterization. With this, the function $SOC(V_{OC})$ was simulated by applying the function to the measured voltage to estimate SOC, generating a set of SOC points distributed along the test timeline, as shown in Figure 7.41.

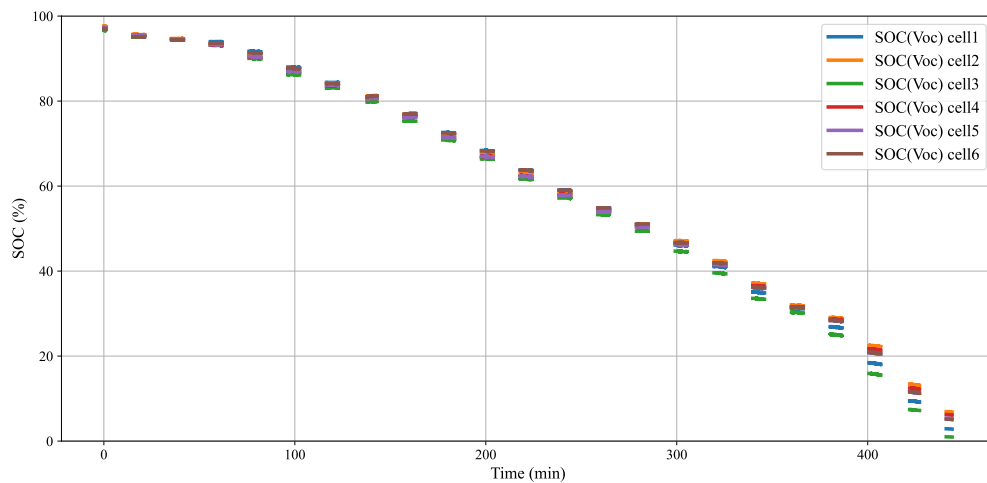


Figure 7.41: SOC points estimated from the $SOC(V_{OC})$ function during relaxed states.

These SOC estimations were then translated into an ideal extracted charge, producing an aligned curve from which the corresponding relationship could be derived. This enabled a direct comparison between the estimated SOC–Voc points and the idealized datasheet curve [43], as shown in Figure 7.42.

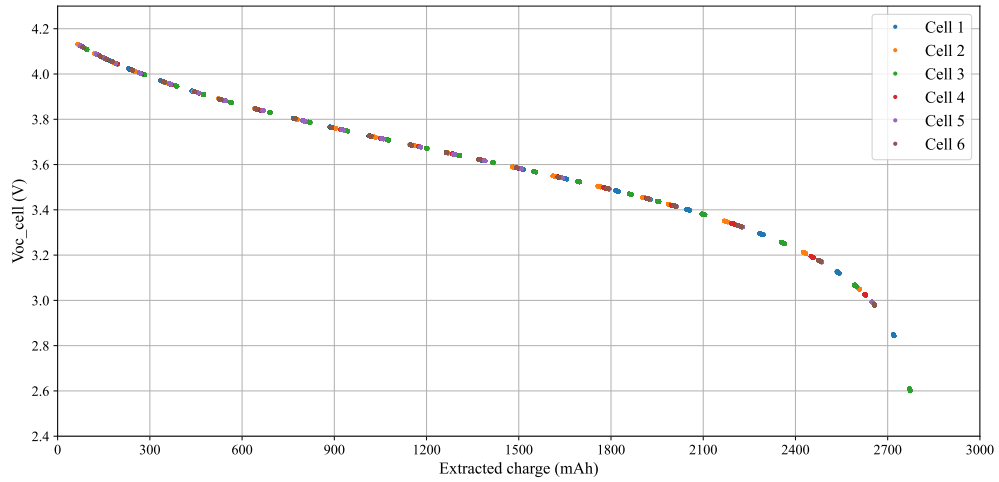


Figure 7.42: Evolution of the V_{OC} as a function of the extracted charge.

To have a clearer perspective on the comparison between the estimated behaviour and the manufacturer's reference, the values extracted from [42] were also considered. In his work, the datasheet discharge curve was processed through curve fitting, producing a continuous analytical representation. Using this fitted curve as a benchmark, the points obtained in this work from the $SOC(V_{OC})$ function could be directly compared against the idealised datasheet profile. The result of this comparison is presented in Figure 7.43.

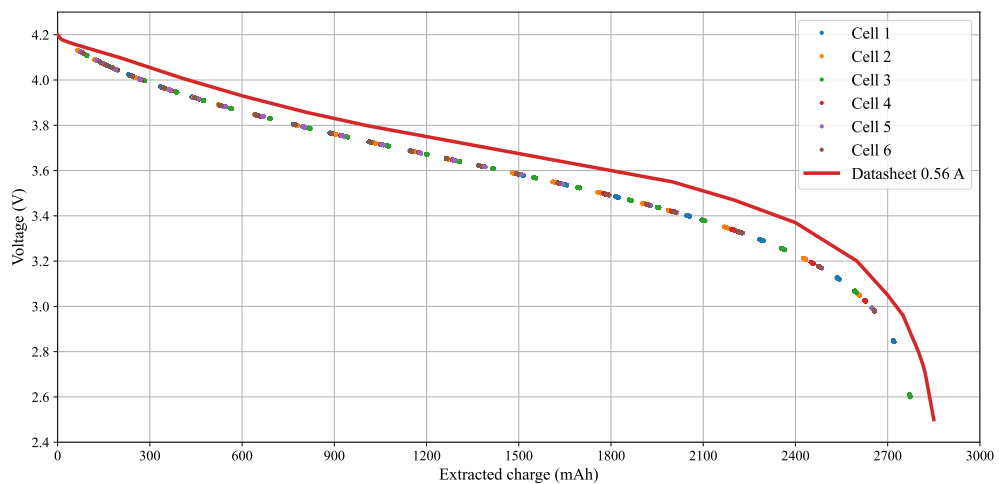


Figure 7.43: Comparison between the $SOC(V_{OC})$ function parametrization and the fitted datasheet curve from cell datasheet.

From this inspection, several conclusions can be drawn. First, it is evident from Figure 7.43 that the two curves follow a broadly similar trajectory, confirming that the adopted parametrization captures the general trend of the $SOC(V_{OC})$ relationship. Nevertheless, it must be emphasized once again that the comparison shown here represents an idealized condition, since it assumes a nominal cell capacity of 2800 mAh and negligible internal resistance. Under real conditions, deviations are expected to be larger due to cell aging, capacity fade, and resistive effects.

A second important observation is that at the beginning of the discharge process, when the voltage is close to 4.15 V, the estimated $SOC(V_{OC})$ curve closely matches the fitted datasheet reference. This validates the decision to rely on $SOC(V_{OC})$ for the initial SOC estimation and then switch to KiBaM for subsequent updates, as the risk of introducing significant error at this stage is minimal. However, as the discharge progresses, the divergence between the curves becomes more apparent, particularly toward lower SOC values, confirming that prolonged reliance on $SOC(V_{OC})$ would indeed accumulate error.

Overall, these results show that using $SOC(V_{OC})$ as a starting point, with the cells voltage close to 4.15V, is a valid strategy, and that the impact of its initial error on subsequent SOC tracking is negligible, since the hybrid approach quickly shifts to KiBaM, which governs the long-term estimation dynamics.

7.2.8.2 Evaluation of $V_{OC}(SOC)$

Following the analysis of the $SOC(V_{OC})$ function, it is clear that the accuracy of the $V_{OC}(SOC)$ parameterization cannot be evaluated in complete isolation, as it directly depends on the SOC values produced by the inversion function, and since this function already has errors, the evaluation of the $V_{OC}(SOC)$ cannot be directly done. Nevertheless, this function plays a key role in the hybrid SOC estimation model, where the initial SOC is estimated from V_{OC} and subsequently updated by the KiBaM model during active current phases. In this configuration, although the initial SOC estimation error can propagate, the SOC decline over time remains primarily dictated by the KiBaM dynamics, meaning that the shape of the $V_{OC}(SOC)$ curve can still be assessed by comparing its evolution against real measurements.

To properly evaluate this behaviour, it is not sufficient to examine only the best or worst performing cell. Instead, the comparison must be extended to all cells in the pack, since dispersion effects can amplify or mask model inaccuracies. Figure 7.44 shows the evolution of both measured V_{cell} and estimated V_{out} across all six cells during the complete discharge process.

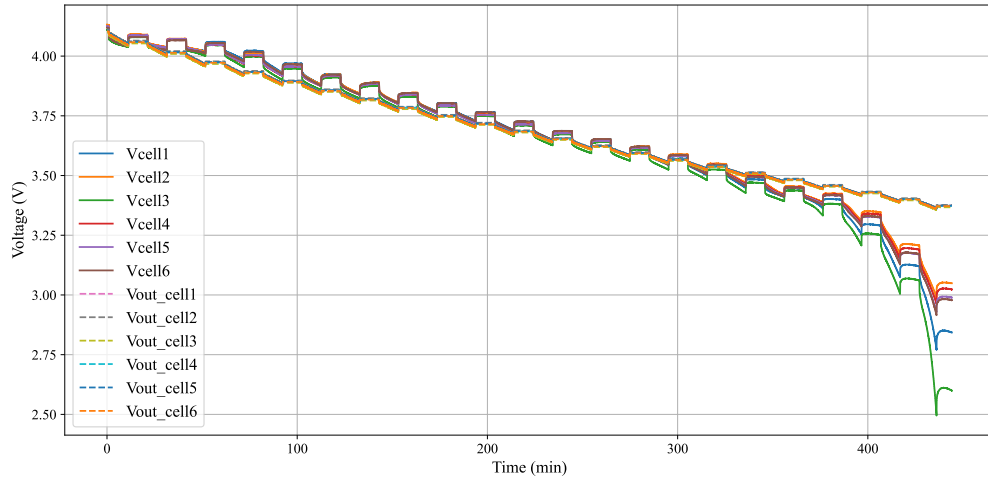


Figure 7.44: Comparison between measured V_{cell} and estimated V_{out} for all six cells during the reference discharge.

Two distinct operating conditions were evaluated. The first corresponds to moments where the discharge current was equal to zero ($I = 0A$), in which case the terminal voltage V_{out} coincides with the open-circuit voltage V_{OC} . These instances allow a direct comparison between the $V_{OC}(SOC)$ values estimated by the model and the measured V_{cell} , eliminating the influence of the internal resistance model.

Figure 7.45 shows the evolution of both measured V_{cells} and estimated V_{out} during rest periods. Overall, the two curves follow the same downward trend, although deviations become more visible toward the end of the discharge.

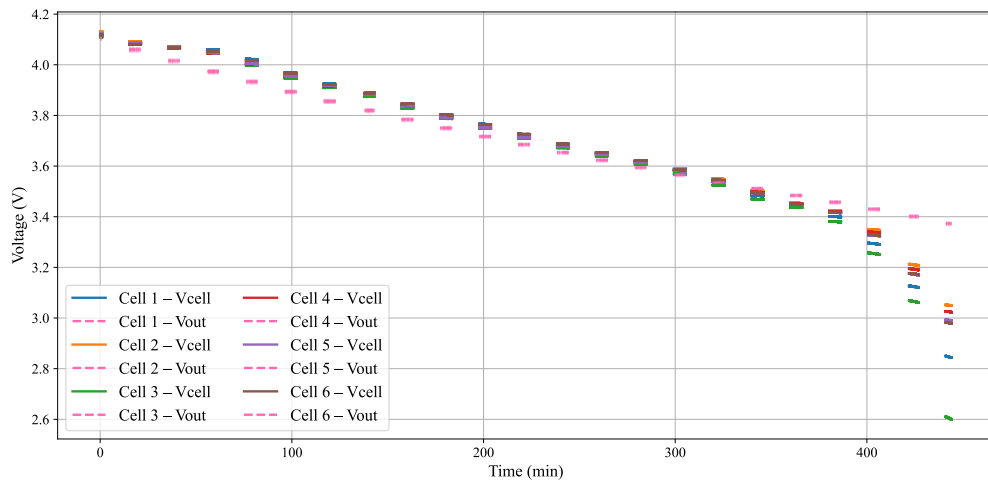


Figure 7.45: Comparison between measured V_{cell} and estimated V_{out} during rest periods ($I = 0A$).

To better visualize these deviations, Figure 7.46 presents the absolute voltage error between V_{cells} and its correspondent V_{out} for the same rest periods.

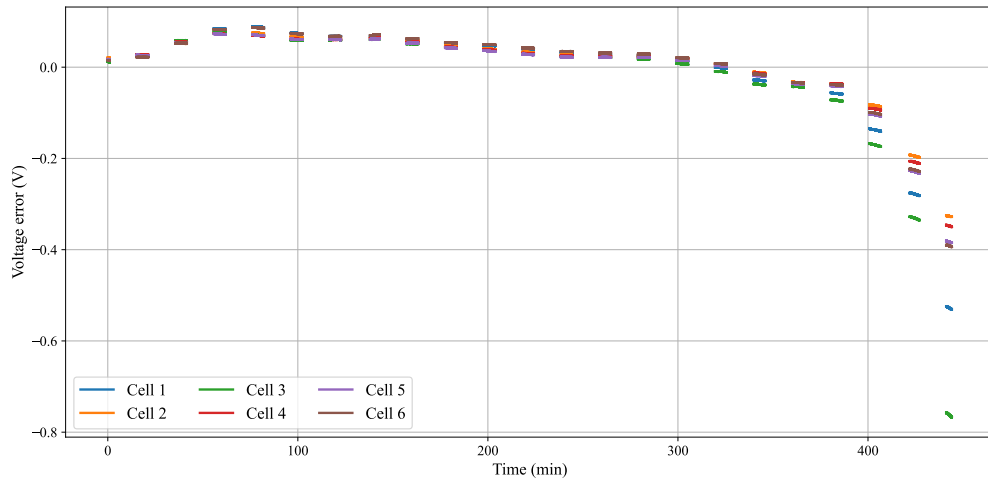


Figure 7.46: Absolute voltage error between V_{cell} and V_{out} during rest periods.

Finally, Figure 7.47 translates this error into a percentage relative to the full OCV swing. This representation makes it clear that the error remains low for most of the test but rises sharply as SOC approaches 0%.

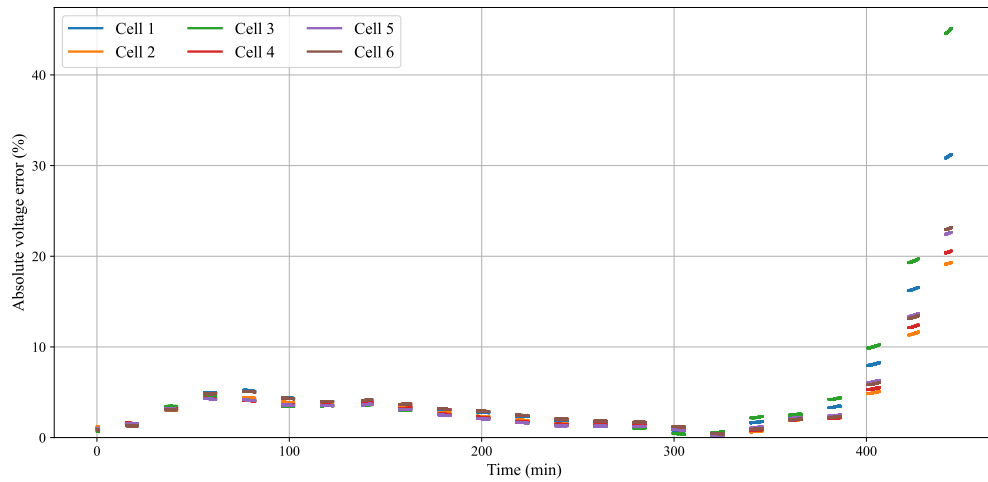


Figure 7.47: Percentage voltage error relative to the total OCV swing during rest periods.

The second operating condition corresponds to active discharge periods with non-zero current ($I \neq 0A$), where the comparison between the estimated V_{out} and the measured V_{cell} reflects not only the $V_{OC}(SOC)$ accuracy but also the combined effect of the internal resistance model. In this work, a fixed $R_{int} = 0.2\Omega$ value from the cell datasheet was used instead of real-time measured values, which inherently limits the precision of this comparison.

Figure 7.48 compares V_{cells} and V_{outs} during these active discharge periods.

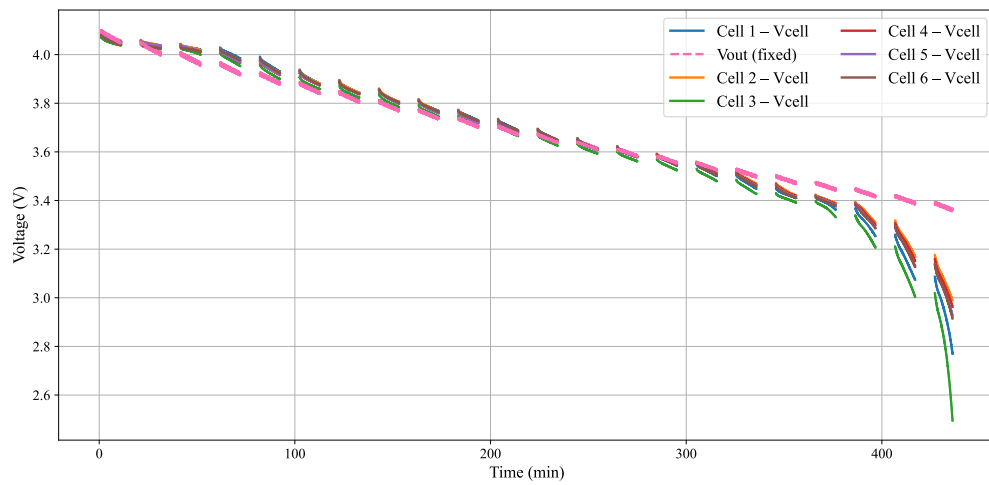


Figure 7.48: Comparison between measured V_{cells} and estimated V_{out} during active discharge periods ($I \neq 0A$).

