POLITECNICO DI TORINO

Master Degree in Mechatronic Engineering

Master Degree Thesis

Design and Implementation of a Battery Management System for Automotive Applications

Supervisor
prof. Stefano Malan

Candidate
Marco Carzedda

Internship Tutor
Ing. Angelo Borneo

October 2024

Contents

List of Figures

List of Tables

Acronyms

BMS	Battery Management System
EV	Electric Vehicle
SOC	State Of Charge
OCV	Open Circuit Voltage
CC	Coulomb Counting
EMF	Electro Motive Force
EIS	Electrochemical Impedance Spectroscopy
KF	Kalman Filter
EKF	Extended Kalman Filter
UKF	Unscented Kalman Filter
NN	Neural Networks
SOH	State Of Health
ASIL	Automotive Safety Integrity Level
HARA	Hazard Analysis and Risk Assessment
E/E	Electrical/Electronic
CC-CV	Constant Current-Constant Voltage

Chapter 1

Introduction

1.1 Battery Management System

Modern societies are nowadays facing monumental and serious issues, which often require an adequate and ingenious response to find fully functional and efficient solutions. Energy sources and their respective management systems are indeed part of said problems: carbon fossil fuels, for instance, are one of the main sources of energy on which all kinds of industrial sectors are now relying on to power their applications. The environmental impact, however, has now become too evident to be ignored, as it poses a great and not-so-distant threat. For this reason, in the past few decades a tendency to replace old state-of-the-art technical solutions with innovative electrical proposals has surfaced. The hybrid/electric vehicle field heavily relies on batteries, perhaps the best-known energy storage system, which would be of no good use if it was not coupled with a management system responsible for a plethora of tasks. Such systems are known as Battery Management System (BMS) and they play a crucial role in the performance, safety, and longevity of all real life applications that require the implementation of batteries. Among the various fields of application, Electric Vehicles (EVs) now represent a challenging frontier, as the automotive industry increasingly shifts towards electrification. In essence, a BMS serves as the brain of an Electric Vehicle (EV) battery pack, monitoring and controlling its various parameters to optimize efficiency and ensure safe operation. The reasons why we mainly couple battery packs with BMSs are:

- to ensure that the battery constantly operates in a safe operating state;
- to increase the reliability of the battery, by avoiding critical conditions which would compromise and shorten the lifespan;
- to better manage the available energy thus increasing the efficiency.

A battery alone would not in fact be able to operate correctly without said system and it would likely fall into unsafe and inefficient operating states. In order to achieve the goals listed above, the BMS oversees critical functions such as:

- Cell balancing, which allows to maintain a similar level of stored energy among all cells;
- State Of Charge (SOC) estimation, to compute an accurate estimate of the available energy in real-time;
- State Of Health (SOH) estimation, which is intended to monitor the aging phenomenon of the battery pack;
- Temperature management, to ensure safe operating conditions inside a specific temperature range while safeguarding against potential hazards like thermal runaway or premature degradation;
- Protection against overcharging or over-discharging.

Furthermore, advancements in BMS technology have led to increased energy density, improved range, and faster charging capabilities for EVs. In summary, Battery Management Systems are indispensable components of electric vehicles, ensuring the optimal utilization, safety, and longevity of battery packs in the pursuit of a sustainable and electrified automotive future.

1.2 Software tools

The purpose of this master thesis is software development and integration oriented and aims at developing an ad-hoc BMS (and all its main functionalities) for a custom-made battery pack model of an EV. The entire work was carried out in a MATLAB environment and, more specifically, it took advantage of additional tools to accomplish the tasks mentioned in the previous paragraph:

- **Simulink:** MathWorks software that allows to model, analyze and simulate dynamical systems. For this thesis, BMS functions were developed by means of appropriate block schemes which, thanks to the close integration between MATLAB and Simulink, were parameterized by suitable variables defined in **.m** scripts.
- **Simscape Battery:** MATLAB add-on which contains tools to design and parameterize battery systems. For this thesis in particular, the Battery Pack Model Builder, a designing tool which allows the user to create simulation models with the desired topology and links to cooling plates to evaluate the electrical and thermal responses, was extensively exploited to generate a virtual model of the battery pack, on which the BMS would act upon.

- **Stateflow:** Mathworks tool developed to implement control logics into dynamical systems by means of flow charts and state machines; some of the Simulink models developed in this work are in fact equipped with a control logic, designed via Stateflow charts.

- **Data Inspector:** the Data Inspector, available in every Simulink file, was exploited to inspect time series data and compare the most significant signals for simulation purposes. This tool fulfills in fact multiple functions, as it allows to view the behaviour in time of every signal, to import data from the MATLAB Workspace, to make graphical comparisons and to even save or share relevant simulation data.

1.3 Motivations

The transition towards a fully electrified automotive field, which has been rapidly progressing in the past few years, has determined an increase in effort to be put into the research for new effective solutions. When thinking about a sustainable future, it is now impossible not to think about electric vehicles, the new frontier for everyday transportation, which can still be improved and perfected, especially when it comes to the energy storage and management systems. In view of the aforementioned considerations, this project aims at developing an overall functioning structure for a Battery Management System, that can potentially supervise an EV battery pack; more specifically the goal was firstly to create a battery pack Simulink block by means of Simscape Battery, with suitable electrical and thermal features that recall real-life storage systems. Secondly, the most pivotal BMS functions would be developed via block schemes on Simulink and tested on the virtual model of the pack by running simulations. Generally speaking, this thesis is not to be intended as a step forward in the development of new technological solutions, but rather as a solid implementing example of already existing techniques, on which future works could be started without needing to begin from scratch.

1.4 Thesis outline

For the sake of making this thesis as orderly and comprehensible as possible, the following paragraph will briefly lay out and describe the topics that will be covered in the next chapters, so that readers can better navigate this piece of work. At the end of this very first introductory chapter is a focus on the analysis of the state of the art techniques that are now put in place to implement the main BMS functions: the description starts with the most widespread battery technologies and their respective cell geometries and electrical quantities, then it shifts to the consolidated BMS algorithms for Cell Balancing, SOC and SOH estimation, Thermal

Management and Fault Detection and, in the end, the Functional Safety concept is described with proper references to the standard ISO 26262. The software development work is discussed in the third and central chapter, which first focuses on the technical aspects of the custom battery pack block realized via Simscape Battery and then on the algorithms and Simulink schemes for each one of the BMS functions. After setting up and running the simulations, the results are presented and validated in the fourth chapter, where the data regarding the unit testing phase and the complete integration are both discussed. To conclude, the final chapter underlines the critical points of the project and provides ideas for possible improving future works.

Chapter 2

State of art

2.1 Battery technologies

Batteries are well known and widespread devices, which allow to stock or provide energy in the form of continuous currents and voltages, by exploiting a complex conversion process between chemical and electric energy. This mechanism is to be found in the working principle of electrochemical cells that make up the battery pack, where oxidation-reduction reactions occur; a cell is in fact composed of two metal plates soaked in a saline solution named electrolyte, which dissociates in positive and negative ions. One of the plates, the so called anode, is made up of a chemical element with the natural tendency of losing electrons, thus making the oxidation reaction happen and liberating positive ions in the electrolyte. On the contrary, the chemical composition of the cathode (the second plate) triggers a reduction reaction, because it naturally acquires electrons and transforms the negative ions into electrically neutral compounds. If we were to close the two ends of a cell with a conductive wire, and perhaps serially connecting a load, an electron current would flow from the anode to the cathode passing through the load, while an ion current would be observed to flow in opposite direction through the electrolyte. No further explanation will be given on the physical-chemical phenomena that govern the working principle of a battery, as it is already extensively discussed in the literature ([1], [2], [3]) and out of the scope of this thesis. For the sake of completeness, however, it is worth listing (and briefly discussing) the technical and functioning solutions for cell technologies available in the market, while also underlining which candidates are more suitable to be implemented for automotive applications or, even better, EVs. The first feature to consider when analyzing and evaluating an electrochemical cell is its geometry. Cell come in three standardized geometries, each with different intrinsic characteristics:

- **Cylindrical:** the anode and the cathode are divided by a separator and manufactured in the form of layers that are rolled together and then sealed in an aluminium or steel case. Cylindrical cells are known to have a bigger capacity compared to the other shape formats (up to 100 Ah) and the lowest manufacturing cost. Despite having a long lifespan (25000 discharge cycles), a high temperature resistance and a cheap and standardized production, this geometry is bound to have a low volumetric energy density: when various cylindrical cells are assembled into a battery pack, the space between the units will be left unoccupied, making this solution poor for the automotive field, where the available space must be optimally exploited.

- **Prismatic:** the electrodes are layered together and enclosed in a rectangular metal case, which gives the cell its typical prismatic shape. Besides taking up the most space, this geometry can reach up to 200 Ah with a single cell and can also be exploited to combine cells with no gaps in between (better volumetric energy density). The biggest issues lie in the likeliness of overheating, in the higher production cost and in a much lower lifespan, which oscillates around 2000 cycles on average. This solution is highly appreciated in the automotive field, as it meets all the needs.

- **Pouch:** these are cells without a proper stiff structure, because they are usually sealed inside a flexible aluminium container. This structural feature provides an incredible advantage over the other competitors, as pouch cells take up the least amount of space as well as weighing considerably less. Moreover, small size and lightweight are highly appreciated features in the automotive field, that is why this geometry is slowly making its way into the car world, as some manufacturers have started introducing it in their vehicles. Despite having other interesting perks, pouch cells have a low lifespan (comparable to prismatic cells'), high production costs, low mechanical resistance and a not negligible risk of overheating.

A more in depth analysis about cell geometry can be read in [2] and [4]. To conclude this brief overview on battery technologies, we must also discuss the types of accumulators available in the market, to better evaluate how weight, power, autonomy and cost influence the overall performance. The most common rechargeable batteries employed in the automotive field are:

- **Lead-acid:** as shown in Fig 2.1, lead-acid batteries have a lead anode (Pb) and a cathode made of a lead alloy (PbO_2), while the electrolyte is sulfuric acid (H_2SO_4). This accumulator dates back to the french physicist Gaston Planté, who invented it in 1859, and for this reason it is now a well-known and consolidated technology, mainly used to kick-start a car thermal engine. With the knowledge matured over the years also comes a big availability and cheap raw materials; on the other hand, lead-acid batteries must not be left discharged or only partially charged for long periods of time, because issues such as internal short-circuit, sulphation or acid leakage might occur. For more details read [5], [6] and [2].

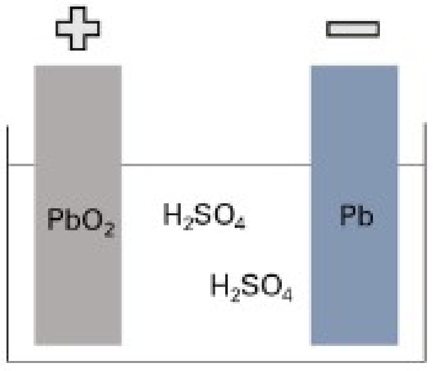


Figure 2.1. Lead-acid cell

- **Nickel-cadmium (NiCd):** in order to overcome the limitations of lead-acid batteries, nickel-cadmium cells were realized to power engines. As deeply discussed in [7], these cells have a lifespan that can oscillate between 1000 and 2000 cycles and can provide a gravimetric energy density of about 45 Wh/kg. As far as the chemical composition is concerned, the cathode, anode and electrolyte are manufactured by respectively using nickel oxide, cadmium and an alkaline substance [8]. Overall, NiCd cells exhibit better features than lead-acid batteries such as faster charge-discharge rates, a longer shell-life and the capability of withstanding idle time spans when left discharged, without the risk of being damaged [9]; however, two drawbacks can also be listed: the first one is the so called “memory effect”, which manifests itself when the user subjects the battery to early charging cycles (meaning that it is not fully discharged before being recharged), causing it to degrade its capacity overtime. The second is strictly related to the natural properties of cadmium, a chemical element which, due to its toxicity, needs to be carefully handled, thus generating managing issues and high costs.

- **Nickel-metal hydride (NiMH):** the development of NiMH cells represents a step forward in the production and development of rechargeable batteries, which constantly aims at finding the technology that better leverages performance and limitations. When it comes to comparing NiMH cells with their predecessor (NiCd cells), huge improvements can be observed: cadmium is replaced with metal alloys (therefore the anode “loses” its toxicity trait), energy density and capacity are improved with respect to NiCd cells of equal dimension and the memory effect becomes less impactful [7]. This technology used to be a protagonist in the production of batteries for EVs and hybrid vehicles, as many manufacturers chose them to power their cars. Over the years, however, thanks to the continuous research in the field, new solutions were developed, and NiMH are now almost entirely replaced by cells of different nature such as lithium-ion [2].

- **Lithium-ion**: even though all the above mentioned energy storage systems have been widely used for automotive applications, the technology that has been representing the most attractive choice in the field for many years is the lithium-ion cell: in spite of its high cost, this battery is in fact consistently being researched on, in order to enhance the performance in terms of durability, safety and reliability [10]. When compared to other types of electrochemical cells, lithium-ion batteries excel under many aspects: weight and volume are tremendously reduced as opposed to previous models (especially lead-acid) and, because the chemical elements that compose the cell are lightweight by nature, increased autonomy and lower energy consumption are beneficial to all EVs that are powered by this technology. Moreover, the volumetric energy density reaches high values (up to 530 Wh/l), the gravimetric energy density is improved along with the power density, the self-discharge rate goes down to a monthly 5% (against a common 30%), while the tedious memory effect, typical of NiCd batteries, does not have any effect. Nevertheless, there are still some safety issues that, to this day, make this battery dangerous if not carefully handled, and for this reason engineers keep researching to strategically enhance the system. Such problems mainly arise from the chemical composition of the cell: the anode is typically made up of a graphite compound while the cathode has been developed with many different materials; the first models used to mount a metallic lithium cathode, which was implicitly chemically unstable and could easily overheat, leading to fusion of the lithium, fires, rapid oxidation or, in worst case scenarios, the explosion of the battery. Overcoming these serious safety hazards was essential, therefore engineers expanded upon their technical knowledge and created various non-metallic lithium solutions for the cathode, which drastically improved the chemical stability of the battery. The schematic and simplified structure of a generic lithium cell is shown in Figure 2.2. For more information refer to [2].

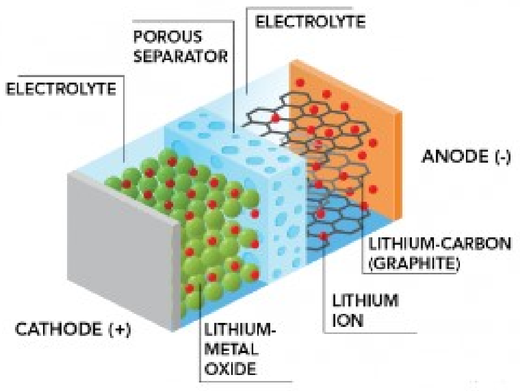


Figure 2.2. Lithium ion cell

The chemical and physical phenomena behind batteries belong to a vast and fascinating subject, which, although essential to deeply understand how energy storage systems such as batteries really work, requires time to be fully grasped on. For these reasons and because these topics are out of the scope of this thesis, readers can deepen their knowledge digging through literature if interested in the subject. Table 2.1 sums up the technical features of battery cells ([10]) and concludes the first paragraph of the chapter.

Technology	**Lead-Acid**	**NiMH**	**Lithium ion**
Temperature [°C]	$-30 \div 60$	$-20 \div 50$	$-20 \div 55$
Efficiency η (%)	85	80	93
Volumetric Energy Density [Wh/l]	$50 \div 70$	200	$150 \div 200$
Gravimetric Energy Density [Wh/kg]	$20 \div 40$	$40 \div 60$	$100 \div 200$
Power Density [W/kg]	300	$1300 \div 500$	$3000 \div 800$
Voltage [V]	2.1	1.2	3.6
Self-Discharge [%/Month]	$4 \div 8$	20	$1 \div 5$
Cycle life	200	> 2500	< 2500
Cost estimation [$/kWh]	150	500	800

Table 2.1. Performance comparison among battery cells

2.2 BMS functions and algorithms

As we learned in the previous chapter, no battery can safely and efficiently operate without a BMS, a piece of equipment that is pivotal for any electric storage system, especially if said system is integrated in a hybrid/electric vehicle and must properly work at all times. The following sections will propose the state of the art solutions and algorithms that are normally put in place to program the software functions of a BMS.

2.2.1 Cell Balancing methods

When dealing with battery powered devices, it is essential to balance the level of charge stored in all the cells that make up the energy storage system. The individual cells are properly connected in series and parallel topologies, to obtain a battery pack with the desired electrical characteristics; even though they might be the same, cells always display manufacturing differences which result in the decrease of the battery overall usable capacity but also lead to possible safety hazards such as failure of the cells and potential overheating ([11] and [12]). Hypothetically, if an imbalanced battery was to be discharged, the useful capacity of the entire pack would be constrained to the cell with the lowest SOC level, as it would reach the lowest tolerable voltage faster than the other cells; on the contrary, when imposing a charging cycle on an imbalanced battery, the constraint would come from the cell with the highest SOC, increasing the chance of overcharging [13]. For the above mentioned reasons, battery balancing becomes pivotal to maximize usable capacity and avoid safety issues, especially in delicate systems such as vehicles. Two basic approaches can be put in place to implement cell balancing:

- **Passive Balancing:** a passive algorithm essentially consists in converting the excessive energy stored in the higher energy cells into heat, by means of shunting resistors. The way this strategy is implemented is pretty simple and straightforward and can be better understood by looking at Fig 2.3: every cell in the pack has its voltage (and hence its SOC), which can easily be monitored by the BMS. When a cell voltage reaches a given threshold, the balancing mechanism is triggered and starts controlling the opening of a switching device, such as a MOSFET, that allows current to be drawn from the imbalanced cell, causing a dissipation of energy in the form of heat across a resistor [13].

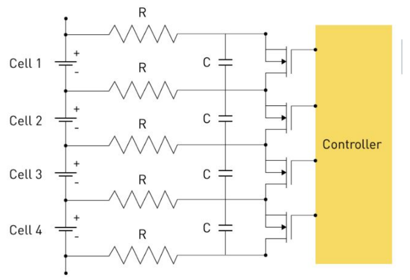


Figure 2.3. Passive Cell Balancing schematic

This solution achieves the desired balancing effect and maintains all cells at the same SOC, while also allowing manufacturers to invest very little money into the implementation, as it is the cheapest solution available in the market. The low cost and easy implementation, however, come at the price of an intrinsic low thermal management, as burning off energy through resistors generates a big amount of heat, and a massive amount of energy loss that results in low transmission efficiency [11].

- **Active Balancing:** active balancing entails moving electrical charges from higher charged cells to lower charged cells, by means of switching devices and reactive components such as capacitors or inductors. An active solution does not squander excess energy like passive balancing, but rather transfers energy throughout the battery and distributes charges in such a way that imbalances are avoided. Fig 2.4 shows the schematic of a practical implementation: first, the central inductor is charged by closing one switch and letting current flow from the higher charged cell; secondly the open switch is closed and the other is opened, allowing the inductor to make the charge flow through the lower charge cell [13].

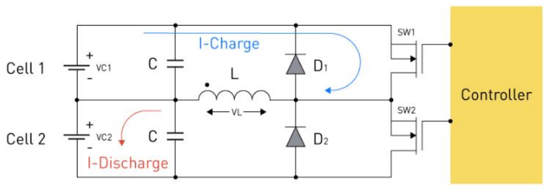


Figure 2.4. Active Cell Balancing schematic

An active algorithm offers great performance when the battery pack contains multiple cells with varying capacities and improves the efficiency because the energy is transferred among cells rather than dissipated; it also enhances the cell life expectancy and reaches balance much quicker compared to a simple passive mechanism. On the other hand, however, energy is unidirectional, as it can only move from higher to lower charged cells, and an expensive production cost must be taken care of by the manufacturer, in order to implement the essential power electronics interface [11].

2.2.2 SoC estimation methods

The first step to fully comprehend the importance of the SOC estimation is to consider the formal definition of said parameter: the State Of Charge of a battery is the available and disposable electrical charge at a certain time instant, expressed in terms of percentage with respect to the nominal capacity:

$$SoC = \frac{Q_0(t)}{Q_{NOM}} \tag{2.1}$$

In terms of analytical formulation, equation 2.1 provides one of the possible mathematical descriptions, but it is perhaps the most straight-forward and easy to understand; in fact, the SOC is hereby defined as the ratio between the residual charge Q_0 at a time instant t (which can therefore be provided by the battery to an external electrical load) and the nominal capacity Q_{NOM}, which represents the maximum value of electrical charge that can be stored in the device. It is also worth mentioning that for better accuracy and precision, the aging phenomenon should be taken into account, as in real life applications the maximum storageable charge gradually decreases, making Q_{NOM} a function of the battery lifetime. Because the SOC cannot be directly measured with a sensor, it needs to be indirectly estimated via estimation methods that allow to recover an SOC value whose accuracy and complexity depend on the method itself. As extensively discussed in [10], the scientific literature provides a variety of different strategies for said purpose, hence for the sake of making the presentation more orderly, they will be divided into five classes:

- **Conventional**
- **Adaptive Filter Algorithms**
- **Learning Algorithms**
- **Non-linear Observers**
- **Others**

To give a full overview of the existing methods that can be implemented, brief descriptions will be provided.

