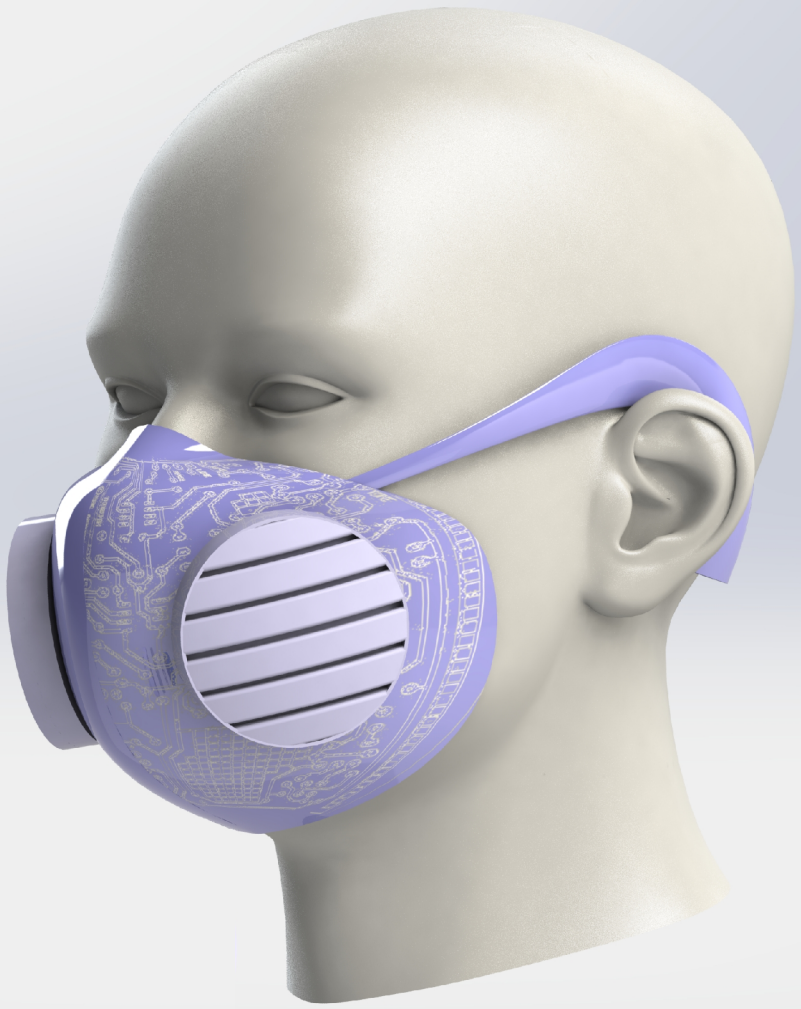
EE3L11

SPPE
Smart Personal Protective Equipment

On-Board Power Management

H.J. Donkers (4475941)
C. Zwart (4430867)



TU Delft

EE3L11

Smart Personal Protective Equipment: On-Board Power Management

by

Huub Donkers
Lotte Zwart

to obtain the degree of Bachelor of Science
at the Delft University of Technology,

Authors:	H.J. Donkers	4475941
	C. Zwart	4430867

Project duration:	April 20, 2020 – July 3, 2020	
Supervisor:	Dr. ing. H. Van Zeijl	
Thesis committee:	Prof. Dr. ir. W.D. Van Driel,	TU Delft
	Dr. ing. H. Van Zeijl,	TU Delft
	Dr. ir. S. Vollebregt	TU Delft

With contribution from the entire SPPE group in Chapter 1.
BSc. Electrical Engineering 2020 BAP group C:

R. P. N. Bakker	4662482
M. J. H. Brouwers	4728181
H. J. Donkers	4475941
M. Goddijn	4666968
J. J. M. Lut	4698207
C. Zwart	4430867

TU Delft

Abstract

The COVID-19 pandemic caused a shortage of Personal Protective Equipment for Healthcare personnel. This project aims to aid in this shortage by extending the lifetime of the filter material used in a mask. This is done in the form of an SPPE, a Smart Personal Protective Equipment. This face mask has two smart filter heads that are modular and contain UVC LEDs to disinfect the filter, a control system to control the LED's radiative power and an on-board power management system. The latter is the focus of this thesis.

The implementation of an on-board power management system for a smart personal protection face mask was designed in three stages: (1) researching existing theory about battery management, (2) implementing and verifying a system design in Simulink and (3) making a PCB design and selecting off-the-shelf components. The goal of this thesis is to make a complete design of a functional battery management system, that supplies required power to the rest of the system, ensuring safe battery operation and aiming to maximize battery life. In this, the design has succeeded as almost all requirements are met. The result is a PCB design that can be made and combined with two other subgroups to create a Smart Personal Protection face mask. The main findings were a different and possibly new approach to estimating the State of Charge of a battery and designing a Battery management system for low power applications in a small form factor as opposed to battery management systems for electrical vehicles, which are common today.

Cover image by Dr. Ing. H. W. van Zeijl

Preface

Before you lies the thesis "On-Board Power management" that is made as a module for the Smart Personal Protective Equipment bachelor graduation project. This thesis has been written to fulfil the requirement for the EE3L11 Graduation Project which is the final course in the Bachelor Electrical Engineering at the Delft University of Technology. We were engaged in researching and writing this thesis from April to June 2020.

This project was not the originally agreed upon project. Due to the COVID-19 pandemic that the world and we alike faced, it became unfeasible to undertake the original project. Therefore, our supervisor, Dr. ing. Henk van Zeijl, inspired by the situation, came up with the idea to make a mask that can neutralize the virus using UVC LEDs. The whole project-group (consisting of 6 people) was very enthusiastic about this and we decided to write a project proposal, that was accepted by the BAP Coordinator.

This project was not straightforward to execute as a boundary condition was set that no physical prototyping or testing was allowed to be done. Therefore, this thesis turned out to be a bit unconventional. The result, normally, is a working prototype that has been tested and tweaked. We, however, have gone through two separate design stages and our final product is a PCB with a test plan. This makes the project very theoretical as every step that you take has been thought out two or three times to ensure that it would work as you cannot test this. For the support and guidance in this, we would like to thank our supervisor Dr. ing. Henk van Zeijl to help guide us in the right direction and come up with ideas that could be implemented. He also made a 3D model of our final product as an artist impression, for which we are very grateful, as we could not have done this on our own in this detail.

We are also grateful to our fellow group members Jasper-Jan Lut, Michael Goddijn, Marcel Brouwers and Roy Bakker which worked on the other submodules. The collaboration was flawless and we could always lay out our problems to the rest of the group to receive feedback.

We hope you enjoy reading this thesis,

Huub Donkers & Lotte Zwart
June 24th, 2020

Glossary

Ampere-hour [Ah]	Unit of electric charge, charge transferred by a steady current of one ampere flowing for one hour, or 3600 coulombs.
Battery Cycle	A round of full charge and full discharge of a battery.
BMS	Battery Management System
CCCV	Constant Current Constant Voltage
Cell charge current [A]	The current supplied to the battery cell during charging.
Cell discharge current [A]	The current supplied by the battery cell during discharge.
Cell temperature [°C]	The internal temperature of the battery cell.
Cell voltage [V]	The voltage difference between the terminals of the battery cell.
Charge-rate	Charge or discharge rate of a battery. 1C = capacity/h [A]
CPCV	Constant Power Constant Voltage
DC	Direct Current
LED	Light Emitting Diode
LIB	Lithium-ion battery
Lifetime [years]	The time during which the product remains integer and usable for its primary function for which it was conceived and produced.
OBPM	On-Board Power Management
Operating time [hours]	The time the power system can deliver power to the complete system from a full battery before it needs to be recharged.
Output voltage ripple [V]	The peak-to-peak voltage difference of the desired output voltage.
PCB	Printed Circuit Board
Power efficiency [%]	The conversion rate of electrical energy from the battery to the load: $\eta = \frac{P_{load}}{P_{bat}}$
PPE	Personal Protection Equipment
PWM	Pulse Width Modulation
SaC	Sensing and Control
SOC [%]	State-Of-Charge
SOC deviation [%]	The percentile deviation of the estimated state-of-charge compared to the true state-of-charge.
SOH [%]	State-Of-Health
SPPE	Smart Personal Protection Equiment
UVC	Ultra violet C (wavelength 200-280 nm)
UVGI	Ultra Violet Germicidal Irradiation

Contents

1 Project Introduction 1
- 1.1 The SPPE project overview. 2
- 1.2 State of the art Analysis 4
- 1.3 Problem definition 5
- 1.4 Thesis synopsis 5

2 Program of Requirements 6
- 2.1 SPPE requirements 6
- 2.2 System requirements 6

3 System Design 8
- 3.1 Battery Types 8
- 3.2 State of Charge 10
- 3.3 State of Health 11
- 3.4 Charge algorithm 12
- 3.5 Battery protection. 12
- 3.6 Output regulation 14
- 3.7 Safety standards. 15

4 System Modelling and Verification 16
- 4.1 Overview 16
- 4.2 Battery Model. 17
- 4.3 Battery Management System 18
- 4.4 Charge module 23
- 4.5 Protective measures. 24
- 4.6 Output conversion 26
- 4.7 Human interaction 28
- 4.8 Simulation results and discussion. 30

5 Design Implementation 32
- 5.1 Battery 32
- 5.2 Battery gauge 33
- 5.3 Microcontroller 33
- 5.4 Battery protection. 34
- 5.5 Charging circuit. 35
- 5.6 Output regulation. 35
- 5.7 Human Interaction 36
- 5.8 Full system integration 37
- 5.9 Test plan 38
- 5.10 Discussion 39

6 Discussion 40
- 6.1 Assessment of requirements 40
- 6.2 Differences Simulink model and implementation. 41

7 Conclusion & Recommendations 42
- 7.1 Recommendations 42

Bibliography 43

A Battery model parameters 46

B LED Truth Table 47

C Load model results 48

D Testbench results 49

E Schematic of the circuit design 50

F Simulink model 51

G Simulink Functions 61

H ATmega328P Code 67

1 Project Introduction

The EE3L11 GRADUATION PROJECT forms the last course in the curriculum of the Bachelor Electrical Engineering of the Delft University of Technology. In the project, a group of six students investigates an electrical engineering challenge and develops a solution to it. Ideally, the solution is an electrical system, which is assembled into a prototype and tested by the students. However, due to the recent developments concerning the COVID-19 pandemic, the university decided that building the prototype is prohibited. Instead, the group must create a solution, i.e. a design, based on literature review and simulations.

In the time that this thesis is conducted the world is captured by the COVID-19 pandemic. The pandemic demonstrates the need for respiratory protection in health institutions to reduce the risk of infection for the workers. Particularly the shortage of face masks [1], generally designed for single-use, leads to the question if microelectronics could be applied to make air filters smarter in order to extend their use and improve their protection.

To design a smart, self-sterilizing air filter targeting a virus, the properties of that virus must be known. However, The virus that causes COVID-19 is a novel coronavirus, called SARS-CoV-2. As it is new to the scientific world, there is currently very limited data available. In this century, two other coronaviruses have caused epidemics. In 2003 the SARS-coronavirus was the cause of the *severe acute respiratory syndrome* (SARS) outbreak. From 2012 until present day, the MERS-coronavirus circulated and caused the *Middle East respiratory syndrome* (MERS) [2].
The authors aim to use the most relevant data in this thesis regarding the novel coronavirus. When it is needed, information regarding the other coronaviruses or viruses of different kinds is used. Which virus is considered in given data will be clearly indicated.

Bacteria, viruses and other pathogens can be killed or inactivated by *ultra violet germicidal irradiation* (UVGI) [3]. Today, several techniques are available to irradiate pathogens with ultraviolet (UV) light in order to disinfect air, surfaces, and drinking water. These techniques created applications such as air filtering systems with UV lighting, cabinets in which hospitals can sterilize their face masks, and continuous overhead UV lighting in laboratories [3]. However, none of these applications contain a small and mobile implementation of UVGI.

It is important to emphasize that the technique of UVGI sterilization can be implemented in different form factors, such as stationary applications where sterile air is crucial. Here, UVGI sources are installed as a filter module in an air conduct. These devices are beneficial in the medical and food industry. Other domains of interest are for example military as a form of protection or water filtration systems to remove any unwanted pathogens. The SPPE design is aimed such that its techniques are easily applied in other UVGI devices.

For this graduation project, the group proposes the use of Smart Personal Protective Equipment, SPPE in short, in order to increase the effectiveness and lifetime of the filters used in standard PPE's. The SPPE is proposed as a filter module with an in-situ disinfection functionality based on UVGI. The key to the SPPE is the integration of a UV source in the filter combined with sensors to monitor the sterilisation process and filter performance. Two SPPE modules are connected to a face mask to create a Smart Personal Protective face mask.

1.1. The SPPE project overview.

In this section, an overview of the SPPE is provided to the reader. The aim is for the reader to obtain a clear overview of the entire project. How the project has been divided into subsystems with its mutual connections, and what the parameters of each subsystem are. For each subsystem, a separate report is written with this chapter as a general introduction. In the following chapters of this report, a detailed explanation is given on the work which has been done to tackle the respective problems of this thesis group.

1.1.1. Design scope

The final design of the SPPE consists of two smart filter modules which are applied to a face mask. Each module can operate autonomously and has a replaceable filter. When the SPPE is operational, the filter material is periodically irradiated with UVC light to neutralize pathogens that have been captured by the filter material. Sensors measure the air pressure, relative humidity, and temperature to compute the required irradiation intensity and monitor the quality of the filter. When the filter has reached the end of its life, the SPPE will notify the user to have it replaced. The filter module is battery powered and can operate for eight hours before needing to be recharged. Figure 1.1 shows the filter module's components implemented in a cross-sectional view.