Figure 7.49 then shows the absolute voltage error in this condition. Compared to the rest-state analysis, the deviations, as expected, were minimized throughout the discharge process, but arose the error in the end.

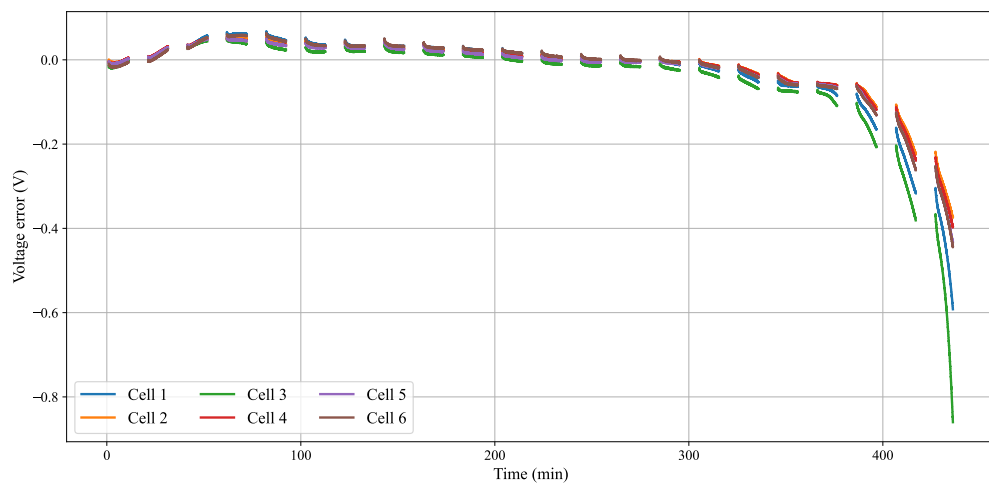


Figure 7.49: Absolute voltage error between V_{cells} and V_{out} during active discharge.

Lastly, Figure 7.50 expresses the same results in percentage form.

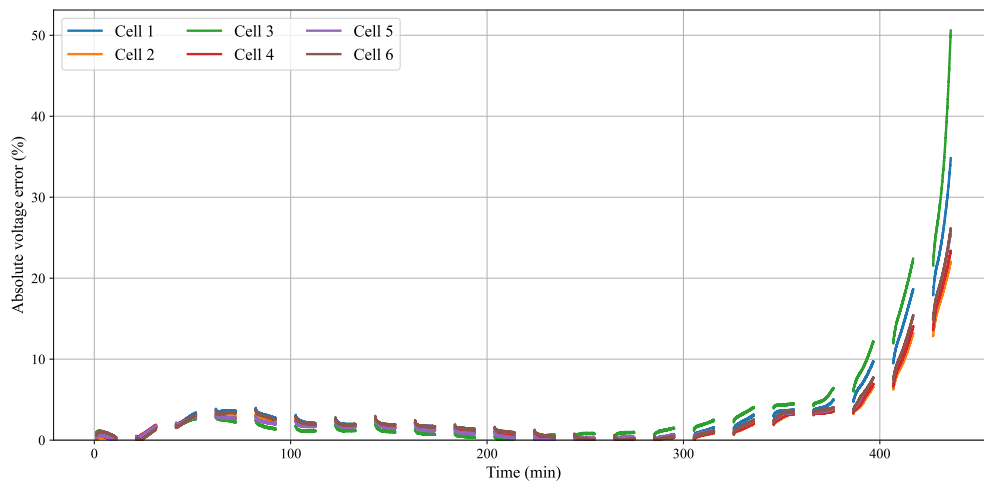


Figure 7.50: Percentage voltage error relative to the total swing, during active discharge.

To better assess the behaviour of the parameterization under real operating conditions, a statistical analysis was carried out for each individual cell. This allows the evaluation of whether the discrepancies are evenly distributed or whether certain cells present higher errors than others.

Table 7.4: Error metrics for all cells during rest and discharge conditions.

Condition	Cell	Mean Error (%)	Max Error (%)	Standard Deviation (%)	Relative Error (%)
$I = 0A$	1	4.121	31.235	5.024	6.498
	2	3.243	19.294	3.174	4.538
	3	4.398	45.176	7.039	8.299
	4	3.253	20.588	3.394	4.701
	5	3.339	22.647	3.782	5.045
	6	3.694	23.176	3.763	5.273
$I \neq 0A$	1	3.548	34.824	5.535	6.575
	2	2.624	22.059	3.812	4.628
	3	3.967	50.588	7.300	8.308
	4	2.670	23.412	4.040	4.843
	5	2.836	25.588	4.411	5.244
	6	3.033	26.176	4.389	5.335

As expected, the average error across all cells remains relatively small, indicating that the overall system estimation is not inaccurate. However, as also reflected in the maximum error values, the deviations toward the end of the discharge are very large. This behaviour confirms that while the parametrization captures the general voltage–SOC trend, it struggles to reproduce the non-linear dynamics that dominate at low SOC.

In summary, the analysis shows that the $V_{OC}(SOC)$ function cannot be perfectly validated in isolation, and that each modelling block introduces errors that, even small, accumulate when combined. The fact that the measured rest voltages did not reproduce the expected decline near the cutoff suggests either limitations in the parametrization or significant dispersion in the cell conditions. This observation motivates a broader validation of the complete hybrid estimation system, which is addressed in the following subsection.

7.2.9 Hybrid System Validation

The hybrid SOC estimation system relies on a combination of $SOC(V_{OC})$ for the initial condition and the KiBaM model to track the SOC during dynamic operation. The accuracy of this approach therefore depends not only on the parametrization of the functions but also on the actual behaviour and condition of the cells under test.

The previous analyses demonstrated that the $SOC(V_{OC})$ function aligns reasonably well with the datasheet values at the beginning of discharge, as shown in Figure 7.43, while the KiBaM model was validated to estimate the extracted charge with high fidelity. In order to translate the estimated SOC into extracted charge, the nominal capacity of 2800 mAh specified in the datasheet was assumed, corresponding to the behaviour of an ideal and undegraded cell. However, the tested cells no longer exhibited this full nominal capacity, meaning that this assumption inevitably introduced a mismatch between the estimated values and the real cell behaviour. Consequently, part of the error observed in the voltage reconstruction can be attributed not only to the model parametrization but also to the difference between nominal and actual usable cell capacity.

To investigate this, the initial SOC obtained from $SOC(V_{OC})$ was converted into an initial depth of charge of approximately 80 mAh, as showed in Table 7.3. This value was then incremented with the extracted charge estimated by KiBaM, which produced around 2120 mAh during the test, as showed in Figure 7.32. The resulting total of about 2200 mAh corresponds to the real usable capacity of the cell, which is significantly lower than the nominal 2800 mAh. This conclusion is consistent with the measured end-of-discharge voltage of 2.6 V, whereas the datasheet indicates that 2200 mAh corresponds to a voltage near 3.35 V.

To further validate this conclusion, the final voltages measured at the end of the discharge were compared against the expected reference value of 3.35 V. The deviation between the measured voltages and the reference highlights how much the tested cells diverge from the nominal behaviour. Table 7.5 presents the final measured voltages for each cell and the associated percentage error relative to the datasheet value.

Table 7.5: Final measured voltages and errors relative to the datasheet reference at 2200 mAh.

Cell	Final Voltage (V)	Error (%)
1	2.848	29.529
2	3.050	17.647
3	2.607	43.706
4	3.024	19.176
5	2.991	21.118
6	2.980	21.765

This analysis ultimately shows that the apparent errors cannot be attributed solely to the parametrization of the functions, but are instead a direct consequence of the gap between the ideal conditions assumed during parametrization and the actual behaviour of real cells, which vary in performance and degradation state. In other words, while the functions were defined for a perfect cell environment, the tested cells displayed significant differences, making it extremely difficult to achieve a parametrization that accurately represents all cells simultaneously. It is likely that if all cells were new and ideal, the current parametrization would better reproduce the full discharge curve, including the steep decline at the end of the process observed in the datasheet.

Because a complete reparametrization of the functions was outside the scope of this work, this limitation highlights a useful direction for future research. A promising approach would be to implement online or adaptive parametrization methods, allowing the model to evolve together with the cells and continuously update its parameters to remain as close as possible to real behaviour. The results of this study also confirm the importance of testing both new and previously used cells, as this strategy exposed the two extremes of the performance spectrum. While the model tracked the discharge process reasonably well from the beginning until around 400 minutes (approximately 30% SOC), the divergence became more pronounced in the later stages, particularly for degraded cells. In fact, the performance of certain cells, such as Cell 3 and Cell 1, suggests that they may no longer be suitable for integration into the final battery pack, as their behaviour would negatively impact the overall reliability and performance of the system.

7.3 Diagnostic and Shutdown Control

The implementation of a robust state machine was a fundamental step in the development of the BMS, ensuring that all control decisions are handled reliably under any operating condition. This mechanism became the cornerstone of the system's logic, as it governs not only standard states like charging, discharging, and balancing, but also encompasses all diagnostic, warning, safety, and shutdown controls.

All the thresholds and conditions that define this state machine were established during the preparatory and planning phase of the project. As described in Subsection 3.2.1, a detailed analysis of both the FS regulations and the team's internal requirements was conducted to extract precise

software constraints. These constraints were then transformed into logical conditions that govern the system's behavior, providing a well-grounded foundation for implementation and validation.

This state machine became the central controller of the BMS firmware. All functionalities, including charging, discharging, balancing, diagnostics, and error detection, are governed through state transitions and controlled by conditional flags set by this logic. The only modules operating independently of this state logic are the acquisition system and the estimation and calculation system, which together form the continuous measurement and computation pipeline. While the acquisition system gathers real-time data, the estimation module processes this data to compute parameters such as SOC, V_{oc} , V_{out} , extracted charge, among others. Some of these outputs are then used to inform and trigger state transitions when specific conditions are met.

The state machine was divided into three logical diagrams for clarity and readability. This modular approach was necessary to visually separate the general operation logic from the more detailed diagnostic and fault-handling paths, which would have been extremely complex to represent in a single diagram.

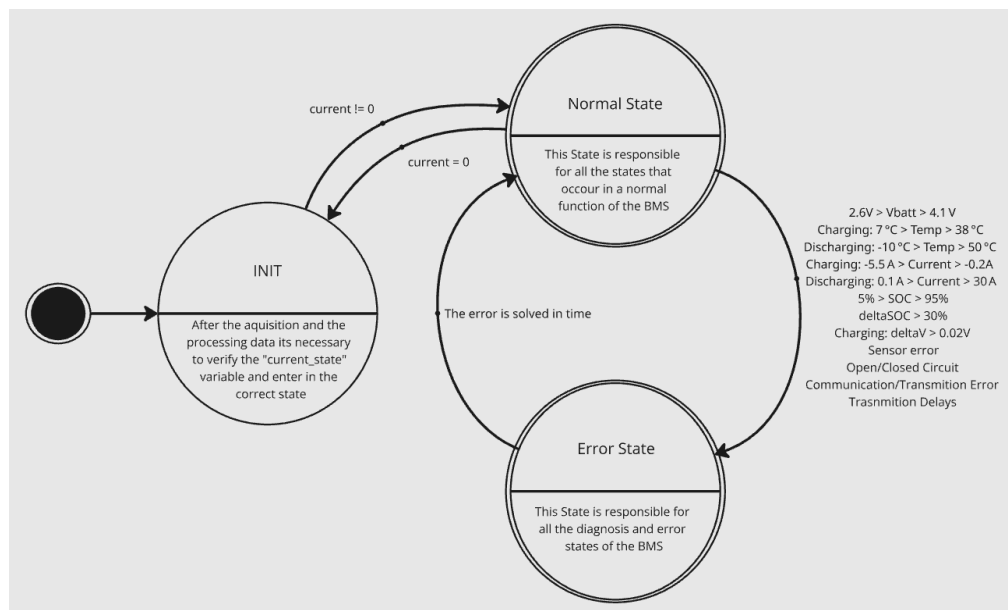


Figure 7.51: Overview of the general BMS state machine.

The general state machine, shown in Figure 7.51, outlines the main operational branches. After the INIT state performs data acquisition and initial validation, the system transitions to the Normal branch.

Figure 7.52 shows that the Normal path controls the battery through Charging, Discharging, and Balancing states. While the Charging and Discharging states are activated based on the current direction, the Balancing state is triggered based on the SOC deviation ($\Delta SOC = max_{SOC} - min_{SOC}$), and charging is temporarily suspended to prioritize balancing in necessary cases that will be explained in 7.5.

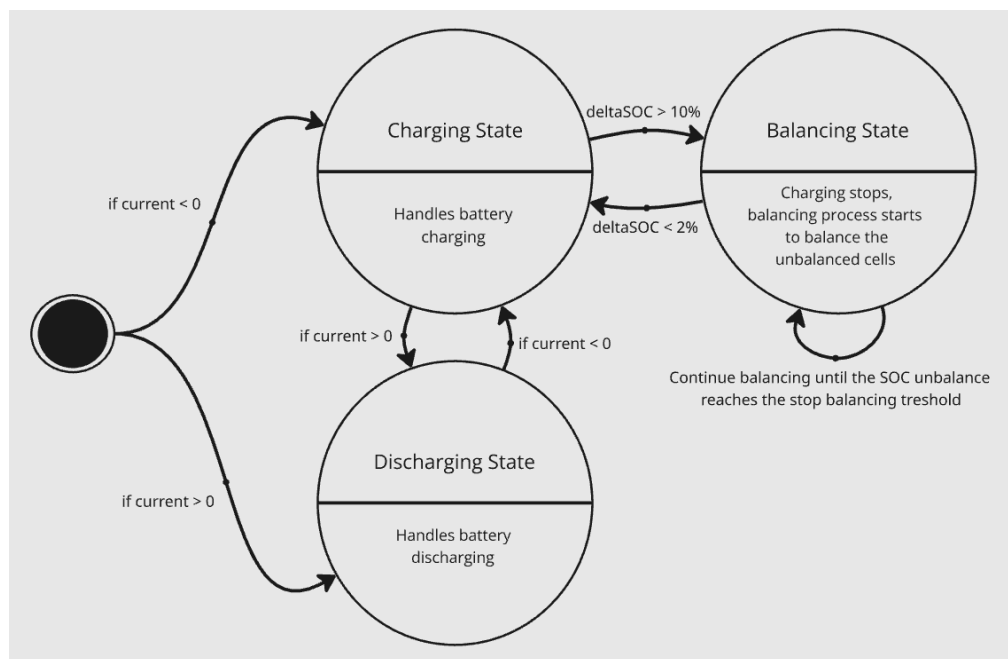


Figure 7.52: Normal operation states: Charging, Discharging, and Balancing.

In contrast, Figure 7.53 presents the logic governing system faults and anomalies. The system can transition directly into one of three dedicated error-handling states whenever any parameter exceeds its defined safety thresholds.

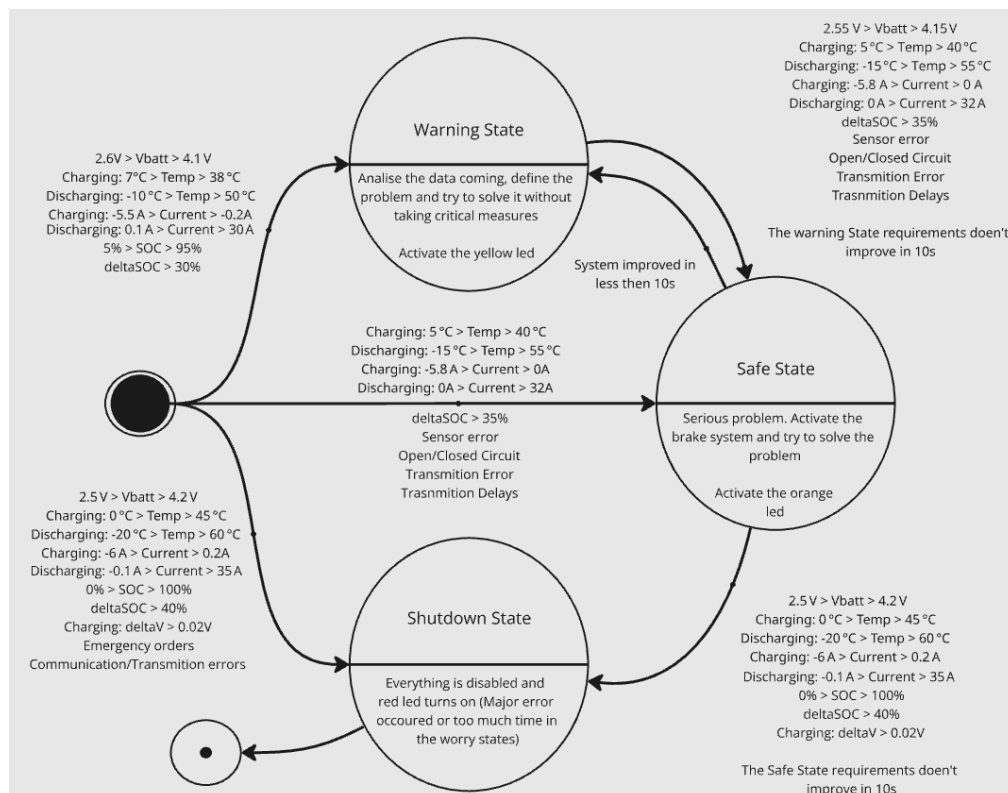


Figure 7.53: Error handling and shutdown logic.

In this prototype, all error and diagnostic information was monitored through the UART interface and visualized via the Tera Term terminal. This software-based approach allowed for easier debugging and a more detailed understanding of system behavior, such as identifying precisely which parameter triggered a warning or fault. Given the limited development time and the need to prioritize functional validation, the LED signaling system was not implemented at this stage. However, it is undoubtedly a valuable feature for future implementations. In a full-scale BMS integrated into a vehicle, especially under FS regulations, physical indicators such as LEDs become essential. They provide immediate feedback on the system's operational status, allowing the driver and technical team to quickly identify anomalies without needing access to the internal software interface.