- **Conventional:** conventional algorithms compute an estimate by manipulating the battery physical characteristics, such as voltage, discharge current, temperature, resistance and impedance.

  1. **Open Circuit Voltage (OCV):** every battery exhibits a specific correlation between its OCV, which depends on the technology adopted for cell-manufacturing, and SOC. For example, a battery with lead-acid cells displays a quasi-linear relation between OCV and SOC whereas lithium-ion batteries have a strong non-linear characteristic. The intrinsic limitation of this method is given by the fact that cells need to reach an equilibrium state before measuring a stable OCV value, and the duration of the transient (which is a function of both temperature and SOC)

is not compatible with real-time applications. Moreover, the hysteresis phenomenon can be observed when considering the measured OCV for a given value of SOC during charging and discharging cycles: higher OCV when charging and lower when discharging.

2. **Coulomb Counting (CC):** the following method is the easiest to be implemented and requires the lowest computational power. It simply consists in applying the mathematical formula for the SOC, by means of integrating the current that flows in or out of the battery according to the equation
$$SoC(t) = SoC(t_0) - \eta \int_{t_0}^{t} \frac{i_L(\tau)}{Q_{NOM}}\, d\tau \tag{2.2}$$
where SOC(t) is the state of charge at time instant t, SOC(t_0) is the state of charge at the initial time instant t_0, η is the efficiency during charge and discharge operations, i_L is the measured current and Q_{NOM} is the nominal capacity. Despite its perks, Coulomb Counting suffers from many disadvantages:
   - the algorithm is in open-loop and is therefore subject to noise, external disturbances and temperature;
   - the final SOC estimate depends on the initial value SOC(t_0), which, if not correctly estimated, generates a cumulative error;
   - the result is strongly dependent on the current sensor measurement.
3. **Internal resistance method:** the internal resistance of a battery is a variable parameter that changes according to the value of SOC. For this reason, the internal resistance can be computed by dividing the internal voltage drop and the measured current, in order to recover an estimate for the SOC. Although easy on a conceptual level, this method is prone to errors which produce a very poor estimate:
   - the internal resistance is in the order of mΩ, which makes measurements with high level of accuracy hard to achieve;
   - the characteristic is highly non-linear, with a small variation of resistance for a broad range of SOC.

   For the above mentioned reasons, this method is rarely implemented.
4. **Electro Motive Force (EMF) method:** a battery EMF is the total voltage drop generated by the battery itself and is the sum of the electrical potential difference on the load (also known as terminal voltage) and on the internal resistance (it models the work per unit of charge that is carried out by the battery to move electrical charges). Such physical quantity can be exploited to estimate the SOC. A common practice to do so is to model the so called OCV relaxation process, which is represented in the graph of Figure 2.5.

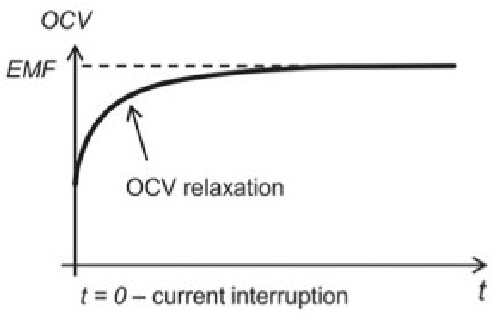


Figure 2.5. OCV relaxation process

This phenomenon occurs when the current in the circuit is suddenly cut off and the OCV starts asymptotically tending to the EMF, in a first order system like fashion. When the transient is extinct and upon reaching the equilibrium state, $OCV = EMF$ and the SOC can be retrieved from it. Despite being easy to implement, this method requires processing times that are much longer than what it is needed for real time estimation algorithms and forecasts of relaxed OCV values are usually quite inaccurate. To adequately implement EMF, more complex algorithms must be integrated.

5. **Electrochemical Impedance Spectroscopy (EIS):** EIS is a non-destructive technique that allows to recover information on the components of a system while preserving its integrity. Generally speaking, EIS can be used to safely operate a procedure that determines the degree of degradation of a battery, but, for the purpose of this discussion, we will only evaluate how these results can be exploited to obtain an SOC estimate. The complexity of a battery equivalent electric circuit can vary depending on the level of accuracy that is needed from the model or on the dynamics that are under study; nevertheless, all models usually include reactive components, thus making the battery a device with multiple components that operate on different time scales. When employing EIS, one can estimate the complex impedance of different parts of a battery, by applying a small *ac* signal over a wide frequency range. This result can then be linked to the SOC of the battery by simply using the relationship between impedance and state of charge. For more info on the subject read [14] and [15].

- **Adaptive Filter Algorithms:** algorithms that take advantage of normal or adaptive filtering techniques are widely employed in the automotive field, as they can be easily implemented by building a mathematical model of a physical system and taking advantage of its state-space representation. Kalman Filters are well suited for this purpose:

1. **Kalman Filter (KF):** it can be used to estimate the SOC of a linear system and it does so by means of filtering the effects of process and measurement errors. A KF is a recursive algorithm that continuously operates between two phases: the first is the so called **prediction phase**, which produces an SOC estimate for the future time step (the system is discretized so we can reason in terms of time steps) that takes into account the past forecasts, while the second is the **correction phase**, which exploits the value obtained in the previous step and computes a filtered value for the SOC of the same time step, by also exploiting the output value directly measured from the physical system. The **correction**, which ends up being a weighted result between the system state estimate and output, is dependent on the gain K, a value that is computed at every iteration by means of minimizing the state error covariance matrix P. A normal KF converges to the real state of the system only if the physical system is linear.
2. **Extended Kalman Filter (EKF):** when the battery mathematical model is non-linear, it is necessary to take advantage of the EKF, a more powerful and demanding filtering algorithm. The system is linearized around given points by computing the first-order Taylor expansion of both the state and output equations; in such a way, the linearized equations can then be rearranged in matrix form by building the Jacobian matrices, so that the new state-space form shapes up to a more familiar and common mathematical representation. This algorithm is therefore executed by following the same phases of a normal KF but, for every iteration, the computation of the Jacobian matrices, which makes the EKF more demanding in terms of computational power, becomes necessary to linearize the system. An EKF diverges if the physical system is highly non-linear and the first-order Taylor expansion is a too stringent approximation, insufficient to correctly capture the real dynamics of the system.
3. **Unscented Kalman Filter (UKF):** for highly non-linear systems, UKFs can be implemented. An arbitrary number of points called *sigma points* is chosen to sample the probability density function of the system state, in such a way that mean value and variance are preserved. Such approach allows to avoid explicitly computing the Jacobian matrix and to improve accuracy levels respect to the EKF; on the other hand, UKFs suffer from poor robustness given by uncertainties in the system modelling and noise that acts upon the system.

More detailed information can be found in [10] and [16].

- **Learning Algorithms:** learning algorithms take advantage of machine learning methods and thus require a large amount of data to train models that are adequate enough to capture the non-linear characteristics of a battery and to correctly estimate the SOC [10]. Many techniques can be listed and put in place to achieve said task, however, because the software development of this work will not revolve around learning algorithms, only Neural Networks will be briefly mentioned and discussed.

  1. **Neural Networks (NN):** in machine learning, a NN is a mathematical model whose structure and functionality are heavily inspired by the behaviour of single neurons and their connections inside animal brains. Each neuron, which represents the elementary unit of the model, belongs to a certain layer along with a set of other neurons, and receives a signal coming from neighbouring neurons to which it is connected; upon receiving a signal, all units elaborate an output signal according to their activation function [17]. In the context of SOC estimation methods, NNs use train data to obtain a prediction, with the advantage of having to know neither the internal structure of the battery nor the initial SOC. The overall structure (Figure 2.6) can be divided into three or more layers, including the input layer (which takes terminal voltage, temperature, discharge current and other typical physical quantities provided by a battery as input signals), the output layer (which provides the SOC estimate) and one or more hidden layers. NNs represent a great tool to be used in this field, as they allow to work in non-linear battery conditions; however, said feature is paid at the cost of needing a large training set storage ([10]).

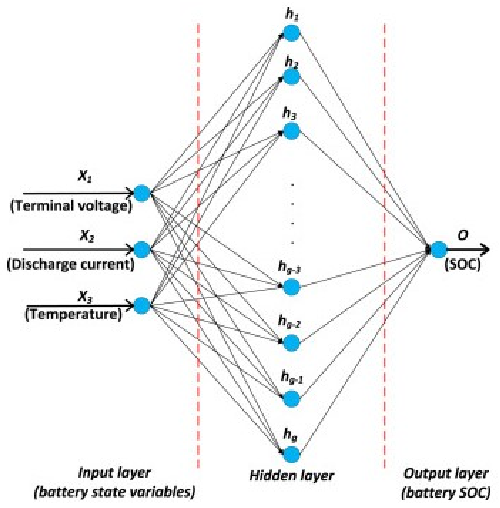


Figure 2.6. Neural Network structure

- **Non-linear Observers:** Non-linear Observers are designed to handle highly non-linear systems.

- **Others:** to the last category belong all the algorithms that do not fit into the previously mentioned categories. The other methods include BI, which takes advantage of two linear interpolations, IR, which exploits linear time invariant systems, and MARS, which uses an extended linear model. For further details, readers may consult [10].

2.2.3 SoH estimation methods

The SOH is a parameter of paramount importance for any energy storage system because it provides information on the battery life and its ageing. Due to a progressive loss of capacity, batteries are capable of storing a limited amount of electrical charge, that decreases as time goes by [18]. The mathematical description of the SOH is

$$SoH = \frac{Q_{MAX}(t)}{Q_{NOM}} \tag{2.3}$$

where Q_{NOM} is the nominal capacity of the battery and $Q_{MAX}(t)$ is the maximum charge that the battery can store at time instant t. As simple as Equation 2.3 might seem, it does not explicit the plethora of parameters on which SOH depends; many studies, in fact, underline an endless list of factors that influence the SOH, among which it is possible to find:

- **Charge/discharge cycles:** the higher the number of charge/discharge cycles the battery undergoes, the lower is the SOH, making the relationship between the former and the last inversely proportional.

- **Overcharging:** upon reaching a fully charged state, the potential across anode and cathode reaches the maximum limit; overcharging the battery causes the potential to overcome the threshold given by the physical limitations of the system and breaks down the electrolyte. As a result, the capacity decreases while the internal resistance increases.

- **Ageing:** the chemical compounds inside the battery electrochemical cells start decreasing as time elapses, which results in a reduced effectiveness of the internal reactions. This phenomenon, along with the formation of a layer over the electrode that prevents it from instantaneously reacting with the electrolyte when it is needed, generates an electric potential barrier to be overcome before the device can provide energy to the external load. This results in a loss of capacity.

- **Maximum amplitude of current pulses:** when withdrawing energy from other external devices, batteries can only tolerate so much current, as the physical limitations of the device impose a maximum and limited value of current (inrush current). If such threshold were to be exceeded, the battery would likely breakdown or suffer from severe damage, causing the State Of Health to worsen.

- **Temperature:** batteries must be operated where influencing parameters are inside an acceptable range, and temperature is no different. For this reason, operating a battery for long periods of time under abnormal temperatures (above or under the maximum or minimum temperatures respectively) can negatively influence the SOH.

In conclusion, it is clear that retrieving an accurate estimate for the State Of Health of a battery is no easy task, as the aging of the cells depends on multiple factors and parameters. The scientific literature offers a multitude of articles and studies that dig deep in this matter: [19] and [20] are both an explicit display of how unused time, number of discharge cycles and temperature are only some of the parameters that influence the aging phenomenon, while [21] and [22] underline the existing dependence between the modelling components of a cell (internal resistance and RC capacitance respectively) and the SOH value, and thus propose estimation techniques for said parameters. Generally speaking, SOH estimation methods are built on mathematical models that take into account the aging phenomenon and, depending on how they are designed, they can fit into two categories:

- **Open-loop models:** the degradation of the capacity and the increase in internal resistance is expected to be provided to the model.

- **Feedback model:** by integrating an estimation algorithm for the internal battery parameters to the already present battery model, an online calculation is performed and feedbacked.

2.2.4 Thermal Management solutions

A thermal management system keeps batteries in safe and efficient operating conditions by regulating the temperature [23]. As time goes by, consumers' needs grow and therefore engineers improve battery energy capacity by stacking an ever increasing number of cells inside a pack, which results in a higher rate of heat generation; it is essential to remove this heat and maintain the battery inside a safe temperature range, otherwise battery failure or phenomena such as thermal runaway might occur [12]. Today's commercial EVs are extremely sensitive to temperature, which can impact the vehicle performance: if carefully controlled, this physical quantity can improve range, charge time, cycle life and voltage efficiency.

To better understand a battery behaviour under different temperature conditions, a brief description of two scenarios is hereby provided:

- Low temperatures slow down the internal chemical reactions and raise the internal resistance, which result in a reduction of the pack power capabilities and charge/discharge capacity. To put some numbers into context, at 0 °C, or lower temperatures, some batteries may stop functioning and below -20 °C they can even undergo irreversible damage.
- High temperatures can lead to the reduction of active material, thus loss of capacity, and increase in internal resistance, thus contributing in power loss. In worst case scenarios, when the temperature reaches values above 60 °C, thermal runaway can potentially trigger explosion and self-ignition [24].

A Battery Thermal Management System controls the operating temperature (ideally it needs to oscillate in the range of 20 °C to 45 °C regardless of ambient temperature), by providing or dissipating heat, depending on whether the pack is too hot or cold. This mechanism is put in place by means of active, passive or hybrid heat transfer solutions [23]:

- **Active solutions:** active solutions require suitable hardware devices such as fans or pumps, which push a fluid (air, water, etc.) through the system to regulate the temperature.
- **Passive solutions:** for passive solutions, a network of pipes or heat sinks are employed along with thermally conductive materials to transfer heat away from the battery.
- **Hybrid solutions:** a combination of the above mentioned solutions is commonly defined hybrid, because it mixes and combines key features of both active and passive mechanisms.

2.2.5 Fault Detection and Protection

A key component in the Battery Management System is the capability of detecting faults and triggering control algorithms to counteract the causes of said faults [25]. These potentially hazardous circumstances affect the battery operational state and prevent it from working in a functioning and safe operational range, as they are the result of various phenomena such as shock and collision, deformation, vibration, formation of a solid electrolyte interface, etc. [26]. Similarly to [25], this thesis will now briefly present some of the causes and mechanisms that lead to faulty operational conditions in an organized and concise manner; for a deeper analysis, do read [27]. For the sake of making order, we can categorize faults in external and internal faults:

- **Internal Battery Faults:** when the fault resides within a cell, it is often difficult to detect its source, as the internal functioning is not yet fully understood. Among the various internal faults, we can mention overcharge, overdischarge, overheating and thermal runaway. Even though more detailed information will shortly be given in the following section, it is worth mentioning that thermal runaway most definitely represents the greatest hazard, since it can significantly damage the battery application or, even worse, cause physical harm to the user [28].

    1. **Overcharge:** overcharge can be caused by capacity variation of cells, incorrect current or voltage measurements, inaccurate SOC estimation and, additionally, when the charger connected to the battery is damaged or breaks down. A sustained overcharge can even lead to more severe faults such as thermal runaway or accelerated degradation.
    2. **Overdischarge:** equivalently to overcharging, overdischarging can be triggered by inaccurate SOC estimation and inaccurate current or voltage measurements. As stated in [29], Electrochemical Impedance Spectroscopy suggests that, during overdischarge, a battery anode undergoes a much larger change in Solid Electrolyte Interface (formation of a layer of decomposed material associated with the electrolyte) compared to the cathode, which results in a smaller anode impedance thus a loss in capacity. Moreover, lifespan and thermal stability of a Li-ion cell can be compromised.
    3. **Overheating:** an overheating fault can be caused by several different circumstances: a hardware malfunction, such as a high amount of voltage sent back to the battery after alternator voltage regulator fails, and internal or external short circuits. Significant capacity loss can be observed along with the potential occurrence of thermal runaway [25].
    4. **Thermal Runaway:** all the aforementioned faults can lead up to thermal runaway, a phenomenon so dangerous that often causes the ignition and explosion of the battery. When the temperature reaches the melting point of the metal that constitutes the cell (for instance lithium in the case of Li-ion), a violent reaction is started: the formation of heat occurs at a much higher rate than its dissipation and, as a consequence, an uncontrolled increase in temperature takes place, besides a substantial increase in pressure that leads to a large amount of flammable and toxic gas being released. Another cause of thermal runaway is restricted air circulation and it has also been observed that the probability of triggering the runaway reaction is directly proportional to the number of charge/discharge cycles [25].

- **External Battery Faults:** when the source of the fault comes from the "outside" rather than the cell and its working principle, the fault is said to be external: it can either be related to faulty temperature, voltage and current sensors or to damaged cell connections and cooling systems. The effects of external faults can be so significant that other BMS functions can be affected and internal battery faults can occur.

1. **Sensor Fault:** when it comes to Battery Management Systems, having a reliable sensor fault diagnostic scheme is imperative to ensure safety: temperature sensor faults can send incorrect data to the BMS, which therefore inefficiently manages the thermal function, needed to keep the battery in safe operating conditions. Moreover, short-circuiting, aging, capacity fade, overheating and thermal runaway can all be a direct consequence of a malfunctioning temperature sensor. As far as voltage sensors are concerned, under normal working conditions, they are employed to measure and monitor cell voltages, which could likely overcome the upper and lower limits specified by the manufacturers, if this type of fault were to manifest. A voltage sensor fault can also result in overcharge, overdischarge and inaccurate SOC and SOH estimation. The last type of sensor fault regards the current, which must be constantly monitored both upon entering or exiting cells. In the case of a current sensor fault, current can bypass the sensor, leading to inaccurate BMS control actions, overcharge, overdischarge or overheat [25].
2. **Cell Connection Fault:** when the electrical connections between cell terminals are corroded overtime or become loose due to vibrations [30], the cell resistance increases drastically while also inducing cell imbalance and overheating [25].
3. **Cooling System Fault:** keeping the battery pack inside the suitable temperature range is among the most important and delicate tasks supervised by the BMS; the cooling system, along with the temperature sensor, is responsible to transfer the heat away from the battery, by means of fans and motors that must be adequately controlled. When the aforementioned piece of equipment fails due to outdated wiring, the temperature sensor starts malfunctioning or a fuse breaks, severe consequences such as overheating or thermal runaway occur [31].

2.3 Functional Safety and ISO 26262 standard

The beginning of the 21st century represented a breakthrough in the automotive field, as manufacturers started extensively introducing electronic systems and software solutions into their commercial vehicles: among the first to start out was Toyota, which adopted an electronic throttle control system in the year 2000. The implementation of programmable electronic control units allowed to achieve a higher degree of functionality along with more flexibility, which means (still up to this day) being able to reprogram a system behaviour by simply writing custom and suitable lines of code, without having to replace the dedicated hardware. Increase of technological complexity and software content, however, comes with risks from systematic and random hardware failures: software is in fact complex and can introduce defects that lead to particularly dangerous and unsafe situations. This evidence brought the necessity to avoid risks and develop safe system processes along with a satisfactory set of evidence that proves all reasonable system safety objectives are met. For the aforementioned reasons, applying a shared set of rules and norms becomes necessary to have car manufacturers follow the same safety procedures. Because standards can often be extremely detailed and hard to follow, a preliminary explanation of technical terminology will be briefly presented:

- **Functional Safety:** it is part of the overall safety of a system that depends on automatic protection operating correctly in response to its inputs or failure in a predictable manner; it is the way to determine the risk of using complex and simple electronic circuit to perform a safety function. The safety function must always be performed under undisturbed and fault conditions. Functional Safety is achieved when there is the absence of unreasonable risk due to hazards caused by the malfunctioning of Electrical/Electronic (E/E) systems.
- **Item:** system, array of systems or a function to which the standard is applied.
- **Harm:** physical injury or damage to the health of people.
- **Severity:** measure of the extent of harm to an individual.
- **Failure:** termination of the ability of an element or an item to perform a function as required.
- **Risk:** combination of the probability of occurrence of harm and the severity of said harm.
- **Controllability:** avoidance of the specified harm or damage through the timely reaction of the people involved.
- **Exposure:** state of being in an operational situation that can be hazardous if coincident with the failure mode under analysis.
- **Automotive Safety Integrity Level (ASIL):** one of the four levels to specify the item necessary requirements of the standard and safety measures for avoiding an unreasonable residual risk, with D representing the most stringent and A the least one.
- **Hazard:** potential source of harm.

- **Hazard Analysis and Risk Assessment (HARA):** method to identify and categorize hazardous events of items and to specify safety goals and ASILs related to the prevention or mitigation of these hazards in order to avoid unreasonable risk.

- **Safety:** absence of unreasonable risk.