Due to COVID-19 restrictions, a physical demonstrator of the SPPE was not realized. However, Figure 1.2 gives an impression of what the SPPE would look like.

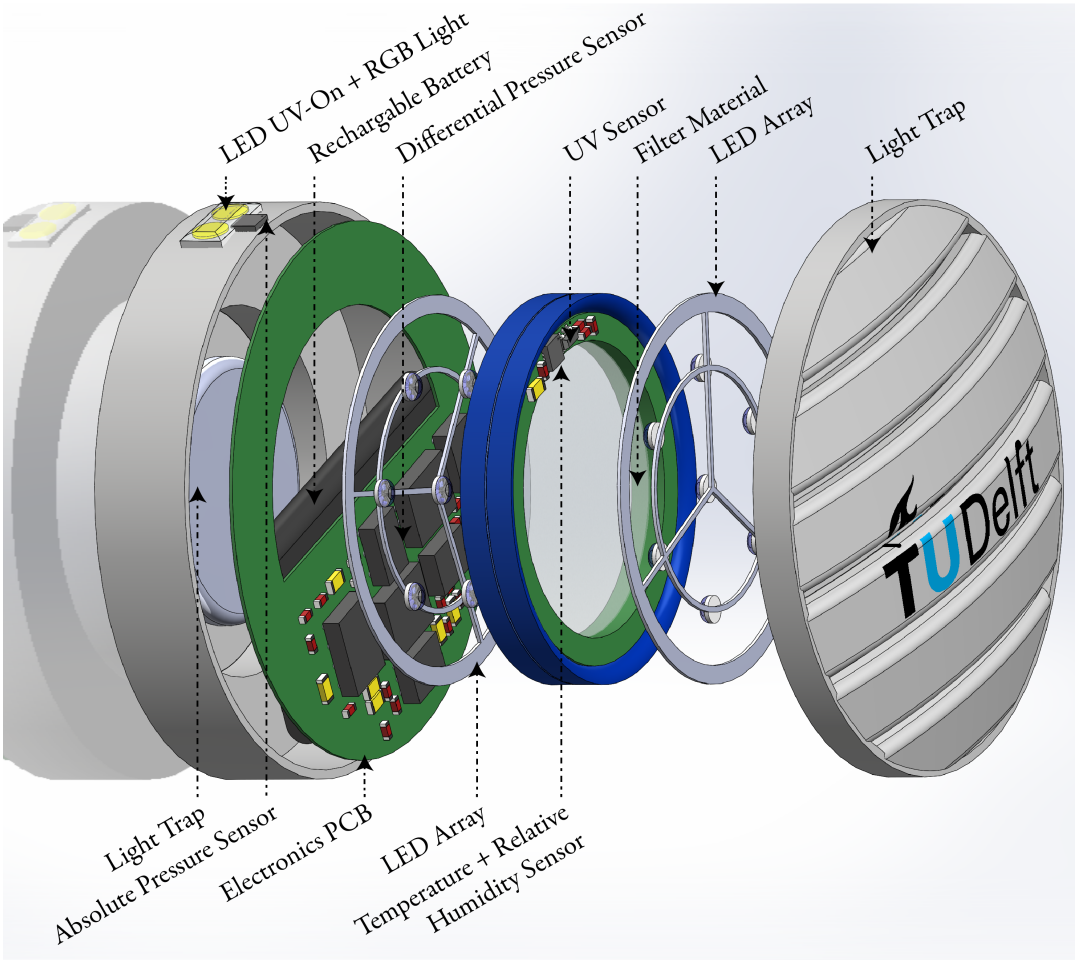
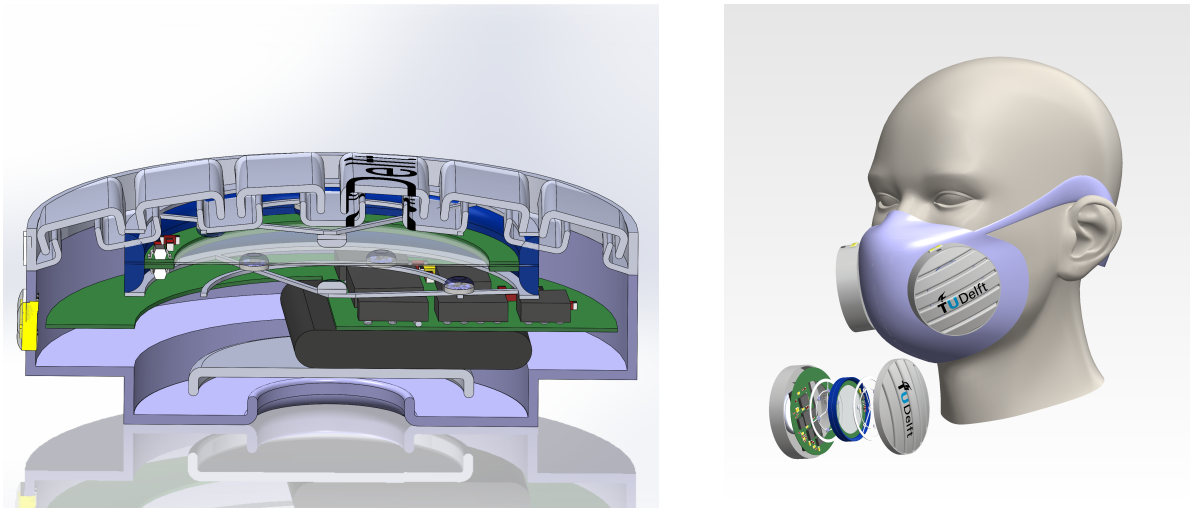


Figure 1.1: Exploded view of the SPPE. It includes the 2 air traps, electronics PCB, both LED arrays and filter material. Combined with integrated pressure, relative humidity, temperature and UV sensors.



(a) Cross-sectional view of the SPPE.

(b) SPPE as connected on a face mask.

Figure 1.2: Artist impression of the SPPE.

1.1.2. Division in submodules

The graduation project group is divided into three teams consisting of two students each. Every team tackles a part of the SPPE design as described above. The teams are organised as follows:

Ultraviolet Germicidal Irradiation (UVGI)	R.P.M. Bakker
	M.J.H. Brouwers
Sensing and Control (SaC)	M. Goddijn
	J.J.M. Lut
On-Board Power Management (OBPM)	H.J. Donkers
	C. Zwart

Below follows a brief introduction on every subgroup.

Ultraviolet Germicidal Irradiation

The Ultraviolet Germicidal Irradiation group is responsible for providing the UV radiation in order to disinfect the filter material in the SPPE. This is achieved through the use of LEDs radiating in the so-called UVC spectrum (wavelengths between 200 and 280 nm). The UVGI group selects LEDs with the most effective wavelength and ensures the most optimal placement of the LEDs to achieve uniform irradiation, with accuracy and power consumption in mind. Additionally, the UVGI group will design the LED drivers. The LED drivers are controlled by signals provided by the SaC group.

Sensing and Control

The Sensing and Control submodule is the controlling body of the SPPE. The submodule measures the temperature, relative humidity, air pressure, and the pressure drop over the filter material. It generates the control signal for the UVGI submodule, thereby regulating the intensity of the UVC radiation. Using a control loop, the Sensing and Control submodule ensures a safe (always sufficient UV intensity) and power-efficient (not an unnecessarily high UV intensity) operation of the SPPE. It also monitors the quality of the filter material.

On-Board Power Management

The aim of the On-Board Power Management is to provide energy for the sensor circuitry and the LED modules. Since the SPPE is a wearable device, the system needs to have on-board power with an energy management system for low-power application and battery safety. This will be implemented by making use of a rechargeable battery and a battery management system that controls and protects the battery. The goal is to design a system that can supply the SPPE with required power in a safe and efficient manner.

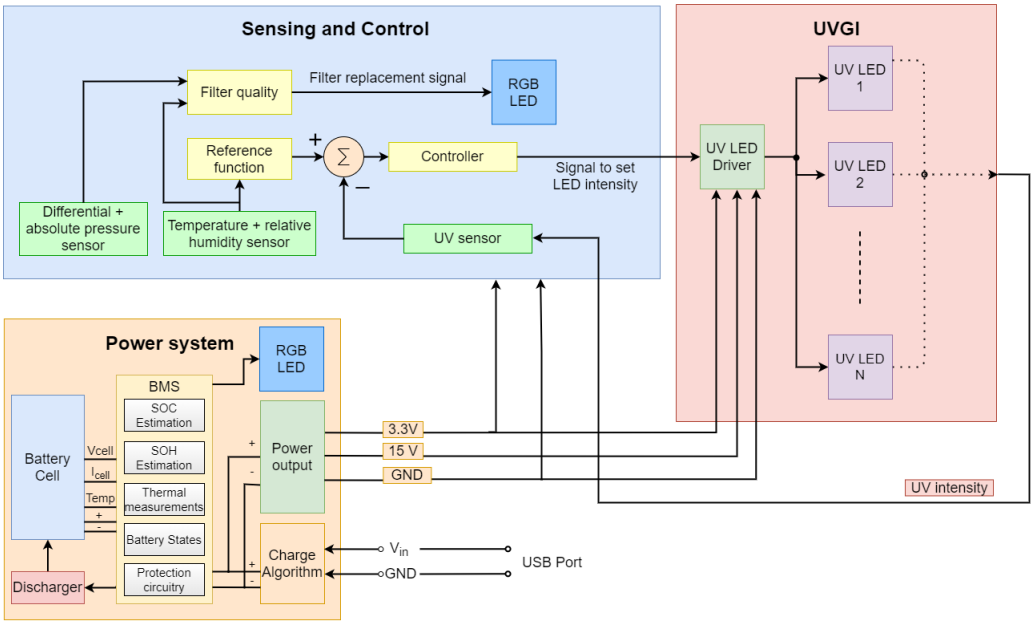


Figure 1.3: System overview divided in submodules.

1.1.3. Submodule interconnections

The submodules are interconnected in a physical way and by means of design constraints. Figure 1.3 shows the physical interconnections, which are the power outputs from the OBPM submodule to the other submodules, the control signal for the LEDs from the SaC module, and the feedback of UV intensity to the SaC module. The non-physical interconnections between the modules are the design constraints. The diameter of the filter material influences the number of LEDs needed to radiate the entire filter with an adequate dose. As one can imagine, this has an influence on how the control loop is calibrated. Using more LEDs also means that more power and higher peak currents are necessary, which influences the battery size. As the battery is the largest component, this has an influence on the filter diameter and the weight of the filter module itself. These interconnections illustrate the need for communication between the subgroups about these parameters throughout the design process. Therefore, communication is key to ensure an optimal SPPE system design.

1.2. State of the art Analysis

The use of rechargeable (e.g. lithium-ion) batteries has become popular for powering electric vehicles, portable devices, tools and utilities, due to their high energy density, lightweight and decreasing market price. Despite these features, today's lithium-ion batteries cannot meet all standards in today's market, such as those of electric vehicles. This is why there is a focus on new battery research which aims to find new lithium compositions that allow for even higher capacity and higher power delivery [4].

A new technology that is entering the battery market is flexible lithium-ion batteries. These type of batteries can be twisted, folded and crumpled with only marginal capacity degradation. They also prove to be very save, as trimming them with scissors or piercing with a nail does not impair the battery to function safely. Research now focuses on integrating these batteries in small and wearable electronic devices [5].

More advanced batteries also demand a battery management system to ensure reliable and safe operation. In order to create such a battery system, an accurate estimation of battery parameters is crucial for state estimation. One approach to this is using battery models, which make use of an equivalent circuit model, usually made up of a recursive least square identifier and a filter algorithm to estimate the state of the battery. More recently, however, some researchers have tried to estimate the battery state based on artificial neural networks and support vector machine [6].

1.3. Problem definition

The SPPE will be designed to integrate UVC LEDs and sensors in a filter package to ensure a safe and controlled protection from airborne pathogens. As the device is meant to be mobile, it cannot depend on a wired power connection. Hence an on-board power system needs to be designed.

The aim of the SPPE project is to create a smart filter module that can be operational for approximately one working day. This means that a power system needs to have enough power storage capacity to ensure this can be achieved. The device should also indicate information about the battery to the user, such as when to recharge the battery and if any errors have occurred in the system.

The UVGI and SaC submodules require a power input. The control systems will be powered by a 3.3V supply and the UVC LEDs will be powered by a 15V supply. These output voltages will be delivered by the power system.

The power system should, of course, ensure safe operations. Rechargeable batteries, especially Lithium cells, need to be accurately monitored and controlled. This means that the on-board power system needs to constraint all components within their electrical limits. The design of the power system should also aim to achieve a lifetime of at least two years for it to be allowed to be sold on the European market.

For this project, the following design goal for this thesis is defined as:
The SPPE needs a complete design of a functional battery management system, that supplies required power to the rest of the system, ensures safe battery operation and aims to maximize battery life.

1.4. Thesis synopsis

This thesis will describe the design process of the on-board power system of the SPPE project. In Chapter 2, the Program of Requirements will be elaborated on for both the complete project of the SPPE, as well as the power subsystem specifically. The system design will be addressed in Chapter 3, where background information about battery and battery management technology will be discussed and where it will be decided which design will be further explored. In Chapter 4 the system design will be modelled using Matlab and Simulink to verify the functionality of the design. Next, in Chapter 5, the model will be translated into real-life components to create an implementation using off-the-shelf components in a PCB design. In Chapter 6 the main differences between the model and the final design will be highlighted and the Program of Requirements will be assessed. Finally, a conclusion about the design process will be formed and recommendations about the further development of the power system will be given in Chapter 7.

2 Program of Requirements

As this thesis is part of a larger project, designing the SPPE, the main goal of the product is to protect human life. Therefore, safety has a high priority. The wearability is also of great importance, as doctors and nurses need to wear the mask for prolonged periods. To aid this, the battery system needs to be very compact.