Although some preliminary tests related to current control were conceptualized, a complete implementation was not possible due to hardware limitations. While time constraints played a role, the primary reason was that the available equipment could not be directly controlled by the BMS to modulate current flow. Implementing this functionality would have required the design and integration of additional hardware components capable of responding to BMS commands, a task that exceeded the scope and timeframe of this prototype. Nevertheless, the state machine was deliberately designed with current control integration in mind. Future developments could incorporate real-time current modulation, enabling the system to respond dynamically to safety conditions. For instance, during charging, if cell temperature approaches a critical limit, the BMS could autonomously reduce the charging current or interrupt the process to prevent further thermal rise. More advanced strategies like CC-CV charging or DCM control, as discussed in the State of the Art, could also be implemented through this framework.

In summary, the proposed state machine not only controls the operational flow and safety mechanisms of the BMS but also provides a flexible and expandable architecture. It lays a solid foundation for future enhancements, including fully autonomous current control strategies that adapt in real time to ensure both safety and efficiency to improve the performance of the car.

7.4 Internal Resistance Reestimation

As introduced in Subsection 2.4.2.3, the estimation of a cell's R_{int} provides a practical and reliable indicator of its SOH. Although the SOH monitoring was not a formal requirement in this project and it was out of the scope, R_{int} estimation was nonetheless explored due to its potential benefits in diagnostic accuracy and estimation improvements.

As previously discussed in 4.2.1, this project used a simple R_{int} model as part of the overall battery modeling framework, and it is possible to see that the V_{out} is influenced by the R_{int} value in the equation 4.15. Initially, fixed R_{int} values from the datasheet were used for all cells, but this assumption proved limiting for precise estimation. Therefore, a dynamic reestimation mechanism was introduced to improve model fidelity and make V_{out} predictions more accurate.

7.4.1 Practical Implementation Strategy

The reestimation of internal resistance was implemented selectively and conservatively to avoid instability and ensure reliable results. Instead of recalculating R_{int} for every cell, the logic was constrained to a single cell at each iteration, the most critical cell for each state. During charging, only the cell with the highest voltage (id_max) was considered, while during discharging, only the cell with the lowest voltage (id_min) was analyzed. This approach focused computational resources where the model errors were likely most impactful.

To ensure that R_{int} reestimation only occurred under stable current conditions, a FIFO-style buffer was implemented to track current readings over time. The system checked whether the variation in current remained below a defined threshold for multiple consecutive samples, rejecting any transient or noisy measurements. Only when the current was stable, and sufficiently large (above 0.2 A), the system considered recomputing R_{int} based on the observed discrepancy between the estimated V_{out} and the measured terminal voltage.

If the difference between the estimated V_{out} and the measured terminal voltage exceeded a defined threshold (50 mV), this indicated that the internal resistance value used by the model was no longer representative of the cell's actual behavior. In such cases, the R_{int} value for the selected cell was reestimated and updated, provided that the resulting value fell within a physically plausible range. This adaptive correction mechanism ensured that voltage prediction errors caused by outdated resistance assumptions did not accumulate, thereby preserving the stability and accuracy of the entire V_{out} estimation process.

7.4.2 Impact on V_{out} Estimation and Model Accuracy

Although internal resistance is often used solely for SOH tracking, in this implementation it served an additional role: correcting voltage estimation errors in the SOC model. Rather than estimating R_{int} from V_{out} with the purpose of analyzing the lifespan of a cell, the control system used real-time voltage and current measurements to reestimate R_{int} , which in turn improved V_{out} predictions and SOC stability. This feedback loop ensured that the electrical model could adapt over time, especially when the voltage prediction diverged from reality due to unmodeled resistance behavior.

After performing the same test previously described in Subsection 7.2.7, but now enabling R_{int} reestimation, the system showed a marked reduction in these errors. To better understand this, it was necessary to analyze the cell with the lower critical voltage, that is, the third cell, as expected.

Figure 7.54 shows the evolution of the reestimated internal resistance for Cell3 throughout the test. As expected, the value remained stable during most of the discharge process, before increasing sharply once the error between the estimated voltage V_{out} and the measured voltage V_{cell} exceeded 50 mV. This threshold was defined as the trigger for enabling a new reestimation of R_{int} .

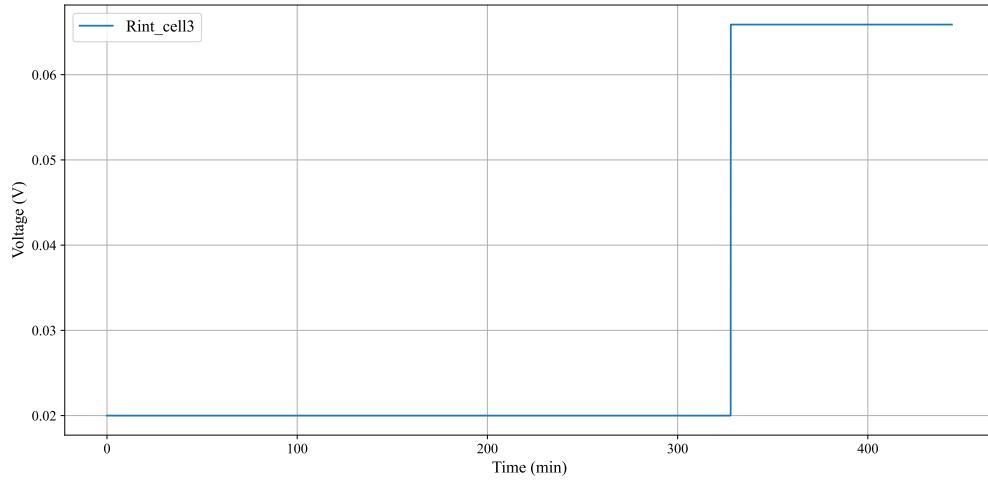


Figure 7.54: Evolution of R_{int} for Cell 3 during the discharge test.

The accuracy of the reestimated value is confirmed by analyzing the zoomed segment in Figure 7.55, which displays the corresponding current variations and voltage responses that justified the update. As mentioned in 2.4.2.3, the real internal resistance is calculated using the voltage drop across the cell caused by a measurable and sudden change in current, using the basic Ohm's Law relation $\Delta V = R \cdot \Delta I$.

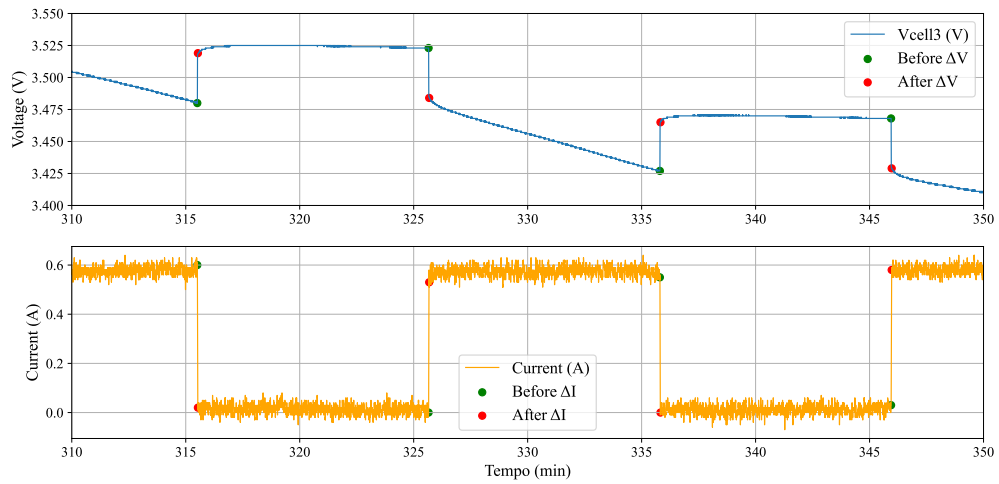


Figure 7.55: Zoomed view of voltage and current variation used to reestimate R_{int} for Cell 3.

It is important to note that obtaining an exact resistance value from this kind of plots is inherently imprecise due to sampling noise. For validation purposes, four manual estimations of R_{int} were made near the timestamp when the system executed its own update. These values are summarized in Table 7.6 and show a strong agreement with the automatic reestimation value of 0.0678Ω , reinforcing the robustness of the implemented approach.

Table 7.6: Manual estimation of R_{int} based on selected current steps.

Sample	Estimated R_{int} (Ω)
1	0.0672
2	0.0736
3	0.0691
4	0.0709
System estimation	0.0688

To avoid erroneous estimations due to noise or outliers, a saturation limit of 150 m Ω was imposed. Beyond this threshold, the system would disregard the estimation to prevent unphysical behavior. This explains why, toward the final part of the discharge curve, no new updates to R_{int} were recorded even though the voltage error continued increasing, the estimated resistance exceeded the upper limit and was discarded.

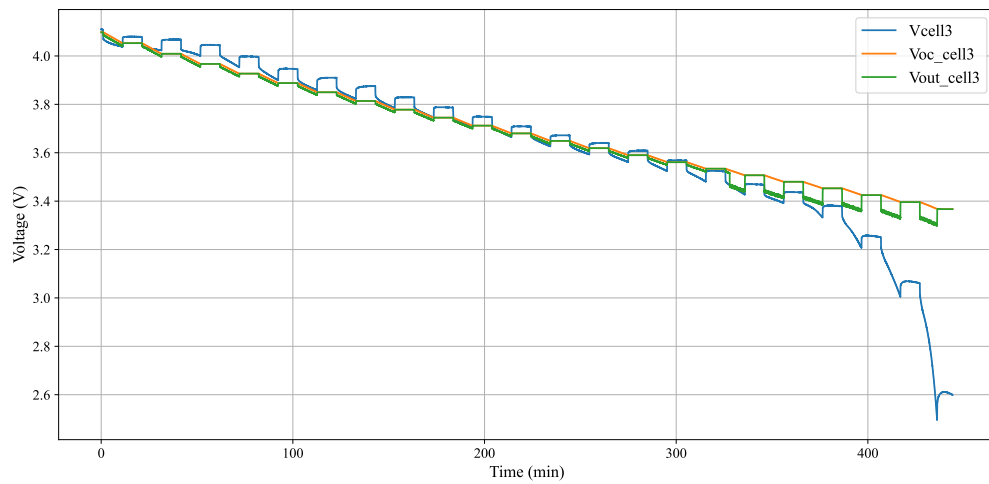
Figure 7.56: Comparison of V_{cell3} , V_{oc_cell3} , and V_{out_cell3} (model output with R_{int} reestimation).

Figure 7.56 shows the evolution of V_{cell3} , V_{oc_cell3} , and V_{out_cell3} over time during the discharge process. In the early stages, the model maintains close alignment between the estimated V_{out} and the real cell voltage. However, as the discharge progresses and the internal behavior of the cell begins to deviate from the initial assumptions, a visible divergence emerges. The reestimation mechanism is triggered precisely when the difference between V_{out} and V_{cell} exceeds 50 mV, resulting in a noticeable correction that restores alignment between the estimated and real values, improving accuracy in the final stages of discharge. In the future stages of this process certainly the code tried to repeat this process but certainly the R_{int} would be higher than the limit threshold condition to this operation.

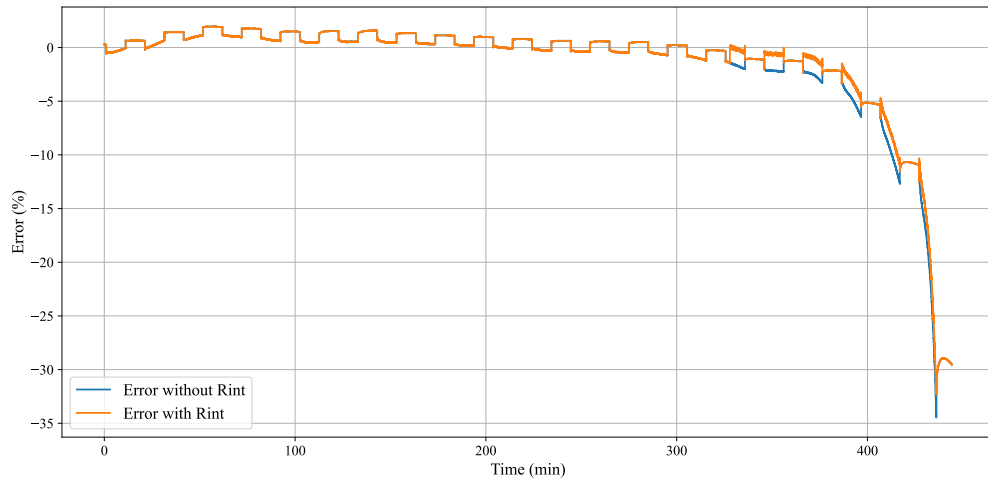


Figure 7.57: Reduction in V_{out} estimation error after enabling R_{int} reestimation.

Figure 7.57 illustrates this improvement by comparing V_{out} error before and after R_{int} correction, across all cells. Specifically, this plot presents the comparison between the V_{cell3} and the V_{out_cell} , expressed as a percentage error.

To better observe the behavior of R_{int} across different conditions and to fully validate the robustness of the reestimation strategy, additional tests were conducted using multiple discharge profiles: 0.56 A (1/5C), 2.8 A (1C), and 10 A. These scenarios were chosen to mimic both typical FS usage and datasheet reference conditions.

7.4.3 Experimental Results and Validation

The results above demonstrated that this dynamic adjustment significantly improved model alignment with measured values under moderate current loads, but to comprehensively assess the robustness and accuracy of the complete BMS prototype, a series of discharge tests were performed under varying current conditions. The next logical step involved subjecting the battery pack to more demanding scenarios. These experiments were designed to test how well the system responded to higher currents and how accurately it could estimate SOC and model behavior using its internal reestimation mechanisms.

Four different discharge profiles were selected:

- Continuous discharge at 2.8 A (1C)
- Pulsed discharge at 2.8 A (1C)
- Continuous discharge at 10 A
- Pulsed discharge at 10 A

Each test includes a full set of plots and analyses to evaluate voltage tracking, SOC estimation, charge integration, and model consistency under dynamic conditions.

7.4.3.1 Results for Four Different Discharge Profiles

Each of the following figures displays the same variable across the four tests. The plots are arranged in a 2x2 grid for a better comparison, with the top row representing 2.8 A and the bottom row 10 A. For each row, the left plot corresponds to continuous and the right to pulsed current.

Figure 7.58 shows the evolution of the six cell voltages throughout the four discharge scenarios. The critical behavior of cell 3 can be clearly observed, especially in the more aggressive discharges.

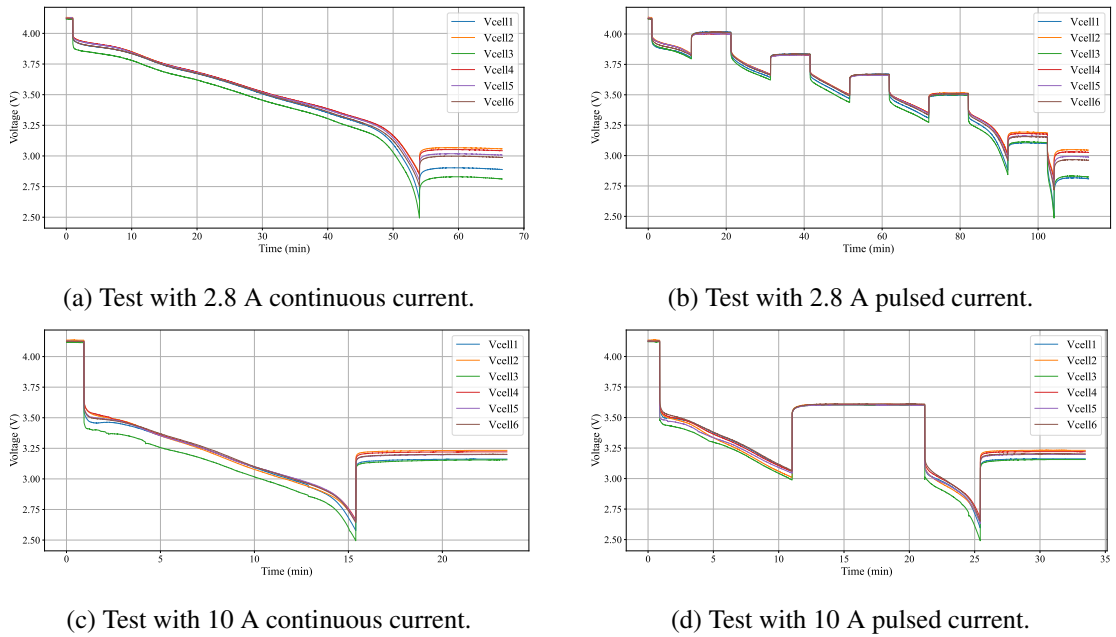


Figure 7.58: Measured voltages for all six cells during the four discharge profiles.

Figure 7.59 presents the SOC evolution over time for each test. The spread between cells increases under stress, validating the importance of accurate modeling and reestimation.

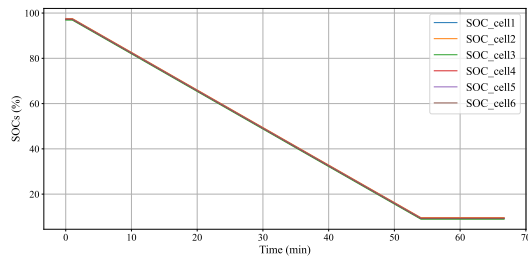
Figure 7.60 compares the real charge extracted with the estimated charge based on SOC for the cells with maximum and minimum SOC. This helps evaluate the accuracy of SOC integration over time.