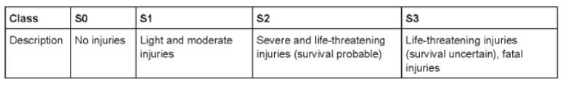
- **Safety Goal:** top level safety requirement as a result of the HARA.

The international standard specific for the automotive industry, that applies to safety-related road vehicle E/E systems, is the ISO26262. It provides requirements for the whole lifecycle of the E/E system and it details how the system has to be conceived, starting from the documents that have to be produced up to the design activities that must be carried out. According the ISO26262, the E/E system development revolves around the risk for the customer, which is identified with an ASIL depending on the level of safety measures required to avoid unreasonable risk. The main focus is therefore the functional safety and the objective is to free people's health from unacceptable risk of physical injury, either directly or indirectly. When a new E/E system has to be embedded in a road vehicle, said system must be compliant with the standard before being put on the market, hence a series of steps must be followed:

- **Item definition:** it describes the item in its entirety, its elements and interactions with other items, it analyzes the functionality provided to and required from other items and the environment.

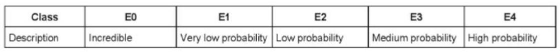
- **HARA:** as previously stated in the paragraph, the Hazard Analysis and Risk Assessment identifies and categorizes hazards derived from item malfunctions and promises to formulate safety goals through a series of intermediate steps.

  1. **Situational analysis:** it describes operating modes and specific situations in which malfunction results in hazardous events. For instance, said situations could be driving at low speed, at high speed, on icy roads, on wet roads, etc.
  2. **Hazard identification and classification:** the first phase consists in systematically identifying hazards by means of appropriate techniques (brainstorming, checklists, FMEA, quality history, field studies) and their respective consequences, while the second consists in classifying and fitting each hazardous event into one ASIL level, based on a score given by three parameters: severity, exposure and controllability.



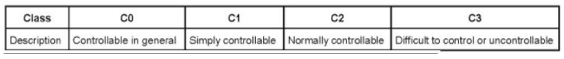
Class	S0	S1	S2	S3
Description	No injuries	Light and moderate injuries	Severe and life-threatening injuries (survival probable)	Life-threatening injuries (survival uncertain), fatal injuries

Figure 2.7. Severity measure of extent



Class	E0	E1	E2	E3	E4
Description	Incredible	Very low probability	Low probability	Medium probability	High probability

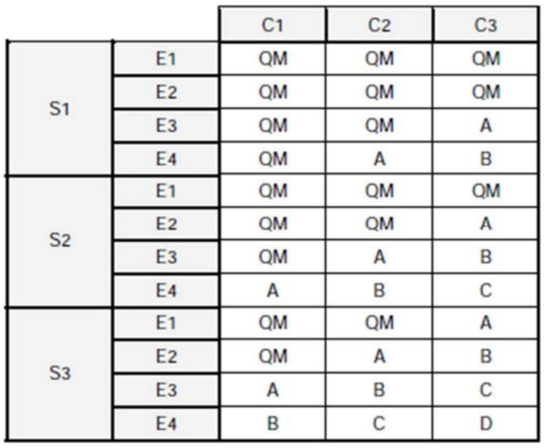
Figure 2.8. Exposure measure of extent



Class	C0	C1	C2	C3
Description	Controllable in general	Simply controllable	Normally controllable	Difficult to control or uncontrollable

Figure 2.9. Controllability measure of extent

As shown in Figures 2.7, 2.8, 2.9, depending on the specific situation under analysis, each parameter is given a score and the ASIL level A ÷ D is finally retrieved according to the criterion shown in Fig 2.10. It is worth mentioning that such table seems to be introducing the level QM, which stands for Quality Management, however, in reality, this label refers to a scenario where the assessed risks can be tolerated from a safety point of view, thus making quality management processes sufficient.



		C1	C2	C3
S1	E1	QM	QM	QM
	E2	QM	QM	QM
	E3	QM	QM	A
	E4	QM	A	B
S2	E1	QM	QM	QM
	E2	QM	QM	A
	E3	QM	A	B
	E4	A	B	C
S3	E1	QM	QM	A
	E2	QM	A	B
	E3	A	B	C
	E4	B	C	D

Figure 2.10. ASIL

3. **Safety goal determination:** determine a safety goal for each hazardous event that has already been labeled with ASIL. It is important to specify

that safety goals are not expressed in terms of technological solutions but in terms of functional objectives.

- **Functional safety requirements:** they are derived from safety requirements and address safety mechanisms, fault detection, failure mitigation and the concept of redundancy.

Following these steps is the implementation phase, which branches out into a path foreseen for hardware development and another for software. For each implementation path, specific prescriptions are given about methods to be used and measures to be collected. This thesis does not intend to go into details regarding ISO 26262, as standards can be very complex, but readers can in fact refer to the ISO for more information.

Chapter 3

Software Development

The following chapter will be the heart of this thesis, as the work that has been carried out to develop the BMS software (which is based on the already discussed numerous state of the art techniques) will be meticulously illustrated and analyzed. Moreover, in order to fully cover all the software functionalities, each important part of the software will be discussed in a dedicated stand-alone section.

3.1 Battery Pack

No BMS software can ever exist without a battery pack to monitor and control. For this reason, the first step to be taken during the development of this software was to build an appropriate model for a battery that could seemingly power an EV. When modelling a mathematical and virtual copy of a battery (or any other physical object) it is fundamental to find the balance between the model complexity, as said model must replicate the physical system behaviour to a certain degree of accuracy, and the computational power required to run it on a given hardware. Depending on the requirements and the tools available for each specific case, many different solutions can be found: when high accuracy levels are required, the model complexity can be arbitrarily increased as long as the equipment at the developers' disposal is powerful enough to run the algorithm; on the contrary, when optimizing the execution time becomes pivotal, model complexity must be sacrificed to make the algorithm less demanding in terms of computational power. Modern real-life battery packs are made up of a variable number of cells, which clearly depends on the type of application they are designed for; in the automotive industry, packs are demanded high electrical performances, especially in the context of hybrid or electric vehicles, where battery packs are composed of a number of cells that can vary from a few hundreds ([32], [33]) to even thousands [34]. An exact software replica with such features could not be developed, as the hardware used to complete this work (a simple laptop) was not suitable for such high demanding computations.

Hence, the model proposed in this thesis is heavily inspired on real-life electrochemical cells and their electrical features, but the overall battery pack is appropriately scaled down to comply with the hardware capabilities. The pack was built in a Matlab environment by exploiting Battery Builder, a free tool available with the Simscape Battery add-on.

Figure 3.1. Samsung SDI 3.7V 94Ah NMC Prismatic Battery Cell

Figure 3.1 shows the Samsung SDI battery cell, an energy storage system often employed for EVs [35], that was reproduced using Battery Builder. This lithium-ion cell is manufactured in a prismatic geometry, it has a high energy density, high nominal capacity and an internal resistance under $1m\Omega$, and it displays no self discharging effect. Because describing the features of this cell is outside the scope of this thesis, Table 3.1 summarizes its most important technical characteristics.

Feature	**Specification**
Technology	Lithium-ion
Nominal Capacity	94 Ah
Nominal Voltage	3.68 V
Internal Resistance	<= 0.75 $m\Omega$
Energy	345 Wh
Battery Weight	2.1 kg
Battery Dimensions	173*125*45 mm

Table 3.1. Samsung SDI cell technical information

The virtual cell recreated in Battery Builder is almost identical to the Samsung SDI cell, as demonstrated in Table 3.2: the geometry is prismatic and the dimensions are perfectly equivalent, the nominal capacity is 94 Ah and the mass equals 2.1 kg. The only significant difference in terms of electrical features is given by a slight increase in energy (345.9 Wh), which was deemed necessary in order to reach a reasonable value of total energy for a pack with a reduced number of cells, as it will shortly be explained. Moreover, the software does not allow to pick and choose the technology the cell is made of (e.g. Lithium-ion), as the electrochemical reactions occurring inside the cell are not of interest, but we rather focus on the electrical behaviour of the system.

Feature	**Specification**
Nominal Capacity	94 Ah
Energy	345.9 Wh
Battery Weight	2.1 kg
Battery Dimensions	173*125*45 mm

Table 3.2. Custom cell technical information

Some of the other options that were enabled and tweaked in the cell model menu are:

- **T_dependence:** some of the battery pack parameters express a dependency on temperature.
- **Prm_age_capacity:** capacity calendar aging is enabled and can be later exploited to run more realistic simulations.
- **Prm_age_modeling:** this property allows the user to select how to mathematically model the internal resistance and capacity aging; for this specific work, this parameter was set to "equations".
- **Prm_age_resistance:** internal resistance calendar aging is enabled and can be later exploited to run more realistic simulations.
- **Prm_fade:** by setting this parameter to "equations", fading phenomena regarding open-circuit voltage, resistance and capacitance will be enabled and modeled by means of suitable equations.

$$v_{0,fade} = v_0(1 - \frac{\delta_{v_0}}{100}\frac{n}{N}) \tag{3.1}$$

Equation 3.1 models the variation of open-circuit voltage across the battery model, which is directly proportional to the constant nominal open-circuit voltage v_0 and fades proportionally with the number of discharge cycles n; δ_{v_0} represents the percent variation of v_0 after N discharge cycles. The charge used for state of charge estimation $q_{nom,fade}$ is, similarly, directly proportional to the constant nominal charge q_{nom} but depends instead on the square root of n, as demonstrated by Equation 3.2.

$$q_{nom,fade} = q_{nom}(1 - \frac{\delta_{AH}}{100}\sqrt{\frac{n}{N}}) \tag{3.2}$$

The same dependencies on R_i and n can be found in the resistance fading $R_{i,fade}$, shown in Equation 3.3.

$$R_{i,fade} = R_i(1 - \frac{\delta_{R_i}}{100}\sqrt{\frac{n}{N}}) \tag{3.3}$$

A more detailed insight on the mathematical description of the battery model can be found in [36].

- **Thermal_port:** the generated Simulink block displays an ambient port, which can be exploited to have the system thermally interact with the environment.

The cell is reported in Figure 3.2.

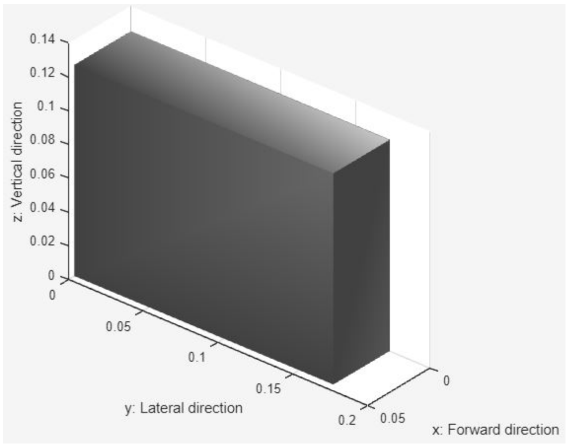


Figure 3.2. Custom battery cell

The custom battery pack is then assembled by combining the cells into hierarchically ordered sequential structures:

- **Parallel Assembly:** a parallel assembly is a block made up of an arbitrary number of cells connected in parallel. During the design phase, it is essential to remember that in order to achieve the desired electrical characteristics of the pack, at the parallel assembly level, one can only obtain an increase in capacity: according to the Kirchhoff laws, the equivalent capacity is in fact the sum of the single cell capacities inside the assembly, whereas the voltage drop across it remains unchanged. For this specific software application, four cells were selected to significantly improve the electrical capacity and the overall energy, which (as it is shown in Table 3.3) have grown respectively to 376 Ah and 1383.68 Wh, at the cost of increasing the cumulative mass up to 8.4 kg.

Number of cells	4
Cumulative mass	8.4 kg
Cumulative cell capacity	376 Ah
Cumulative cell energy	1383.68 Wh

Table 3.3. Custom parallel assembly technical information

To further characterize the model, some of the additional options were enabled:

1. **Model Resolution:** this parameter was set to "Detailed" to achieve finer data; during simulation, each cell is in fact considered as an individual electrical entity to be simulated, rather than having the entire parallel assembly simulated as a single equivalent cell. This choice benefits users as they can gather more detailed information from simulations, but a heavier computational effort is required.
2. **Balancing Strategy:** being set to "Passive", this option exposes an input port in the generated Simulink block, which is meant for cell balancing purposes.
3. **Coolant Thermal Path:** because the assembly can also be thermally characterized, it is possible to select the coolant thermal path, which, for the work carried out in this thesis, was set to "Cell Based Thermal Resistance".
4. **Ambient Thermal Path:** similarly to the coolant, the ambient thermal path was set to "Cell Based Thermal Path".
5. **Cooling Plate:** to conclude the thermal aspect of the assembly, Battery Builder allows developers to integrate and position a cooling plate either on top, at the bottom or even on both surfaces. The cooling plate for this custom parallel assembly was positioned at the bottom.

The custom parallel assembly is shown in Figure 3.3.

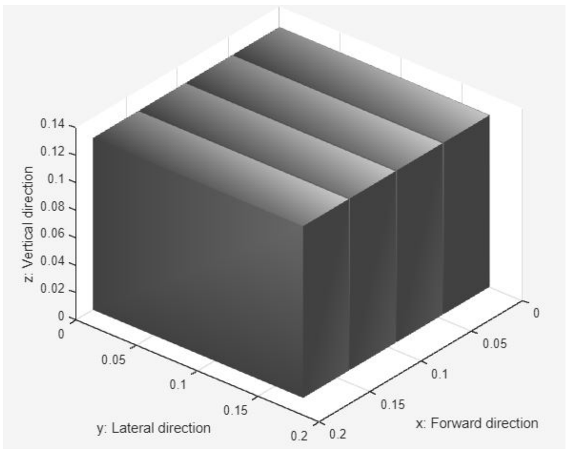


Figure 3.3. Custom battery parallel assembly

- **Module:** a module is a serial connection of parallel assemblies, characterized by an overall cell capacity equivalent to one assembly and an equivalent voltage drop given by the sum of the voltages. The custom module is composed of 8 parallel assemblies, for a total of 32 cells, and, as it is reported in Table 3.4, the updated mass is 67.2 kg and the total energy is 11069.44 Wh. Moreover, a comparison between Tables 3.4 and 3.3 shows that the cumulative capacity remains unchanged.

Number of parallel assemblies	8
Cumulative mass	67.2 kg
Cumulative cell capacity	376 Ah
Cumulative cell energy	11069.44 Wh

Table 3.4. Custom module technical information

A module, similarly to the parallel assembly, can be further characterized by arbitrarily setting the additional options:

1. **Model Resolution:** the resolution of the custom module was set to "Detailed" to gather finer simulation data.
2. **Balancing Strategy:** a passive balancing strategy was chosen for the custom module.
3. **Coolant Thermal Path:** the thermal path for the coolant was set to "Cell Based Thermal Resistance".
4. **Ambient Thermal Path:** "Cell Based Thermal Resistance" was selected as ambient thermal path.

5. **Cooling Plate:** one cooling plate was placed on the bottom surface of the module.

6. **Cooling Plate Block Path:** for the specific type of cooling plate, a geometry with u-shaped channels was selected among the alternatives made available by Battery Builder.

The graphical representation of the custom module is displayed in Figure 3.4.

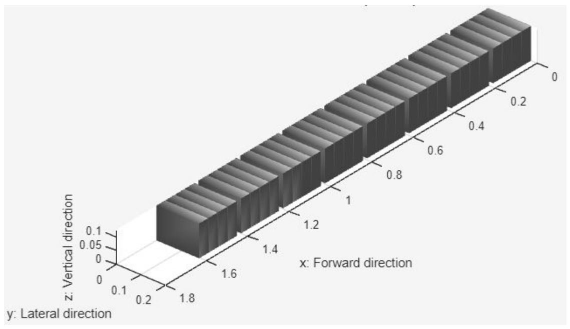


Figure 3.4. Custom battery module

- **Module Assembly:** a module assembly is an arbitrary connection of modules that can either be organized in serial or parallel topologies. In the hierarchy of battery blocks, this structure precedes the pack and helps building sets of numerous modules, which can then be easily replicated when finalizing the battery pack. Because no additional cells were needed to build the custom battery pack, the module assembly for this specific work was only made of one module, making this structure equivalent to the already defined module. It is also worth mentioning that all the additional options were left untouched with respect to what was set for the custom module. For technical information and a graphical representation of the module assembly refer to Table 3.4 and Figure 3.4.

- **Pack:** the pack sits on top of the hierarchical pyramid and represents the complete battery pack. As it was mentioned in the section relative to the module assembly, the custom battery pack is only made of one single module and, for this reason, no additional changes and modifications were made to the already defined blocks. To sum up the electrical features of the custom battery pack, Table 3.5 collects all the important technical information, while Figure 3.5 shows what the pack looks like inside Battery Builder.

Number of module assembles	1
Cumulative mass	67.2 kg
Cumulative cell capacity	376 Ah
Cumulative cell energy	11069.44 Wh

Table 3.5. Custom pack technical information

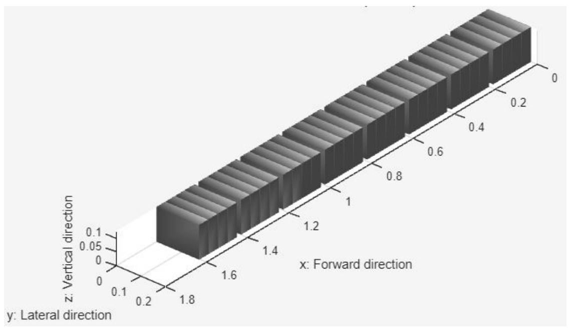


Figure 3.5. Custom battery pack

After completing the definition of the custom battery pack via graphical interface, the "Create Library" option allows the user to generate a Simulink block with all the defined features, which can be implemented in a Simulink schematic for simulation purposes. The software automatically generates three files and leaves the developer the choice of generating an additional script. The four files generated for this specific software development are:

- **filename_param.m:** the .m file is a Matlab script that gathers the parameters defined in Battery Builder. They are divided into ModuleType, ParalleAssemblyType, CoolingPlate and BatteryInitialTarget and range from cell state of charge, temperatures, change in cell capacity and change in cell resistance to initial cell current, terminal voltage and discharge cycles. This script can come in very handy because parameters can quickly be modified, without having to reopen Battery Builder, and because the generated Simulink block is already parametrized with the parameter names defined in the script.

- **filename.mat:** a second file with the .mat extension is generated and contains the values of all the pack parameters saved in the Matlab Workspace.

- **filename.slx:** the .slx file contains the generated Simulink block which represents the custom battery pack.

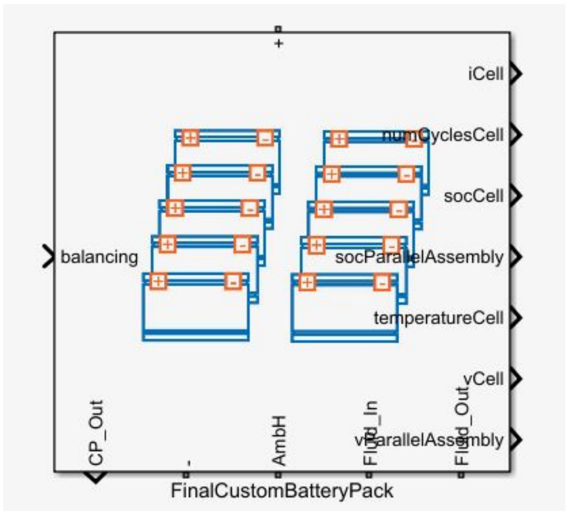


Figure 3.6. Simulink block of the custom battery pack

Upon opening the file, the block shown in Figure 3.6 appears on screen and is equipped with several input and output ports that can be exploited to monitor and control the pack during the simulation phase:

1. **iCell:** Simulink port that outputs cell currents.
2. **numCyclesCell:** Simulink port that outputs the cell number of discharge cycles as a function of time.
3. **socCell:** Simuink port that outputs the state of charge of all cells.
4. **socParallelAssembly:** Simulink port that outputs the state of charge of all parallel assemblies.
5. **temperatureCell:** Simulink port that outputs cell temperatures.
6. **vCell:** Simulink port that outputs cell voltages.
7. **Fluid_In:** Simscape input port for the coolant.
8. **Fluid_Out:** Simscape output port for the coolant.
9. **AmbH:** Simscape port that can be exploited to thermally couple the battery pack with the surrounding environment.
10. **CP_Out:** port that outputs three physical signals, the first being the temperature of the cooling plate Tp, the second being the fluid temperature change dT and the third being the pressure drop of the fluid dP [37].
11. **balancing:** Simulink input port for cell balancing purposes.
12. $+$ and **-:** positive and negative battery terminals.

- **filename_lib.slx:** this last Simulink file contains module and parallel assembly models.

To conclude the discussion on the development of the custom battery pack, it is essential to underline that, due to limited hardware capabilities, the battery model had to be downscaled to reach an acceptable compromise between model accuracy and simulation time. Nonetheless, the model as it is shows a behaviour that justifies the juxtaposition with a small electric vehicle and could potentially be upscaled to better simulate a commercial EV on the market.