2.1. SPPE requirements

The SPPE will be designed for use in a medical environment to protect against airborne pathogens. The finished product as a whole has the following general requirements:

1. The SPPE uses UVGI to sterilize the filter material.
2. The SPPE's mechanical filter is replaceable.
3. The SPPE's electronics are reusable after the mechanical filter is replaced.
4. The SPPE's battery operation time is at least eight hours.
5. The components' lifetime is at least two years.
6. The SPPE is designed preferably using off-the-shelf components unless it is necessary.
7. The SPPE is designed circularly: if one component breaks-down, either it could be replaced or if this is not possible, the other components can be salvaged.
8. The SPPE should communicate internal conditions to the user.
9. The SPPE filter module should be airtight, apart from the airflow channel.
10. The SPPE should be small enough to fit the side of a PPE.

2.2. System requirements

The On-Board Power Management system will be designed according to the requirements mentioned in Section 2.1. For the OBPM specifically, the following criteria are defined to support those requirements. These can be subdivided in Functional requirements, what the system must do, and Non-Functional requirements, what the system must have.

2.2.1. Key performance indicators

The performance of the on-board power system will be quantified by making use of the following key performance indicators:

- Lifetime
- Operating time
- State-Of-Charge deviation
- Power efficiency
- Cell voltage
- Cell charge current
- Cell discharge current
- Cell temperature
- Output voltage ripple

2.2.2. Functional requirements

1. The operating time of the power system is at least eight hours.
2. The power system must inform the user about the state of the battery.
3. The battery cell must operate within its specified cell voltage range.
4. The cell charge current cannot exceed the maximum rated charge current.
5. The cell discharge current cannot exceed the maximum rated discharge current.
6. The battery cell must operate within its specified temperature range.
7. The State-of-Charge deviation must not exceed 5%.
8. The power output efficiency must be at least 90 %.
9. The ripple voltage of the output terminals should preferably be lower than 0.1%.

2.2.3. Non-Functional requirements

1. The battery management system should have protection circuits that consist of:
    - Overcharge and overdischarge protection
    - Short circuit and quick discharge protection
    - Temperature protection
2. The battery system should adhere to medical safety standards.
3. The battery should be charged via a 5V micro USB connection.
4. The output of the battery system shall contain a 15 volt and 3.3 volt terminal.
5. The battery controller has a separate (micro)controller.
6. The PCB design should preferably make use of surface mounted devices.
7. The PCB design should not exceed 8 cm in diameter.

3
System Design

This chapter will focus on the design and theoretical background of the power system. Different types of battery technologies will be discussed in Section 3.1 and a battery chemistry will be chosen. Next, in Section 3.2 the methods for estimating the State of Charge (SOC) will be investigated and a conclusion will be drawn which estimation is the most suitable. Additionally, in Section 3.3 it will be mentioned what the influence of battery degradation is and how it is defined. Furthermore, in Section 3.4 methods for charging are listed as well as the best method for different charge speeds. In Section 3.5 different battery failure mechanisms are explained together with suggestions for protection circuits to prevent them. Subsequently, in Section 3.6 the output regulation will be discussed. Lastly, the medical safety standard that the system should adhere to will be explained in Section 3.7.

3.1. Battery Types

Batteries are electrochemical devices that allow for both storage of energy and a source of direct current (DC) electricity. All batteries consist of a positive electrode (cathode), negative electrode (anode), and an ionic conductor (electrolyte). All these components together form a battery cell. When multiple cells are stacked together, they form a battery. Cells can be classified into four groups [7]:

1. Primary Cells: Cells are discharged once and then discarded.
2. Secondary Cells: Can be easily recharged, mostly used in cell phones and tablet computers.
3. Reserve Batteries: Designed to be stored for long periods of time and to avoid chemical deterioration. Often used in weapon systems.
4. Fuel Cells: Converts chemical energy to electrical energy as long as fuel is supplied into the reaction chamber.

As stated in the system requirements, the battery has to be rechargeable, hence a secondary type cell is needed. The following types of secondary battery technologies are most commonly used or show the most promise in future usage [7].

Lead-Acid Batteries

A lead-acid battery is made of a positive PbO_2 and negative Pb electrode in a diluted H_2SO_4 electrolyte [8]. Power can be delivered from these batteries when suddenly needed but should be charged slowly. Lead-Acid batteries are typically low cost but have a low energy density when compared to other rechargeable batteries. Also, these batteries can be very heavy due to lead being the main component [7].

Alkaline Batteries

The two mainly used alkaline batteries are the nickel-cadmium (NiCd) and nickel-metalhybrid (NiMH) batteries. They both have a positive NiOOH electrode but differ in the negative electrode, which are Cd and H_2 respectively. The electrolyte for both is potassium hydroxide (KOH) [8]. NiMH batteries were designed to replace NiCd batteries. They have advantages over NiCd such as more specific power [W/kg] and longer battery life. All alkaline-based batteries have a more stable voltage over their discharge range than lead-acid batteries but are more sensitive to temperature (operating range of -20 to 40 °C) [7].

Lithium-ion Batteries

Lithium is the lightest known metal and can thus be used to create very light batteries, lithium-ion batteries (LIBs). However, these kinds of batteries is characterised by high specific capacity and high specific energy [Wh/kg]. They have to be treated carefully since they are susceptible to overcharging and heat [7]. The positive electrode consists of lithium-metal-oxide and the negative electrode of lithium-graphite. The electrolyte is a salt solution in an organic compound [8]. A subset of lithium-ion batteries is the lithium polymer batteries. These have a solid electrolyte made of a polymer instead of a liquid compound [7].

Sodium Batteries

Sodium batteries are low mass and low cost due to the abundance of sodium. These batteries are proposed for grid storage. They are, however, very explosive, so they need to be handled with care. In order for these types of batteries to operate, a high temperature needs to be reached (270 °C for a NaS battery) [7].

For the SPPE, a wearable design needs to be realised. Hence, the application of lead-acid or sodium batteries will no be practical, due to their larger size (both), weight (lead-acid) and operating temperature (sodium) compared to lithium and nickel batteries. Because of this, the two types of batteries will not be further considered. In Table 3.1, commonly used characteristics of NiCd, NiMH and lithium-ion batteries are displayed.

Table 3.1: Average ratings of commercial batteries. Speciality batteries with above average ratings are excluded [9].

Specification	NiCd	NiMH	Lithium-ion		
			Cobalt	Managnese	Phosphate
Specific Energy (Wh/kg)	45-80	60-120	150-250	100-150	90-120
Internal resistance	Very Low	Low	Moderate	Low	Very Low
Cycle life (80% DoD)	1000	300-500	500-1000	500-1000	1000-2000
Charge Time (hours)	1-2	2-4	2-4	1-2	1-2
Overcharge tolerance	Moderate	Low	Low. No trickle charge		
Self-discharge per month	20%	30%	<5%		
Cell Voltage (Volts)	1.2	1.2	3.6	3.7	3.2-3.3
Peak load current	20C	5C	2C	>30C	>30C
Charge temperature (°C)	0 to 45		0 to 45		
Discharge temperature	-20 to 65		-20 to 60		
Maintenance requirement	Full discharge every 90 days		Maintenance free		
Safety requirements	Thermally stable, fuse protection		Protection circuit mandatory		
Coulombic efficiency	~70% fast charge ~90% slow charge		99%		
Cost	Moderate		High		

3.1.1. Battery selection

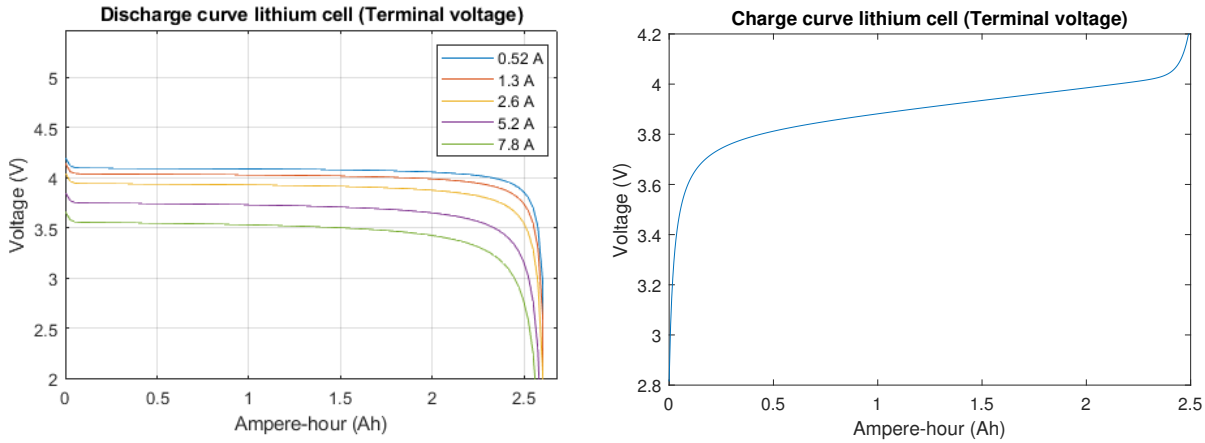
Based on the previous subsections, a lithium-ion cell is chosen to be implemented in the OBPM design. The LIB provides a higher energy density over the NiCd and NiMH batteries, allowing for a smaller design. The lifetime of lithium-ion is also longer, providing a product that can be operable for at least two years. A drawback of the lithium-ion battery is that it requires a strict protection circuit, as LIBs can potentially be hazardous due to their flammable electrolyte with a powerful oxidizer. Overcharge, external heating, short-circuits and crushing of the battery are the main reason battery failure leads to explosions or rupture of the battery. This requires LIBs to have safer design during charging and operation [10].

3.1.2. Lithium-Ion cell behaviour

A lithium battery cell consists of graphite or graphene anode, an electrolyte with lithium salts in solvents to create lithium ions and a cathode that can be made from different types of metals, dependent on the desired battery response. Both electrodes are made of materials that can intercalate or absorb lithium ions. For this battery type, lithium ions are bound to electrodes in the anode. When the battery discharges, the intercalated lithium ions are released from the anode. They then travel through the electrolyte solution to be absorbed (intercalated) in the cathode [11].

The lithium cell has a different discharge voltage dependant on the output current of the cell. Figure 3.1a shows this as the voltage curves differ for different discharge currents. It can be seen that when the output current increases, the terminal voltage drops. This is due to the internal resistance of the battery, which causes a voltage drop according to Ohm's Law, $V = IR$. Hence, when the current increases, the voltage drop across the internal resistance increases as well.

When charging the battery, a constant current source is used. This will be explained in more detail in Section 3.4. For charging the battery, usually a fixed charge rate (C) is used. This is expressed in C, where 1 C expresses the current with which the battery would charge its total capacity in one hour. Typically, the charge rate is either 0.2 C or 1 C. From Figure 3.1 it can be seen that in the beginning and end of the charge or discharge cycle the curve is highly non-linear. The reason for this is that the curve slopes at the beginning of the charge curve because in the first 10% of the SOC, the internal resistance is quite high. Applying Ohms law means that therefore the voltage should also be low with a constant discharge current. However, after the first 10% the internal resistance decreases enormously and remains stable, therefore the voltage can increase near-linear until the end 10% of the charge is reached. Here the constant current stage ends as the battery is nearly saturated. Now a constant voltage source needs to take over to slowly trickle-charge the battery to full (Section 3.4). Here a constant current stage is not advised for risk of the potential of the graphite decreasing to 0 V versus the Li^+/Li electrolyte, which leads to metallic plating of the graphite, reducing the capacity of the cell [12].



(a) Discharge characteristics for different discharge rates.

(b) Charge characteristics with a 0.3 C charge rate.

Figure 3.1: Charge and discharge curves for a battery with a nominal 3.7 V and maximal 4.2 V rating.

3.2. State of Charge

The State of Charge (SOC) relays the amount of energy that is left in the battery. The State of Charge is necessary to know for the control logic, as to protect the battery and elongate the lifespan, but more importantly, to relay to the user when it is time to recharge. Estimating the State of Charge is an ongoing research topic, which has not yet yielded 'the definitive way'. Therefore, different methods can be used, depending on the application of the SOC. The State of Charge can be estimated in an offline or online mode. Offline meaning that the battery needs to be taken out of service and charged and discharged via a specific protocol. Since the application of this project requires the SOC to be known in real-time, this is not a viable option. The online mode estimates the SOC, while the battery is in use. In the coming subsections, three different methods for estimating the SOC online are discussed.

3.2.1. Coulomb Counting

Coulomb counting uses a current sensor to measure current entering and leaving the battery pack. This is then integrated and counted, by using the formulas of Equation 3.1.

$$SOC = SOC_0 \pm \frac{1}{C_n} * \int_{t_0}^{t} |I| dt \tag{3.1}$$

Here, C_n is the nominal capacity, t_0 is the start time of measurement, I is the battery current and SOC_0 the initial SOC. However, to use this method an accurate current measurement needs to be done and a historic knowledge of the initial SOC_0 is required [13]. This could be solved by calibrating to the fully charged point, to create a correct reference point SOC_0. The downside of using this method is that when the SOC_0 is not known, for instance after a cold start, which is when the system has been off, the SOC is not trustworthy. Furthermore, C_n changes over time as the State of Health (SOH) degrades, making the SOC inaccurate.

3.2.2. Open Circuit Voltage

Another computational low option to estimate the SOC is by measuring the output voltage of the cell and referencing that to a 3D look-up table of the SOC, voltage and current. However, the voltage measurement suffers from signal noise and disturbances. If the battery is in charging mode, the SOC will always be estimated higher and vice versa for discharging mode.