Figure 7.61 shows the errors between estimated and real extracted charge, proving that this method has a very good accuracy.

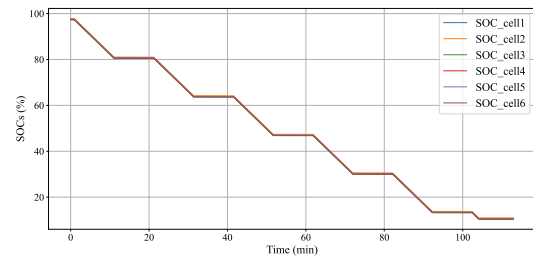
Figure 7.62 shows the comparison between the V_{cell*} , V_{oc_cell*} , and the V_{out_cell*} of a cell that didn't get the R_{int} reestimation, in this case Cell 1. This reflects how the system simulates a non-critical cell.

The model error for cell 1 is shown in Figure 7.63, giving a sense of prediction stability across the four tests.

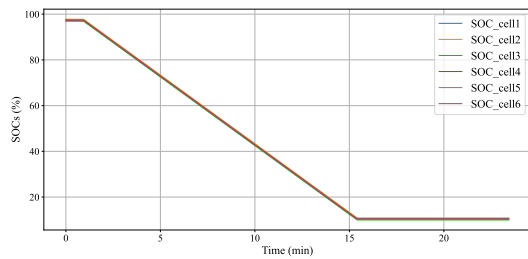
Figure 7.64 presents the same comparison for cell 3, which is consistently the most critical during discharges. The benefit of R_{int} reestimation is especially evident here.



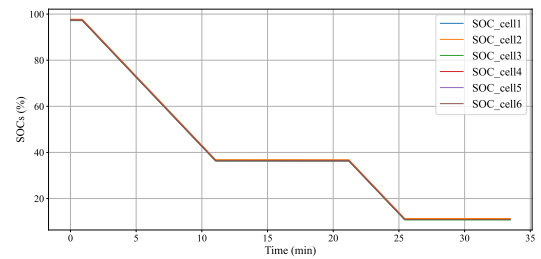
(a) Test with 2.8 A continuous current.



(b) Test with 2.8 A pulsed current.

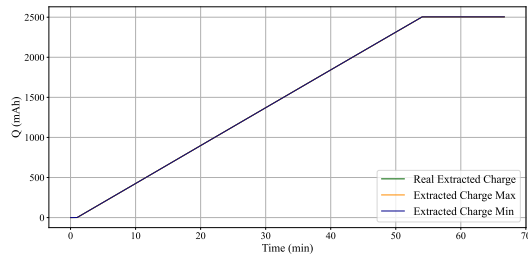


(c) Test with 10 A continuous current.

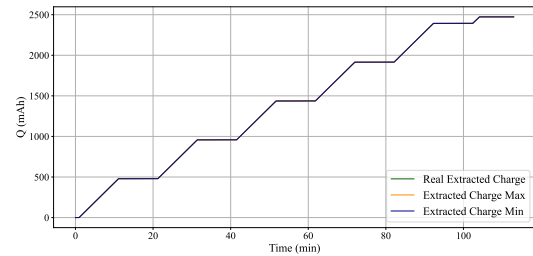


(d) Test with 10 A pulsed current.

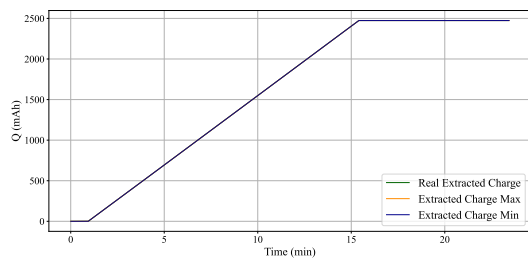
Figure 7.59: Estimated SOC's for all six cells during the four discharge profiles.



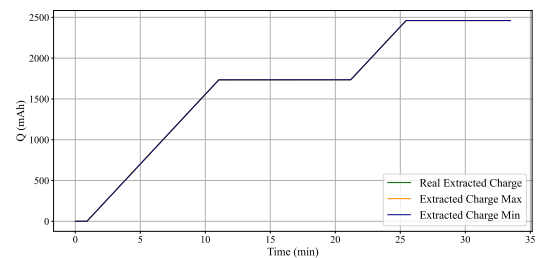
(a) Test with 2.8 A continuous current.



(b) Test with 2.8 A pulsed current.

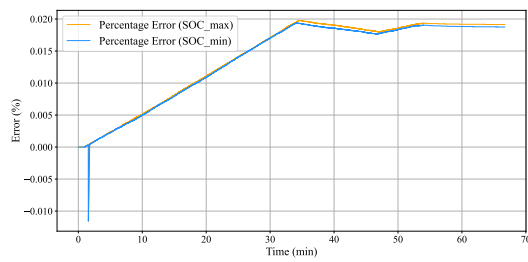


(c) Test with 10 A continuous current.

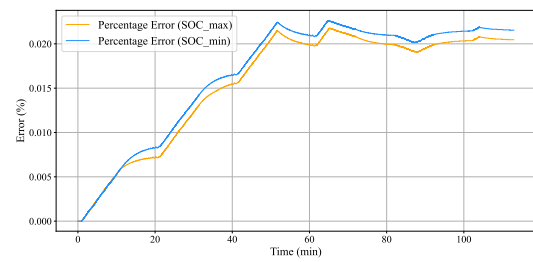


(d) Test with 10 A pulsed current.

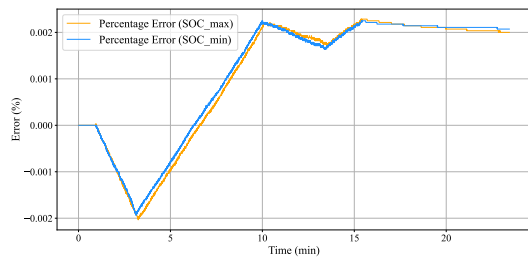
Figure 7.60: Comparison between the real extracted charge and critical cells extracted charge.



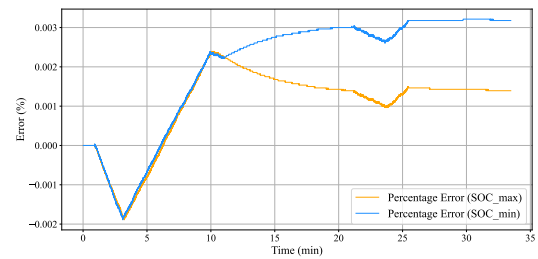
(a) Test with 2.8 A continuous current.



(b) Test with 2.8 A pulsed current.

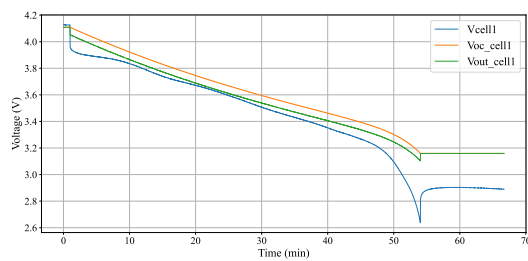


(c) Test with 10 A continuous current.

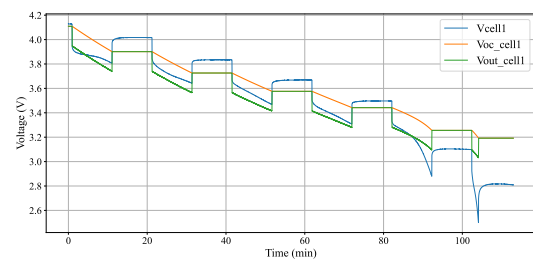


(d) Test with 10 A pulsed current.

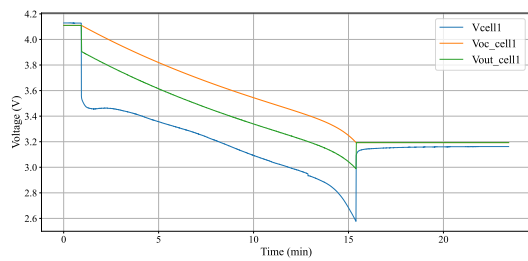
Figure 7.61: Errors between the real extracted charge and critical cells extracted charge estimations.



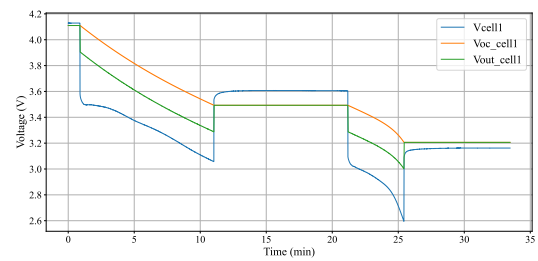
(a) Test with 2.8 A continuous current.



(b) Test with 2.8 A pulsed current.

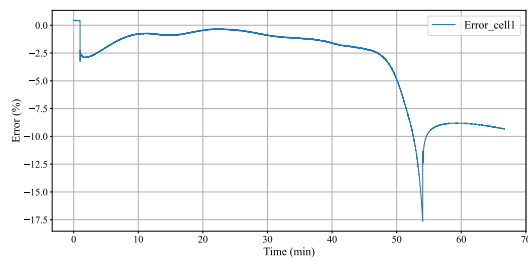


(c) Test with 10 A continuous current.

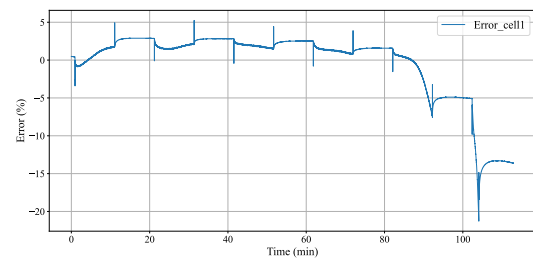


(d) Test with 10 A pulsed current.

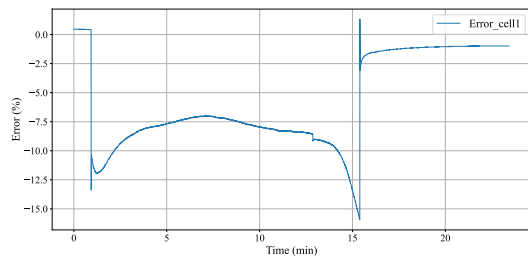
Figure 7.62: Comparison between the V_{cell1} , V_{oc_cell1} , and the V_{out_cell1} , that's a cell that didn't get the Rint reestimation.



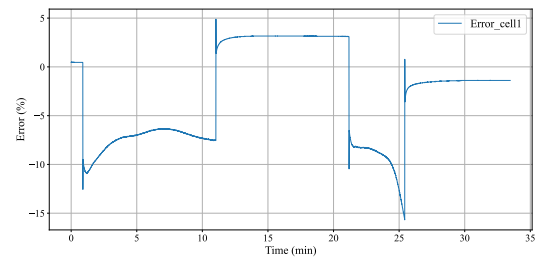
(a) Test with 2.8 A continuous current.



(b) Test with 2.8 A pulsed current.

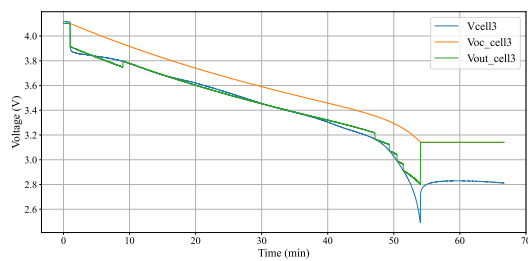


(c) Test with 10 A continuous current.

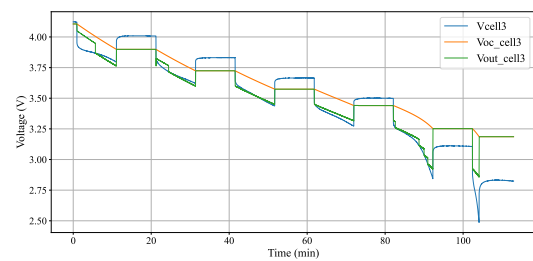


(d) Test with 10 A pulsed current.

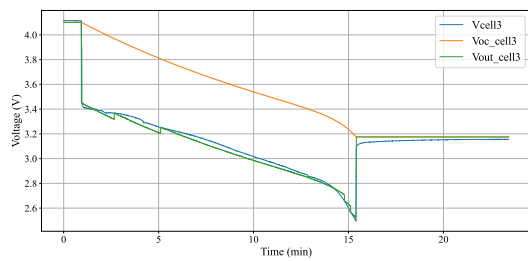
Figure 7.63: Error between the V_{cell1} , V_{oc_cell1} , and the V_{out_cell1} , that's a cell that didn't get the R_{int} reestimation.



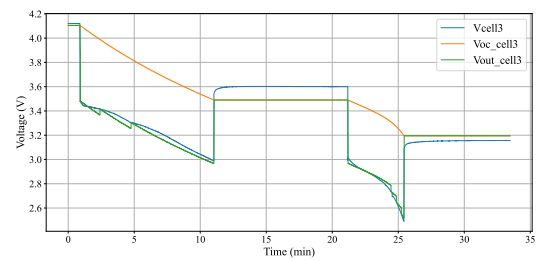
(a) Test with 2.8 A continuous current.



(b) Test with 2.8 A pulsed current.

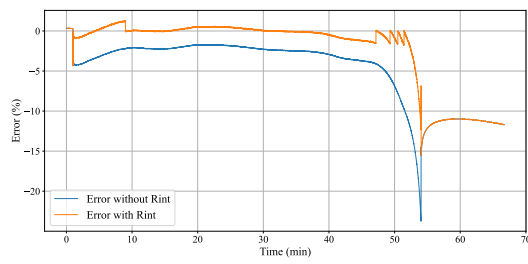


(c) Test with 10 A continuous current.

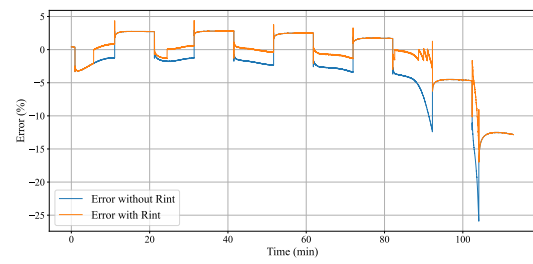


(d) Test with 10 A pulsed current.

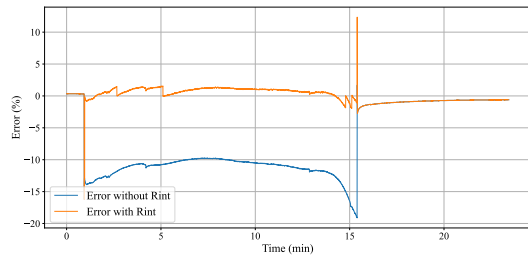
Figure 7.64: Comparison between the V_{cell3} , V_{oc_cell3} , and the V_{out_cell3} , that's a cell that get the R_{int} reestimation.



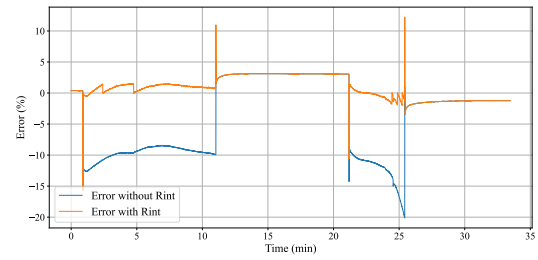
(a) Test with 2.8 A continuous current.



(b) Test with 2.8 A pulsed current.

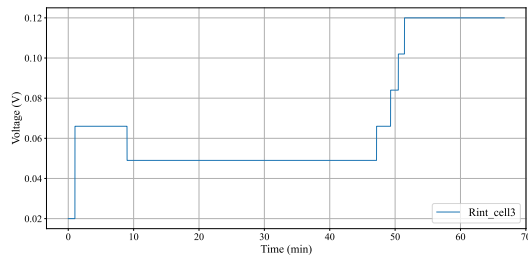


(c) Test with 10 A continuous current.

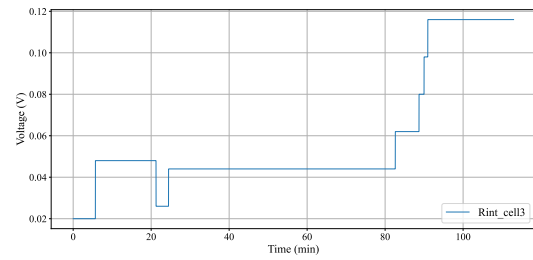


(d) Test with 10 A pulsed current.

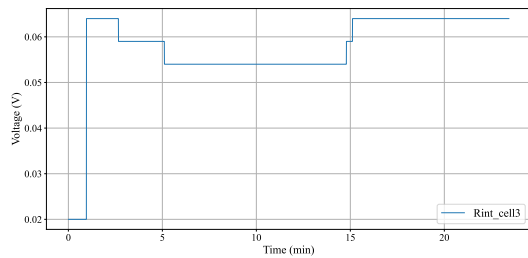
Figure 7.65: Comparison between V_{cell3} and V_{out_cell} , expressed as an error percentage, with and without the Rint reestimation.



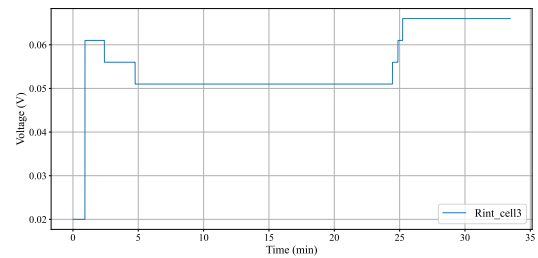
(a) Test with 2.8 A continuous current.



(b) Test with 2.8 A pulsed current.



(c) Test with 10 A continuous current.



(d) Test with 10 A pulsed current.

Figure 7.66: Rint reestimated for cell 3.