3.2 Passive Cell Balancing implementation

For this thesis work, the approach taken to develop a cell balancing solution was necessarily a passive algorithm, as a fully functioning and efficient active algorithm requires the implementation of reactive electrical components (capacitors and inductors), directly and appropriately connected to the pack cells. However, because the Simulink block allows the user to only access the parallel assembly, without granting the possibility to view the cells inside, active mechanisms were put aside in favor of passive ones.

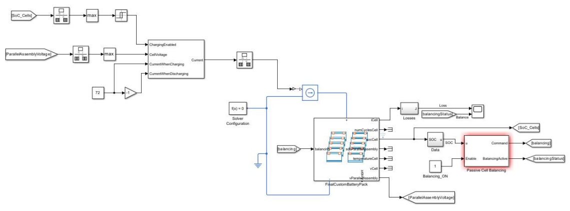


Figure 3.7. Passive Cell Balancing Simulink schematic complete overview

Figure 3.7 shows the complete overview of the Simulink schematic, which, for explanation purposes, can be divided in three sections:

- **Charging/Discharging algorithm:** on the top left corner (Figure 3.8) is a set of blocks connected to the battery, which forces the pack to charge and discharge indefinitely, according to a Constant Current-Constant Voltage (CC-CV) algorithm. Because a deeper insight into CC-CV and a proper description of all the building blocks will be provided in the next chapter, now it is sufficient to say that the CC-CV imposes a constant current phase followed by a constant voltage during charging and a constant current during discharging [38].

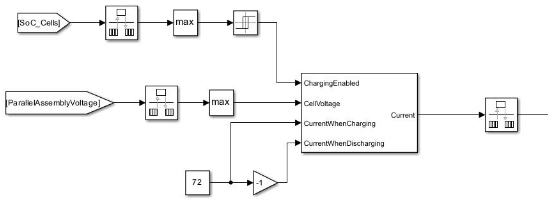


Figure 3.8. Constant Current-Constant Voltage Simulink schematic

- **Custom battery pack:** the battery pack to be tested is placed in the center of the schematic and is appropriately connected to both the CC-CV and the rest of the blocks that make up the actual cell balancing mechanism. In Figure 3.9, it is possible to observe that the battery terminals are connected to a Controlled Current Source, a block that models an ideal current generator that can be externally controlled; such block comes in fact with a physical signal input port, where the value of the current computed by the CC-CV is conveyed, and two additional ports through which it is connected to the battery terminals. It is also worth mentioning that the virtual wire that closes the circuit has two more connections, one being the necessary link to an Electrical Reference, while the other links the model to the Solver Configuration, a block employed every time the Simscape library is recalled in the schematic, which defines the solver settings to use for simulations. As far as the other battery ports are concerned, iCell, socCell and vParallelAssembly are exploited to implement cell balancing, via direct connections and a GoTo block, while the others are left untouched because irrelevant for balancing purposes (numCyclesCell, socParallelAssembly, temperatureCell and vCell are connected to terminators while AmbH is left unconnected).

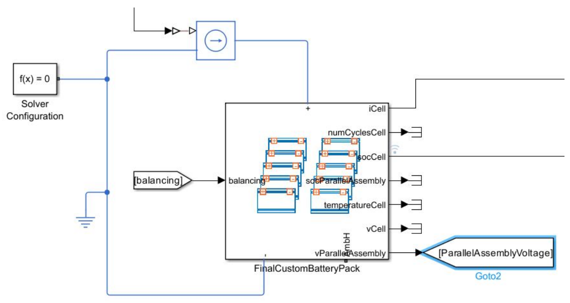


Figure 3.9. Passive Cell Balancing battery pack Simulink schematic

- **Passive Cell Balancing:** the heart of the balancing mechanism lies in the connection between the blocks placed in the right-hand side of the Simulink schematic.

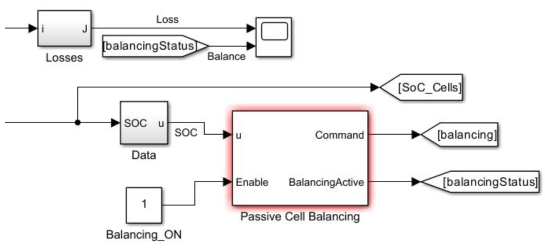


Figure 3.10. Passive Cell Balancing Simulink schematic

Figure 3.10 zooms in on such section and can be very useful to have a clear visual representation of the algorithm; the bottom branch is made up of all the necessary blocks for balancing purposes and is organized as follows:

1. **Data subsystem:** the state of charge vector is taken as input from the socCell output port of the battery pack and is conveyed to a simple Matlab function, which computes one SOC value for each parallel assembly as the average SOC of all the cells belonging to the same assembly. However, because the overall system is discrete, a Unit Delay with a zero initial condition is connected to the Matlab function, to provide a one time step delayed parallel assembly SOC.
2. **Balancing_ON constant:** a simple constant block set to 1 in order to enable the Passive Cell Balancing block to which it is connected.
3. **Passive Cell Balancing block:** this block is equipped with a **u** input port, where the accepted value can either be a cell voltage vector or an SOC vector, and an **Enable** input port, where a logic boolean signal can be set to control the balancing mechanism; in this specific implementation, upon receiving the Balancing_ON signal, the block starts balancing the battery cells until the procedure is complete and it does so by monitoring the SOC levels provided to the **u** port.

$$f(x) = \begin{cases} 0, & \text{if } u - min(u) \leq \text{Threshold} \\ 1, & \text{if } u - min(u) \geq \text{Threshold} \end{cases} \tag{3.4}$$

Equation 3.4 describes the output generated by the block at the **Command** port. The output **Command** specifies the balancing command provided to the battery through the **balancing** input port and it is expressed in the form of a vector of elements equal to 0 or 1. When one vector element is equal to 0, the difference between the corresponding cell SOC and the lowest SOC at a given time step is lower or equal to a certain threshold, while it switches to 1 as soon as this inequality is no longer true. In simple words, the algorithm discharges some of the cells until their SOC is equal or lower to the state of charge of the cell with the lowest value [39]. The second and last output port is the **BalancingActive** port, which returns a true scalar value if any cell of the battery exceeds the threshold.

The upper branch, on the contrary, does not influence the balancing algorithm, but it simply allows the user to have a visual representation of how the balancing mechanism progresses during the entire simulation time. The main components are:

1. **Losses subsystem:** the following subsystem takes cell currents and the value of the cell balancing resistance as inputs, it computes the power loss by means of a simple Matlab function and retrieves the energy loss by integrating the power through an integrator.
2. **Scope:** the scope displays the behaviour of the energy loss in time, along with the **balancingStatus**, allowing developers to estimate how long it takes to reach a fully balanced state, while still keeping an eye on the system losses.

In addition to the Simulink schematic, a Matlab script was developed to retrieve finer data during simulations: with **filename_PassiveBalancing_Simulation.m**, developers are able to set an arbitrary value for the balancing SOC threshold, along with a set of balancing resistor values, which can all be tested by simply running the code. For each of these defined values, the software will run a simulation and display resistor rating, balancing time and power loss on the Command Window. More information on this topic will be provided in the next chapter, when discussing the results obtained during the unit testing phase.

3.3 Thermal Management implementation

The second BMS function developed for this work relates to the thermal management of the battery pack, a pivotal aspect to take into consideration when dealing with energy storage systems. The goal was to build a software solution that would appropriately control a cooling system in such a way that the cell temperature would be kept inside an ideal working temperature range, in order to avoid all the issues related to a prolonged use of the battery under abnormal temperature conditions (especially thermal runaway).

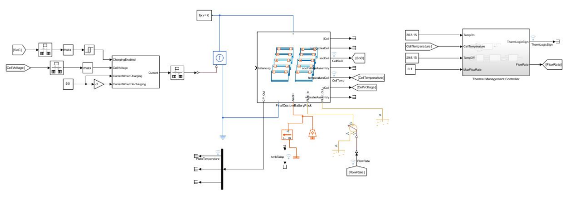


Figure 3.11. Thermal Management Simulink schematic complete overview

Figure 3.11 shows the complete Simulink schematic, which can be divided in three functional sections:

- **Charging/Discharging algorithm:** on the left-hand side there is the already mentioned Constant Current-Constant Voltage (CC-CV) algorithm, chosen to have the pack undergo consequent charging and discharging cycles during simulations.

- **Custom battery pack:** the battery pack is placed in the center of the schematic (Figure 3.12) and is subject to charging and discharging currents, by means of a current source connected to the + and - terminals and externally controlled by the CC-CV output. For thermal control purposes, the cell state of charge, cell temperature and cell voltage vectors are provided to the thermal management controller located on the right-hand side of the schematic, while the pack temperature ports are connected to a variety of different sets of Simscape blocks, as they fulfill specific tasks:

  1. **AmbH:** the ambient thermal port is connected to a temperature source, needed to impose and simulate the temperature of the surrounding environment in which the battery operates, and a temperature sensor connected to a thermal ground, to measure and visualize the ambient temperature in the simulation data.
  2. **Fluid_In:** the inlet port for the coolant needed to be connected to a block that could somehow emulate the behaviour of a fluid pump and, for this reason, a flow rate source was implemented. Said block allows the developer to arbitrarily impose a volumetric flow rate, which (in this thesis) is specified by the signal **FlowRate** coming from the thermal management controller output, applied to port **V**. In conclusion, the cooling fluid flows from the reservoir at port **A** to the battery pack connected to port **B**.
  3. **Fluid_Out:** after flowing through the cooling plate inside the battery, the fluid comes out and reaches a reservoir block, which is essential to close the circuit, thus allowing the coolant to continuously flow as long as the mechanism is active.

4. **CP_Out:** the last thermal port is exploited for simulation purposes only; the three element vector provided by this port is in fact split into single signals, by means of a demux, and can be very useful to visualize data after running simulations.

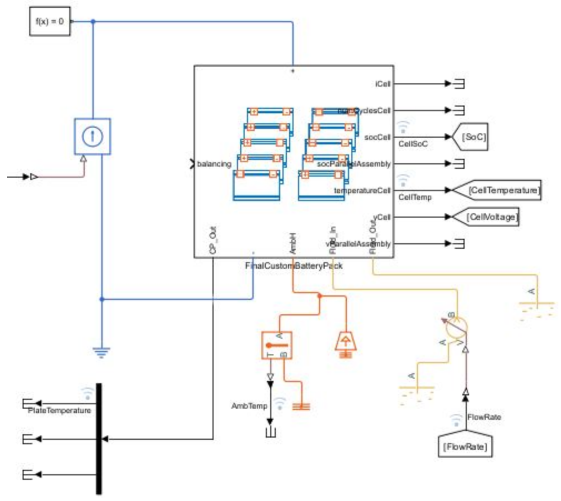


Figure 3.12. Thermal Management battery pack Simulink schematic

- **Thermal Management Controller:** on the right-hand side of the schematic is the subsystem called Thermal Management Controller (Figure 3.13), which is developed to implement a temperature-related logic, with the aim of imposing a flow rate depending on the relative value of the cell temperature, with respect to an activation temperature. The input/output ports of the subsystem are:

  1. **Cell Temperature:** the cell temperature vector comes from the battery pack and is fed to the controller.
  2. **Temp On:** an activation temperature is chosen by the user and fed to the controller.
  3. **Temp Off:** temperature at which the mechanism is deactivated.
  4. **Max Flow Rate:** arbitrary value for the flow rate that is conveyed to the block.
  5. **ThermLogicSign:** logic signal coming out of the controller, which is equal to 1 as long as the coolant flows in the battery, while it is reset to 0 by the time the cell temperature is inside the safe range.
  6. **Flow Rate:** value of flow rate computed by the controller.

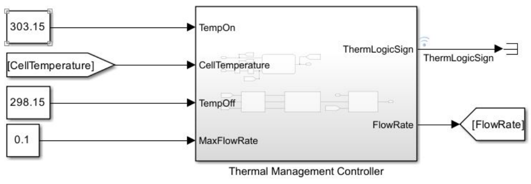


Figure 3.13. Thermal Management Controller Simulink schematic

Because the Thermal Management Controller, as shown in Figure 3.14, is a complex structure made up of several subsystems, a proper understanding of the working principle of such algorithm requires an explanation of all the building blocks.

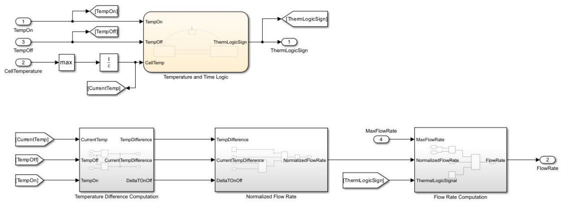
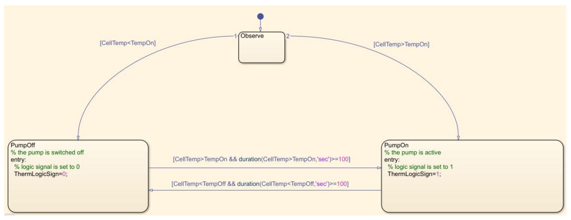


Figure 3.14. Overview of the Thermal Management Controller Simulink schematic

- **Temperature and Time Logic:** a Stateflow chart was developed to implement the fundamental activation logic behind the thermal management of the battery, which only requires the activation and deactivation temperatures and the maximum value of cell temperature.


Observe
1
2
[CellTemp<TempOn]
[CellTemp>TempOn]
[CellTemp>TempOn && duration(CellTemp>TempOn,'sec')>=100]
[CellTemp<TempOff && duration(CellTemp<TempOff,'sec')>=100]

```
PumpOff
% the pump is switched off
entry:
  % logic signal is set to 0
  ThermLogicSign=0;
```

```
PumpOn
% the pump is active
entry:
  % logic signal is set to 1
  ThermLogicSign=1;
```

Figure 3.15. Temperature and Time Logic Stateflow chart

Figure 3.15 shows how simple the structure behind this logic really is; the chart is in fact divided into three states, the first one being a default state to which the system transitions only during turn-on, while the other two characterize the system for the residual simulation time.

1. **Observe state:** when running the simulation, the system must initially be given time to determine whether the pack is operating in a safe temperature range or not, thus requiring the activation of the algorithm. For this reason, upon executing the chart, the system transitions by default to the **Observe** state and then compares the maximum cell temperature with the activation temperature TempOn, to transition to either the **PumpOff** or **PumpOn** state. By the time the system transitions from **Observe**, the default state cannot be reached for the rest of the simulation.
2. **PumpOff state:** firstly reached when the maximum cell temperature is lower than the activation temperature TempOn, the **PumpOff** system state describes a safe thermal operating condition and corresponds to a scenario where the pump is switched off. Upon entering this state, the logic signal ThermLogicSign is set to 0. After every simulation step, the software verifies the validity of the transitioning condition to the **PumpOn** state and, if true, proceeds with the transition; in this case, the transitioning condition is a set of two conditions, one related to temperature and the other related to time, linked by an AND logic operator. The thermal condition requires that the maximum cell temperature is strictly greater than TempOn, while the second requires that the first condition is true for a time interval in the order of hundreds of seconds. Only applying a temperature type logic would not be effective because the number of pump activations would be far greater than what is needed, making a real application inefficient in terms of energy consumption. That being the case, an additional time condition, which can be interpreted as a relaxation of the thermal condition, is considered; thermal phenomena are in fact slow if compared to electrical phenomena, which means that transients take much longer to reach steady-state. By letting the temperature overcome the arbitrary threshold for a time interval in the order of hundreds of seconds (typical order of magnitude for thermal time constants), the cell temperature will only slightly exceed the threshold value, but the number of pump activations will be significantly reduced.
3. **PumpOn state:** when the battery pack is operating under abnormal thermal conditions, the system is in the **PumpOn** state, which corresponds to the scenario where the coolant is flowing through the cooling plate and exchanging heat with the battery to bring the temperature down to safe values. This state can either be reached initially, if the system is in **Observe** state and the maximum cell temperature overcomes TempOn, or alternatively if, starting from a **PumpOff** state, the two aforementioned conditions are true. On the contrary, moving away from this state requires the cell temperature to reach values lower than the deactivation temperature TempOff and to stay in that range for hundreds of seconds. As long as the system stays in the **PumpOn** state, the logic signal ThermLogicSign

is set to 1.

- **Temperature Difference Computation subsystem:** the Stateflow chart is essential to implement the overall activation and deactivation logic of the thermal function, which, as it was stated countless times at this point, must be carefully designed to avoid hazards and safety risks. During the development of this thesis, however, it was deemed appropriate to further improve the function and introduce a flow rate modulation to reduce the energy consumption and match the flow rate value to the temperature condition of the battery. The general idea is to set an arbitrary maximum flow rate value, which is decreased by means of a linear relation, depending on the maximum cell temperature. This portion of the thermal function is developed in three different subsystems, the first of which is the **Temperature Difference Computation**.

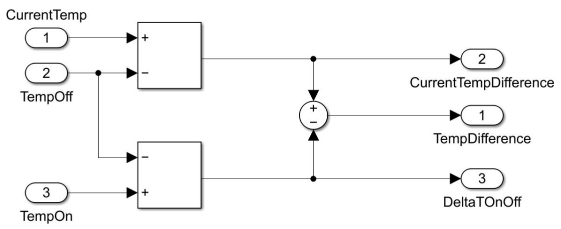


Figure 3.16. Temperature Difference Computation Simulink schematic

The first step in developing the flow rate modulation is shown in Figure 3.16. By simply considering the one time step delayed maximum cell temperature CurrentTemp, the deactivation and activation temperatures TempOff and TempOn respectively, three temperature differences are computed:

1. **CurrentTempDifference:** difference between CurrentTemp and TempOff.
2. **DeltaTOnOff:** difference between TempOn and TempOff.
3. **TempDifference:** difference between CurrentTempDifference and DeltaTOnOff.

These values will be exploited in the following subsystems.

- **Normalized Flow Rate subsystem:** the outputs of the **Temperature Difference Computation** are provided to the **Normalized Flow Rate** subsystem as inputs, with the aim of computing a multiplying coefficient lower or equal to 1 for the MaxFlowRate, which acts as a NormalizedFlowRate and is obtained via linear temperature relation.

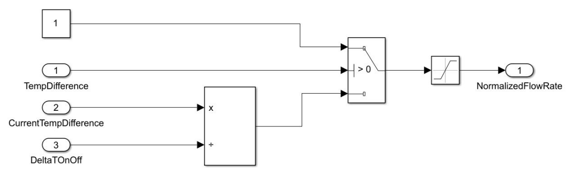


Figure 3.17. Normalized Flow Rate Simulink schematic

The block schematic is shown in Figure 3.17 and can be described as follows: depending on the sign of TempDifference, the central switch lets through one of the two inputs specified at ports 1 and 3. The first scenario occurs when TempDifference is positive, meaning that the maximum cell temperature measured for that time step is higher than the activation temperature TempOn. When this happens, we want the function to provide the strongest reaction to counteract and decrease the increasing cell temperature as fast as possible. Hence, a constant value of 1 passes through the switch. When the TempDifference switches sign (the cell temperature is varying between TempOn and TempOff), the value at port 3 is let through. Such value is the ratio between CurrentTempDifference and DeltaTOnOff, which linearly decreases with respect to temperature. Before being conveyed to the next subsystem, the NormalizedFlowRate is constrained to a lower limit of 0.01 with a saturation block.

- **Flow Rate Computation subsystem:** in the last subsystem, the MaxFlowRate is multiplied by the NormalizedFlowRate to obtain the FlowRate value that commands the flow rate source at the Fluid_In port. For obvious reasons, this is true if the ThermLogicSignal is equal to 1, while it is set to 0 if the logic signal is equal to 0. The Simulink scheme of this subsystem is shown in Figure 3.18.

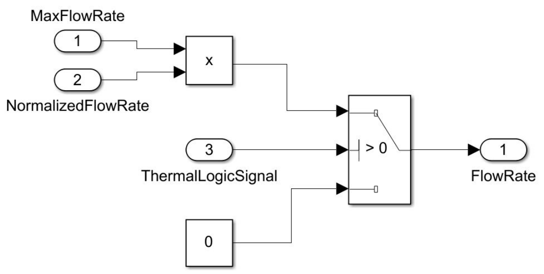


Figure 3.18. Flow Rate Computation Simulink schematic

3.4 SoC and SoH estimation

When designing a BMS, countless different functions can be implemented to optimize the behaviour of the battery and monitor its operations, but no innovative or efficient algorithm can properly function without solid SOC and SOH estimation methods, as they are often required as inputs for other purposes. Because both quantities cannot be directly measured by means of a sensor, it is essential to recover accurate estimates, by exploiting other battery characteristics.