The SOC will be relatively close to the real SOC while charging, as the current is stable in the largest portion of the charge, which is the constant current stage (explained in Section 3.4). But while it is discharging it will not be accurate as the algorithm must interpolate between different current and voltage vs SOC curves, making it inaccurate. From Figure 3.1a it can be seen that in the largest portion of the discharge, the voltage is relatively flat. Therefore, if noise is introduced, the error can be significant as misreading 4 V as 3.9 V already means that the SOC could be off by more than 20 %. Also, battery charge and discharge curves differ for different temperatures, introducing yet another variable. Since the application of this system while discharging is close to a persons face in a changing environment, this cannot be called a stable temperature environment. However, when charging this would be a stable environment and this technique would yield adequate results. The only other downside is that this algorithm needs enough memory available to store the look-up table.

3.2.3. Kalman Filter

A Kalman Filter provides a theoretical method to filter measurements of a system's input and output. It aims to produce an intelligent estimation of a dynamic system's state on the basis of a least mean squared error. The main assumption in Kalman filtering is that the measuring noise and process noise are Gaussian, independent of each other, and have a mean of zero. However, Kalman filtering assumes the system is linear, which the power system is not. Therefore, a linearization process should be employed to estimate the non-linear system with a linear time-varying system. This variation would be an Extended Kalman filter (EKF), although in a highly non-linear system the answer does not converge. Therefore, an Unscented transformation could be used, resulting in an Unscented Kalman Filter (UKF). The basis for a Kalman filter can be found in Equation 3.2 [13].

$$\begin{aligned} \boldsymbol{x}_k &= A\boldsymbol{x}_{k-1} + B\boldsymbol{u}_{k-1} + \boldsymbol{a}_{k-1} \quad \text{process} \\ \boldsymbol{z}_k &= C\boldsymbol{x}_k + \boldsymbol{b}_k \quad \text{measurement} \end{aligned} \tag{3.2}$$

Here **x** is the system's state variable, for which the SOC can be used and **z** is a vector used for comparison, for which the voltage is used [13].
Kalman filters are able to run in real-time. They also do not require an initial SOC and they already incorporate noise compensation. However, they are computational very heavy and require a complex algorithm to be implemented.

3.2.4. SOC estimation method selection

In the modelling chapter (Chapter 4), only the Coulomb Counting and Open Circuit Voltage method will be further explored as they are a simple yet accurate approximation at a low computational cost. Therefore, the Kalman Filter approach will not be used as the expectation is that there is not enough computing power to run this filter in the desired small form package of the SPPE.

3.3. State of Health

The maximum capacity of a battery decreases with the number of charge cycles and with changing temperature. The SOH is defined as the actual capacity compared to the initial capacity. This can be seen in Equation 3.3, where Q is the nominal capacity of the battery [mAh] at a given life cycle and k is the number of life cycles [14]. Here a life cycle is defined as a full charge and discharge.

$$SOH(k) = \frac{Q_{nom}(k)}{Q_{nom}(0)} \tag{3.3}$$

Q_{nom} can be seen as the amount of cycles k times the degradation per cycles. In order to keep estimating the SOC accurately, it is important that the SOH is accurately estimated.

3.4. Charge algorithm

Charging a battery can be done in different ways. This depends on the chemistry of the battery. For lithium-ion batteries there are several ways to charge it correctly such that the SOH (State of Health) is minimally affected, therefore increasing the battery life. In charging a lithium-ion battery, special notice should be given to the beginning and end of the charging cycle. As explained in Subsection 3.1.2: when the SOC is below 10%, the internal resistance of the cell is very high and a very low charging current should be used. When the SOC is higher than 90%, again a low current should be used, because otherwise there is metallic lithium plating in the graphite, which reduces the capacity of the battery [12].

3.4.1. Charge profiles

There are several charge profiles that can be followed, all having their benefits and drawbacks. The two profiles that were considered for the system where:

- A Constant Power-Constant Voltage (CPCV) profile. This profile starts with a high current value (as the voltage is still low), which gradually decreases with the charging time following a correlation of constant power until the voltage of the cell reaches to 4.2 V.
- A Constant Current-Constant Voltage (CCCV), which is the standard as of yet. It starts with a constant current until a predefined voltage level is reached and then charges using a constant voltage.

The choice of charge profile depends on the speed of charging. For small charge rates (0.5 C) the best option is CCCV as this leads to the smallest capacity fading and the current in the beginning stages is small enough that it does not lead to metallic plating. For larger charges rates (1C) it is advised to use the CPCV profile as it benefits the beginning phase the most and in the final stage, it limits the current, such that it does not lead to metallic plating in the graphite [12].

3.5. Battery protection

A vital part of the whole on-board power management system is the protection equipment. This makes sure that the battery remains in the specified operation region, preventing capacity degradation and short circuits leading to breakage of the battery and other circuitry.

3.5.1. Short Circuit and Quick Discharge protection

There are multiple mechanisms that, when allowed to occur, draw a current which is too large for the battery, leading to thermal runaway. These mechanisms will be explained below.

External Short Circuit

Short circuit protection is necessary since when the output of the battery is shorted for some reason, the battery will also partly discharge via its internal resistance. The relation for this can be seen in Equation 3.4 [15].

$$I = \frac{E}{(R_i + R_s)} \tag{3.4}$$

Where R_i is the internal resistance and R_s is the external short circuit resistance. The circuit can be seen in Figure 3.2. If R_s is zero, so a perfect external short circuit, the short circuit current is at its maximum. The consequence of this is heat generation. This happens via the relation of Equation 3.5 [15].

$$W_i = I^2 R_i$$
$$W_i = \frac{E^2 R_i}{(R_i + R_s)^2} \tag{3.5}$$

Here W_i is the energy dissipation in Joule in the form of heat. This form of short circuit can be prevented by using an external protection circuit. An example of this would be a fuse.

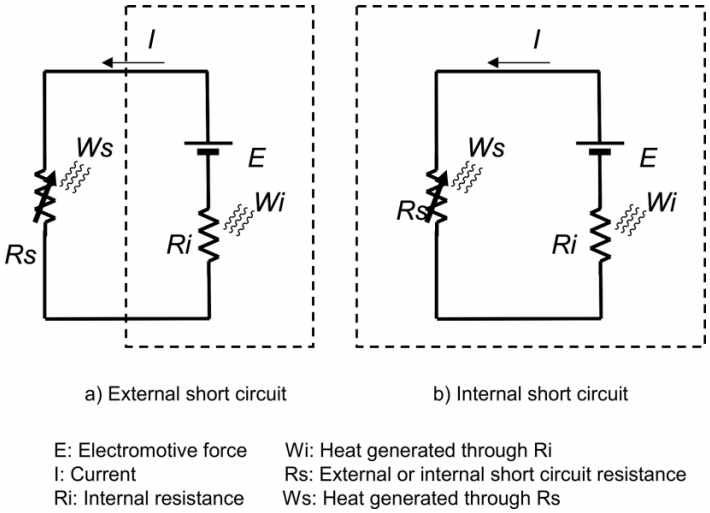


Figure 3.2: Simple equivalent circuit for shorted battery [15]

Internal Short Circuit

When an internal short circuit occurs, the relation for the current remains equal to Equation 3.4. However, for the energy, now the following Equation 3.6 holds [15].

$$W_s = I^2 R_s$$

$$W_s = \frac{E^2 R_s}{(R_i + R_s)^2} \tag{3.6}$$

Where W_s is the internal short circuit heat generation in Joules. Here W_s has the highest value of $\frac{E^2}{4R_s}$ when the internal resistance equals the short circuit resistance. Both the internal and external short circuit heat generation versus the short circuit resistance R_s can be seen in Figure 3.3a. Reasons for internal short circuits can include [15]:

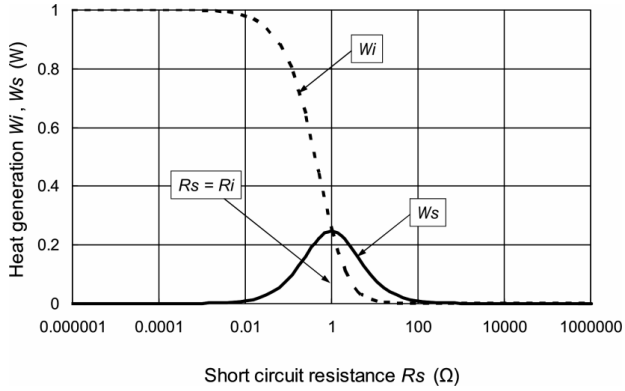
- Alien electroconductive substances that invade the battery during manufacturing.
- Changes in the battery caused by dropping or crushing it.
- Dendrite shorts caused by the extraction of lithium metal due to the reduction of lithium storage capability in the negative electrode.

Although internal short circuits are rare, the probability ranging from 1 in 1 million to 1 in 5 million, protection for them is necessary, as the result of a battery overheating and venting or exploding are enormous [16]. However, internal short circuits cannot be detected by outside circuitry. The only way to detect this short circuit is by implementing a temperature sensor close to the battery.

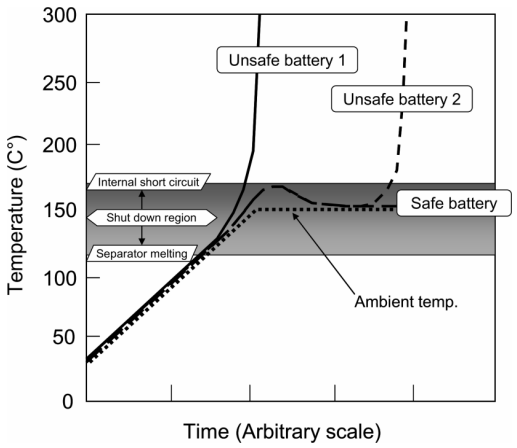
A standard battery test as Ichimura [15] mentions, shows that a safe battery should not exceed 150°C. However, in normal ambient temperatures a battery will already not exceed 100°C. This can also be seen from Figure 3.3b. Therefore, introducing a temperature measurement in this area with a way to decouple the battery from the rest of the circuit, will be the best practise.

Quick Discharge protection

Quick discharge should be prevented as dendrites may form inside the battery, which in turn could lead to internal short-circuiting [17]. Discharging should, in general, not happen with more than 1C. By quick discharging the total capacity of the battery is degraded [18]. Quick discharge can be prevented by limiting the amount of current that can exit the battery, for example by implementing a fuse, although that would also prevent the battery from being used again before the fuse is replaced.



(a) Joule heat W_i and W_s as a function of short circuit resistance R_s [15].



(b) Behaviour of Lithium-Ion batteries in overheating test [15].

3.5.2. Overcharge and overdischarge Protection

Lithium-ion batteries should not be overcharged with a voltage that is higher than 0.1 V above the maximum rated voltage. For applications that invite a serious situation due to overcharging, the overcharge measures should be doubled. When a lithium-ion battery keeps being overcharged, structural destruction of positive active material or reduction decomposition of electrolyte may occur and lithium metal may be deposited on the negative electrode surface. Finally, thermal runaway would ensue, inducing a vent with smoke, fire, or an explosion [15].

To protect against this means of failure, a measure should be built in to prevent the cell from discharging when it reaches the cut-off voltage. Since capacity degradation already starts a bit earlier than at the cut-off voltage, the safety-measure could also kick in sooner, to spare the battery further. This is often done in mobile applications.

3.6. Output regulation

The battery cell will output a variable voltage between 4.2 and 3.0 V. This output voltage needs to be converted to a stable 3.3 V and 15 V output. This will be done by using a step-up and a step-down DC-DC converter.

3.6.1. Step-up converter

To boost the DC voltage, a buck converter is used, of which a typical configuration can be seen in Figure 3.3a. During normal operation of the boost power stage, Q1 is repeatedly switched on and off by a drive circuit. This switching action creates a train of pulses at the junction of Q1, CR1, and L. Inductor L and output capacitor C form an effective LC output filter. This filters the train of pulses to produce a dc output voltage, V_O. The voltage can then be increased by adjusting the duty cycle D of the pulse train, according to Equation 3.7 [19].

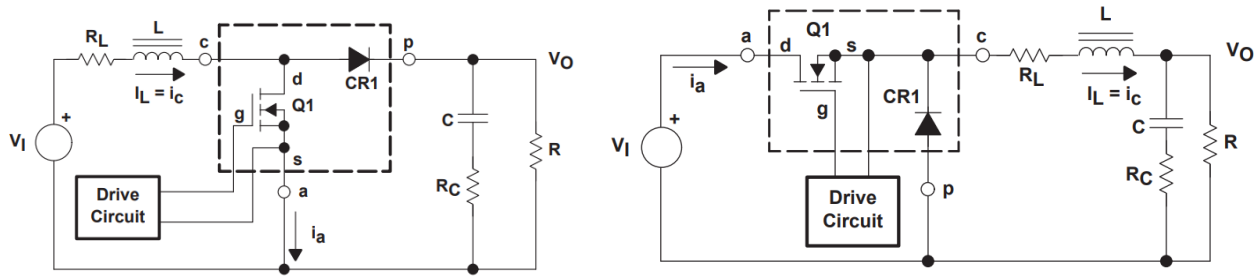
$$V_O = \frac{V_I}{1 - D} \tag{3.7}$$

3.6.2. Step-down converter

To maintain a constant voltage at the output of the power system, a Low-Dropout Regulator (LDO) or fast switching regulator can be used. LDOs are more efficient than common linear regulators as they have a very low voltage dropout. This voltage dropout results in a waste of power, according to Equation 3.8.

$$W_{loss} = (V_{in} - V_{out}) \cdot I_{Load} \tag{3.8}$$

Normal linear regulators also require a higher input than output voltage to function. A switch-mode regulator is not bound by this requirement, which means that the output voltage can exceed the input voltage whilst also achieving a higher conversion efficiency (>90%) [21]. Hence, a buck converter is used, of which a typical



(a) Boost converter [19].