To highlight the improvement, Figure 7.65 compares the model error for cell 3 using the updated R_{int} and a static placeholder value, using the R_{int} values from datasheet. This plots present the comparison between the V_{cell3} and the V_{out_cell} , expressed as a percentage error.

Finally, Figure 7.66 shows the evolution of the internal resistance estimated for cell 3.

7.4.3.2 General Analysis and Discussion

The results obtained from the four discharge profiles confirm the reliability and robustness of the implemented SOC estimation strategy, even under highly dynamic and demanding conditions. As shown in Figure 7.59, the SOC trajectories remained consistent and coherent across all cells. Furthermore, when comparing the estimated extracted charge with the real extracted charge in Figure 7.60, the deviation remained minimal throughout the tests. This validates that the system was effective in maintaining accuracy even at higher discharge rates.

A key enhancement to this system was the implementation of internal resistance reestimation. As demonstrated in the voltage error plots and in Figure 7.57, the R_{int} correction is not only known to be an essential step toward reliable SOH estimation but also proved to be a good contribution to improving the V_{out} estimation. Notably, Cell 3, identified across all tests as the most degraded or “critical cell”, benefited significantly from the R_{int} update, showing a clear reduction in voltage estimation error when compared to other cells that did not receive reestimation, especially when, as previously demonstrated, initial model parametrization may be imprecise. A properly tuned R_{int} reestimation mechanism greatly enhances model precision, which is fundamental for future SOH tracking and degradation monitoring.

Throughout all discharge scenarios, Cell 3 consistently exhibited the most critical behavior in terms of voltage drop and SOC depletion. This consistent pattern highlights the necessity of SOH-aware control strategies in BMS design. Without active SOH management, the system risks becoming limited by a single weak cell, reducing the overall pack performance and lifespan. Implementing early detection and replacement mechanisms for such cells becomes crucial, especially in applications demanding high reliability.

Additionally, one relevant outcome from the experiments is the clear indication that the internal resistance of the cells is not constant during discharge. Further experiments, which were not considered in this prototype, are needed to understand whether internal resistance varies significantly throughout a single discharge cycle or remains relatively constant.

Finally, for even greater battery modeling accuracy, the current approach, based on a simplified R_{int} electrical model, could be extended to include more advanced representations such as Thevenin models. These models can capture transient behaviors more effectively, making them ideal candidates for future development and integration into the hybrid estimation framework.

7.5 Balancing Test

As discussed in Section 2.7.3, passive balancing relies on dissipating excess energy from higher-SOC cells through internal resistors, converting it into heat. While this technique is relatively simple to implement and does not require complex hardware, it inherently wastes energy and can contribute to thermal stress within the battery pack and is also very time consuming. For this reason, it is not a desirable operation to be used continuously or aggressively. Instead, the strategy adopted in this work prioritizes minimal and efficient use of balancing, activating it only when strictly necessary and under optimal conditions.

Furthermore, passive balancing effectiveness is influenced by the voltage of the cell to be discharged. The balancing current can be estimated by the following equation:

$$I_{balance} = \frac{V_{cell}}{R_{balance} + R_{int}} \quad (7.4)$$

This means that cells with higher voltage will dissipate more current when balancing is active. Therefore, cells that begin with a significantly higher SOC benefit from faster correction during balancing. This behavior was experimentally confirmed in [42], and reinforces the importance of selectively targeting cells the latter as possible in the charging process.

It is also important to emphasize that balancing in this system is performed exclusively during the charging process. This choice is based on the fundamental principle that, during discharge, the most critical constraint is the voltage of the weakest (i.e., lowest-voltage) cell, which defines the system's lower operating limit. Discharging a pack while actively removing charge from cells would be counterproductive, as it would reduce the available energy without any gain in system longevity or performance. In contrast, during charging, the upper limit is determined by the most charged cell. To extract the maximum possible energy from the battery during the next discharge cycle, it is essential that all cells reach their full charge capacity. Balancing at this stage ensures that high-voltage cells are slightly discharged to allow lower-voltage cells to catch up, resulting in a uniformly charged pack. This behavior aligns with the performance-oriented goals of the system and the passive balancing philosophy described in Section 2.7.3.

7.5.1 Balancing Capability in the Slave Module

The balancing strategy implemented in this BMS prototype relies entirely on the hardware capabilities of the BQ79616EVM-021 slave board. This evaluation module, integrates all necessary circuitry for passive cell balancing, including switching MOSFETs and internal dissipation resistors. This feature eliminates the need for external balancing hardware and significantly simplifies the system design.

Each cell channel on the BQ79616 device is connected to a dedicated CBFET and a 100 Ω internal resistor. When a particular channel's balancing is enabled through a command sent from the master to the slave, the corresponding switch closes, allowing a current path from the positive terminal of the cell to ground through the resistor. The typical balancing current achieved through

this mechanism is approximately 200 mA. This current is sufficient to gradually discharge higher-voltage cells during specific operating moments and conditions, which will be detailed throughout this section.

Figure 6.3 illustrates the internal balancing structure provided by the BQ79616, including the switch–resistor arrangement embedded in the IC. All control signals required to activate or deactivate balancing are sent via UART from the master microcontroller to the slave. The BMS determines which cells should be balanced based on a comparison of their SOC values, and then sends the corresponding bitmask to the correct register, so that the balancing begins.

7.5.2 Balancing Configuration Steps

There are four key steps required to configure the BQ79616EVM-021 for manual cell balancing. As previously mentioned, each step involves sending a specific command from the master microcontroller to the slave via UART. Once all the prerequisites are satisfied, the slave will then activate the internal balancing MOSFETs and begin discharging the selected cell(s).

The four configuration steps are as follows [59]:

1. Define the number of active cells to be monitored and balanced
2. Enable the AUX ADC so it is possible to measure the current in every individual cell
3. Determine which channel to enable for cell balancing
4. Select the cell balancing control methods, auto or manual balancing control with or without adjacent cell support

7.5.2.1 Step 1: Definition of Active Cells

This configuration step was previously described in Section 5.4.3. It consists of sending the correct ACTIVE_CELL register during system initialization to specify how many cells are present and should be actively monitored. This setting not only enables voltage acquisition but also determines which CB (cell balancing) channels can be activated later. Without defining active cells, balancing will be disabled for all channels, and channels not selected are bypassed during cell balancing.

7.5.2.2 Step 2: Continuous AUX ADC Operation

Balancing requires accurate temperature monitoring of the MOSFETs, the internal dissipation resistors and PCB, which is done via the AUX ADC subsystem of the BQ79616. To activate this function, the master sends a command to write the value 0x02 to register ADC_CTRL3 (address 0x030F), which configures the AUX ADC to run in continuous mode, as presented in figure This ensures that GPIO-based temperature readings are updated in the background at all times and are valid for balancing safety checks.

Address	0x030F							
RW	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	RSVD	AUX_GPIO_SEL[3:0]				AUX_GO	AUX_MODE[1:0]	
Reset	0	0	0	0	0	0	0	0
RSVD = Reserved								
AUX_GPIO_SEL[3:0] = Selects which GPIO channel(s) will be multiplexed through the AUX ADC to use for temperature measurement diagnostic. If this selection is not set to 0x00, the AUX ADC will lock onto a single GPIO channel and the measurement result is output to the <i>AUX_GPIO_HI/LO</i> registers. 0x00 = AUX ADC cycles through all GPIO channel(s) that are configured as ADC only or ADC and OTUT. 0x01 = Lock to GPIO1 0x02 = Lock to GPIO2 : 0x08 = Lock to GPIO8 All other codes are RSVD. NOTE: If GPIO is not configured for ADC measurement or RSVD codes are selected, device will not perform AUX ADC conversion on the GPIO slot and the <i>AUX_GPIO_HI/LO</i> registers will be kept in reset value.								
AUX_GO = Starts AUX ADC conversion. When this bit is written to 1, all AUX ADC inputs are sampled. Once the AUX ADC is started, any change to the AUX ADC control setting has no effect until this bit is written to 1 again. This bit is cleared to 0 in read. 0 = Ready. Writing 0 has no effect. 1 = Start AUX ADC								
AUX_MODE[1:0] = Sets the Main ADC run mode. In continuous run, if user would like to stop ADC, user must read all the ADC conversion results, then stop it. ADC results are not valid before ADC is reenabled next time. 00 = AUX ADC not running 01 = Single run. Run the AUX ADC round robin once and then stop. 10 = Continuous run. Continually run the AUX ADC round robin until host sends command to stop. 11 = 8-round-robin run to measure all eight GPIOs once.								

Figure 7.67: ADC_CTRL3 register configuration, from [50].

The complete command sent from the master to enable continuous AUX ADC operation is structured as follows:

0xD0 0x03 0x0F 0x02 CRC1 CRC2

7.5.2.3 Step 3: Determining the Channels for Cell Balancing

The BQ79616 provides an individual timer for each of its 16 possible cell balancing channels. These timers define how long a given cell's CBFET should remain activated during a balancing cycle. The timers are controlled through the CB_CELL*_CTRL registers, where * = 1 to 16 corresponds to balancing channels 1 to 16. Setting a non-zero value in any of these registers enables that channel for balancing, though the balancing operation itself will only begin once the master sends the comand to star balancing.

Each timer value is 5 bits wide (bits [4:0]) and corresponds to a specific duration, as described in the datasheet. A timer value of 0x00 disables balancing for that channel.

Address	0x0318 to 0x0327							
RW	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	RSVD	TIME[4:0]						
Reset	0	0	0	0	0	0	0	0
RSVD = Reserved								
TIME[4:0] = Sets the timer for cell* balancing. The selection is sampled whenever [BAL_GO] = 1 is set by the host MCU. 0x00 = 0 s = stop balancing 0x01 = 10 s 0x02 = 30 s 0x03 = 60 s 0x04 = 300 s 0x05 to 0x10 = range from 10 min to 120 min in 10-min steps 0x11 to 0x1F = range from 150 min to 540 min in 30-min steps and 600 min								

Figure 7.68: Cell balancing time configuration register (CB_CELL_CTRL), from [50].

To configure the balancing timers for all 16 possible channels, the master sends two sequential write commands following the standard BQ79616 frame structure. Each command targets a contiguous block of 8 CB_CELL*_CTRL registers, writing the desired balancing time for each individual channel. The first command configures channels CB9 through CB16, and the second

configures CB1 through CB8. It is important to note that the data bytes must be sent in descending order, starting from the highest-numbered CB channel down to the lowest, as per the device's register mapping, illustrated in Figure 7.68.

The command frame used to configure CB channels 16 down to 9 is structured as follows:

```
0xD7  0x03  0x18  CB16_TIME  CB15_TIME  CB14_TIME  CB13_TIME
      CB12_TIME  CB11_TIME  CB10_TIME  CB9_TIME  CRC1  CRC2
```

Similarly, the command frame to configure CB channels 8 down to 1 is:

```
0xD7  0x03  0x20  CB8_TIME  CB7_TIME  CB6_TIME  CB5_TIME
      CB4_TIME  CB3_TIME  CB2_TIME  CB1_TIME  CRC1  CRC2
```

Unlike the previous two configuration steps, which are executed once during system initialization, the commands used to configure the CB_CELL_CTRL registers are issued dynamically during runtime. These commands are not static, they are updated and transmitted by the master microcontroller in the middle of program execution, depending on the current balancing requirements. This allows the system to adapt the balancing duration and target cells at each cycle based on the latest SOC readings. Consequently, this step serves as the final preparatory action before triggering the actual balancing process. The real-time decision logic that governs when and how these commands are sent will be detailed in the following subsection, which explains the balancing control strategy implemented in the BMS firmware.

7.5.2.4 Step 4: Enabling Manual Balancing for Non-Adjacent Cells

By default, the BQ79616 supports both automatic and manual cell balancing modes. In automatic mode, the master microcontroller specifies which cells should be balanced and for how long, and the slave executes the balancing sequence autonomously. However, in this configuration, the master loses real-time control over individual balancing channels during execution. All selected channels balance for the full timer duration, and it is not possible to interrupt or selectively stop individual cell balancing midway.

For this project, a manual balancing strategy was adopted to maintain full control over which cells are balanced and when. This approach enables the system to dynamically choose which cell to discharge in each iteration, based on real-time SOC evaluation, and to stop balancing when necessary.

To implement manual balancing, there are two possible options: with or without adjacent channel support. The next figures illustrate the difference. The left image shows a scenario where two adjacent CB FETs are active simultaneously, resulting in the balancing current being shared between them. The right image illustrates the selected configuration for this project, a non-adjacent scheme in which only one CB is active at a time.

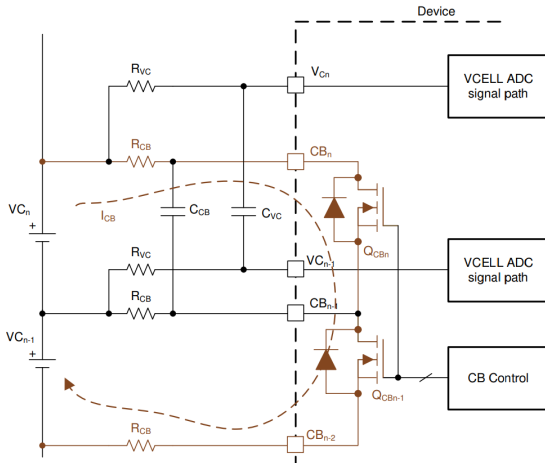


Figure 7.69: Cell balancing with 2 consecutive CBFET's on, from [50].

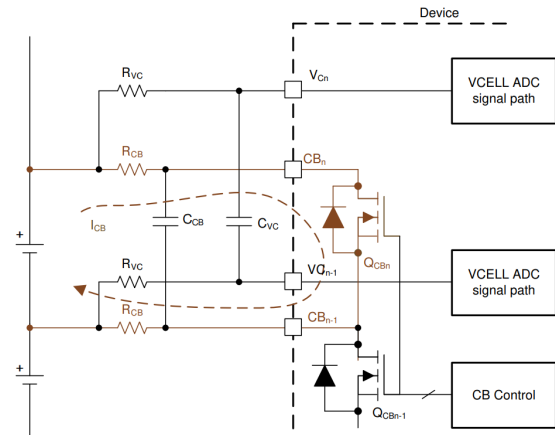


Figure 7.70: Cell balancing with one internal CBFET, from [50].

In the adjacent balancing mode, the device allows two neighboring CB FETs to be activated simultaneously. While this could potentially speed up the overall balancing process, it introduces several challenges from a control perspective. First, the balancing current is shared between the two adjacent cells, meaning each cell receives only a portion of the available discharge current, which reduces the effectiveness of balancing per cell. Second, the firmware would need to constantly check whether adjacent cells simultaneously require balancing and ensure that enabling them together does not violate current distribution constraints. This increases complexity and makes it more difficult to track the effect of balancing on each individual cell, especially when analyzing the system's behavior in real time.

For these reasons, the non-adjacent mode was selected. In this configuration, the BQ79616 ensures that only one CB FETs can be turned on at the time. This ensures that the full balancing current, limited to approximately 200 mA by the internal circuitry of the slave board, is directed to a single cell, improving the discharge rate and making voltage drops easier to observe. It also simplifies the balancing logic on the master side, allowing greater control and visibility during tests and debugging.

This behavior is configured via the DEV_CONF register. To prevent adjacent CB FETs from being activated simultaneously, the bit DEV_CONF[NO_ADJ_CB] must be set to 1. Since this bit is already set in the default reset configuration (0x54), no additional customization was required.

Address	0x0002							
NVM	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	RSVD	NO_ADJ_CB	MULTIDROP_EN	FCOMM_EN	TWO_STOP_EN	NFAULT_EN	FTONE_EN	HB_EN
Reset	0	1	0	1	0	1	0	0
RSVD = Reserved								
NO_ADJ_CB = Indicates the device will not allow an adjacent CB FET to be turned on in manual CB control. If MCU has enabled an adjacent CB FET, device will not start CB even if host sends [BAL_GO] = 1. 0 = Device will allow two adjacent CB FETs to be enabled. 1 = Device will not allow adjacent CB FET to be enabled.								
MULTIDROP_EN = Defines if the device is used in a multidrop or daisy chain configuration. The TX and RX for COML and COMH will be enabled or disabled based on the configuration. 0 = Daisy chain of base device 1 = Multidrop								
FCOMM_EN = Enables the fault state detection through communication in ACTIVE mode. 0 = Disable 1 = Enable								
TWO_STOP_EN = Enables two stop bits for the UART in case of severe oscillator error in the host and device. 0 = One STOP bit 1 = Two STOP bits								
NFAULT_EN = Enables the NFAULT function. 0 = NFAULT always pulled up 1 = NFAULT pulled low to indicate an unmasked fault is detected. Note: This bit setting does not affect the FAULT_SUMMARY register.								
FTONE_EN = Enables FAULT TONE transmitter when device is in SLEEP mode. 0 = Disable 1 = Enable								
HB_EN = Enables HEARTBEAT transmitter when device is in SLEEP mode. 0 = Disable 1 = Enable								

Figure 7.71: Device configuration register (DEV_CONF), from [50].

The command sent from the master to configure the device explicitly in this mode is:

0xD0 0x00 0x02 0x54 CRC1 CRC2

Once all configuration steps are completed and valid CB timers have been loaded into the CB_CELL*_CTRL registers, the system is ready to start the balancing operation.