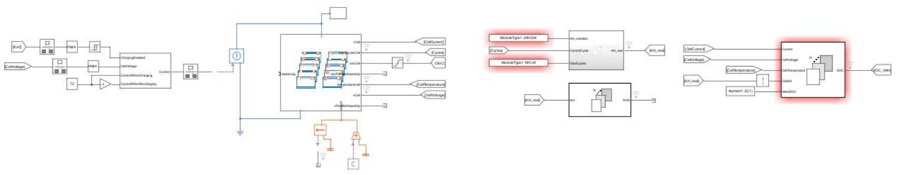


Figure 3.19. SoC and SoH estimation Simulink schematic complete overview

Figure 3.19 shows the complete Simulink schematic, which, as usual by now, is divided as follows:

- **Charging/Discharging algorithm:** the CC-CV algorithm, which is set equally to all the previous functions, is placed on the left-hand side of the schematic.
- **Custom battery pack:** the battery pack is connected to the CC-CV output via controlled current source, while the other ports, as shown in Figure 3.20, are either connected to terminators, in which case the corresponding quantities are not needed to compute the estimates, or to Goto blocks. The only exception is represented by the ambient thermal port AmbH, where a controlled thermal source allows the user to set an arbitrary temperature, while a temperature sensor measures it and reports it in the simulation data.

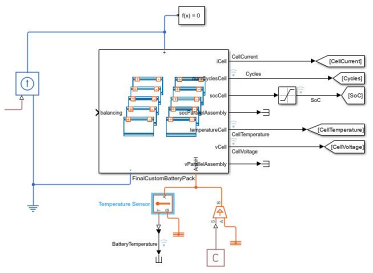


Figure 3.20. SoC and SoH estimation battery pack Simulink schematic

- **SOC and SOH estimation:** on the right-hand side of the schematic (Figure 3.21) is a set of three blocks, properly arranged to compute state of charge and state of health estimates.

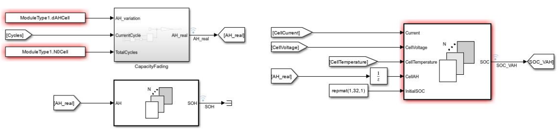


Figure 3.21. SoC and SoH estimation Simulink schematic

1. **Capacity Fading subsystem:** The first step in setting up the algorithm was to evaluate the phenomenon of capacity fading, due to which the capacity of the battery pack decreases as the number of discharge cycles increases. In this regard, the **Capacity Fading subsystem** was created to reproduce the mathematical description of this phenomenon, which can be observed in Equation 3.2.

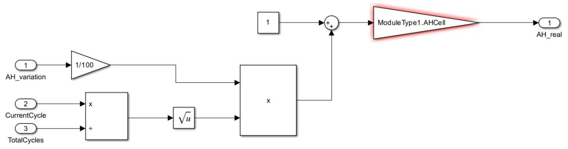
   

   Figure 3.22. Capacity Fading Simulink schematic

   The block scheme in Figure 3.22 is in fact a reproduction of said equation and, for obvious reasons, it is necessary to provide the three battery variables as inputs to compute the value of capacity. AH_variation δ_{AH} and TotalCycles N are two variables defined in the **filename_param.m** file, under the **dAHCell** and **N0Cell** parameters respectively; the first is the change in cell capacity after N discharge cycles, while the second is the number of discharge cycles, which can be freely set by the user, after which a capacity reduction equal to δ_{AH} occurs. The third input is the number of current discharge cycles n and is obtained from the pack output port numCyclesCell. Modelling the fading mechanism becomes imperative, firstly to keep track of the amount of energy that can be stored in the battery and also to estimate SOC and SOH with a higher degree of accuracy, obtaining, as a result, a more reliable BMS.

2. **SOC Estimator:** the Simscape library allows developers to choose from a rich variety of blocks, that are divided into categories, depending on the purpose or function they are designed for. For the development of

this thesis, the estimation of the cell state of charge was built around the SOC Estimator block, a KF unit whose specific aim is to provide an SOC estimate, given the state-space representation of the battery model. When picking one of the estimators available in the library, the choice fell (as shown in Figure 3.23) on a variant of the block with variable capacity, which was needed to take into account the already modelled capacity fading phenomenon.

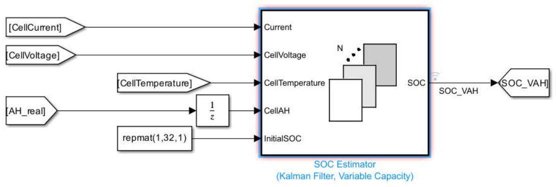


Figure 3.23. SoC Estimator Simulink schematic

The MathWorks documentation [40] shows that the equivalent battery model, upon which state and output equations are computed, is a one-time-constant dynamics circuit with a serial connection of a voltage source, an ohmic resistance and an RC cell. The equations for the equivalent circuit are:

$$\frac{dSOC}{dt} = -\frac{i}{3600AH} \tag{3.5}$$

$$\frac{dV_1}{dt} = \frac{i}{C_1(SOC,T)} - \frac{V_1}{R_1(SOC,T)C_1(SOC,T)}$$

$$V_t = V_0(SOC,T) - iR_0 - V_1$$

where SOC is the state of charge, i is the current, V_0 is the open circuit voltage, V_t is the terminal voltage, AH is the variable ampere-hour rating, R_1 is the polarization resistance, C_1 is the parallel RC capacitance, T is the temperature and V_1 is the polarization voltage. Equations 3.5 only represent the starting point in estimating the SOC because, after double clicking on the block in Figure 3.23 and selecting the type of filter to be used, different mathematical operations are carried out, depending on the algorithm selection. For this custom BMS development, the highly non-linear behaviour of the system pointed towards a UKF, which, despite being more demanding than other algorithms in terms of computational power, is accurate to the second order of the Taylor expansion. For this reason, to apply the unscented transformation on the system, which consists in deterministically choosing a set of sigma points and calculating mean and covariance weights for each one of them, the SOC Estimator block allows to set the α, β and κ parameters, along with the process noise covariance Q, the measurement noise covariance R, the initial state error covariance P_0 and the sample time. In addition, the terminal resistance R_0, the polarization resistance R_1 and the time constant τ can be customized and modified to adjust the equivalent circuit to match the pack dynamics. After correctly setting up the filter, the block is fully functional and can provide a state of charge estimate by generating the desired number of sigma points and their respective weights, predicting the system state and then correcting it. For this algorithm exploits the Kalman theory, the quantities that must be iteratively fed to the block are:

- **Current:** the following port can either be exploited to input a vector of cell currents or the pack current, depending on whether the user intends to compute the battery overall SOC or a state of charge vector, whose elements refer to every single cell. For this particular software development, the cell current vector was provided to the estimator.
- **Cell Voltage:** the vector containing the voltage provided by each battery cell.
- **Cell Temperature:** the vector containing the temperature of all battery cells.
- **Cell AH:** because this specific estimator accounts for a variable capacity, one of the required inputs to make the algorithm work is the ampere-hour rating of the battery, which can be retrieved from the **Capacity Fading subsystem** output. It is also worth mentioning that to have the filter run in real time, the capacity must be delayed by one time step.
- **Initial SOC:** coherently with all kalman filters, even this estimator needs the initial conditions of the SOC.

3. **SOH Estimator:** the last block of the right-hand side of the Simulink schematic is dedicated to SOH estimation algorithm. Similarly to what was done for the SOC, the algorithm for the state of health estimation was built on one of the blocks available in the Simscape library: the Capacity-Based SOH Estimator.

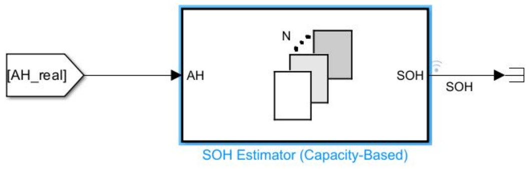


Figure 3.24. SoH Estimator Simulink schematic

This estimator, which can be observed in Figure 3.24, is a very simple block, whose working principle is based on capacity fade [41]; the state of health is in fact computed as a function of the cell capacity, according to Equation 3.6

$$SOH_Q = \frac{Q - Q_{eol}}{Q_{eol} - Q_{new}} \tag{3.6}$$

where Q is the cell capacity in ampere-hour, while Q_{new} and Q_{eol} are the cell capacity at the beginning and end of the battery life respectively. The default value of the last two parameters can be modified by double clicking on the estimator.

3.5 Fault Detection implementation

All physical systems are subject to failures and hence require a detection mechanism to first locate where the fault originated from and a control logic to counteract the issue and try to bring said system from a faulty to a normal operational state. Batteries are no different in this regard, as failures can occur for a plethora of different reasons, leading to dangerous hazards for the health of users or people around them. As a consequence, to complete the development of this BMS, a fault detection function was implemented to monitor the well-being of the pack and to transition from a "healthy" to a faulty operational state, in case issues of different nature arise.

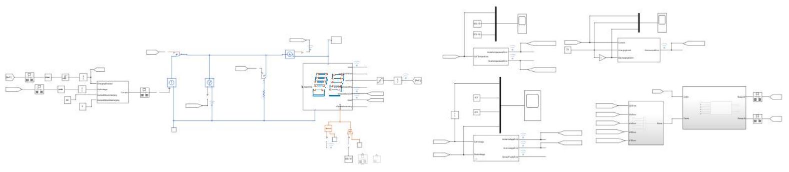


Figure 3.25. Fault Detection Simulink schematic complete overview

It should be clear by now that during the development of every single BMS function, the Simulink schematic, which for the Fault Detection is shown in Figure 3.25, is divided into the following three functional blocks:

- **Charging/Discharging algorithm:** the CC-CV algorithm is located on the left-hand side of the schematic and, unlike its previous implementations, for Fault Detection it only exerts charging currents on the battery pack, as the CurrentWhenDischarging input port is provided a null current by means of a constant block.
- **Custom battery pack:** the battery pack is placed at the center of the schematic and the physical quantities needed to implement the algorithm (cell SOC, cell temperature, cell voltage, pack current and voltage) are directly connected to Goto blocks, to obtain a cleaner scheme. The ambient thermal port Amb_H is configured similarly to the previous configurations, as a thermal source and a temperature sensor are connected to externally control the temperature of the surroundings and measure such temperature respectively.

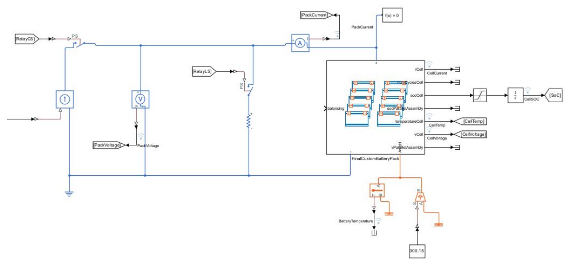


Figure 3.26. Fault Detection battery pack Simulink schematic

As it can be observed in Figure 3.26, an additional network of blocks is included in the battery pack; more specifically, two sensors are implemented in the scheme, one current sensor serially connected to the positive pack terminal to measure the current flowing in and out of the battery and a voltage sensor to measure the pack voltage across the two terminals. Moreover, a set of two switches is introduced in the schematic as a safety measure to counteract fault occurrences. The first switch is placed in a branch parallel to the battery pack and in series with a resistor that models the vehicle load; depending on the driving scenario and type of fault, the switch can be opened or closed, letting through or cutting the current flow. The second is serially connected to the controlled current source that provides the charging current coming from the CC-CV algorithm; in critical cases, the switch can be actuated to open, thus disconnecting the system from the charging algorithm.

- **Fault Detection:** the actual detecting mechanism is developed on the right-hand side of the schematic and, as it can be seen in Figure 3.27, it is divided into four functional units, three for detecting purposes and one to manage and reset faults.

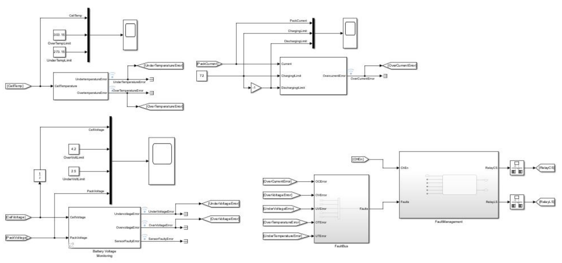


Figure 3.27. Fault Detection Simulink schematic

When it comes to developing a fault detection function, it is pivotal to first determine the nature of the faults the battery might encounter while being used and, only then, design the actual mechanism. For this thesis work, the faults that can be detected relate to temperature, voltage and current anomalies and the detection is implemented through blocks of the Simscape library. The mechanism responsible for actuating the switches and solving the faults is instead custom made and revolves around a Stateflow chart.

- **Battery Temperature Monitoring:** the first monitoring functional unit is built around the Battery Temperature Monitoring block shown in Figure 3.28.

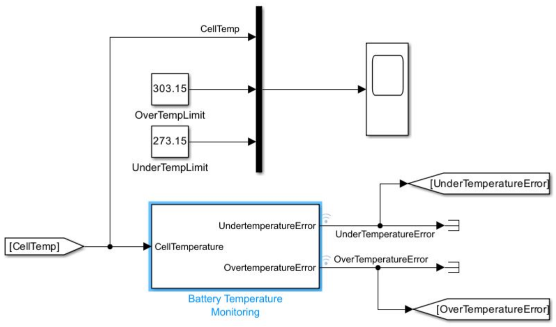


Figure 3.28. Battery Temperature Monitoring Simulink schematic

The central block receives the cell temperature vector as input and computes two output signals: the UnderTemperatureError and the OverTemperatureError. The outputs are two simple logic signals, which are set to 0 as long as the cells are working under normal temperature conditions, but immediately switch to 1 upon the occurrence of a temperature fault. The definition of the fault characteristics is left to the developer, who is free to set an UnderTemperatureLimit and OverTemperatureLimit, with which the block verifies these inequalities:

$$\begin{cases} min(Temperature) \leq UnderTemperatureLimit \\ max(Temperature) \geq OverTemperatureLimit \end{cases} \tag{3.7}$$

The block actually computes two logic signals, the UnderTemperatureSymptom and the OverTemperatureSymptom, which are set to 1 if the inequalities 3.7 are true. The symptoms are then passed to a Fault Qualification block: if the symptom is 1 for as long as the qualification time (arbitrarily set by the user), the error occurs and can only be reset if the symptom is null for an interval of time equal to at least the disqualification time [42]. To have a visual representation of how the cell temperature might end up triggering a fault, a scope displays all the temperature curves, including the thresholds chosen by the user.

- **Battery Voltage Monitoring:** the Battery Voltage Monitoring block of Figure 3.29 is based on the same working principle of the Battery Temperature Monitoring block, meaning that a fault is generated or reset depending on how long a symptom is either present or absent, according to qualification and disqualification times [43].

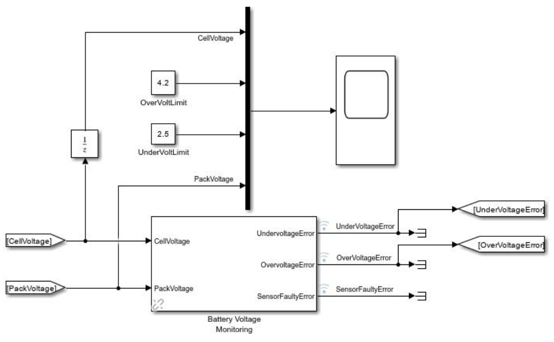


Figure 3.29. Battery Voltage Monitoring Simulink schematic

The cell voltage vector and the pack voltage are provided as inputs, while, on the output side, three values are computed: an UnderVoltageError and an OverVoltageError, whose values depend on the limits declared in the block parameter menu according to the inequalities

$$\begin{cases} min(CellVoltage) \leq UnderVoltageLimit \\ max(CellVoltage) \geq OverVoltageLimit \end{cases} \tag{3.8}$$

and a SensorFaultyError, which is triggered when the discrepancy between the measured pack voltage and the theoretical voltage overcomes an adequate threshold. This fault, whose behaviour is controlled by the inequality 3.9, simulates the damage of the voltage sensor in the system.

$$|PackVoltage - \sum_{i=1}^{8} ParallelAssemblyVoltage(i)| \geq FaultySensorThreshold \tag{3.9}$$

Once again, a scope is at the user disposal to compare the voltage behaviour with the over and under voltage limits.

- **Battery Current Monitoring:** the last functional unit with detecting purposes regards current faults and is displayed in Figure 3.30.

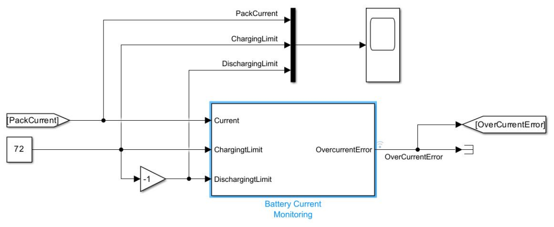


Figure 3.30. Battery Current Monitoring Simulink schematic

The Battery Current Monitoring block requires the pack current and charging and discharging current limits in order to compute the OverCurrentError, which, similarly to the aforementioned detecting systems, computes a symptom and passes it to a Fault Qualification block [44]. The comparison between the pack current and the two limits respects the inequality

$$(i - ChargeLimit) > ErrorLimit \vee (DischargeLimit - i) > ErrorLimit \tag{3.10}$$

The scope above the monitoring block can be opened to check if the pack current stays inside the safe region delimited by the two limits.

- **Fault Bus and Fault Management:** the core of the fault detection function resides in the last functional unit on the right-hand side of the schematic.

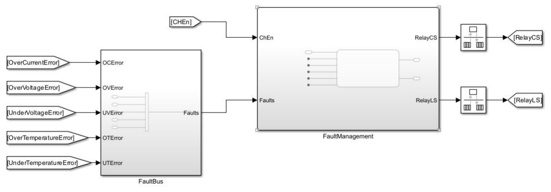


Figure 3.31. Fault Bus and Fault Management Simulink schematic

Figure 3.31 shows that the composition is divided into two sections:

1. **Fault Bus subsystem:** because the function deals with a considerate number of faults, the first subsystem is designed to create a single bus that carries all errors, hence reducing the number of connections in the scheme. This is made possible by providing all errors to a bus creator block, which outputs one "line", as shown in Figure 3.32, containing all the faults.

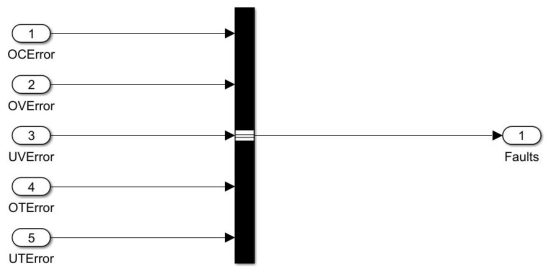


Figure 3.32. Fault Bus subsystem Simulink schematic

2. **Fault Management subsystem:** the last subsystem is based upon a control logic, developed by means of a Stateflow chart, which takes the bus elements as inputs and computes a logic signal to control the state of the two switches.

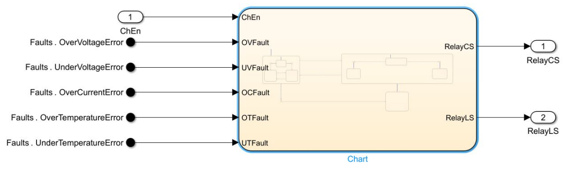

1
ChEn
Faults . OverVoltageError
Faults . UnderVoltageError
Faults . OverCurrentError
Faults . OverTemperatureError
Faults . UnderTemperatureError
ChEn
OVFault
UVFault
OCFault
OTFault
UTFault
RelayCS
RelayLS
Chart
1
RelayCS
2
RelayLS

Figure 3.33. Fault Management subsystem Simulink schematic

In Figure 3.33 is a clear representation of how the system works: the bus elements are extracted from the bus with the Bus Element In block and are given as inputs to the Stateflow chart; moreover, an additional input called ChEn is provided to the switch control logic and represents the activation signal of the battery charging phase, taken from the CC-CV algorithm. The Stateflow then computes a RelayCS signal, which commands the switch connecting the charging algorithm with the pack, and a RelayLS signal, which, on the contrary, controls the switch serially connected to the resistive load.

The aforementioned Stateflow chart contains a complex logic made up of nested states and composed transitioning conditions and, for this reason, deserves a separate analysis.

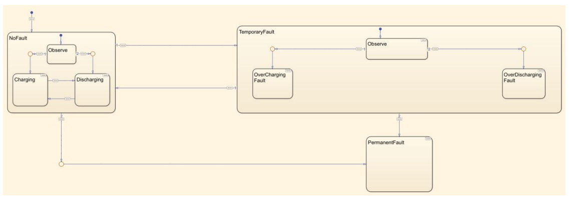


Figure 3.34. Fault logic Stateflow chart

By looking at Figure 3.34, it can be observed that the chart is developed around three states, two of which are parent states to multiple other states:

1. **NoFault state:** default state of the control logic, which describes a regular operational state where no battery failure has occurred and therefore no fault is currently triggered. Upon entering in **NoFault** for the first time, the local variable FaultCounter is set to 0, as no fault has yet occurred, and the logic develops into three children states.

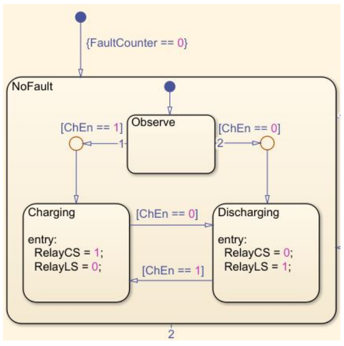


Figure 3.35. NoFault state Stateflow chart

As shown in Figure 3.35, the default secondary state is called **Observe** and it is necessary to determine whether the battery pack is under charging or discharging conditions; for this reason, by checking the value of the input variable ChEn, an internal transition takes place. If the charging logic signal is equal to 1, the system transitions to the **Charging** state and it sets RelayCS to 1 (connecting the pack to the charging circuit) and RelayLS to 0 (causing the switch to disconnect from the load). On the contrary, when ChEn is equal to 0, the system enters the **Discharging** state, setting opposite values for the output variables. Needles to say that transitions can also happen between **Charging** and **Discharging**, depending on the value of ChEn.