(b) Buck converter [20].

Figure 3.3: Typical configurations of a Boost and Buck converter.

configuration can be seen in Figure 3.3b. During normal operation of the buck power stage, Q1 is repeatedly switched on and off. This switching action causes a train of pulses at the junction of Q1, CR1, and L which is filtered by the LC output filter to produce a dc output voltage, V_O. The output voltage can then be set according to Equation 3.9 [20].

$$V_O = V_I D \tag{3.9}$$

3.7. Safety standards

For (battery-powered) medical devices there are multiple international standards and amongst these, the most notable is the IEC 60601-1 [22] (the full relevant list can be found in Annex A of [23]). This standard provides basic safety en performance requirements. However, there is a volume of standards and these are very long, detailed and with a lot of legal terms. Therefore, for both the CE marking as well as the IEC 60601-1 standard, it is considered that this falls beyond the scope of this thesis. A document by AdvaMed focused more on battery safety and is more relevant to be consulted [23]. This document has the disclaimer that it does not provide the specifications for a Standard or give legal advice, however it aims to give a best practice advice. Although, this mostly also falls beyond the scope of this thesis as the aim of this thesis is not to manufacture batteries. In almost all cases, the manufacturer of the battery has provided information and data on safety tests.

3.7.1. CE marking

If the SPPE product is to be sold in the European Union, it needs to pass certain safety regulations to be eligible for a CE marking. To obtain this certification, the product must comply with Directive 2007/47/EC [24] of the European Union on the safety and performance of medical devices. This Directive aims to protect human health and safety and ensures that medical devices can be freely and fairly traded throughout Europe. To obtain a CE marking, one should:

1. Determine the classification of the medical devices. In the case of the SPPE probably IIa.
2. Implement a quality management system, such as ISO 13485.
3. Draw up a Technical File for Class I/II devices, with detailed information on their design, function, composition, use, claims and clinical evaluation.
4. Appoint an Authorised Representative in the European Union.
5. Apply for an audit by a Notified Body to receive a CE Marking Certificate.
6. Draw up a Declaration of Conformity stating that the device complies with the relevant Directive.
7. Affix the CE marking to your medical device.

Also, it is required to use the correct labelling [25]. Notable labels include the address of the manufacturer, identification number and batch code.

4

System Modelling and Verification

This chapter will focus on the theoretical implementation of the on-board power management system. This has been done by modelling the complete system and verifying its functionality through simulation. The modelling software used is Simulink and the Simscape electrical library [26]. This library offers a large set of electrical components and can be used to simulate the discharge rates, temperature effects and ageing of a battery. The simulation results using Simulink give a clear indication of the functionality of the power system and its subsystems.

The chapter starts with a general overview of the design in Section 4.1. Next, the Simscape battery model and the Battery Management System (BMS) are explained in Sections 4.2 and 4.3. In the BMS of Section 4.3, the method for estimating the SOC and SOH is also explained. Subsequently, in Section 3.4 the charge model is introduced and the results are shown. After this, in Section 4.5 it is explained how the protective measures have been implemented. Then, the output conversion is explained in Section 4.6, this includes the simulation output and the implemented model output. The last implementation that is discussed is the human interaction in Section 4.7. Lastly, the simulation is run for all possible states and the results are discussed in Section 4.8.

4.1. Overview

The power system layout is based around the typical design of a Battery Management System (BMS) [14], as seen in Figure 4.1. The system has two input terminals (V_{in} and ground) and three outputs (15V, 3.3V and ground) which supplies the rest of the SPPE system with the required power. The system can be divided into four subsystems:

1. Battery cell and sensors: Contains the battery cell and sensors that measure the terminal voltage, output current and temperature of the cell.
2. Battery Management System: The logic core of the system. Estimates the SOC and SOH, controls the state of the battery and protects against out-of-scale operating conditions.
3. Charge algorithm: Uses the CCCV method to charge the battery.
4. Power output: Converts the variable input voltage of the battery to the desired constant voltages at the output of the power system.

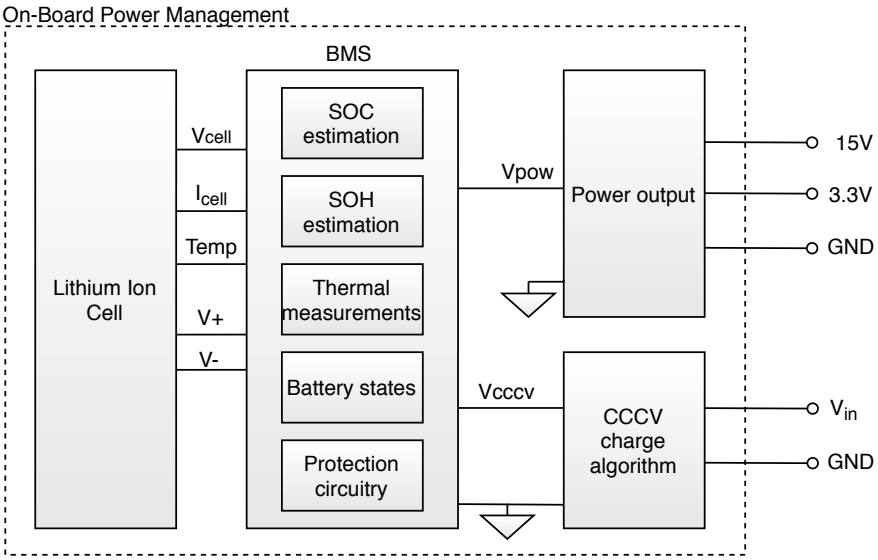


Figure 4.1: Overview of the power system

4.2. Battery Model

Simscape offers a model of a battery which can be easily set up by inputting parameters from a datasheet [27]. The equivalent circuit of this model is a controlled voltage source in series with a resistor. The controlled voltage source models charge and discharge dynamics, thermal effects and capacity deterioration due to ageing. Figure 4.2 shows the implementation om the battery model and sensors in Simulink.

The Simulink model of the subsystem has two electrical connections for the positive and negative terminal of the cell. The ambient temperature of the cell can be set, so the behaviour of the cell can be modelled for different temperature environments. The model outputs three measured data connections which will be fed to the BMS. The model also outputs a battery info vector that contains seven data sets, namely the life cycles, capacity, ambient temperature, internal temperature, SOC, terminal voltage and current of the cell.

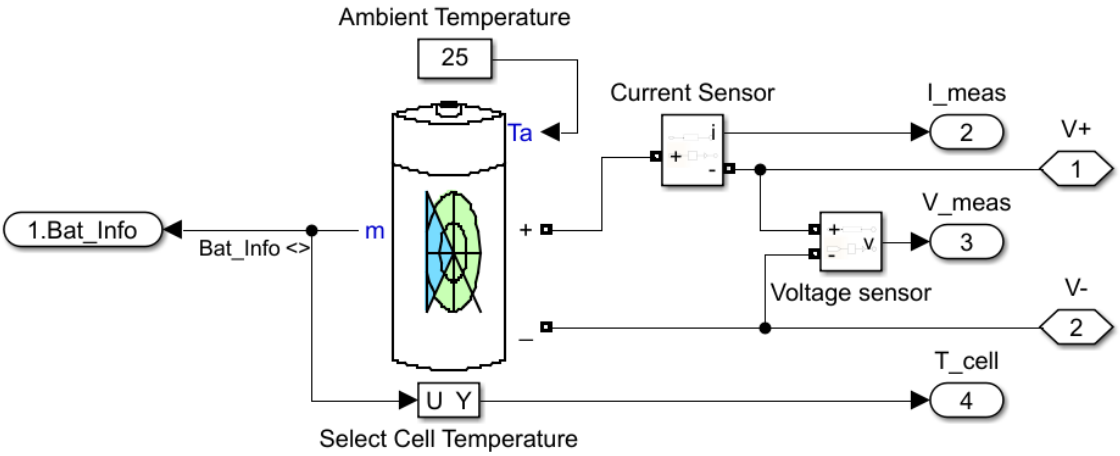


Figure 4.2: Simulink model of the lithium cell and sensors.

4.3. Battery Management System

The BMS makes all logic decisions of the power system. It controls the state of the battery, decides when to charge and when to enable the output model to supply power to the load. The BMS has five data inputs, two data outputs and five connectors for electrical flow. These are listed in Table 4.1 and are illustrated in Figure F.4 for the interested reader.

Table 4.1: Signals of the BMS

Signal	Direction	Property
Charger_detect	Input	Signifies whether the charger is plugged in
System_enable	Input	Signifies whether OBPM system is enabled
I_cell	Input	Measured output current from battery sensor
V_cell	Input	Measured terminal voltage from battery sensor
T_cell	Input	Measured cell temperature from battery sensor
CCCV_Enable	Output	Enables or disables the charging module
Power_Enable	Output	Enables or disables the power module
Vbat+	Bidirectional	Connects to the positive terminal of the battery cell
Vbat-	Bidirectional	Connects to the negative terminal of the battery cell
Vpow	Bidirectional	Connects to positive terminal of output module
Vcccv	Bidirectional	Connects to positive terminal of charging module
GND	Bidirectional	Connects to ground

4.3.1. Battery Finite State Machine

The BMS is implemented as a Finite State Machine (FSM) (Figure 4.3) which has three states:

1. Idle: The battery does nothing, it is disconnected from the load and there is no charging.
2. Charge: The charger is connected and the FSM enables the charging module. The battery is disconnected from the load.
3. Discharge: The battery is connected to the load and discharges.

The FMS starts in the idle state. Here it checks the SOC and whether the charger is plugged in. If the charger is plugged in and the SOC is below 95%, the BMS will move to the charge state. If the charger is not plugged in and the SOC is above 10%, the BMS will move to the discharge state. The state change is dependant on the SOC, such that the battery will have a maximum DOD of 90%, which will increase its lifetime.

In the charge state, the BMS is charged by the charging module. If the charger is unplugged, the BMS will move to the discharge state. If the charger is still plugged in, but the battery has been fully charged, the BMS will remain in the charge state as the charging module will stop charging the battery automatically.

From the discharge state, the BMS will move to the idle state if the SOC drops below 10%. If the charger were to be plugged back in while discharging, the BMS will move back to the charging state again.

4.3.2. Noise robustness

When doing measurements for the BMS, the sensors can suffer from noisy measurements. The way this was implemented in Simulink was by adding a random normally Gaussian distributed signal with a 0 mean. The variance differs for the voltage and the current. The reason for this, is that when selecting the components for the final product, the noise of these components was fed back in Simulink, to simulate the effect of this on the system. For the voltage, this meant a variance of 0.007^2 and for the current a variance of 0.002^2. The implementation can be found in Figure F.5 in the Appendix. After this implementation, the error was at most 3 mV. This doesn't seem much, however from the SOC in Figure 4.22 it can be seen that in charging the battery, when the OCV method is used, there is an error due to the noisy voltage measurement. Although, this stays within 3.5 % of the actual SOC

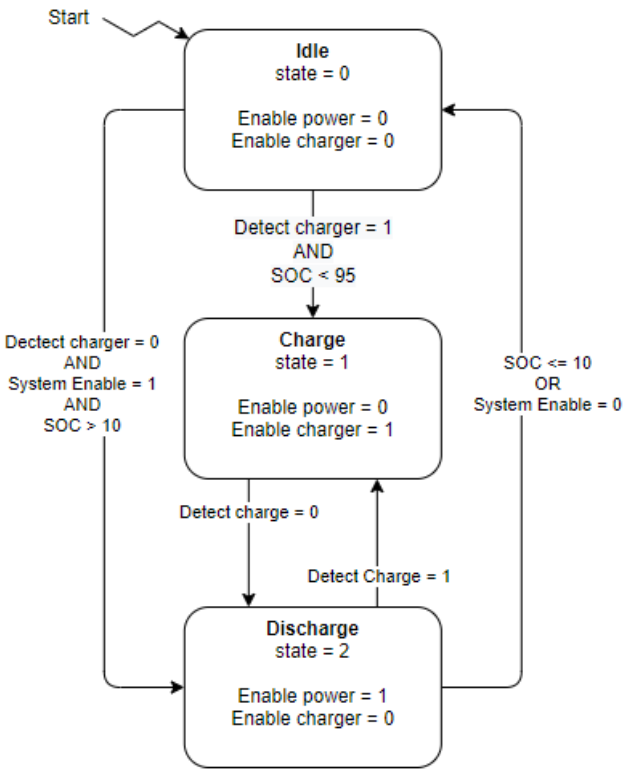


Figure 4.3: Finite State Machine of the BMS

For the temperature, it was decided not to add a noise filter to the system. This is because the temperature readings may be inaccurate up to 5 °C and the signal would need to be extremely noisy to reach this. The temperature sensor selected for the final product was accurate enough as it fell far below this 5 °C threshold, so no need was found to implement a filter in the Simulink model.

4.3.3. SOC Estimation algorithm

The State of Charge can be estimated from the voltage and current measurements of the battery cell. The first method implemented is Coulomb counting, based on Equation 3.1, and can be seen in Figure 4.4. This method takes the measured output current of the cell and uses an integrator to count the charge.