Address	0x032F							
RW	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	RSVD	CB_PAUSE	FLTSTOP_EN	OTCB_EN	BAL_ACT[1:0]		BAL_GO	AUTO_BAL
Reset	0	0	0	0	0	0	0	0
RSVD = Reserved								
CB_PAUSE = Pauses cell balancing on all cells to allow diagnostics to run. 0 = Normal cell balancing operation 1 = Pause all cell balancing								
FLTSTOP_EN = Stops cell or module balancing if unmasked fault occurs. The selection is sampled whenever [BAL_GO] = 1 is set by the host MCU. 0 = Balancing is continuous regardless of fault condition (excluding thermal shutdown) 1 = All CB balancing stops when any unmasked fault occurs								
OTCB_EN = Enables the OTCB detection during cell balancing. The selection is sampled whenever [BAL_GO] = 1 is set by the host MCU. 0 = Disable OTCB detection 1 = Enable OTCB detection								
BAL_ACT[1:0] = Controls the device action when the MB and CB are completed. These bits are samples whenever [BAL_GO] = 1 is set by the host MCU. The action is valid. 00 = No action 01 = Enters SLEEP 10 = Enters SHUTDOWN 11 = Reserved								
BAL_GO = Starts cell or module balancing. When written to 1, all balancing configuration registers are sampled. Any change to the configuration registers has no effect until this bit is written to 1 again. The bit is self-clearing. 0 = Ready 1 = Start balancing								
AUTO_BAL = Selects between auto or manual cell balance control. The selection is sampled whenever [BAL_GO] = 1 is set by the host MCU. 0 = Manual cell balancing 1 = Auto cell balancing								

Figure 7.72: Cell balancing control register (BAL_CTRL2), from [50].

In manual mode (BAL_CTRL2[AUTO_BAL] = 0), balancing begins when the master sets BAL_CTRL2[BAL_GO] = 1 in the BAL_CTRL2 register. The internal logic of the BQ79616 then enables the CB FETs for the channels with non-zero timer values. The activation is immediate and strictly based on the master's instructions, with no internal scheduling or round-robin logic applied.

The command used to trigger balancing is:

0xD0 0x03 0x2F 0x02 CRC1 CRC2

It is critical that this command is sent only after confirming a valid configuration. Otherwise, balancing will not start and the slave will set the BAL_STAT[INVALID_CBCONF] flag. This flag is raised under three conditions:

- More than eight CB channels are configured with non-zero timers.
- Adjacent channels are selected for balancing while DEV_CONF[NO_ADJ_CB] = 1.
- More than two adjacent CB FETs are active when DEV_CONF[NO_ADJ_CB] = 0.

To monitor whether balancing has successfully started, the master can read the BAL_STAT register, because this provides real-time information on CB status.

Address	0x052B							
Read Only	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	INVALID_CBCONF	OT_PAUSE_DET	CB_INPAUSE	MB_RUN	CB_RUN	ABORTFLT	MB_DONE	CB_DONE
Reset	0	0	0	0	0	0	0	0
	R	R	R	R	R	R	R	R
INVALID_CBCONF = Indicates CB is unable to start (after /BAL_GO = 1) due to improper CB control settings. Incorrect settings include: • More than eight cells are enabled for CB. • More than two adjacent cells are enabled for CB if DEV_CONF[NO_ADJ_CB] = 0. • Any adjacent cells are enabled for CB if DEV_CONF[NO_ADJ_CB] = 1. This bit is updated every time /BAL_GO = 1. 0 = Valid CB setting 1 = Invalid CB setting								
OT_PAUSE_DET = Indicates the OTCB is detected if /OTCB_EN = 1. The bit is also set if CB TWARN is detected, which will also pause CB. Valid only after /BAL_GO = 1. 0 = No OTCB or CB TWARN is detected 1 = Any NTC thermistor measurement is greater than /OTCB_THR[3:0] setting, or die (CBFET) temperature is greater than CB TWARN								
CB_INPAUSE = Indicates the cell balancing pause status. 0 = CB is running or not started 1 = Paused (can be caused by OTCB detection, or host sets /CB_PAUSE = 1)								
MB_RUN = Indicates module balancing, controlled by the device, is running. Only valid if /MB_TIMER_CTRL is not 0x00 and after /BAL_GO = 1. Does not indicate the cell balancing status. 0 = Completed or not started 1 = Module balancing, controlled by the device, is running								
CB_RUN = Indicates cell balancing is running. Only valid after /BAL_GO = 1. Does not indicate the module balancing status. This bit remains as 1 even if CB is in pause state. 0 = Completed or not started 1 = At least 1 cell is in active cell balancing								
ABORTFLT = Indicates cell balancing is aborted due to detection of unmasked fault. Cleared when BAL_CTRL1/BAL_GO = 1. CB abort does not trigger if CB is in pause (/CB_INPAUSE = 1) even if an unmasked fault is detected. The abort at fault function will resume if CB is no longer in pause state. 0 = Not aborted or cell balancing not running 1 = Aborted								
MB_DONE = Indicates module balancing is completed. Cleared when BAL_CTRL1/BAL_GO = 1. 0 = Module balancing is still running or has not started 1 = Module balancing completed								
CB_DONE = Indicates all cell balancing is completed. Cleared when BAL_CTRL1/BAL_GO = 1. 0 = Cell balancing is still running or has not started 1 = All cell balancing is completed								

Figure 7.73: Cell balancing status register (BAL_STAT), from [50].

The read command used to access this information is:

0xC0 0x05 0x2B 0x00 CRC1 CRC2

This register is polled periodically by the master firmware to track the progress of balancing and take corrective action if necessary.

7.5.3 Balancing Control Strategy

The balancing control logic was fully implemented in the STM32 firmware and is designed to act dynamically and selectively, targeting only one cell at a time. This approach minimizes energy waste and maximizes control and visibility throughout the balancing process, as mentioned before. The decision to activate balancing is governed by two core system conditions: `flag_balancing` must be set to 1, indicating that balancing mode is active, and `flag_relax` must also be 1, confirming that the battery is in a relaxed state, meaning there has been no charging or discharging activity for a sufficient period. Initially, the `flag_balancing` condition required the maximum SOC in the pack to exceed 90% in order to avoid premature or unnecessary balancing. However, this threshold was later removed during test execution to simplify the process of manually disbalancing cells and observing the time required for their correction.

Once both conditions are met, the system scans all cells and selects one that meets the criterion, defined as having at least a 10% higher SOC than the cell with the minimum SOC.

$$\text{deltaSOC} = \text{SOC}_{\text{cell}^*} - \text{SOC}_{\text{min}} \quad (7.5)$$

If there is no cell that satisfies this condition the balancing process is stopped, but when such cell is found, the selected cell index is stored in a variable, and the following actions are triggered:

- The system is forced to continuously estimate SOC based on `Voc` during the balancing period.
- The timer for the selected CB channel is set to 0x02, corresponding to 30 seconds of active balancing.
- The appropriate `CB_CELL*_CTRL` commands are sent to the BQ79616 to update the internal timer for that cell, and pass everything else to 0x00.
- The bit `BAL_CTRL2[BAL_GO]` is set to 0x02, triggering the start of the balancing operation, in manual mode.

After activation, the firmware enters a monitoring loop. Every 35 seconds (30 seconds of balancing + 5 seconds of wait), the controller checks the delta SOC between the currently balancing cell and the minimum SOC cell. If the gap has dropped below 5%, the balancing is considered successful:

- All CB timers are cleared.
- `BAL_CTRL2[BAL_GO]` is set to 0x00 to stop the balancing process.
- All the variables, and internal timers are reset.

If the battery still requires further correction for other cells, a new balancing cycle is initiated with the same logic and updated timers. This approach ensures that no unnecessary balancing cycles are run, and each activation is based on a valid SOC gap evaluation.

Additionally, the system constantly reads the BAL_STAT register to confirm the internal state of the slave. This robust, logic-based approach combines safety, adaptability, and energy efficiency, ensuring that balancing only occurs when needed and under optimal conditions.

7.5.4 Test Results and Analysis

As mentioned previously, the condition requiring the maximum SOC to exceed 90% for balancing activation was intentionally removed during this tests. The goal of the tests was to manually create a moderate imbalance between the cells and then observe whether the implemented balancing logic would autonomously trigger balancing and progressively restore uniformity across the pack. This approach enabled the system to be evaluated under repeatable and observable conditions, with clear transitions in SOC and voltage confirming the effectiveness of the control strategy.

7.5.4.1 Test 1: Manual Overvoltage of a Single Cell

In this first experiment, the objective was to validate the correct triggering and execution of the balancing logic under a controlled imbalance scenario. To do this, the voltage of Cell2 was increased by manually injecting current through an external CC power source, forcing its SOC to rise above the rest of the pack, particularly when compared to Cell4, which was selected as the reference minimum SOC cell. The goal was to confirm that the balancing algorithm would autonomously detect this imbalance, activate the internal balancing CBFET for Cell2, and maintain the process until the difference in SOC between the two cells dropped below 5%, at which point balancing would be stopped.

To simplify observation, only Cell2 (the overcharged cell) and Cell4 (the lowest SOC cell) were monitored during this test, as the purpose was to prove the logic in a single-cell imbalance scenario. Figure 7.74 shows the evolution of their voltages over the course of the test. It is clear that Cell2 experienced a continuous voltage drop over time due to active balancing, confirming that it was being discharged by the balancing circuit, while Cell4's gradual decrease was expected since, as previously mentioned, the battery is the power source for the slave.

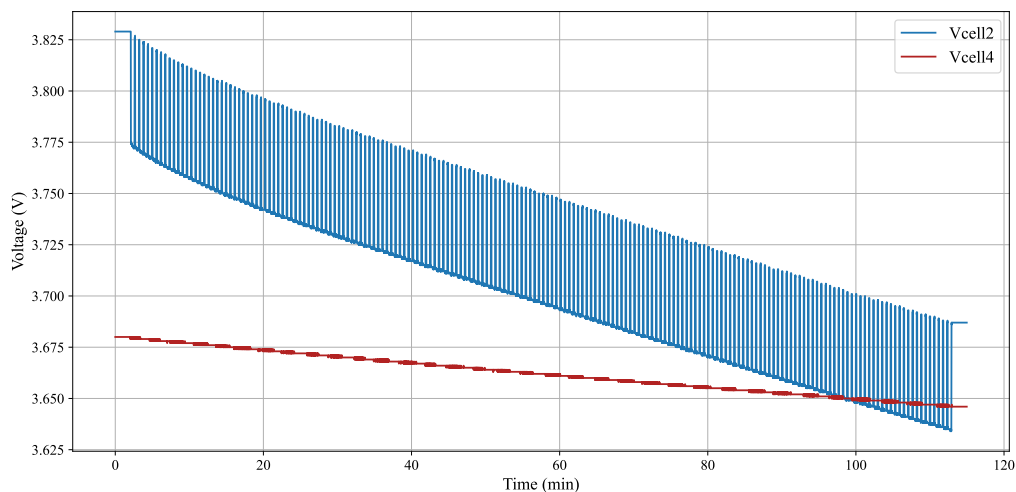


Figure 7.74: Voltage evolution of Cell2 and Cell4 during balancing.

To better illustrate the behavior of the balancing process, Figure 7.75 presents a zoomed-in view of the voltage pattern. The periodic voltage drops observed on Cell2 confirm the operation of the balancing logic in the expected 30-second on and 5-second off cycle. These characteristic steps result from the internal resistor being periodically activated and then deactivated to allow thermal relaxation.

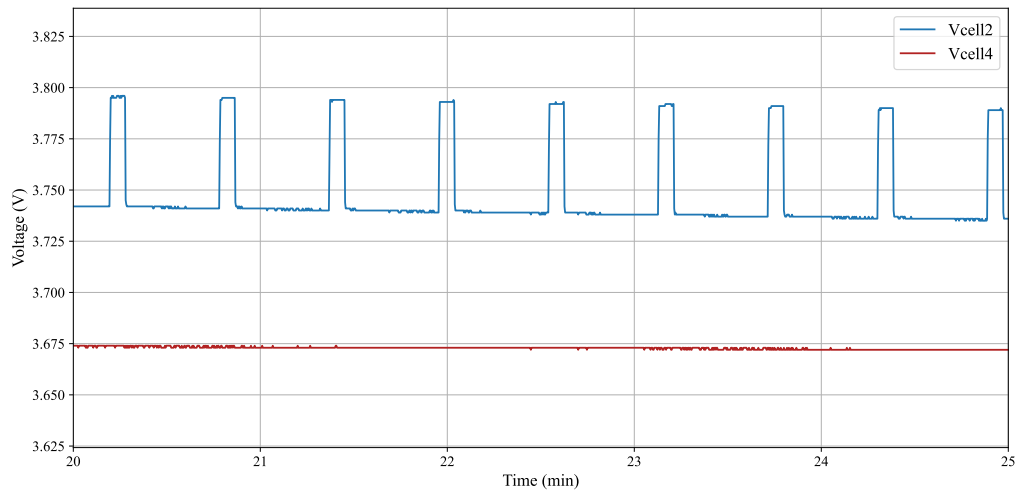


Figure 7.75: Zoomed view of voltage profile for Cell2 and Cell4 showing periodic balancing steps.

Finally, Figure 7.76 presents the SOC progression for both Cell2 and Cell4. The graph shows a clear convergence of the two SOC curves over time. The point at which the SOC difference drops below the configured 5% threshold is marked with a vertical line, highlighting the precise moment when the balancing logic automatically deactivated the balancing signal for Cell 2. This confirms the correct termination behavior of the firmware.

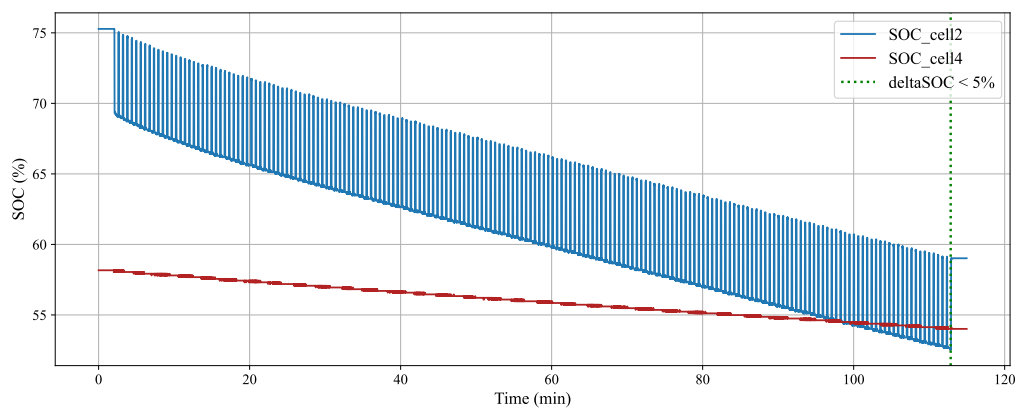


Figure 7.76: SOC evolution of Cell2 and Cell4 during balancing test.

This test successfully demonstrated the core functionality of the balancing strategy. It confirmed that the STM32 firmware could autonomously detect cell-level imbalance, configure the correct CB channel, initiate balancing using the BQ79616, and terminate the process precisely when the stopping condition was met.

7.5.4.2 Test 2: Full Balancing Cycle with Physical Imbalance

This final experiment aimed to validate the full balancing control strategy under realistic operating conditions. The objective was to manually create a physical imbalance across all six cells and observe how the system autonomously handled the correction process using the balancing control strategy from 7.5.3. This test was designed to simulate a real-world scenario where some cells have higher SOC levels than others, requiring sequential balancing actions based on the criteria defined in the firmware.

Figure 7.77 shows the voltage profile of all six cells over time during the balancing process. It is clear that the voltages of the cells progressively converged, approaching similar values by the end of the test. While this convergence suggests that the system achieved a balanced state, a more rigorous confirmation will be provided using the SOC values in a subsequent analysis. From the voltage behavior, it is evident that Cell5 consistently had the lowest voltage and was not selected for balancing. As defined in the strategy, only cells with deltaSOC greater than 10% relative to the minimum SOC are targeted for balancing. Accordingly, as predicted, Cell1 was not balanced due to its small deltaSOC and the balancing sequence followed was: Cell2, then Cell3, followed by Cell4 and Cell6.

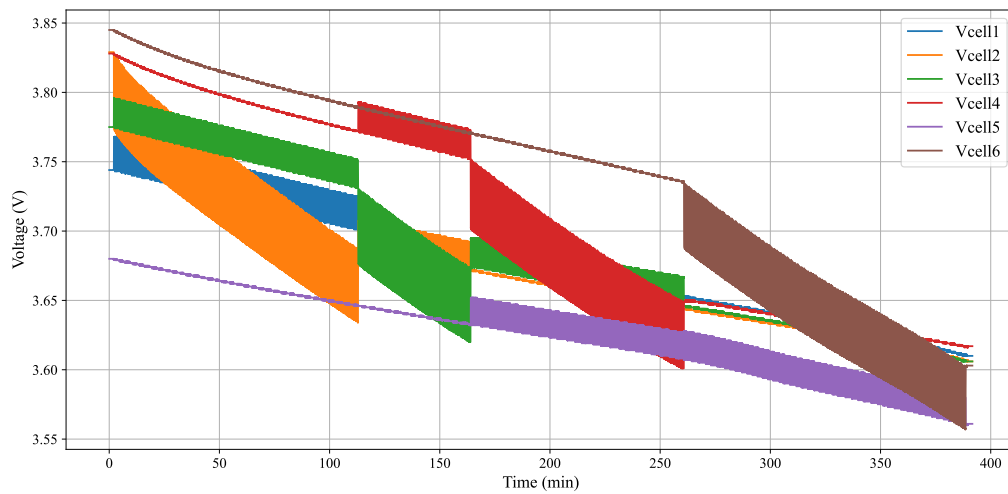


Figure 7.77: Voltage evolution of all six cells during full balancing test.

To better visualize the fine dynamics of the balancing process, Figure 7.78 shows a zoomed-in view of the voltage behavior during a portion of the test. By comparing the timing of this figure with the full voltage profile in Figure 7.77, it is possible to identify when each cell is being balanced.