2. **TemporaryFault state:** by the time one voltage or current related fault occurs, the system immediately transitions to the **TemporaryFault** state and the FaultCounter variable is increased by one. In practical terms, this means that a vehicle, upon detecting a fault regarding voltages or currents, enters a logic state that allows the control system to trigger the countermeasures needed to reset the fault itself. It is also worth mentioning that this Stateflow chart implements a sort of security mechanism, according to which **TemporaryFault** can be accessed as long as the number of current or voltage fault occurrences is lower than five.

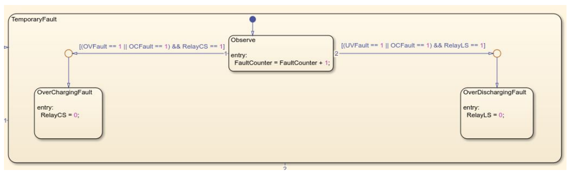

TemporaryFault
Observe
entry:
FaultCounter = FaultCounter + 1;
[(OVFault == 1 || OCFault == 1) && RelayCS == 1]
[(UVFault == 1 || OCFault == 1) && RelayLS == 1]
OverChargingFault
entry:
RelayCS = 0;
OverDischargingFault
entry:
RelayLS = 0;

Figure 3.36. TemporaryFault state Stateflow chart

In Figure 3.36, it can be observed that **TemporaryFault** is divided into three secondary states, the default one being a common **Observe** state and the other two being **OverchargingFault** and **OverDischargingFault**. When the battery pack is connected to the charging circuit ($RelayCS = 1$) and an over voltage or over current fault occurs, the system transitions from the **Observe** state to the **OverChargingFault** state, where the closed switch is opened in order to avoid reaching dangerous levels of battery charge. If an under voltage or over current fault takes place but the pack is instead connected to the load ($RelayLS = 1$), the system enters the **OverDischargingFault** state and the battery is disconnected from the load to avoid excessive battery discharge. If all faults are solved, the system transitions back to the **NoFault** state.

3. **PermanentFault state:** the last state to be discussed is the **PermanentFault**, which describes the most extreme scenarios where the battery pack might end up. Temperature faults are extremely dangerous, as they can easily lead to potential hazards for the driver or passengers aboard. For this reason, during the design phase of this software, it was decided to develop an additional state related to temperature anomalies.

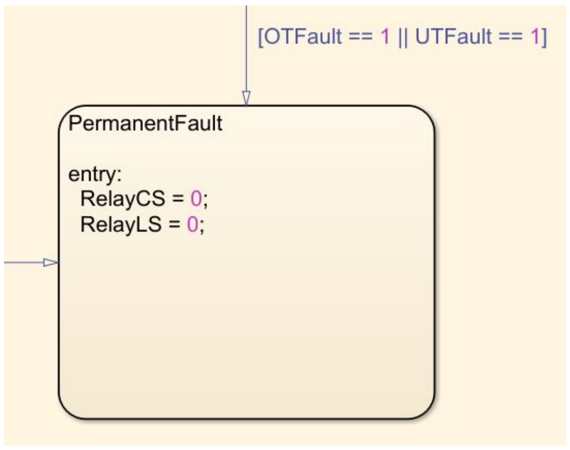


Figure 3.37. PermanentFault state Stateflow chart

As it is shown in Figure 3.37, the **PermanentFault** state is entered if an over or under temperature fault occurs regardless of the starting state, or if FaultCounter reaches a value of 5 and the system is in **NoFault** state. When the battery pack transitions to this state, both switches are opened so that the system is disconnected from both the CC-CV algorithm and the load.

Chapter 4

Testing and Results

Software development is certainly not an easy task, as developers must come up with ideas to technically implement the designing concepts and also to validate the work through extensive testing. For this thesis, the designing phase has already been carried out and discussed in the previous chapter, while testing and validation of the single functions first and of the complete BMS later will be examined in the following paragraphs.

4.1 Battery pack testing

Before testing out the BMS functions, it is essential to validate the battery pack for which the software was originally designed, as some of its basic electrical features and characteristics require testing to verify if they match the expected behaviour. The aim of these tests was mainly to check the real electrical capacity of the model, comparing it with the expected results, and to have the model undergo discharging cycles with variable loads to analyze and measure currents, voltages and temperatures and to critically observe whether such measurements can be tolerated by real-life systems.

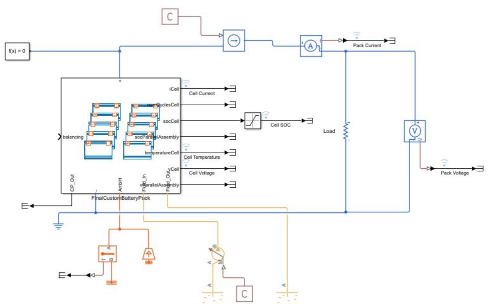


Figure 4.1. Battery testing Simulink schematic

Figure 4.1 shows the testing circuit built around the model: for starters, the output ports are connected to terminators and cell currents, state of charge, temperatures and voltages are all logged, so that, after running simulations, data can be visualized in the Data Inspector to validate the results. Secondly, the temperature of the surrounding environment is simulated by connecting a temperature source to the Amb_H port, to which a temperature sensor is also connected, while the inlet and outlet coolant ports let the fluid flow thanks to the connection of a flow rate source and a reservoir block. To conclude, a resistor that models the battery load closes the circuit across the positive and negative terminals of the pack. The test "harness" was first used to measure the real electrical capacity of the system, whose technical definition was exploited to set up the simulation; one of the possible ways to measure the capacity is in fact via the ampere-hour unit of measurement, which simply represents the current (in Ampere) that can be continuously provided by the system for an hour, before completely discharging. A controlled current source was for this reason connected in series with the positive terminal and the value of current was set to 376 A, same as the ampere-hour rating of the pack (for more details check Table 3.5). Two sensors were then added, one current sensor to measure the imposed current value and a voltage sensor placed in parallel with the load to measure the pack voltage.

Initial cell SOC	1
Simulation time	5000 s
Experimental capacity	376 Ah
Load	50 Ω

Table 4.1. Custom battery pack capacity test

As it can be observed in Table 4.1, the initial condition for the cell state of charge was set to 1, so that the battery could undergo a deep and complete discharge cycle without having unbalanced cells; besides, the resistive load was set to 50 Ω and the simulation time to 5000 s, a little over an hour.

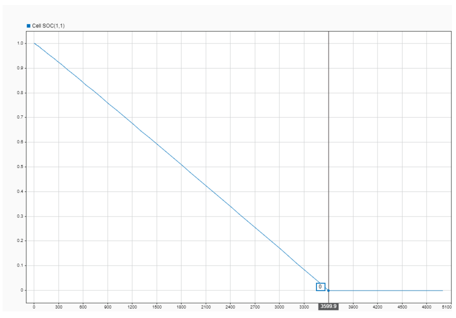


Figure 4.2. Battery testing discharge curve

The simulation results can be analyzed in the Data Inspector, where all the logged signals can be visualized as functions of time. In Figure 4.2, the curve representing one cell SOC is shown: the cell starts out at 0 s with an SOC value of 1, which corresponds to a fully charged unit, as it was specified in the **filename_param.m** file. As soon as the simulation starts, the curve starts decreasing because the current source forces the battery to discharge, by having electrical charges flow through the load. After continuously decreasing in a strictly monotonic fashion, the curve reaches a zero value at 3600 s (one hour), demonstrating that the maximum capacity of the battery is 376 Ah as expected. It is also worth mentioning that, even though Figure 4.2 only represents one cell state of charge, the other curves in the simulation data are perfectly equivalent. The second test simply consisted in having the pack discharge on a resistive load, without imposing a fixed discharge current through a current source.

Initial cell SOC	1
Simulation time	5000 s
Load	0.1 Ω

Table 4.2. Custom battery pack 0.1 Ω discharge test

For this simulation, as it can be read in Table 4.2, the settings remain untouched with respect to the previous simulation run, but the load value is modified to 0.1 Ω.

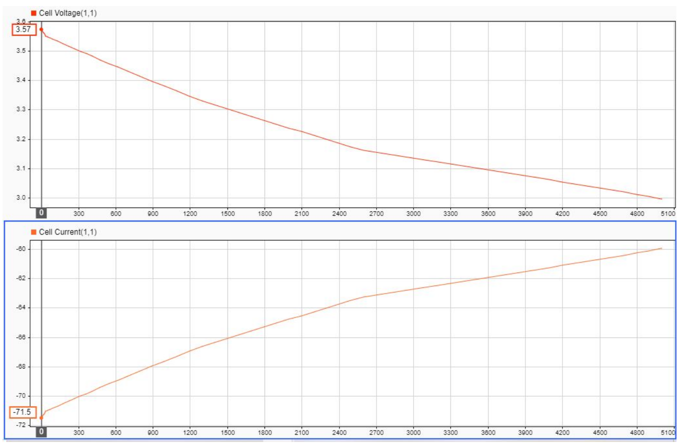


Figure 4.3. Battery testing on 0.1 Ω load

Figure 4.3 shows two graphs taken from the Data Inspector, which depict the cell voltage and current. The former shows that the cell initially provides 3.57 V, a value that is comparable to the real cell nominal voltage this development is inspired by, while, by the end of the simulation, the voltage reaches 3 V; the latter exhibits the same behaviour, despite a sign inversion: the cell current starts out at -71.5 A and, with different rates along the way, decreases in absolute value till reaching -59.9 A at the end of the simulation. The overall results are acceptable as real-life storage systems can easily provide said values. To conclude the tests on the battery and to verify the consistency of the acquired data, the pack voltage curve was analyzed.

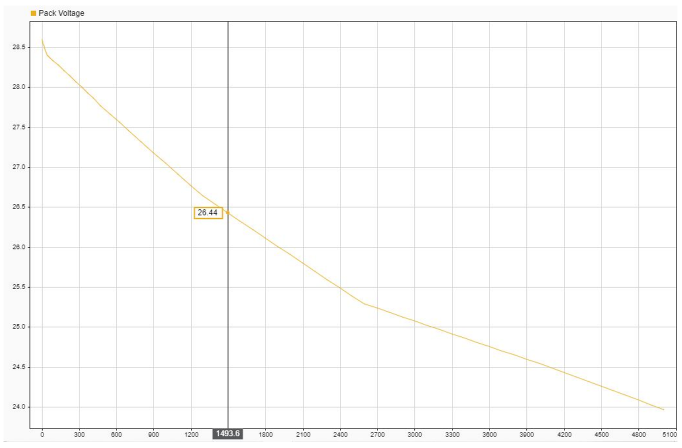


Figure 4.4. Pack voltage across 0.1 Ω load

In Figure 4.4 the behaviour over time of the pack voltage is shown, it starts at 28.6 V and constantly decreases till halfway through the simulation, where the value settles at 25.3 V. From that point on, the voltage decreases with a lower rate and stops at 23.97 V. In conclusion, all the tests and simulations performed on the battery pack proved that the main model electrical features are consistent with the values declared during the design phase and that the cells provide voltage and current values that are compliant with real-life physical systems.

4.2 Constant Current-Constant Voltage charge cycle

After validating the system model, it becomes essential to test the BMS software through the so called unit testing phase, which consists in running simulations and gathering data, while applying only one function at the time. However, before doing so, a brief description of the charging/discharging algorithm that the battery undergoes during these tests will be presented, as anticipated in the previous chapter. The algorithm in question is the CC-CV, which has already been mentioned countless times by now, and is implemented through the Battery CC-CV block, available in the Simscape library.

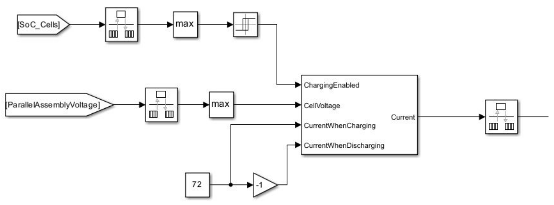


Figure 4.5. Constant Current-Constant Voltage block scheme

Figure 4.5 represents the circuit, which can be described as follows: in the upper branch, the maximum cell state of charge element is provided to a relay block, which outputs 1 when said value reaches 0.63 while it outputs 0 when it reaches 0.9; in this way, the ChargingEnabled port activates or deactivates the charging phase according to the SOC levels of the cells. The middle branch is related to a voltage monitoring activity because the maximum parallel assembly or cell voltage is conveyed to the CellVoltage input port and compared to a voltage threshold, arbitrarily set by the user. The lower and final branch is instead implemented to specify the values of charging and discharging currents, provided to CurrentWhenCharging and CurrentWhenDischarging ports respectively. The output of the block is a current, which is used to drive a controlled current source. The value of such current is described by the Equations

$$Current = \begin{cases} \text{Maximum charge current}, & \text{if battery is charging and} v_{meas} < v_{max} \\ (K_p + K_i \frac{1}{s})(v_{max} - v_{meas}), & \text{if battery is charging and} v_{meas} \geq v_{max} \\ \text{Maximum discharge current}, & \text{if battery is discharging} \end{cases} \tag{4.1}$$

where v_{max} is the voltage threshold, v_{meas} is the voltage of the highest cell, K_p and K_i are the proportional and integral constants of a PI controller inside the block. To put in simple words what is written in Equations 4.1, it can be said that when charging, the current provided by the battery coincides with the value specified at the CurrentWhenCharging input port, as long as the maximum cell voltage is below the threshold; by the time the threshold value is reached, the current is lowered by the PI controller according to the written mathematical law, while the voltage is maintained constant. During discharge, the current is instead always kept constant and coincident with the value specified at the CurrentWhenDischarging input port. Despite being a purely descriptive section, this paragraph has been placed in this chapter because the next unit testing discussions will be based on results obtained via CC-CV application.

4.3 Passive Cell Balancing unit testing

When testing the passive cell balancing function, the simulation data must prove that a pack in a state of imbalance among its cells eventually reaches a shared level of SOC, once the mechanism is triggered. The first phase in the unit testing was to correctly set up the simulation.

Initial cell SOC first assembly	0.69
Initial cell SOC second assembly	0.715
Initial cell SOC third assembly	0.7
Initial cell SOC fourth assembly	0.7
Simulation time	18000 s
Shunt resistance	6 Ω
Balancing threshold	10^{-3}

Table 4.3. Passive cell balancing simulation setup

Table 4.3 sums up the most important values that were set before running the simulations. For starters, the cell SOCs needed to be initialized to different values in order to recreate a virtual scenario of charge imbalance; for this reason, the cells of the first parallel assembly started at a value of 0.69, the cells of the second assembly started from 0.715, while those belonging to the third and fourth assemblies were set to 0.7. This SOC pattern was then repeated for the remaining parallel assemblies. The shunt resistor used to dissipate through heat generation the excessive charge from highly charged cells was set to 6 Ω and the balancing threshold which determines the activation of the mechanism was set to 10^{-3}.

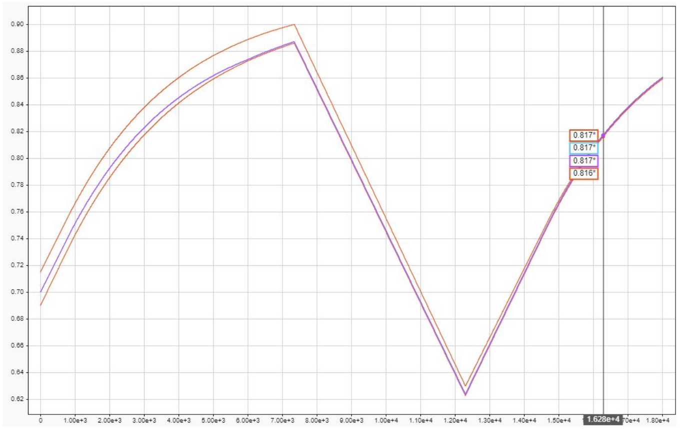


Figure 4.6. SOC curves with passive cell balancing

In Figure 4.6 there are four curves representing the SOC of one cell for each of the first four parallel assemblies and it can be observed how they evolve and converge during the course of the simulation. The curves all start at different values, as it is stated in Table 4.3, and, because the battery undergoes a charging phase, they immediately start rising in a non-linear fashion. At this point, the SOC values are clearly unbalanced and, for this reason, the mechanism is activated because the difference between the cell with the lowest charge and the one with highest charge overcomes the defined threshold. At the end of the charging phase, two cells already converge to the lowest SOC value and reach a balanced status because the difference settles precisely at 10^{-3}. The pack is then subject to a discharging phase, during which the curves descend in a quasi-linear fashion: the remaining unbalanced cell starts out with a difference of 0.0139 with respect to the low charged cell at the beginning of the descent and finishes with a discrepancy of 0.007. When the SOC reaches a value of 0.62, the curves start rising again and a fully balanced status is reached after an overall time interval of 16280 s, after which all cell states of charge stay inside the tolerance band given by the threshold.

4.4 Thermal Management unit testing

For the thermal management unit testing, a longer simulation runtime was required as this type of physical phenomena has slower dynamics with respect to electrical ones, therefore in order to better appreciate and capture the behaviour of the temperature and the control logic behind it a bigger time scale was exploited.

Simulation time	43200 s
Temperature at ambient port	298.15 K
Activation temperature	303.15 K
Deactivation temperature	298.15 K
Maximum flow rate	0.1 m^3/s

Table 4.4. Thermal management simulation setup

As shown in Table 4.4, the simulation time has been extended to 12 hours in order to get a clear picture of how the thermal side of the pack is managed by the algorithm over a long period of time. As far as the custom control logic setup is concerned, the activation temperature TempOn and the deactivation temperature TempOff are respectively set to 303.15 K and 298.15 K, while the maximum rate at which the coolant can flow through the cooling plate is 0.1 m^3/s. The setup is then completed with a temperature source connected to the pack ambient thermal port Amb_H, which simulates a temperature of 298.15 K for the surroundings.

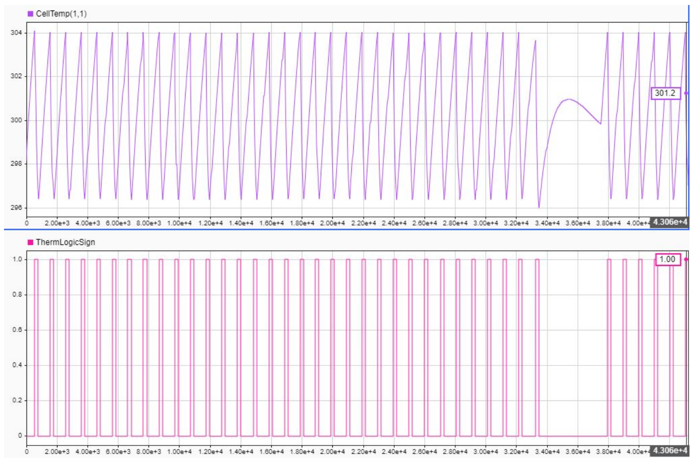


Figure 4.7. Cell temperature and thermal logic signal curves under thermal management

The first step in analyzing the algorithm is by taking a look at Figure 4.7, where the graph of the cell temperature behaviour is coupled with the logic signal ThermLogicSign. The former develops along the time axis with a series of spikes, which are limited by temperature values that slightly exceed TempOn and TempOff: this experimental evidence, along with the graph placed at the bottom of the Figure, proves that, qualitatively speaking, the mechanism works fine, as the logic signal continuously switches between 1 and 0 and follows the cell temperature accordingly; moreover, because a time delay of 100 s was introduced in the Stateflow chart, the temperature reaches peaks that overcome the activation and deactivation thresholds as expected. A rough efficiency estimate can also be computed by counting a total of 39 activations in 12 h, which averages to 3.25 activations per hour, an acceptable number for a real-life system. The only strange looking behaviour occurs at around 36000 s (10 hours) when the cell temperature starts increasing non-linearly with a lower average rate and then reaches a local maximum, after which it decreases for around 2000 s: the reason why this happens is due to the fact that, around that time, the battery pack is undergoing the final stage of a charging phase, which, according to the CC-CV algorithm, first foresees a constant current phase and then, once a certain voltage threshold is reached, a constant voltage phase, during which the current is decreased according to the equation of the PI controller. Because the current provided by the CC-CV circuit to the battery is decreasing over time, the temperature decreases as well and, as demonstrated by the zero value of the ThermLogicSign, never reaches the activation temperature.