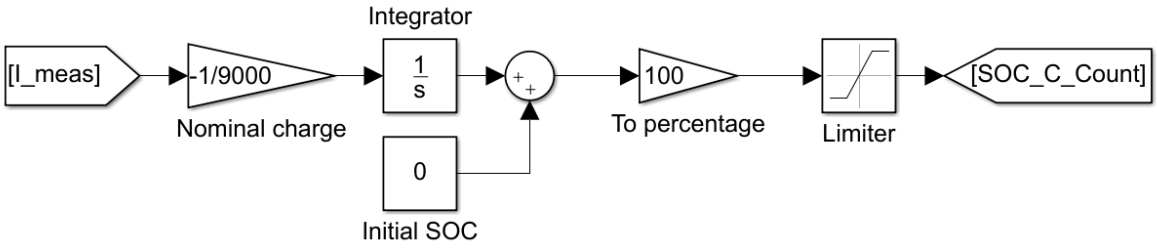


Figure 4.4: Implementation of coulomb counting in Simulink

The second method is the Open Circuit Voltage (OCV) estimation, which is modelled in Figure 4.5. The state switch selects which curve to read by looking at the polarity of the current, which indicates charging or discharging. This method relies on accurate data from the battery cell and therefore needs to be tailored to a single cell. Since the power module will only be using one cell, charge and discharge measurements have been executed to obtain these characteristic curves.

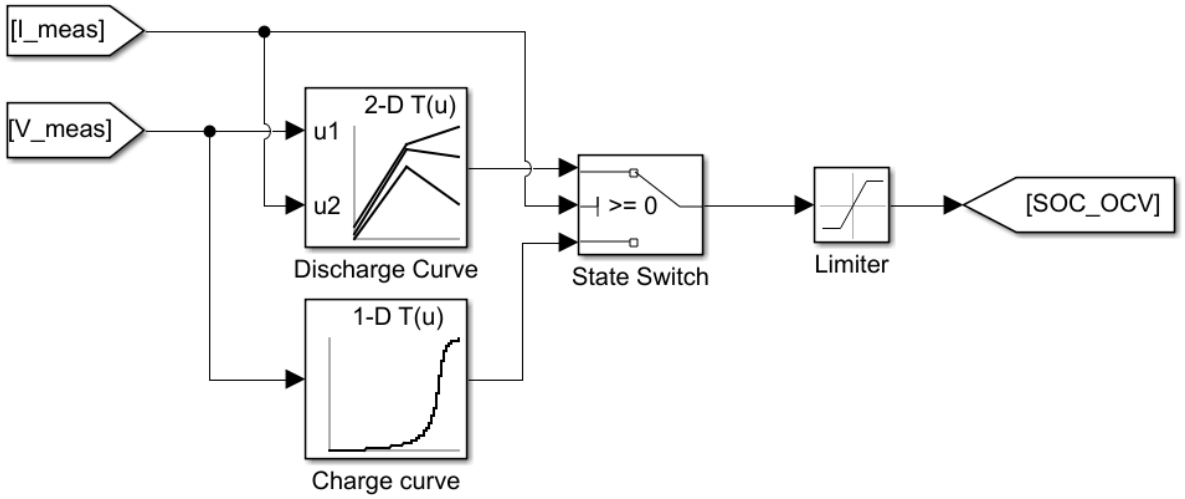


Figure 4.5: Implementation of OCV estimation in Simulink

Lastly, a combination of both SOC estimation methods has been implemented in Figure 4.6. This estimates the SOC in the charge state based on the charge curve and is stored in a memory unit. The estimate in the memory unit is used to initialize the coulomb counting method in the discharge state. This negates the problem of the unknown initial condition in the coulomb counting method, as long as the battery is charged when the power system is first started. It also has no interpolation errors because the charge curve is identical for every charge cycle. When the battery is charged again, the integrator is reset and the memory unit is updated with the new SOC estimate from the charge curve.

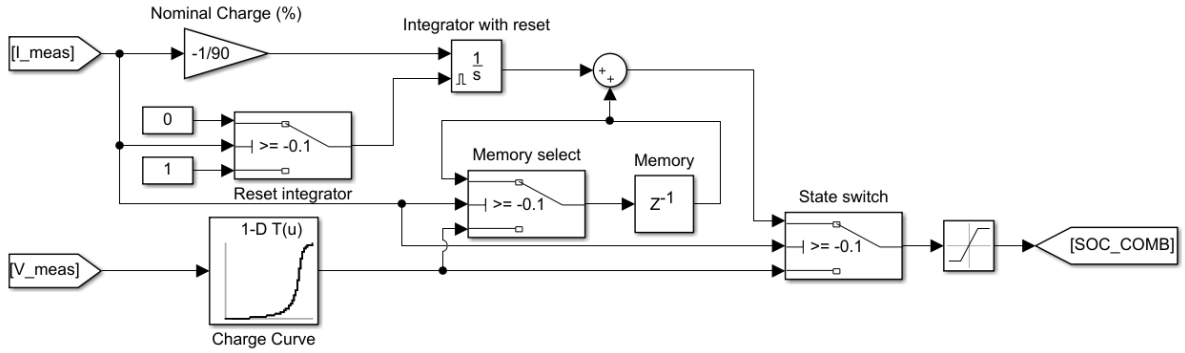


Figure 4.6: Implementation of combination estimation in Simulink

In Figure 4.7 the SOC estimations can be seen compared to the real SOC. It can be seen that the coulomb counting method represents the SOC very closely when charging and discharging, however, it has an offset, because it does not know the initial condition. The OCV method works well when charging, but drifts when discharging. This can be explained by the fact that the battery output voltage doesn't behave linearly with the output current and can therefore not be accurately interpolated from the discharge curve. The combined method has comparable results with the coulomb counting method, but did not require a known initial condition and also has the advantage that it can correct itself if an error would ever occur within the estimation algorithm. This is not the case with normal coulomb counting. For these reasons, it is recommended to combine both the SOC estimation methods for a more accurate result.

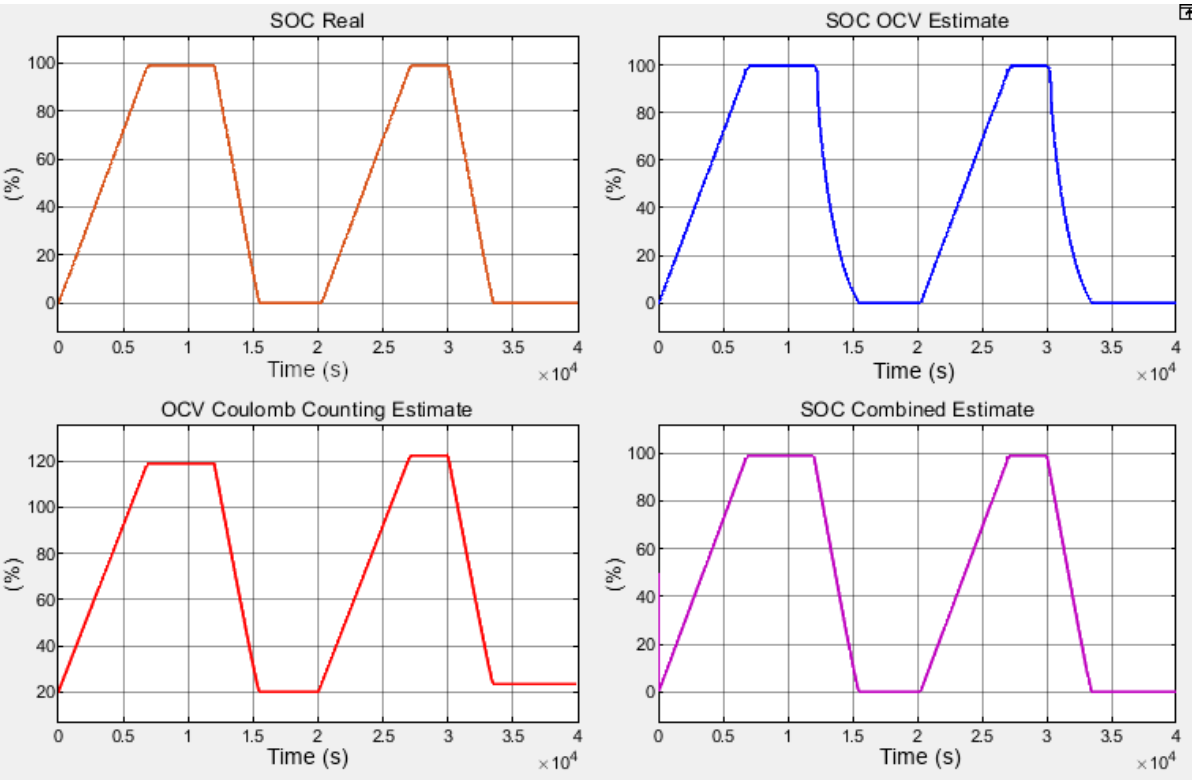


Figure 4.7: Comparison of SOC estimation methods.

4.3.4. SOH Estimation algorithm

The SOH algorithm, which can be seen in Figure F.7 in the appendix, functions similarly as the SOC Coulomb counting method. It integrates the absolute value of the measured current to estimate the current cycle of the battery. As a cycle is defined as one full charge and discharge, this means that one cycle is equal to twice the capacity of the battery. By dividing the measured current by this capacity, the cycle can be estimated.

According to the datasheet of the battery [28], the initial capacity is 1.05 Ah and has a lifetime of 500 cycles. The lifetime of a battery is defined as the number of complete charge/discharge cycles a battery can perform before its nominal capacity falls below 80% of its initial rated capacity [29]. The capacity loss per cycle then becomes:

$$Q_{loss} = \frac{1.05 \cdot 0.2}{500} = 0.00042 \qquad [Ah/cycle] \tag{4.1}$$

This capacity loss per cycle can then be used to estimate the current capacity of the battery. The complete formula for the capacity estimation can be seen in Equation 4.2. Here the cycles are estimated by integrating the absolute value of the current and dividing it by the previously estimated capacity. The estimated cycles are then multiplied with the loss per cycle and subtracted from the initial capacity.

$$Q[n] = Q[0] - \frac{Q_{loss}}{7200} \int_0^t |I| dt \tag{4.2}$$

With the estimation of the capacity when the battery has aged, the SOC algorithm can be improved by feeding the capacity estimate into the SOC estimation algorithm. This is can be seen in Figure 4.8.

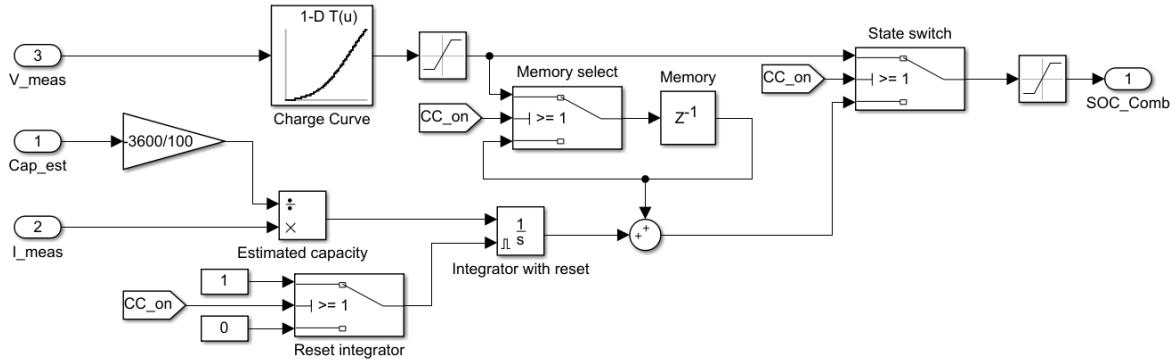


Figure 4.8: Improved model of the SOC estimation algorithm

The results of the SOH algorithm can be seen in Figure 4.9. When compared to the data from the battery model, it can be observed that the estimate of the cycle deviates very little from the actual life cycle, as indicated by the red line in the error graph of Figure 4.9. The capacity estimate also has a slight deviation. This is because the actual capacity of the battery not only depends on the frequency of charge and discharge but also on other factors such as temperature, Depth of Discharge, charge and discharge rate, charge methods, internal battery dynamics, etc. The algorithm as described in this section does not factor in these parameters but is still suitable for this application.

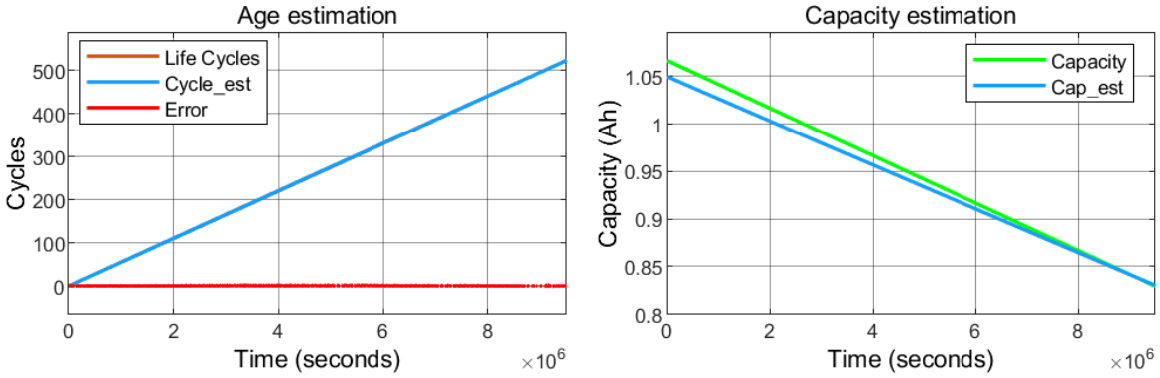


Figure 4.9: Estimation of the State of Health

The estimated capacity is used to have a better SOC estimation over time. In Figure 4.10 the error between the real SOC and estimated SOC is plotted over time. After 500 cycles (9000000s) the estimation error stays below 5%.