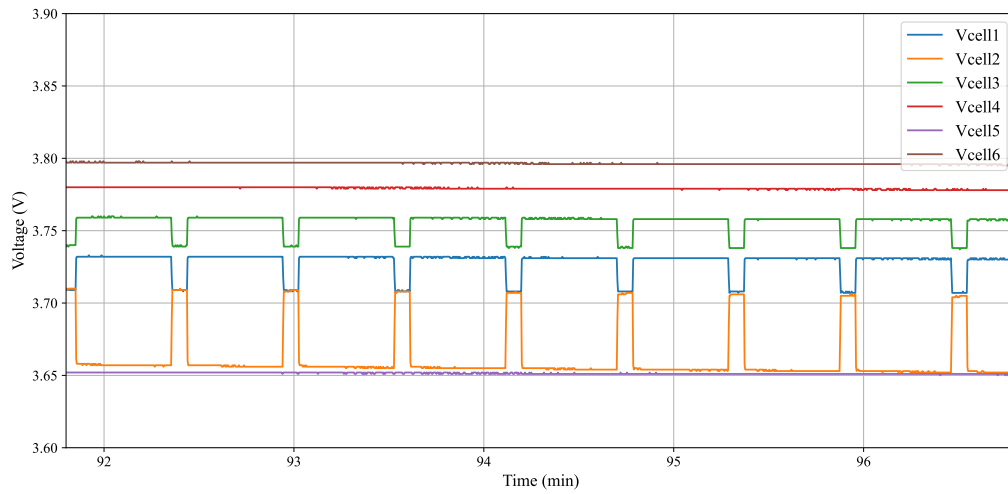


Figure 7.78: Zoomed voltage view showing interaction between Cell 2, Cell 1, and Cell 3 during balancing.

A particularly interesting phenomenon is observed during the intervals when the cells are being actively balanced: while the voltage of the selected cell decreases due to the activation of the internal discharge path, the voltages of adjacent cells simultaneously increase slightly.

This effect is a natural consequence of the differential measurement architecture used in the BQ79616, as shown in Figure 6.3. In this configuration, each cell voltage is measured as the difference between two consecutive tap voltages, such that $V_{cell2} = VC1 - VC2$. When Cell2 is balanced, the node VC1 (shared between Cell1 and Cell2) drops in potential due to current flowing through the internal balancing resistor. This localized drop causes two simultaneous effects: a reduction in V_{cell2} , as expected, and a perceived increase in $V_{cell1} = VC0 - VC1$, since VC1 is now lower. Similarly, if VC2 is slightly pulled down by internal current paths, $V_{cell3} = VC2 - VC3$ may also appear to increase.

Although these changes in adjacent cells are smaller than the drop in the balanced cell, they are consistent with the physical structure of the acquisition system. Interestingly, the voltage gain in neighboring cells often approximates the drop seen in the balanced cell, maintaining overall voltage conservation across the stack. While not perfectly symmetric, due to sampling delays, parasitic effects, and ADC resolution, this interaction confirms that the observed behavior is not a measurement error but rather a predictable outcome of the series-connected tap-based architecture. To support this analysis with quantitative data, Table 7.7 presents the measured cell voltages at selected key moments, specifically, at the end of each cell's balancing period. These values illustrate how the adjacent cells experience only minor voltage changes across time, which remain consistent with the system's background consumption rather than active discharge.

Table 7.7: Measured cell voltages during balancing sequence.

Time (min)	0.0	112.96	163.99	260.78	389.67
Vcell1	3.744	3.701	3.683	3.653	3.610
Vcell2	3.829	3.687	3.672	3.644	3.606
Vcell3	3.775	3.731	3.673	3.646	3.606
Vcell4	3.828	3.772	3.752	3.649	3.617
Vcell5	3.680	3.646	3.633	3.608	3.561
Vcell6	3.845	3.789	3.771	3.735	3.603

To confirm that the system achieved its balancing objective, Figure 7.79 presents the SOC evolution for all six cells.

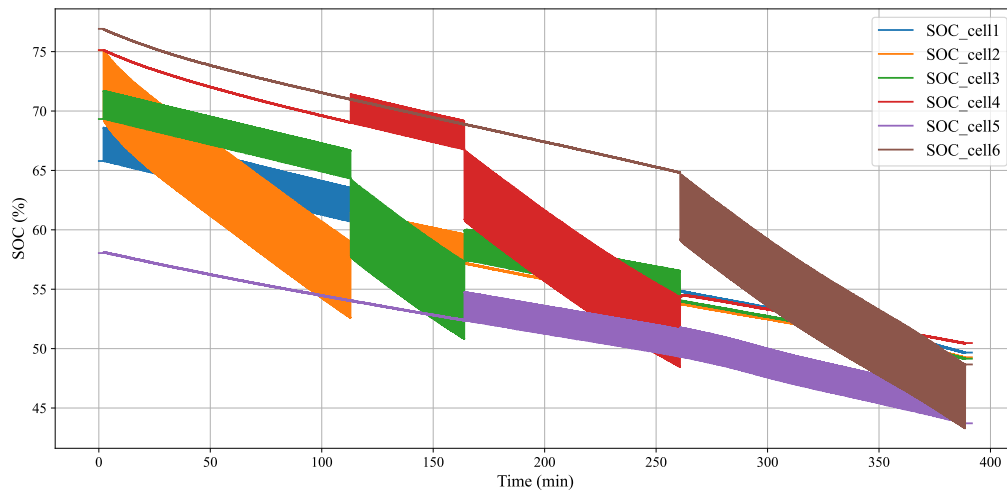


Figure 7.79: SOC progression of all six cells during the full balancing cycle.

To reinforce this observation, Table 7.8 presents the SOC values of each cell at the same key moments throughout the test. At the end of the balancing cycle, all cells converged to a SOC range within approximately 5 and 10% of the lowest SOC cell (Cell5). This confirms that the firmware correctly identified which cells required correction, applied the balancing logic sequentially, and successfully reached a uniform state within the defined threshold.

Table 7.8: SOC values of all cells during the balancing process.

Time (min)	0.0	112.96	163.99	260.78	389.67
SOC_cell1	65.791	60.732	58.522	54.851	49.668
SOC_cell2	75.110	58.988	57.203	53.774	49.269
SOC_cell3	69.327	64.329	57.299	54.031	49.152
SOC_cell4	75.130	69.048	66.787	54.328	50.470
SOC_cell5	58.044	54.031	52.395	49.317	43.713
SOC_cell6	76.917	70.999	68.895	64.812	48.667

7.5.5 Analysis and Conclusion of the Balancing Test

While the test results confirmed the effectiveness of the implemented balancing logic, several important considerations must be highlighted. First, it was observed that voltage readings taken during the balancing of a cell are not reliable for further processing or control decisions. This is due to the differential measurement architecture of the BQ79616, where balancing one cell influences the reference nodes of adjacent cells. As such, only the voltage measurements captured immediately before or after a balancing event (as shown in the previous tables) should be considered valid for system analysis or diagnostics.

Additionally, the implemented strategy introduced a fixed 5 second cooling delay between each balancing cycle. While this thermal pause helped mitigate heat buildup in the slave board, an alternative approach could be explored in future revisions: cycling between different cells rather than pausing entirely. By alternating the active CB channels instead of relying on a fixed resistor, the thermal load could be distributed more evenly, alternating from cell to cell, potentially improving balancing speed and system reliability.

Most importantly, this study confirms that passive balancing, although functionally effective, presents significant drawbacks in terms of energy efficiency and system wear. The process inherently wastes energy as heat, adds thermal stress to the accumulator, and demands long execution times. In the present system, energy is not only lost during balancing itself but also continuously consumed by the slave board throughout the operation. This reality reinforces that balancing should only be performed when absolutely necessary, and under ideal conditions.

Finally, it is worth noting that the application context strongly influences the necessity and frequency of balancing. In electric vehicles designed for extended daily use, where the battery undergoes hundreds of charge cycles, periodic balancing may be essential to preserve long-term performance. However, in the case of a FS prototype, where the car is not used very often and the number of complete charge cycles is very low, balancing may do more harm than good. For such scenarios, it is advisable to minimize balancing activity to preserve battery longevity and avoid unnecessary energy losses.

Chapter 8

Conclusions

This chapter presents the final reflections and outcomes of the project, divided into two key sections: an overall assessment of the objectives achieved during the development of the BMS prototype, and recommendations and insights for future work that can evolve from the foundation laid by this dissertation.

8.1 Project Achievements and Reflections

Although the development process was not without setbacks, ranging from communication issues to prototype constraints, the main objectives outlined at the beginning of the project were successfully addressed. This dissertation marks the conclusion of the second phase of the FSFEUP BMS development initiative, where the focus was to bring theoretical concepts into real-world validation through physical prototyping.

Despite some errors, some minor and others more structurally relevant, the implemented system proved capable of acquiring real cell data, estimating SOC values, controlling system behavior via a custom embedded software architecture, and validating the core architecture through testing. The lessons learned throughout this phase will be essential for the continued development of the BMS into a production-ready version.

Despite several implementation challenges, the developed prototype met the expectations in terms of technical feasibility, performance, and architectural flexibility. The project succeeded in translating simulations into embedded logic, creating a real-time acquisition and control system capable of operating with real hardware. This transition from theoretical models and simulation environments to the physical world inevitably revealed discrepancies and practical limitations, as expected in any complex engineering system. Nevertheless, the system's behavior and structure provide a solid foundation for continued development, serving as a reliable and extensible platform for future exploration and refinement of the various modules and strategies addressed in this work.

The following table summarizes this project's objectives and their respective fulfillment status:

Table 8.1: Summary of BMS objectives and implementation status.

Objectives	Status
Avoid centralized architecture	Fulfilled
Comply with FS rules and requirements	Fulfilled
Create a scalable BMS solution	Fulfilled
Create a configurable BMS solution	Fulfilled
Ensure low-error acquisition of voltages	Fulfilled
Ensure low-error acquisition of temperatures	Fulfilled
Ensure low-error acquisition of current	Fulfilled
SOC estimation capabilities	Fulfilled
SOH estimation capabilities	Partially implemented
Safety protocols integrated and functional	Fulfilled
Diagnostics integrated and functional	Fulfilled
Current control algorithm for charging/discharging	Implemented but not validated
Balancing control algorithm	Fulfilled
HMI for external monitoring	Fulfilled

It is important to clarify that certain aspects, such as the development of a complete and user-friendly HMI designed to reflect the final quality of a production system, the implementation of full current control capabilities to actively regulate the maximum battery current, or the reparametrization of the battery models, were not excluded due to lack of relevance, but rather due to time constraints inherent to this project phase. These elements were deprioritized to maintain focus on the core development goals and ensure a robust and functional prototype. Nonetheless, they remain essential components for future iterations and are strongly aligned with the long-term vision of this system.

8.2 Future Work and Recommendations

This project revealed several areas where further development and refinement are both necessary and promising. While the current prototype validates core functionality, future work should focus on:

- **Development of a complete human–machine interface (HMI):** In this project, the interaction with the BMS was carried out primarily through the Tera Term terminal, which provided access to all relevant variables for data acquisition and configuration. Although this simple text-based interface was sufficient for testing and validation, a production-ready system requires a more advanced and user-friendly HMI. Future development should focus on creating a complete interface capable of presenting all BMS functionalities in an intuitive

way. Such an interface would significantly improve usability and reflect the quality expected of a finished product. This would allow rapid adaptation of the system when changing cell types or chemistries, without requiring firmware modifications.

- **Reparameterization of the estimation models:** Although the SOC estimation methods used in this project performed well for the majority of the time, their precision can be improved by conducting dedicated parameterizations for a specific battery, to extract more accurate model parameters for both the KiBaM and Voc-based approaches. This solution can be done by doing specific test or doing online parameterization
- **Development of advanced current control logic:** The current limitation strategy developed in this work can be expanded into a comprehensive current control framework applicable across all BMS states. This includes implementing a fully functional CC-CV charging algorithm during charging phases, integrating DCM strategies during discharge, and enabling dynamic current modulation in response to fault conditions. For example, in warning or safe states, the system could autonomously reduce the current to mitigate escalating issues, while in the shutdown state, the current would be cut entirely to 0 A to ensure complete circuit isolation. These control mechanisms could be realized by interfacing the BMS with programmable chargers or electronic loads, allowing real-time current regulation and contributing to a safer and more intelligent battery system.
- **Integration of a complete SOH estimation module:** Based on the work done with internal resistance (R_{int}) reestimation, the next step is to implement a formal SOH estimation method. This would allow tracking cell degradation over time and predicting maintenance or replacement needs.
- **Investigation of non-linear R_{int} behavior:** Several tests revealed that internal resistance does not remain constant over time or across SOC levels. A more detailed study of its evolution could improve SOH tracking and the precision of the electric model.
- **Exploration of higher-order battery models:** Instead of a simple one-RC Thevenin equivalent model, future work could explore the use of n -order models to more accurately reflect battery dynamics and allow better customization for different chemistries.
- **Design and production of custom PCB's for master and slave modules:** To transition from the evaluation boards used in this project into an automotive-grade solution, it is necessary to design, manufacture, and validate dedicated PCB's for both the master and slave modules. Relying on evaluation hardware is neither cost-efficient nor sufficiently reliable for long-term use, as it does not align with the scalability and architectural robustness required of a final system. The development of custom PCB's will therefore be a critical step toward achieving a fully integrated, replicable, and production-ready BMS.

- **Scaling of the system with multiple slave modules:** A full-scale prototype should include multiple slave modules communicating via daisy-chain, to demonstrate that the architecture supports the management of a complete high-voltage battery pack.
- **Implementation of CAN for external communication:** In this project, UART was adopted both for internal communication between the master and slave modules and for external communication with the PC, mainly to simplify development and testing. While this approach proved effective for prototyping, it does not align with the communication architecture of the FS vehicle, where the ECU and other subsystems exchange data via the CAN protocol. Therefore, for future integration of the BMS prototype into the car, the external communication interface must be adapted from UART to CAN. This change will ensure compatibility with the vehicle's network, enable real-time data exchange with the ECU, and allow full validation of the BMS under real operating conditions.
- **Expansion of real-world test scenarios:** The current validation relied primarily on controlled discharge tests. Future iterations should include testing under dynamic conditions such as acceleration, regenerative braking, and varying thermal environments to assess system robustness, this would only be possible when the full prototype is done and prepared.

The modularity and flexibility of the developed prototype offer a strong base for the continued evolution of FSFEUP's BMS. The results achieved not only meet the immediate goals but also suggest that a fully custom solution can indeed rival and potentially surpass commercial alternatives in performance, adaptability, and cost-effectiveness.

References

- [1] United Nations. *United Nations' Sustainable Development Goals (SDGs)*. <https://sdgs.un.org/goals>. Accessed: 2024-07-26. 2016.
- [2] Institution of Mechanical Engineers. *Formula Student Website*. <https://www.imeche.org/events/formula-student/about-formula-student>. Accessed: 2025-04-14. 2025.
- [3] Formulastudent FEUP. *About Us | FSFEUP*. <https://formulastudent.fe.up.pt/aboutus-2/>. Accessed: 2025-03-10. 2025.
- [4] Formulastudent FEUP. *Formulastudent FEUP | Home*. <https://formulastudent.fe.up.pt/>. Accessed: 2025-03-10. 2025.
- [5] Gregory L Plett. *Battery management systems, Volume II: Equivalent-circuit methods*. Artech House, 2015.
- [6] *Battery Management Systems*. URL: <https://www.monolithicpower.com/en/learning/mpscholar/battery-management-systems> (visited on 12/04/2024).
- [7] Fazel Mohammadi and Mehrdad Saif. "A comprehensive overview of electric vehicle batteries market". In: *e-Prime - Advances in Electrical Engineering, Electronics and Energy* 3 (Mar. 2023), p. 100127. ISSN: 2772-6711. DOI: 10.1016/j.prime.2023.100127. URL: <https://www.sciencedirect.com/science/article/pii/S2772671123000220> (visited on 12/04/2024).
- [8] Davide Andrea. *Battery Management Systems for Large Lithium Ion Battery Packs*. en. Artech House, 2010. ISBN: 978-1-60807-105-0.
- [9] Bruno Manuel Baraças dos Santos. *Battery management system applied to projecto fst's ev prototype*. pt. 2024. URL: <http://www.fenixedu.org>.
- [10] Jorge Varela Barreras. "Practical Methods in Li-ion Batteries: for simplified modeling, battery electric vehicle design, battery management system testing and balancing system control". In: (2017).
- [11] A. K. M. Ahasan Habib et al. "A review: Energy storage system and balancing circuits for electric vehicle application". In: *IET Power Electronics* 14.1 (2021), pp. 1–13. DOI: <https://doi.org/10.1049/pel2.12013>. eprint: <https://ietresearch.onlinelibrary.wiley.com/doi/pdf/10.1049/pel2.12013>. URL: <https://ietresearch.onlinelibrary.wiley.com/doi/abs/10.1049/pel2.12013>.
- [12] Zachary Bosire Omariba, Lijun Zhang, and Dongbai Sun. "Review on Health Management System for Lithium-Ion Batteries of Electric Vehicles". In: *Electronics* 7.5 (2018). ISSN: 2079-9292. DOI: 10.3390/electronics7050072. URL: <https://www.mdpi.com/2079-9292/7/5/72>.