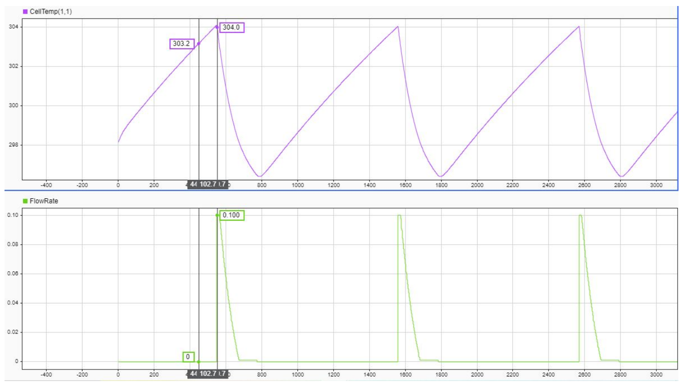


Figure 4.8. Focus on cell temperature and flow rate curves under thermal management

A quantitative confirmation of such observations comes from Figure 4.8, where the top graph is a focus on the temperature behaviour during the first 3115 s, while the bottom graph is the corresponding variation of the flow rate. At the beginning of the simulation run, the top curve starts at 298.15 K and rapidly increases in a quasi-linear fashion, without triggering the mechanism (as it is demonstrated by the null value of the flow rate). Along the rising slope, the TempOn threshold is then reached and, after a time interval of 100 s, the temperature exceeds the limit by only one degree and reaches its maximum value, thus activating the thermal management function. The most evident change occurs in the flow rate, which almost instantly varies from 0 to the maximum value of 0.1 and consequently causes the temperature to decrease. During the temperature descent, the flow rate is linearly scaled down until the system is brought back to a safe thermal range and it can therefore be set to 0 again. Similarly to what happens during the temperature increase phase, once the deactivation value TempOff is reached, the mechanism switches off after a time interval of 100 s elapses. At that point the system can operate without needing to be cooled down and the temperature starts rising again starting from the minimum value of 296.4 K.

4.5 SOC and SOH estimation unit testing

Computing accurate estimates for the state of charge and state of health of a battery pack is a delicate but necessary task that must be carefully carried out, especially in the automotive field. For this reason, extensive testing was performed during the development phase to make sure the results are the best one can possibly obtain from this custom algorithm.

Simulation time	25200 s
Number of discharge cycles N	100
Change in cell capacity after N cycles	-25 %
Initial SOC	1

Table 4.5. SOC and SOH estimation simulation setup

As usual by now, Table 4.5 sums up the most important parameter settings for the simulation setup: the simulation time is set to 7 hours and all cells have an initial SOC value of 1, as there is no need to recreate an imbalanced battery state for this specific function testing. An important physical phenomenon to take into account in this phase, however, is the capacity fading, for which two parameters were modified in the **filename_param.m** file; the first is a fixed number of discharge cycles N, which was set to 100, and the second is the capacity variation of the battery after N cycles, which was set -25 %. The first test aimed at finding the best SOC estimate by means of UKF parameter tuning, which was essential to later test and validate the SOH function, given that, mathematically speaking, the state of health is a function of the state of charge.

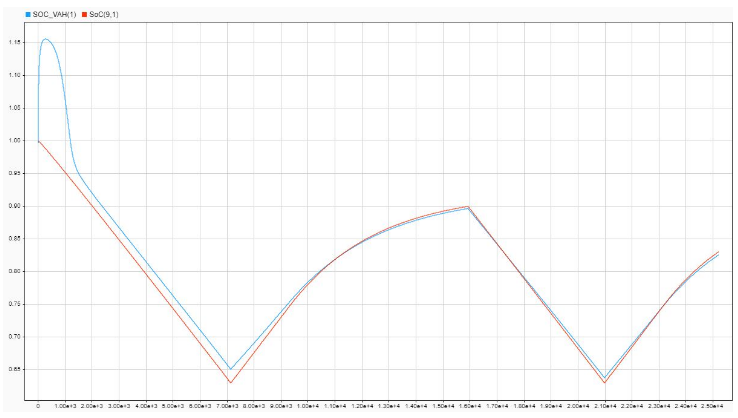


Figure 4.9. Comparison between real SOC and SOC estimate curves

Figure 4.9 shows the graphical comparison between the real state of charge evolution in time (red curve) and the computed estimate via UKF (light blue curve). At the beginning of the simulation, both curves start out at a value of 1, but the estimate quickly diverges to a maximum peak of 1.156: this behaviour is not to be feared, as a transient phase before convergence is typical of kalman filters; besides a saturation block could easily be implemented in the Simulink scheme in order to limit the estimate to a maximum value of 1. At around 1600 s, while the battery is in the middle of the first discharge phase of the simulation run, the estimate settles at a value that only differs by 2 % from the real SOC, making the computation already acceptable. Once the battery reaches a 63 % autonomy level, the CC-CV algorithm switches from discharge to charge mode and the light blue curve rapidly converges to the red one: by the time the simulation reaches the time instant 9300 s, the difference between estimate and real value is under one percent.

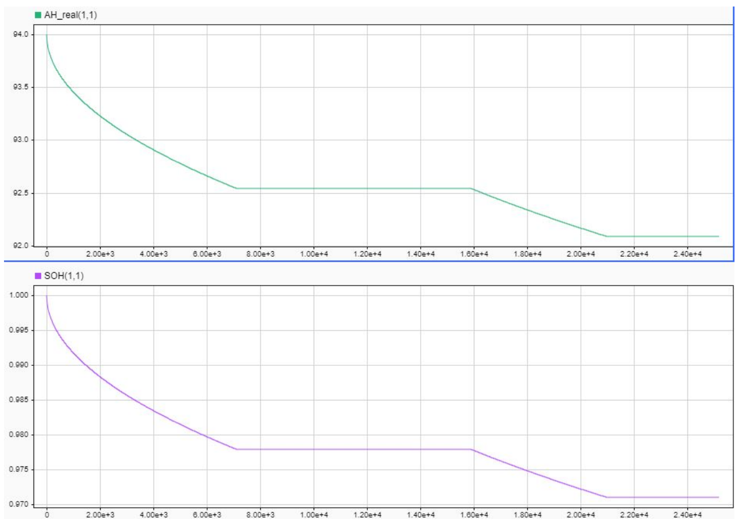


Figure 4.10. Comparison between battery capacity and SOH estimate curves

The two curves in Figure 4.10 represent the degradation of the capacity due to the fading phenomenon and the SOH estimate at the top and bottom respectively. Even though they look exactly the same, these two graphs have completely different scales: the capacity is equal to 94 Ah at $t = 0s$ and decreases depending on the value of $\sqrt{n}$ (where n is the actual number of discharge cycles) when the battery is being discharged, while it stays constant during charging. Similarly, the SOH decreases non-linearly during the first discharge phase, reaching a value of 0.9779, and starts decreasing again at $t = 15900s$, settling at a final value of 0.9710.

4.6 Fault Detection unit testing

The Fault Detection unit testing phase aimed at inducing all faults in the system to prove that the algorithm is capable of recognizing the errors under suitable conditions and triggering the switching mechanism to bring the pack back to a safe operating range and to reset to 0 the error logic signals. To do so, many simulation runs have been performed, in order to underline the faults one by one. The first simulation shows the occurrences of multiple overcurrent errors when the current provided by the battery pack to the load is uniformly distributed; it is also worth mentioning that for the first test run, the switching mechanism was deactivated because the goal was to observe whether the algorithm would detect the fault, without needing to act upon it. Table 4.6 collects the simulation setup parameters.

Simulation time	21600 s
Limit values for the uniformly distributed pack current	±80 A
Charging limit	72 A
Discharging limit	-72 A
Qualification time	4 s
Disqualification time	2 s

Table 4.6. Overcurrent fault detection simulation setup

The simulation time was set to 6 hours and the battery current was implemented by means of a uniform random number block, which outputs uniformly distributed values in the range between $\pm 80A$; in this way, the Battery Current Monitoring block can detect the fault when the current overcomes the charging and discharging limits of respectively 72 A and -72 A. Besides, to make sure the error does not occur due to the presence of noise, a debouncing mechanism was added: the fault, in fact, occurs only if the threshold is exceeded and an internal counter reaches its maximum value in a time interval of 4 s (qualification time). On the contrary, the fault is reset if the current stays below the threshold for at least 2 s (disqualification time).

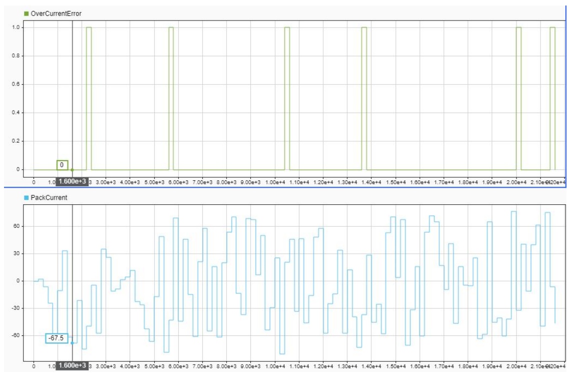


Figure 4.11. Overcurrent logic signal and battery pack current curve

Figure 4.11 shows the results obtained from the simulation: the top graph represents the overcurrent fault logic signal, which oscillates between 1 and 0, whereas the bottom curve represents the behaviour of the pack current along the time axis. Throughout the entire simulation, the fault occurs for a total of 6 times, every time the pack current value, for the same time instant, overcomes the limit, while it is reset to zero when the current comes back to the safe range. To give a numerical result we can focus on the first occurrence of the error and exploit the cursors of the Data Inspector tool: the first rising edge of the logic signal sits at 2200 s when the battery current has been sitting at a constant value of -74.6 A (below the discharging limit) for a few seconds. If we employ the cursors to measure the time interval between the two events, we obtain a value of roughly 4 s, which proves the effectiveness of the debouncing mechanism. By following the same procedure, it is possible to observe that when the current later reaches a value of -49.3 A (above the discharging limit), the logic signal switches to 0 after 2 s. The second simulation was performed with the aim of inducing an overvoltage fault and Table 4.7 collects the corresponding simulation setup parameters.

Simulation time	21600 s
Limit values for the uniformly distributed pack current	±80 A
Undervoltage limit	2.5 V
Overvoltage limit	4.2 V
Overvoltage/Undervoltage qualification time	4 s
Overvoltage/Undervoltage disqualification time	2 s

Table 4.7. Overvoltage fault detection simulation setup

The simulation time was set to 6 hours and the strategy implemented to induce high cell voltages was obtained by means of a current uniform random number block, with limit values equal to $\pm 80A$. The thresholds for under and overvoltage were instead set to 2.5 V and 4.2 V respectively, while the qualification and disqualification times were left untouched.

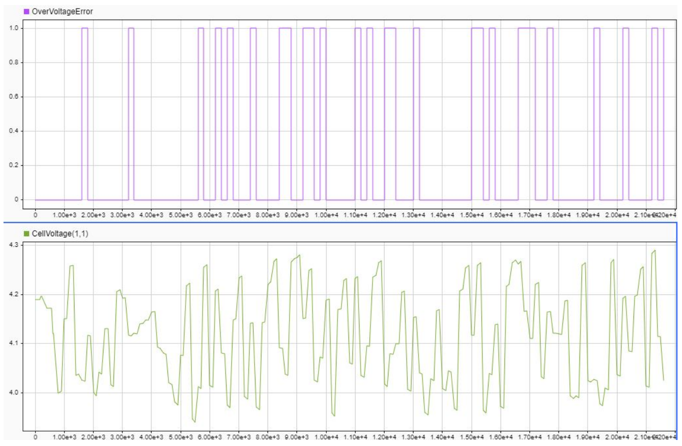


Figure 4.12. Overvoltage logic signal and cell voltage curve

Figure 4.12 reports the two most important graphs that are needed to analyze and validate the results: the overvoltage logic signal on top and the cell voltage curve at the bottom. During 6 hours, the fault occurs 21 times because, as it can be observed in the bottom graph, the voltage provided by the cells exceeds the thresholds imposed by the developer. More specifically, the fourth fault occurrence sits at 6200 s, when the voltage is at 4.21 V, thus above the overvoltage limit; if we take the cursors to measure the time needed by the logic signal to switch from 0 to 1, it can be observed that 4 s after the voltage reaches the overvoltage threshold, the fault occurs. The same fault is reset 2 s after the voltage decreases below 4.2 V.

Simulation time	21600 s
Undertemperature limit	273.15 K
Overtemperature limit	313.15 K
Ambient temperature	300.15 K
Overtemperature/Undertemperature qualification time	4 s
Overtemperature/Undertemperature disqualification time	2 s

Table 4.8. Overtemperature fault detection simulation setup

The third simulation focused on the detection of overtemperatures, along with the control logic which puts the system in a state of permanent fault, by disconnecting the battery pack from both the resistive load and the charging circuit; for this reason, during this test run the CC-CV algorithm and the switching network were enabled. The setup parameters are displayed in Table 4.8: the simulation time is 6 hours and the under and over temperature limits are 273.15 K and 313.15 K respectively. The simulated temperature for the battery surroundings is 300.15 K and the qualification and disqualification times for both errors are 4 s and 2 s.

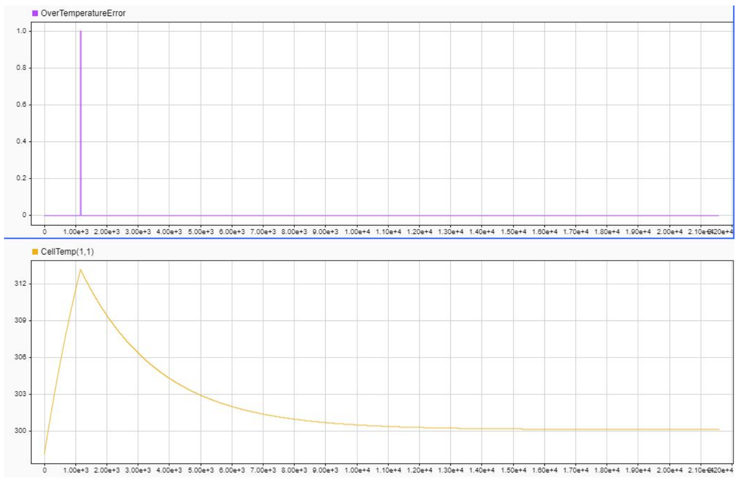


Figure 4.13. Overtemperature logic signal and cell temperature curve

The graphs represented in Figure 4.13 show the effectiveness of both the detection algorithm and the switching network: the top curve represents the error logic signal, which appears to be an impulse at 1156 s; when zooming in, we can quickly realize that, in reality, the curve instantaneously increases from 0 to 1, only to settle back to 0 for the rest of the run at 1170 s. As far as the cell temperature is concerned, it starts at 298.15 K and rapidly increases until it reaches the overtemperature limit at 1152 s, exactly 4 s before the fault is triggered; shortly after, a peak temperature of 313.21 K is reached and, as soon as the logic signal is reset to 0, the temperature curve starts decreasing in a non-linear fashion and it tends asymptotically to 300.15 K, the environment temperature. The graphical data confirms that the detection mechanism is perfectly functioning and respects the threshold values set by the user; moreover, the presence of only one fault occurrence, along with the temperature tendency to equalize the temperature of the surroundings, indicates that one temperature related error is enough to trigger the permanent fault state, which opens both switches and isolates the pack from both the CC-CV and the load, inducing the cell temperature to come back to a safe operating range. In order to test the second condition that makes the system transition to a permanent fault state, a persistent overcurrent request was induced on the battery.

Simulation time	21600 s
Charging limit	72 A
Discharging limit	-72 A
Load	0.04 Ω
Qualification time	4 s
Disqualification time	2 s

Table 4.9. Overcurrent fault detection and permanent fault simulation setup

Table 4.9 collects the simulation setup parameters: the duration of the simulation is kept fixed at 6 hours and the charging and discharging limits are set to $\pm 72A$. The qualification and disqualification times are set to 4 s and 2 s respectively, while the load value is fixed 0.04 Ω, a value low enough to theoretically induce the cells to provide a current that overcomes the threshold.

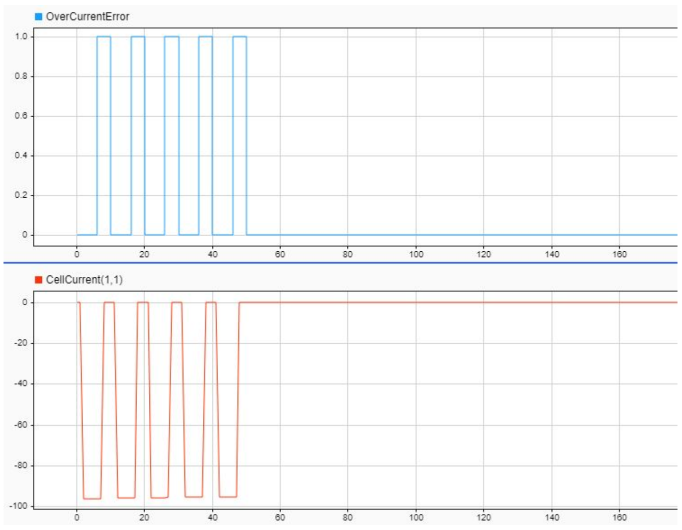


Figure 4.14. Overcurrent logic signal and cell current curve

The data collected from this simulation run is depicted in Figure 4.14, where the overcurrent logic signal is represented by the top graph while the bottom one is the corresponding behaviour of the cell current. The cell current starts out at a value of 0 A but rapidly changes to -96 A, a value that is over the defined discharging limit, thus triggering the occurrence of an overcurrent fault, which switches from 0 to 1. At that point, the pack transitions to a temporary fault state, where the battery is isolated from the load as demonstrated by the fact that the current is cut to 0 A, and, after a time interval equal to the disqualification time, the fault is reset. This cycle is repeated for a total of 5 times, meaning that the overcurrent fault is triggered just as many times. As it was declared in the Stateflow chart of the detection logic, a counter keeps memory of how many voltage or current related errors occur and forces the system into a permanent fault state if said value reaches 5. After the fifth occurrence, the pack does in fact transition to such state, because the error remains constant to a logic value of 0 and, more importantly, the cells stop providing current for the rest of the simulation.

4.7 Complete model testing and validation

After completing the unit testing phase, a final step for the validation of the work carried out in this thesis was performed by putting together all the developed functions, in order to verify their effectiveness when being activated simultaneously, as it happens in a real-life BMS inside an EV.

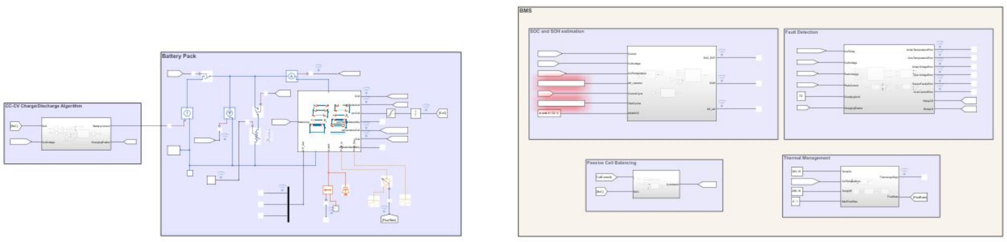


Figure 4.15. Complete model Simulink schematic

Before running the simulations, a new Simulink file was created and, as shown in Figure 4.15, all the needed components were placed and labeled appropriately: on the left-hand side of the image is an area that contains the CC-CV algorithm, in the middle is the battery pack along with the switching network, while on the right is an area labeled BMS, which is made up of four subsystems, one for each of the major functions that were developed. The first test aimed at triggering the simultaneous activation of the SOC and SOH estimations and the passive cell balancing mechanism.

Simulation time	43200 s
Balancing resistor	6 Ω
Load	50 Ω
First parallel assembly initial SOC	0.69
Second parallel assembly initial SOC	0.715
Third and fourth parallel assembly initial SOC	0.7

Table 4.10. Complete model first test simulation setup

All the information regarding the simulation setup can be found in Table 4.10: the simulation time was set to 12 hours, while the balancing resistor, through which the high charge cells release a portion of their energy, and the resistive load were set to 6Ω and 50Ω respectively. Moreover, to have the cell balancing mechanism activate, different initial SOC values were chosen to recreate an imbalanced scenario: 69% for the first assembly, 71.5% for the second and 70% for the third and fourth.

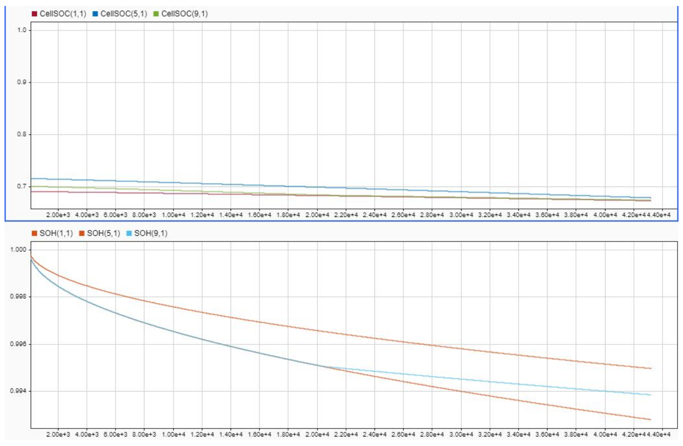


Figure 4.16. Complete model first test passive cell balancing and SOH estimates

In Figure 4.16, it can be observed in the top graph that the passive cell balancing mechanism is correctly activated, as the initial SOC imbalance among cells is compensated for, inducing the curves to converge to the lowest SOC value. The graph at the bottom represents instead the estimates of the state of health for one cell of each parallel assembly: all curves start at 1 and then decrease non-linearly with respect to the number of discharge cycles n. It is also clear how the curves descend at a different rate, due to the fact that the cell balancing mechanism forces the cells to discharge differently, in order to achieve the desired balance.