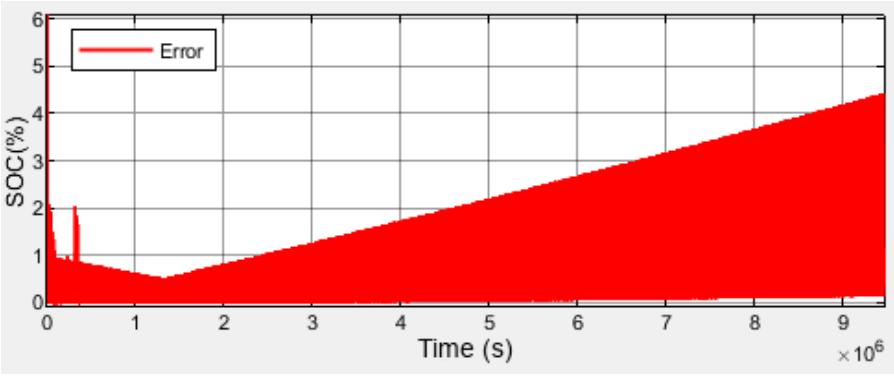


Figure 4.10: Estimation of the State of Charge after 500 cycles

4.4. Charge module

The battery cell will be charged with the Constant Current Constant Voltage method. This module takes the power from the input source of the SPPE system and converts it to the appropriate voltage to charge the battery.

4.4.1. Charge algorithm

The CCCV charge model supplies a constant current of 0.2 C (210 mA) to the battery in the first charge phase. When the battery is charged to its maximum terminal voltage, the algorithm switches to its second phase, in which it charges the battery with a constant voltage (4.2 V). When the battery is fully charged, the module disables the charging. This algorithm is implemented in the Finite State Machine depicted in Figure 4.11.

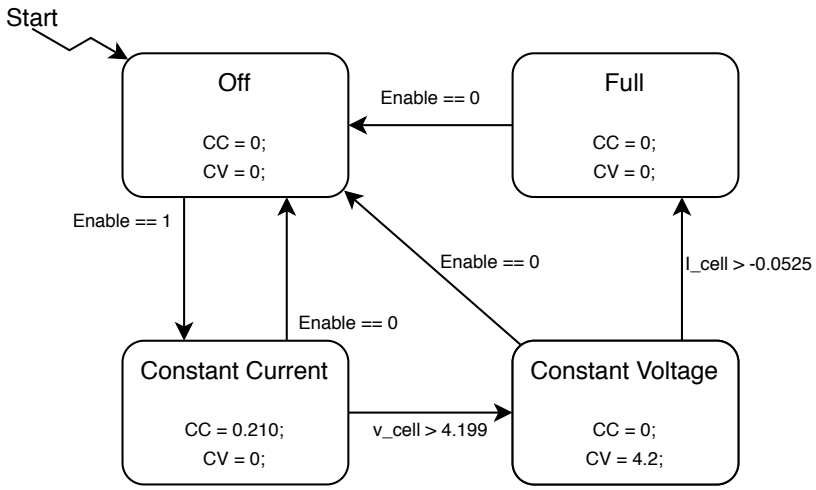


Figure 4.11: Finite State Machine of the CCCV charge module

4.4.2. Charge model

This model is implemented in Simulink using a controllable current and controlled voltage source, as seen in Figure 4.12. A diode is placed in series with the voltage source to prevent a short circuit when the source is turned off (0 V). For this model, the diode is considered to be ideal, hence its forward voltage is 0.
The system also signals when it is in Constant Current mode to the SOC estimation algorithm to indicate what estimation method should be used. A small delay is used for stability reasons in de SOC estimation.
In Figure 4.13 the results of the CCCV charge module can be seen. The charging first starts at constant charging current indicated by a negative -0.21 A current. When the threshold voltage of 4.2 V is reached, the algorithm switches to a constant 4.2 V to charge the last stage of the battery.

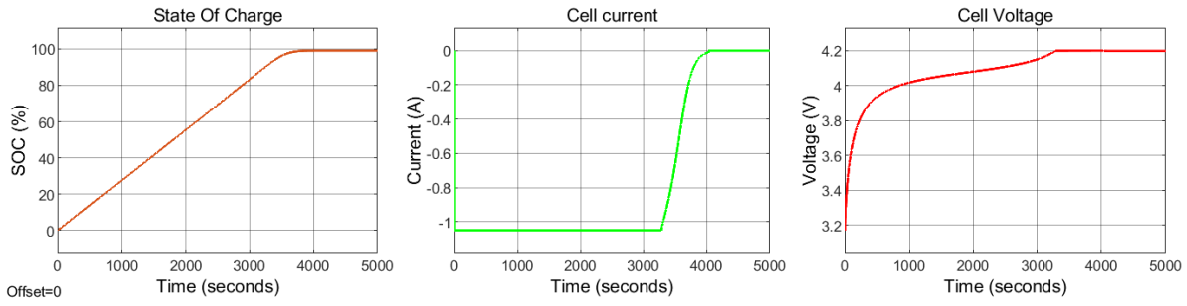


Figure 4.13: CCCV charge results at 1C charge.

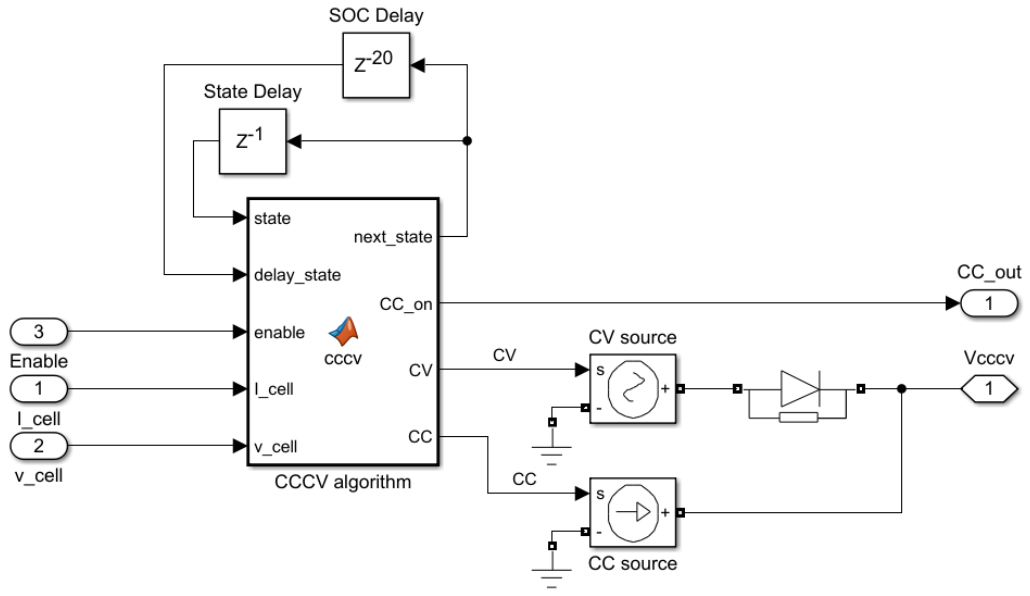


Figure 4.12: Simulink implementation of the CCCV charge module

4.5. Protective measures

Other blocks that are vital to the batteries function are the protection blocks. These ensure that if the battery starts to work outside of its operating zone, the operation is altered or shutdown. Some of these protective measures are implemented in the BMS FSM (Figure 4.15) and others are implemented as physical components.

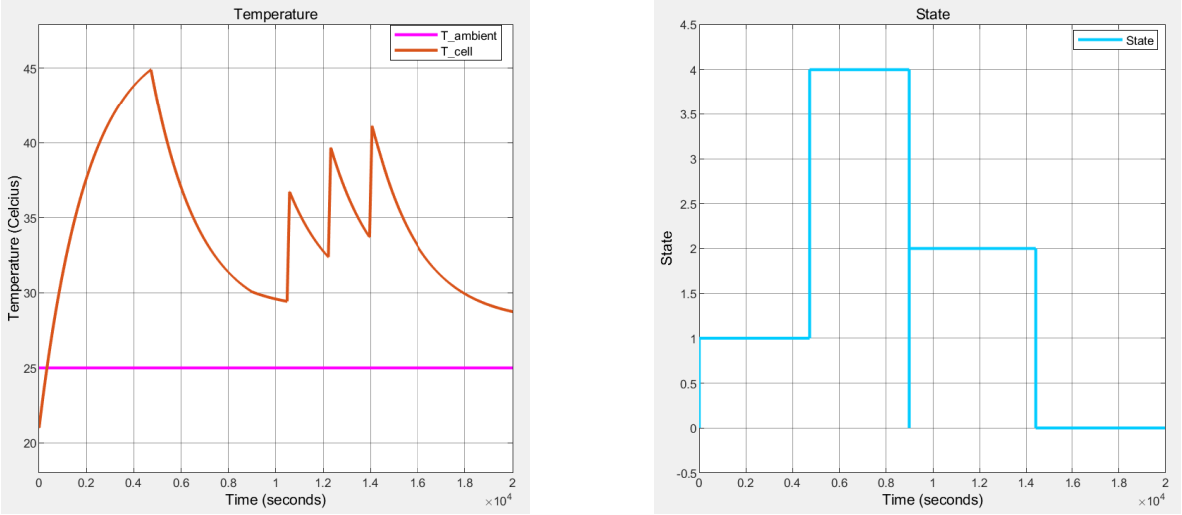
4.5.1. Overcharge protection

As can be read in Section 3.5.2, it is unwise to charge a battery with more than 0.05 V above the maximum charging voltage.
Therefore, to protect the system from this, the battery management system, as a Finite State Machine, has a fault state called "Charge error". When charging, the BMS senses the terminal voltage over the battery cell, and when this exceeds 4.25 V, which is the maximum battery voltage + 0.05 V leeway, it stops the charge cycle and shows an error. This state can only be exited when the charger is disconnected, as the problem is with the charger and not with the battery.

4.5.2. Thermal protection

Another protection that is done directly via the BSM is thermal protection. The normal operating temperature range of a Lithium-Ion type battery is within the range of 0 - 45 °C when charging and -20 - 60 °C when discharging [28]. Therefore, when the temperature is above 45 °C when the battery is charging, the BMS goes into the "Heat error state", which is signalled to the user. At this point, the SPPE is unequipped to allow it to cool down. For discharging the threshold could be set at 60 °C. However, that would not be comfortable next to your face, therefore, in the interest of human safety and comfort, the threshold is lowered to 45 °C for both charging and discharging. The heat error state can only be exited when the temperature is below 30 °C to the "Idle" state where it will sense which state is the appropriate state to go into.

The results of the thermal protection can be found in Figure 4.14. Here, the battery starts with charging, until a temperature of 45 °C is reached. After this, the battery goes into the "Heat error" state and waits until the battery is under the 30 °C threshold. At this time, the charger is no longer present, so after shortly being in the idle state, the battery starts discharging. As can be seen from the graph, discharging does not generate as much heat as charging. To test this protection, the battery setting Thermal resistance [C/W] was set to 40 instead of the usual 0.6. This device will usually stay within 1 °C of the ambient temperature and only produce this heat when in thermal runaway, due to high ambient temperature or due to an internal short circuit, which isn't resolved by the fuse.



(a) Battery Temperature °C

(b) BMS state

Figure 4.14: Battery temperature and state showing the thermal protection functioning within one charge and discharge cycle

4.5.3. Overdischarge protection

The overdischarge protection is the protection that prevents the battery from discharging after the battery reaches its cut-off voltage of 2.75 V. As stated in Section 3.5.2, overdischarging leads to the degradation of the battery, which in turn could make the battery prone to internal short circuits making it unsafe. Therefore, the BMS of this system, as seen in Figure 4.15 has a shutdown state. If the battery reaches this state, all the load will be decoupled from the battery. This is an end state and it can only be exited by a reset in the form of switching on after a power outage.

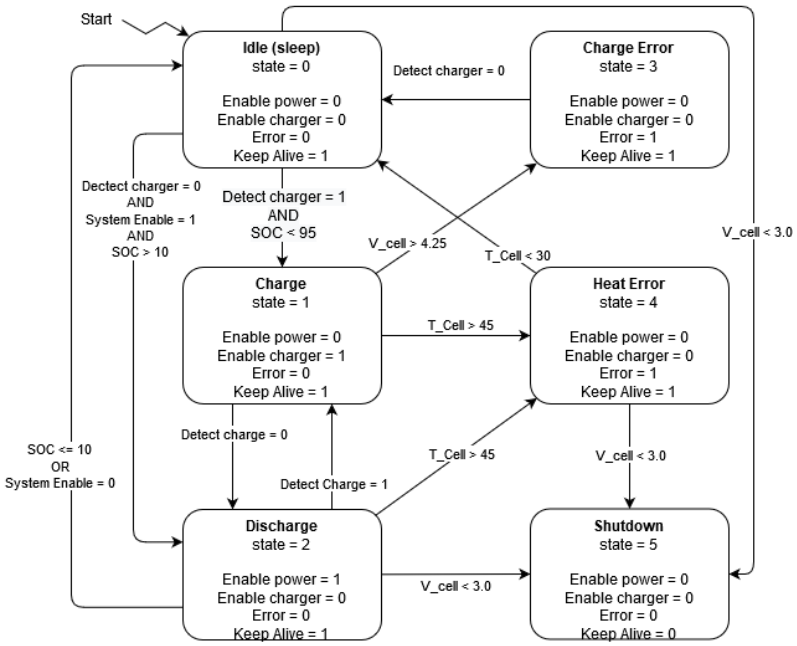


Figure 4.15: Finite state machine diagram of the complete BMS with fault states.

4.5.4. Short circuit and Quick discharge protection

As the battery is not expected to discharge with a rate larger than 2 C, everything above this is above the nominal region. As the UV-C LEDs require short power peaks, there should be some leeway for discharge currents larger than 1.5 C. However, a discharge rate larger than 2.5 C is unexpected and unwanted as it causes damage to the battery. Therefore, a fuse is implemented, to prevent this from happening. As a discharge with more than 2 C is unexpected from the system, this will most likely be caused by a short circuit of the battery, whether internal or external. Taking this into account, a fuse is a logical choice, since when it would have been, for instance, a reset button, the user would keep running into the same issue and the fabricator needs to be contacted nonetheless.