- [13] Shichun Yang et al. *Advanced Battery Management System for Electric Vehicles*. Springer Nature, 2023.
- [14] Sónia Carinaso Lopes da Costa and Armando Luís Sousa Araújo. “Análise e Desenvolvimento de um método de Estimação de Estado de Carga de Baterias Baseado em Filtros de Kalman”. MA thesis. Universidade do Porto (Portugal), 2014.
- [15] Dickson N. T. How et al. “State of Charge Estimation for Lithium-Ion Batteries Using Model-Based and Data-Driven Methods: A Review”. In: *IEEE Access* 7 (2019). Conference Name: IEEE Access, pp. 136116–136136. ISSN: 2169-3536. DOI: [10.1109/ACCESS.2019.2942213](https://doi.org/10.1109/ACCESS.2019.2942213). URL: <https://ieeexplore.ieee.org/document/8843918/?arnumber=8843918> (visited on 12/13/2024).
- [16] Ruifeng Zhang et al. “State of the Art of Lithium-Ion Battery SOC Estimation for Electrical Vehicles”. In: *Energies* 11.7 (2018). ISSN: 1996-1073. DOI: [10.3390/en11071820](https://doi.org/10.3390/en11071820). URL: <https://www.mdpi.com/1996-1073/11/7/1820>.
- [17] Stefano Floridia. “Analysis and development of Battery Management System for electric vehicles”. PhD thesis. Politecnico di Torino, 2022.
- [18] Rui Xiong, Linlin Li, and Jinpeng Tian. “Towards a smarter battery management system: A critical review on battery state of health monitoring methods”. In: *Journal of Power Sources* 405 (2018), pp. 18–29. ISSN: 0378-7753. DOI: <https://doi.org/10.1016/j.jpowsour.2018.10.019>. URL: <https://www.sciencedirect.com/science/article/pii/S037877531831111X>.
- [19] M. Bercebar et al. “Critical review of state of health estimation methods of Li-ion batteries for real applications”. In: *Renewable and Sustainable Energy Reviews* 56 (2016), pp. 572–587. ISSN: 1364-0321. DOI: <https://doi.org/10.1016/j.rser.2015.11.042>. URL: <https://www.sciencedirect.com/science/article/pii/S1364032115013076>.
- [20] Daniel Filipe Pereira Magalhães. “Modelo de Baterias com Aplicação em Sistemas de Gestão de Baterias (BMS) de Veículos Elétricos (EVs)”. por. In: (Mar. 2013). Accepted: 2019-02-02T07:46:19Z. URL: <https://repositorio-aberto.up.pt/handle/10216/98136> (visited on 12/12/2024).
- [21] Valentina Lucaferri et al. “A Review on Battery Model-Based and Data-Driven Methods for Battery Management Systems”. In: *Energies* 16.23 (2023). ISSN: 1996-1073. DOI: [10.3390/en16237807](https://doi.org/10.3390/en16237807). URL: <https://www.mdpi.com/1996-1073/16/23/7807>.
- [22] Qi Zhang et al. “A Fractional-Order Kinetic Battery Model of Lithium-Ion Batteries Considering a Nonlinear Capacity”. In: *Electronics* 8.4 (2019). ISSN: 2079-9292. DOI: [10.3390/electronics8040394](https://doi.org/10.3390/electronics8040394). URL: <https://www.mdpi.com/2079-9292/8/4/394>.
- [23] Masoomah Karami et al. “An Extension of the Kinetic Battery Model for Optimal Control Applications”. In: *2023 IEEE 32nd International Symposium on Industrial Electronics (ISIE)*. 2023. DOI: [10.1109/ISIE51358.2023.10227986](https://doi.org/10.1109/ISIE51358.2023.10227986).
- [24] Gustavo Piske Fenner et al. “Comprehensive Model for Real Battery Simulation Responsive to Variable Load”. In: *Energies* 14.11 (2021). ISSN: 1996-1073. DOI: [10.3390/en14113209](https://doi.org/10.3390/en14113209). URL: <https://www.mdpi.com/1996-1073/14/11/3209>.

- [25] Xiaosong Hu et al. “Advanced Fault Diagnosis for Lithium-Ion Battery Systems: A Review of Fault Mechanisms, Fault Features, and Diagnosis Procedures”. In: *IEEE Industrial Electronics Magazine* 14.3 (Sept. 2020). Conference Name: IEEE Industrial Electronics Magazine, pp. 65–91. ISSN: 1941-0115. DOI: [10.1109/MIE.2020.2964814](https://doi.org/10.1109/MIE.2020.2964814). URL: <https://ieeexplore.ieee.org/document/9205673> (visited on 12/13/2024).
- [26] Manh-Kien Tran and Michael Fowler. “A Review of Lithium-Ion Battery Fault Diagnostic Algorithms: Current Progress and Future Challenges”. en. In: *Algorithms* 13.3 (Mar. 2020). Number: 3 publisher: Multidisciplinary Digital Publishing Institute, p. 62. ISSN: 1999-4893. DOI: [10.3390/a13030062](https://doi.org/10.3390/a13030062). URL: <https://www.mdpi.com/1999-4893/13/3/62> (visited on 12/13/2024).
- [27] Mahammad A. Hannan et al. “State-of-the-Art and Energy Management System of Lithium-Ion Batteries in Electric Vehicle Applications: Issues and Recommendations”. In: *IEEE Access* 6 (2018). Conference Name: IEEE Access, pp. 19362–19378. ISSN: 2169-3536. DOI: [10.1109/ACCESS.2018.2817655](https://doi.org/10.1109/ACCESS.2018.2817655). URL: <https://ieeexplore.ieee.org/document/8320763/?arnumber=8320763> (visited on 12/11/2024).
- [28] Weixiang Shen, Thanh Tu Vo, and Ajay Kapoor. “Charging algorithms of lithium-ion batteries: An overview”. In: *2012 7th IEEE Conference on Industrial Electronics and Applications (ICIEA)*. ISSN: 2158-2297. July 2012, pp. 1567–1572. DOI: [10.1109/ICIEA.2012.6360973](https://doi.org/10.1109/ICIEA.2012.6360973). URL: <https://ieeexplore.ieee.org/document/6360973> (visited on 12/11/2024).
- [29] A. K. M. Ahasan Habib et al. “Lithium-Ion Battery Management System for Electric Vehicles: Constraints, Challenges, and Recommendations”. en. In: *Batteries* 9.3 (Mar. 2023). Number: 3 Publisher: Multidisciplinary Digital Publishing Institute, p. 152. ISSN: 2313-0105. DOI: [10.3390/batteries9030152](https://doi.org/10.3390/batteries9030152). URL: <https://www.mdpi.com/2313-0105/9/3/152> (visited on 12/11/2024).
- [30] Zachary Bosire Omariba, Lijun Zhang, and Dongbai Sun. “Review of Battery Cell Balancing Methodologies for Optimizing Battery Pack Performance in Electric Vehicles”. In: *IEEE Access* 7 (2019). Conference Name: IEEE Access, pp. 129335–129352. ISSN: 2169-3536. DOI: [10.1109/ACCESS.2019.2940090](https://doi.org/10.1109/ACCESS.2019.2940090). URL: <https://ieeexplore.ieee.org/document/8834858/?arnumber=8834858> (visited on 12/04/2024).
- [31] Mina Naguib, Phillip Kollmeyer, and Ali Emadi. “Lithium-Ion Battery Pack Robust State of Charge Estimation, Cell Inconsistency, and Balancing: Review”. In: *IEEE Access* 9 (2021). Conference Name: IEEE Access, pp. 50570–50582. ISSN: 2169-3536. DOI: [10.1109/ACCESS.2021.3068776](https://doi.org/10.1109/ACCESS.2021.3068776). URL: <https://ieeexplore.ieee.org/document/9386065/?arnumber=9386065> (visited on 12/04/2024).
- [32] Muhammad Uzair, Ghulam Abbas, and Saleh Hosain. “Characteristics of Battery Management Systems of Electric Vehicles with Consideration of the Active and Passive Cell Balancing Process”. en. In: *World Electric Vehicle Journal* 12.3 (Sept. 2021). Number: 3 Publisher: Multidisciplinary Digital Publishing Institute, p. 120. ISSN: 2032-6653. DOI: [10.3390/wevj12030120](https://doi.org/10.3390/wevj12030120). URL: <https://www.mdpi.com/2032-6653/12/3/120> (visited on 12/04/2024).
- [33] “Energy-efficient system and charge balancing topology for electric vehicle application”. In: 53 (Oct. 2022), p. 102516. ISSN: 2213-1388. DOI: [10.1016/j.seta.2022.102516](https://doi.org/10.1016/j.seta.2022.102516). URL: <https://www.sciencedirect.com/science/article/pii/S2213138822005665>.

- [34] Sonu Kumar et al. “Switched-Resistor Passive Balancing of Li-Ion Battery Pack and Estimation of Power Limits for Battery Management System”. en. In: *International Journal of Energy Research* 2023.1 (2023), p. 5547603. ISSN: 1099-114X. DOI: 10.1155/2023/5547603. URL: <https://onlinelibrary.wiley.com/doi/abs/10.1155/2023/5547603> (visited on 12/11/2024).
- [35] *Review of Battery Balancing Techniques based on Structure and Control Strategy* | IEEE Conference Publication | IEEE Xplore. URL: <https://ieeexplore.ieee.org/document/9597839> (visited on 12/01/2024).
- [36] VDA QMC and intacs™. *Automotive SPICE® Process Assessment Model Version 4.0*. Yellow Print Release. Automotive Special Interest Group (SIG). 2023. URL: <https://vda-qmc.de/wp-content/uploads/2023/06/Automotive-SPICE-PAM-40-Gelbbandrelease.pdf>.
- [37] Formula Student Germany. *Formula Student Germany — Rules*. <https://www.formulastudent.de/fsg/rules/>. Accessed: 2025-04-20. 2025.
- [38] Ewert Energy Systems, Inc. *Orion BMS Standard - Lithium-Ion Battery Management System*. <https://www.orionbms.com/products/orion-bms-standard>. Accessed: 2025-05-19. 2025.
- [39] Ewert Energy Systems, Inc. *Orion BMS2 Operational Manual*. https://www.orionbms.com/manuals/pdf/orionbms2_operational_manual.pdf. Accessed: 2025-05-19. 2025.
- [40] STMicroelectronics. *STM32F446xC/E Microcontroller Datasheet*. <https://www.st.com/resource/en/datasheet/stm32f446mc.pdf>. Accessed: 2025-03-26. 2025.
- [41] Digi-Key Electronics. *STMicroelectronics NUCLEO-L152RE Development Board*. <https://www.digikey.pt/pt/products/detail/stmicroelectronics/NUCLEO-L152RE/4695528>. Accessed: 2025-03-26. 2025.
- [42] Francisco Oliveira Amorim Plácido Teixeira. “Development of a Battery management system for electric vehicles applied to formula student”. In: (2024).
- [43] Molicel. *INR-18650-P28A Lithium-Ion Cell Datasheet*. <https://www.molicel.com/wp-content/uploads/INR18650P28A-V1-80093.pdf>. Accessed: 2025-02-16. 2025.
- [44] NKON. *Molicel INR-18650-P28A 2800mAh 35A Product Page*. <https://www.nkon.nl/pt/molicel-inr18650-p28a-2800mah-35a.html>. Accessed: 2025-07-16. 2025.
- [45] *Datasheet Battery*. URL: <https://www.mathworks.com/help/autoblks/ref/datasheetbattery.html> (visited on 12/13/2024).
- [46] Sónia Carina Lopes da Costa. *Análise e desenvolvimento de um método de estimação de estado de carga de baterias baseado em filtros de kalman*. (Visited on 2014).
- [47] Daniel F. Magalhães, Armando S. Araújo, and Adriano S. Carvalho. “A model for battery lifetime calculation implementable in circuit simulators”. In: *2013 World Electric Vehicle Symposium and Exhibition (EVS27)*. 2013, pp. 1–6. DOI: 10.1109/EVS.2013.6915012. URL: <https://ieeexplore.ieee.org/document/6915012>.
- [48] Texas Instruments. *BQ79616EVM-021 Evaluation Module User Guide*. <https://www.ti.com/lit/ug/sluuc37d/sluuc37d.pdf>. Accessed: 2025-02-09. 2025.

- [49] Texas Instruments. *BQ79616EVM-021 Evaluation Module Product Page*. <https://www.ti.com/tool/BQ79616EVM-021>. Accessed: 2025-03-13. 2025.
- [50] Texas Instruments. *BQ79616-Q1 Automotive 16-Channel Battery Monitor Datasheet*. <https://www.ti.com/lit/ds/symlink/bq79616-q1.pdf>. Accessed: 2025-02-09. 2025.
- [51] Murata Electronics. *NXFT15XH103FEAB050 NTC Thermistor Datasheet*. <https://www.murata.com/products/productdetail?partno=NXFT15XH103FEAB035>. Accessed: 2025-02-13. 2025.
- [52] Digi-Key Electronics. *Murata Electronics NXFT15XH103FEAB050 Product Page*. <https://www.digikey.pt/pt/products/detail/murata-electronics/NXFT15XH103FEAB050/10376890>. Accessed: 2025-02-13. 2025.
- [53] Allegro MicroSystems. *ACS759: Current Sensor IC Datasheet*. <https://www.allegromicro.com/-/media/files/datasheets/acs759-datasheet.pdf>. Accessed: 2025-03-07. 2025.
- [54] Robbe Modellsport. *ISDT T8 BattGO Smart Charger Product Page*. <https://www.robbe.com/en/RC-ELECTRONICS/CHARGERS-POWER-SUPPLIES/Chargers/Chargers-DC-12-50V/ISDT-T8-BATTGO-SMART-CHARGER-1-8S-1000W-30A-12-40V-DC/T8>. Accessed: 2025-02-03. 2025.
- [55] Robbe Modellsport. *ISDT T8 BattGO Smart Charger User Manual*. https://www.robbe.com/media/5f/0a/48/1588084547/9748933_EN_Manual.pdf?ts=1588084547. Accessed: 2025-02-03. 2025.
- [56] Mean Well Enterprises Co., Ltd. *HRPG-1000 Series Power Supply Datasheet*. <https://www.meanwellusa.com/upload/pdf/HRPG-1000/HRPG-1000-spec.pdf>. Accessed: 2025-02-04. 2025.
- [57] BK Precision Corporation. *BK Precision 8510 DC Electronic Load Product Page*. <https://www.bkprecision.com/products/dc-electronic-loads/8510>. Accessed: 2025-03-10. 2025.
- [58] BK Precision Corporation. *BK Precision 8500 Series DC Electronic Loads Datasheet*. https://bkpmedia.s3.us-west-1.amazonaws.com/downloads/datasheets/en-us/85xx_datasheet.pdf. Accessed: 2025-03-10. 2025.
- [59] Texas Instruments. *BQ79616-Q1 Software Development Application Note*. <https://www.ti.com/lit/an/slvae86b/slvae86b.pdf>. Accessed: 2025-04-29. 2025.

Appendix A

Use Case Diagram

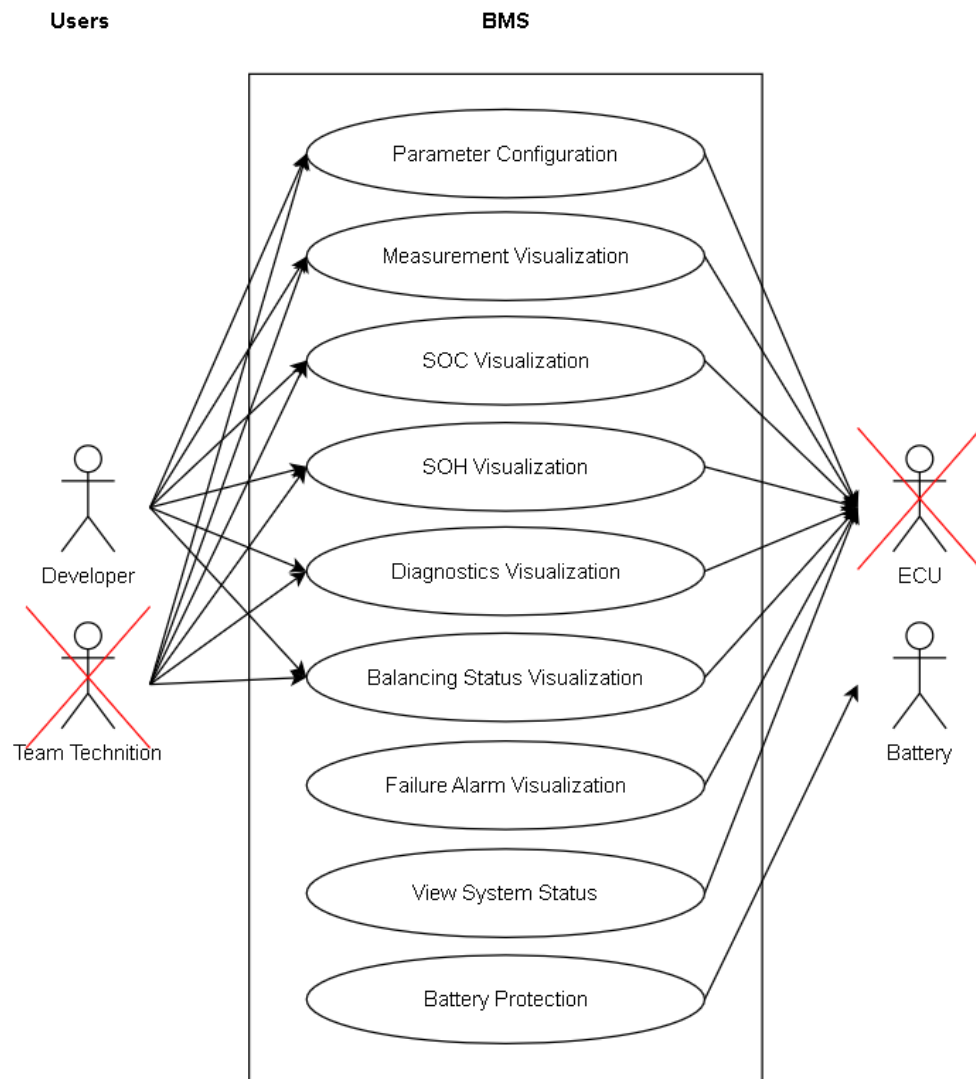


Figure A.1: Use Case Diagram

Appendix B

Work Breakdown Structure and Development Plan

This appendix presents the Work Breakdown Structure (WBS) used to organize the development of the BMS prototype, along with the corresponding Gantt chart that details the execution timeline of each task. The prioritization of modules was based on their functional criticality and integration dependencies.