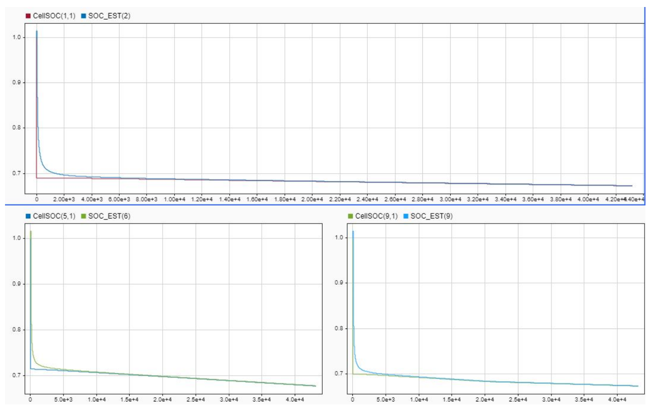


Figure 4.17. Complete model first test SOC estimates

As far as the SOC estimation is concerned, Figure 4.17 shows that the estimates obtained from the UKF rapidly converge to the real values of cell state of charge: by exploiting the cursor available in the Data Inspector tool, it can be measured that after 900 s, the SOCs are overestimated by only 1% while after 1500 s the discrepancy is already below it. A second test was later performed to test the simultaneous activation of the fault detection and the thermal management.

Simulation time	43200 s
Balancing resistor	6 Ω
Variable load	[50, 0.01, 6] Ω
Max flow rate	0.1 m^3/s
Ambient temperature	298.15 K

Table 4.11. Complete model second test simulation setup

Table 4.11 collects the simulation setup parameters for the second and final test of the complete model: the simulation time was set to 12 hours and the balancing resistor to 6 Ω; the load was made variable by connecting a repeating sequence stair block to a variable resistor and the max flow rate for the thermal management was set to 0.1 m^3/s, along with an ambient temperature of 298.15 K.

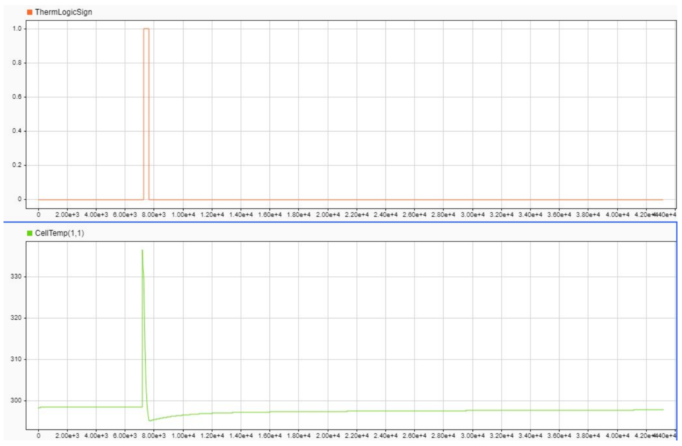


Figure 4.18. Complete model second test thermal logic signal and cell temperature curve

The correct activation of the thermal management is observable in Figure 4.18, where the top graph represents the thermal logic signal while the bottom one the behaviour of the cell temperature in time. At $t = 7300$ s, the thermal logic signal toggles from a value of 0 to a value of 1, when the cell temperature has already overcome the activation temperature, thus activating the mechanism; the temperature then proceeds to decrease and at $t = 7640$ s, when the temperature is at 295 K, the logic signal is reset to 0 and the mechanism is switched off.

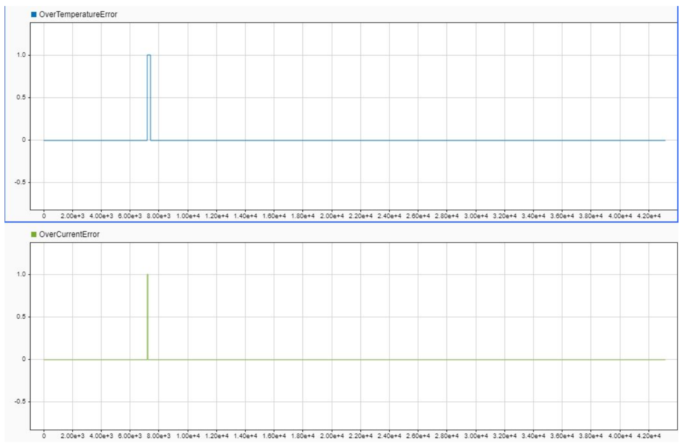


Figure 4.19. Complete model second test overtemperature and overcurrent logic signals

As far as the fault detection is concerned, Figure 4.19 shows that two errors occur consequently: the first one, represented in the bottom graph, is the overcurrent error, which occurs at $t = 7200$ s, while the second is the overtemperature error, which is triggered immediately after.

Chapter 5

Conclusions and future works

5.1 Conclusions

The development work carried out in this thesis and explained in Chapter 3 was heavily tested via countless simulations, whose results have already been reported in Chapter 4. In order to conclude the project, however, it is necessary to discuss and critically analyze data, underlining the highlights and the best assets of the software, as well as the weaknesses that have been detected. The key-points will be divided into different sections depending on the BMS function they are referred to and will be briefly discussed and broken down.

- The passive cell balancing function proved to be effective in bringing balance among the battery pack cells; the simulation data shows in fact that a simple 6 Ω balancing resistor can dissipate the excessive energy from highly charged cells and allow the overall system to reach a balanced status in roughly 4.5 hours. Even though this time interval might seem too long, the simple nature of the mechanism, which dissipates the electrical charge into heat rather than redistributing it among cells, and the stringent threshold constraint, which foresees an SOC discrepancy of only 10^{-3}, must be taken into account. A note to be taken also regards the initial SOC values that were set during the testing phase to simulate an imbalanced state: the maximum difference in terms of state of charge was of only 2 % (as it can be observed in Chapter 4) because some of the preliminary tests showed that an initial SOC difference in the order of unitary percentage points could be compensated for by the algorithm in reasonable times, while the mechanism would fail, meaning that it would not fulfil its purpose, with an imbalance in the order of tens of percentage points.

- The control logic developed for the thermal management of the battery pack, despite being conceptually simple, is capable of maintaining the system in a safe operating range and, in view of a possible real-life implementation, keeps the number of pump activations feasible. A purely thermal logic that compares the cell temperature with a threshold value induces the pump to activate too many times, making the implementation not feasible. Integrating two additional criteria, as it was done in this work, is therefore a good solution to solve said issue: because thermal phenomena have slower dynamics with respect to electrical ones, the temperature constraints have been loosened, by integrating a timer that measures how long the system persists in a state where the temperature is over the activation threshold and eventually activates the pump when a time interval in the order of hundreds of seconds is reached. Even though the cells will always overcome the activation temperature before being cooled down, the aforementioned solution grants the cooling system an additional one hundred seconds before switching on the pump, thus reducing the number of pump activations. The second important piece of the algorithm regards the coolant flow rate modulation, a mechanism that reduces the rate at which the coolant flows through the cooling plate linearly with the temperature. As a result, once the mechanism is active the system takes longer to reach the deactivation temperature (especially if compared to a scenario where the flow rate is always maximum) but, once again, the pump switches on and off a fewer times.

- The estimation of the state of charge, which is pivotal for the development of any BMS, was implemented by means of a Kalman Filter, whose aim was to compute an SOC with a satisfactory accuracy. It appeared clear from the beginning that a regular KF could not be employed due to the non-linearity of the system, therefore the solution was to either opt for an EKF or a UKF. After some initial failed tests, the Extended Kalman Filter, despite being the least demanding option in terms of computational power, was discarded as the first order Taylor expansion approximation that it performs results in poor SOC estimates. On the other hand, the implemented UKF, performs an approximation that is equivalent to a second order Taylor expansion and, even though it requires a greater computational effort, it outputs an estimate that converges to the real SOC in under 3 minutes with an error of less than 1 %. As far as the SOH is concerned, the estimation was implemented via analytical formulation of the fading battery capacity; a Simulink subsystem was in fact created to output the pack capacity, which degrades overtime as a function of $\sqrt{n}$, where n is the number of discharge cycles. The actual estimate is then computed as a simple ratio between the different capacity values reported in Equation 3.6.

- The fault detection function proved to be effective in recognizing errors regarding voltage, current or temperature anomalies. The algorithm built around the blocks available in the Simscape library can in fact toggle logic error signals depending on the relative value of said physical quantities with respect to suitable thresholds. Because in real-life applications the noise is overimposed on any signal, possibly generating oscillations across the threshold, a debouncing mechanism was implemented by means of a counter, which sets the errors to 1 or resets them to 0 if and only if a symptom persists until a certain value is reached. In this way, not only can the BMS detect a fault, but it can be sure that it was not triggered by noise. The additional switching network, along with its control logic developed via Stateflow chart, was then added to have the battery pack switch between the charging and discharging states, but also to react to the presence of faults by forcing the system back into a safe operating state. In the presence of overvoltage, undervoltage or overcurrent the battery is in fact either disconnected from the load or the CC-CV circuit, until it is safe to go back to a normal operating condition, while upon the occurrence of overtemperature, undertemperature or five consecutive voltage or current related faults, all switches open, leaving the system completely isolated and in a permanent fault state.

5.2 Future works

The work carried out in this thesis represents a solid foundation for possible future works aimed at improving the custom BMS model or even produce the software for implementation purposes.

- In the presence of more powerful hardware, the first logical step would be to run the unmodified version of this BMS with a rescaled model of the battery pack employed in this activity; as it was discussed in Chapter 3, the system that simulates the behaviour of the battery was modelled as a system with a 8s4p topology, meaning that the total number of cells is only 32. The system is therefore underdimensioned to compensate for the limited computational power, however, to faithfully simulate a real life EV battery pack, an improvement could be made by creating a system with a number of cells in the order of thousands and with an appropriate cell topology.

- For balancing purposes, a more efficient algorithm in terms of energy management should be prioritized. The current function implements a simple passive solution, therefore an updated active mechanism could be added to allow energy to flow from overcharged cells to other sections of the pack, via suitable networks of reactive electrical components (capacitors and inductors) appropriately arranged.

- A different approach to estimating the SOC of the battery could be to replace the filtering algorithm with a machine learning technique. A Neural Network, for instance, could be implemented to better capture the non-linear characteristics of the system and compute a more accurate estimate of the state of charge. However, this can only be possible provided that a huge amount of data is available to train the model.

- Aside from a brief theoretical analysis of the standard ISO 26262, this work does not include the Functional Safety requirements. A possible future improvement of the project might address said requirements.

- The final step to possibly implement the BMS in a real-life system is to generate the code from the Simulink file of the complete model: by exploiting the Simulink Coder tool, it is in fact possible to recover C or C++ code which can be eventually built and run.

Bibliography

[1] Wiki, “Automotive battery,” Wikipedia, Tech. Rep. [Online]. Available: https://en.wikipedia.org/wiki/Automotive_battery#:~:text=An%20automobile%20battery%20is%20an,solution%2C%20which%20is%20the%20electrolyte

[2] D. Lo Re, “Progettazione del layout per la produzione di batterie al litio: analisi dello stato dell’arte,” Master’s thesis, Politecnico di Torino, 2022.

[3] Wikipedia, “Electric vehicle battery,” Wikipedia, Tech. Rep. [Online]. Available: https://en.wikipedia.org/wiki/Electric_vehicle_battery

[4] Keyence, “Comparison of different types of electric vehicle battery cells,” Keyence, Tech. Rep. [Online]. Available: https://www.keyence.co.in/products/marker/laser-marker/resources/laser-marking-resources/comparison-of-different-types-of-electric-vehicle-battery-cells.jsp

[5] Wikipedia, “Lead-acid battery,” Wikipedia, Tech. Rep. [Online]. Available: https://en.wikipedia.org/wiki/Lead-acid_battery#:~:text=The%20lead%2Dacid%20battery%20is,have%20relatively%20low%20energy%20density.

[6] Varta, “Come funziona una batteria e come è composta?” Varta, Tech. Rep. [Online]. Available: https://batteryworld.varta-automotive.com/it-it/come-funziona-una-batteria-auto

[7] S. Licoccia, “Batterie,” Università degli Studi di Tor Vergata, Tech. Rep. [Online]. Available: https://didattica.uniroma2.it/files/index/insegnamento/148670-%0AChimica-Per-Lenergia

[8] Wikipedia, “Nickel-cadmium battery,” Wikipedia, Tech. Rep. [Online]. Available: https://en.wikipedia.org/wiki/Nickel%E2%80%93cadmium_battery

[9] STIHL, “Le batterie moderne sono soggette all’effetto memoria?” STIHL, Tech. Rep. [Online]. Available: https://www.stihl.it/it/diy-stihl/manutenzione-attrezzi-da-giardino/informazioni-batterie/effetto-memoria-batteria#:~:text=L’effetto%20memoria%20%C3%A8%20una,solo%20quella%20quantit%C3%A0%20di%20energia.

[10] M. Hannan, M. Lipu, A. Hussain, and A. Mohamed, “A review of lithium-ion battery state of charge estimation and management system in electric vehicle applications: Challenges and recommendations,” *Renewable and Sustainable Energy Reviews*, 2017. [Online]. Available: https://doi.org/10.1016/j.rser.2017.05.001.

[11] Bacancy, "Everything you need to know about cell balancing," Bacancy Systems, Tech. Rep. [Online]. Available: https://bacancysystems.com/blog/cell-balancing-and-its-types

[12] NAZSolarElectric, "Lithium batteries: Bms theory - cell balancing," Solar Electric, Tech. Rep. [Online]. Available: https://www.solar-electric.com/learning-center/bms-theory-cell-balancing/

[13] MonolithicPowerSystems, "Battery balancing: A crucial function of battery management systems," Monolithic Power Systems, Tech. Rep. [Online]. Available: https://www.monolithicpower.com/en/learning/resources/battery-balancing-a-crucial-function-of-battery-management-systems

[14] BioLogic, "Why use electrochemical impedance spectroscopy (eis) for battery research?" BioLogic, Tech. Rep., 2023. [Online]. Available: https://www.biologic.net/topics/why-use-electrochemical-impedance-spectroscopy-for-battery-research/

[15] L. Middlemiss, A. Rennie, R. Sayers, and A. West, "Characterisation of batteries by electrochemical impedance spectroscopy," *Energy Reports*, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2352484720303103

[16] Wikipedia, "Kalman filter," Wikipedia, Tech. Rep. [Online]. Available: https://en.wikipedia.org/wiki/Kalman_filter

[17] ——, "Neural network (machine learning)," Wikipedia, Tech. Rep. [Online]. Available: https://en.wikipedia.org/wiki/Neural_network_(machine_learning)

[18] D. Faverato, "Virtual sensing for the estimation of the state of health of batteries," Master's thesis, Politecnico di Torino, 2020.

[19] D. Haifeng, W. Xuezhe, and S. Zechang, "A new soh prediction concept for the power lithium-ion battery used on hevs," *2009 IEEE Vehicle Power and Propulsion Conference*, 2009. [Online]. Available: https://doi.org/10.1109/VPPC.2009.5289654

[20] B. Bharat and M.-Y. Chow, "The state of the art approaches to estimate the state of health (soh) and state of function (sof) of lithium ion batteries," *2015 IEEE 13th International Conference on Industrial Informatics (INDIN)*, 2015. [Online]. Available: https://doi.org/10.1109/INDIN.2015.7281923

[21] A. Zenati, P. Desprez, and H. Razik, "Estimation of the soc and the soh of li-ion batteries, by combining impedance measurements with the fuzzy logic inference," *IECON 2010*, 2010. [Online]. Available: https://doi.org/10.1109/IECON.2010.5675408

[22] Z. Chenghui, Z. Yun, and L. Yan, "A novel battery state-of-health estimation method for hybrid electric vehicles," *IEEE/ASME transactions on mechatronics*, 2015. [Online]. Available: https://doi.org/10.1109/TMECH.2014.2371919

[23] MathWorks, "What is a battery thermal management system?" MathWorks, Tech. Rep. [Online]. Available: https://it.mathworks.com/discovery/

battery-thermal-management-system.html

[24] YDP, "How it works: Battery thermal management system," MODINE, Tech. Rep., 2023. [Online]. Available: https://www.modineev.com/how-it-works-battery-thermal-management-system/

[25] M.-K. Tran and M. Fowler, "A review of lithiu-ion battery fault diagnostic algorithms: Current progress and future challenges," *ResearchGate*, 2020. [Online]. Available: https://doi.org/10.3390/a13030062

[26] D. Ouyang, M. Chen, J. Liu, R. Wei, J. Weng, and J. Wang, "Investigation of a commercial lithium-ion battery under overcharge/over-discharge failure conditions," *RSC Advances*, 2018. [Online]. Available: http://dx.doi.org/10.1039/C8RA05564E

[27] D. Lyu, B. Ren, and S. Li, "Failure modes and mechanisms for rechargeable lithium-based batteries: a state-of-the-art review," *Acta Mechanica*, 2018. [Online]. Available: https://doi.org/10.1007/s00707-018-2327-8

[28] X. Feng, J. Sun, M. Ouyang, F. Wang, X. He, L. Lu, and H. Peng, "Characterization of penetration induced thermal runaway propagation process within a large format lithium ion battery module," *Journal of Power Sources*, 2015. [Online]. Available: https://doi.org/10.1016/j.jpowsour.2014.11.017

[29] Y. Liu and J. Xie, "Failure study of commercial lifepo4 cells in overcharge conditions using electrochemical impedance spectroscopy," *Journal of the Electrochemical Society*, 2015. [Online]. Available: https://doi.org/10.1149/2.0911510jes

[30] L. Yao, Z. Wang, and J. Ma, "Fault detection of the connection of lithium-ion power batteries based on entropy for electric vehicles," *Journal of Power Sources*, 2015. [Online]. Available: https://doi.org/10.1016/j.jpowsour.2015.05.090

[31] G. Xia, L. Cao, and G. Bi, "A review on battery thermal management in electric vehicle application," *Journal of Power Sources*, 2017. [Online]. Available: https://doi.org/10.1016/j.jpowsour.2017.09.046

[32] E. V. Database, "Nissan leaf," ev-database.org, Tech. Rep. [Online]. Available: https://ev-database.org/car/1106/Nissan-Leaf

[33] ——, "Bmw i3 120ah," ev-database.org, Tech. Rep. [Online]. Available: https://ev-database.org/car/1145/BMW-i3-120-Ah

[34] Quora, "How many cells are there in a fully electric car battery?" Quora, Tech. Rep. [Online]. Available: https://www.quora.com/How-many-cells-are-there-in-a-fully-electric-car-battery#:~:text=A%20standard%20range%20about%203%2C000,and%20rectangular%20rather%20than%20cylindrical

[35] EVlithium, "Samsung sdi 3.7v 94ah nmc prismatic battery cell," EVlithium, Tech. Rep. [Online]. Available: https://www.evlithium.com/nmc-battery/samsung-sdi94-94ah-nmc-battery.html

[36] MathWorks, “Battery (table-based),” MathWorks, Tech. Rep. [Online]. Available: https://it.mathworks.com/help/sps/ref/batterytablebased.html

[37] ——, “U-shaped channels,” MathWorks, Tech. Rep. [Online]. Available: https://it.mathworks.com/help/simscape-battery/ref/ushapedchannels.html

[38] ——, “Battery cc-cv,” MathWorks, Tech. Rep. [Online]. Available: https://it.mathworks.com/help/simscape-battery/ref/batterycccv.html

[39] ——, “Passive cell balancing,” MathWorks, Tech. Rep. [Online]. Available: https://it.mathworks.com/help/simscape-battery/ref/passivecellbalancing.html

[40] ——, “Soc estimator (kalman filter, variable capacity),” MathWorks, Tech. Rep. [Online]. Available: https://it.mathworks.com/help/simscape-battery/ref/socestimatorkalmanfiltervariablecapacity.html

[41] ——, “Soh estimator (capacity-based),” MathWorks, Tech. Rep. [Online]. Available: https://it.mathworks.com/help/simscape-battery/ref/sohestimatorcapacitybased.html

[42] ——, “Battery temperature monitoring,” MathWorks, Tech. Rep. [Online]. Available: https://it.mathworks.com/help/simscape-battery/ref/batterytemperaturemonitoring.html

[43] ——, “Battery voltage monitoring,” MathWorks, Tech. Rep. [Online]. Available: https://it.mathworks.com/help/simscape-battery/ref/batteryvoltagemonitoring.html

[44] ——, “Battery current monitoring,” MathWorks, Tech. Rep. [Online]. Available: https://it.mathworks.com/help/simscape-battery/ref/batterycurrentmonitoring.html