The implementation can be seen in Figure 4.16. Since the Simulink electrical power library does not have a DC fuse, this is a manual implementation. The current through the battery is measured and when it is larger than 2 C, the system opens the switch. The code for the fuse block can be found in Appendix G.

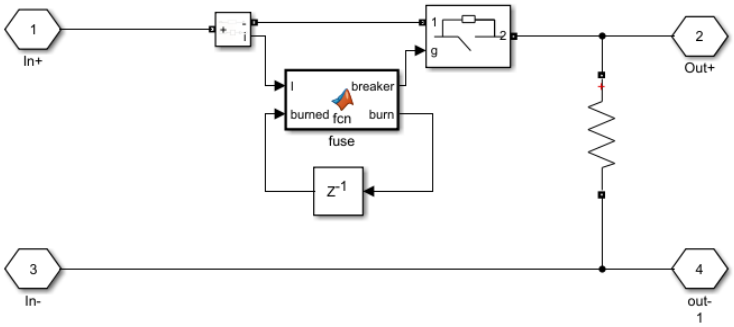


Figure 4.16: Subsystem of the fuse.

4.6. Output conversion

As mentioned in Subsection 3.1.2 the battery output voltage can vary over time. To ensure a stable output voltage to the rest of the system, an output conversion module is implemented. This module consists of two switch-mode power converters with a variable duty cycle; a buck and boost converter.

4.6.1. Buck converter

The buck converter takes the battery voltage as input and converts it to a constant 3.3 V. The model can be seen in Figure 4.17. The switch is implemented by using a MOSFET driven by a PWM generator. The duty cycle of the PWM signal is determined by measuring the input voltage of the buck converter. The output voltage over the load can be determined according to Equation 4.3. This means that the duty cycle can be adjusted by dividing the desired output voltage, as seen in Figure 4.17.

$$V_{out} = DV_{in} \tag{4.3}$$

The buck converter is tested by modelling the battery voltage curve from 4.2 V to 3.3 V to the input of the converter. The result can be seen in Figure 4.18. The input voltage of the converter is modelled after the discharge curve of the battery, starting at a fully charged voltage of 4.2 V and dropping to 3.3 V when the cell is nearly empty. We can see that when the voltage difference between V_{in} and V_{out} is larger, less current is drawn at the input. The efficiency of the converter can be determined by looking at the power input and output. In this case, the buck converter has a nominal power input of 0.85W and power output of 0.83 W. When the voltage difference is larger at t = 0, the input power is larger, 0.89 W, while the output power remains 0.83 W. This gives a nominal efficiency of 97.6% and minimal efficiency of 93.3% for the buck converter model. The ripple voltage of the buck converter resulted in 25 mV peak-to-peak.

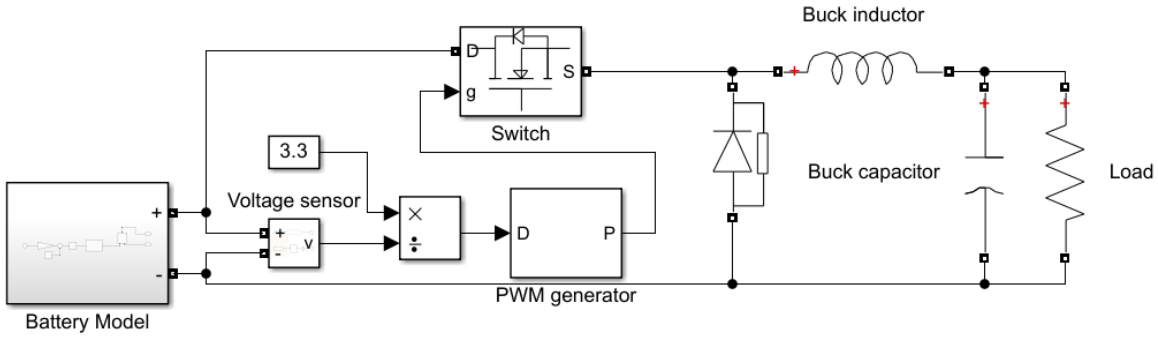


Figure 4.17: Simulink model of a buck converter

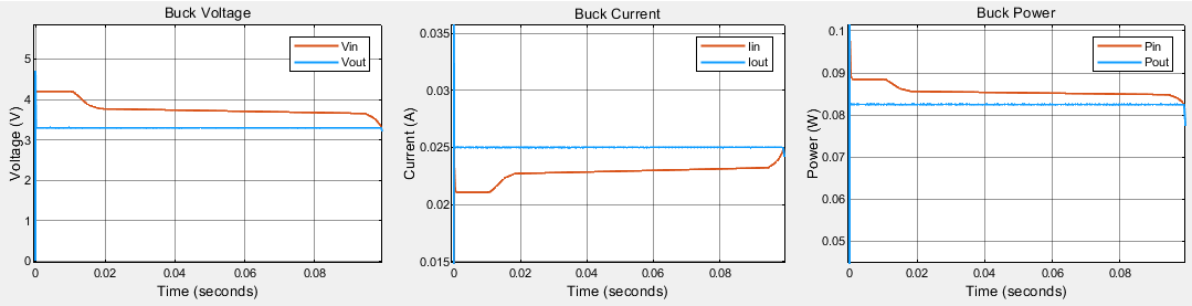


Figure 4.18: Results of the buck converter

4.6.2. Boost converter

The boost converter boosts the battery voltage to 15V DC to be supplied to the UVGI system, Figure 4.19. The boost converter works similarly as the buck converter but converts the input voltage to an output voltage according to Equation 4.4.

$$V_{out} = \frac{V_{in}}{1-D} \tag{4.4}$$

By measuring the input voltage of the boost converter the duty cycle can be altered to ensure a stable 15V output. Notice that in the figure, the duty cycle is set to ensure a 15.6V output. This is done to compensate for the forward voltage drop of the diode.

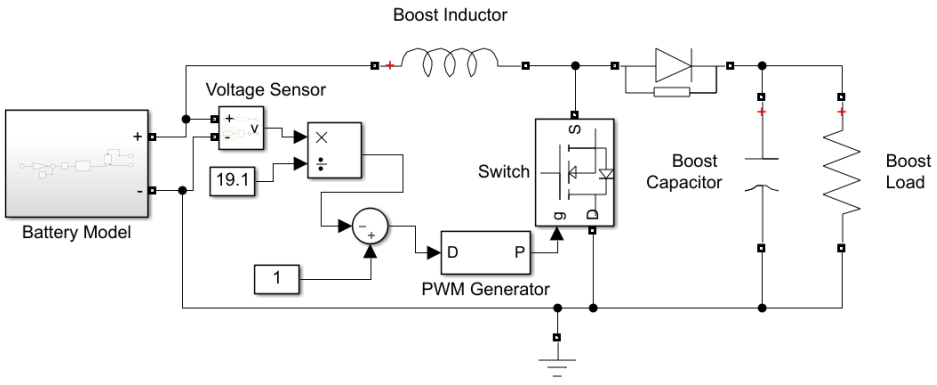


Figure 4.19: Simulink model of a boost converter

The boost converter is tested by applying the same linear voltage curve from 4.2 V to 3.3 V to the input of the converter. The result can be seen in Figure 4.20. Again, the input voltage of the converter is modelled after the discharge curve of the battery. Since the voltage difference between the input and output is much

larger than the buck converter, an oscillation pattern at t = 0 can be observed, as the inductor and capacitor of the boost converter are charged. Within 0.01 s, this oscillation stabilizes to the desired constant 15V. For the boost converter, the power input is 7.03W and the power output is 6.73 minimally and is 6.75W nominally. This gives a minimal efficiency of 95.7% a nominal efficiency of 96.0% for the boost converter model. The voltage ripple of the boost converter was found to be 0.6mV peak-to-peak.

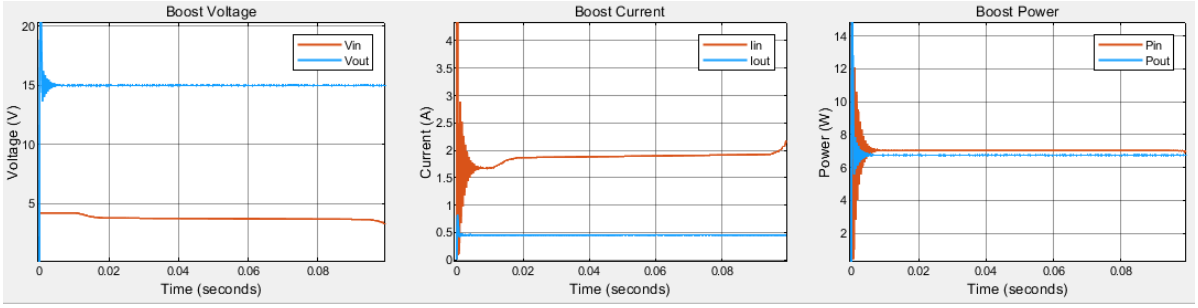


Figure 4.20: Results of the boost converter

4.6.3. Conversion model

The buck and boost converter can now be integrated with the complete model of the power system. However, due to the high switching frequency of the MOSFETs (>100 kHz), these models are not suitable for a long simulation. Instead, they are implemented as a DC-DC power converter, see Figures F.13, F.14 & F.15, with an efficiency factor as derived from the results in the previous subsections.

These converters have a set output voltage (15V or 3.3V) and measure the output current over the load. At the input, the voltage is measured as well. This measured current and voltage are then used to determine the input current according to Equation 4.5.

$$I_{in} = \frac{V_{out} I_{out}}{\eta V_{in}} \tag{4.5}$$

where η is the efficiency of the converter.

From this equation, it can be seen that the input current increases as the input voltage drops since the output voltage is held constant. This will be an important factor in determining the maximum amount of current the system will draw and henceforth the necessary capacity of the battery.

4.6.4. Load modelling

The load is modelled as resistors that draw current that is expected from the UVGI and SaC submodules. Since these modules are not active all the time, there is a low constant current draw with periodic current peaks when the other submodules are activated. The SaC module (3.3 V) requires a 0.57 A continuously and 0.25 A every 30 minutes for 2.75 minutes. The UVC LEDs (15V) require 0.263 A every 30 minutes for 1.75 minutes. This load behaviour is modelled to accurately estimate the peak currents from the battery. The load model behaviour can be seen in Appendix C.

4.7. Human interaction

To signal the state of the battery to the user, a red and green LED will be used. The LEDs are controlled by a LED driver, as seen in Figure 4.21.

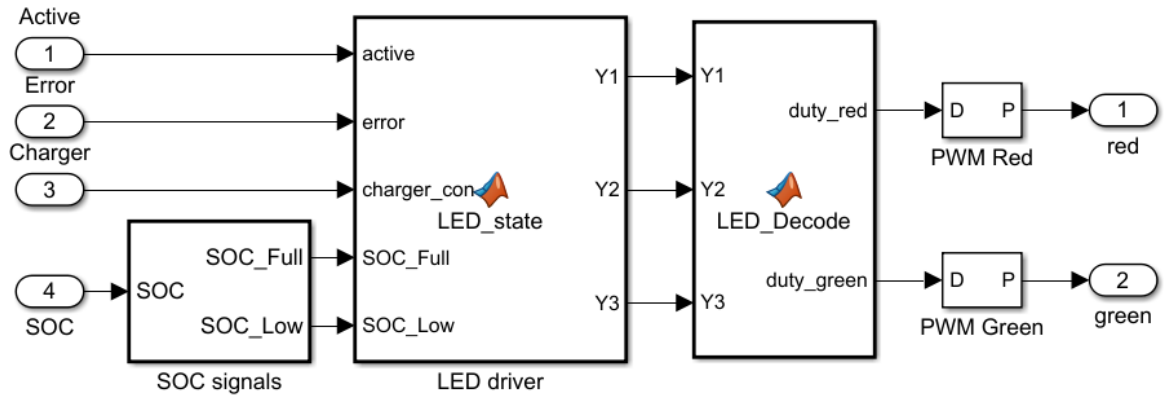


Figure 4.21: Simulink model of the LED driver

The different states of the battery and the corresponding LED action can be found in Table 4.2. The led driver determines the state of the battery with five input booleans:

1. Active (A): Indicates if the system is enabled.
2. Error (B): Indicates if an error in the system has occurred.
3. Charger (C): Indicates if the charger is connected.
4. SOC Full (D): Indicates an SOC > 95%.
5. SOC Low (E): Indicates an SOC < 25%.

By making use of a truth table, all inputs correspond to the desired outputs. The complete table can be found in Appendix B. The table can be simplified to a logic expression for each output signal according to Equations 4.6, 4.7 and 4.8.

$$Y1 = A\overline{BCDE} \tag{4.6}$$

$$Y2 = A\overline{CE} + CD + BC + AD + AB \tag{4.7}$$

$$Y3 = C\overline{D} + BC + AB \tag{4.8}$$

These codes can then be translated into a PWM signals for the LEDS with a frequency of 2 Hz to allow them to blink. When the duty cycle is 50% the LED will blink and when it is 100% the LED will be continuously on.

Table 4.2: LED signals for different system states

State	**LED action**	**Y1 Y2 Y3**
Device off	Off	000
Charging	Green blink	001
Fully charged/On	Green still	010
Heat or charge error	Red blink	011
Battery Low	Red still	100

The result of the test driver can be found in Appendix D, where it can be seen that the LED behave according to the state the BMS is in and the current level of the SOC.

4.8. Simulation results and discussion

In this section the complete Simulink model will be simulated with a testbench. This testbench will control the input signal of the BMS to have it cycle through all its states. If the BMS behaves correctly, the cycles should follow the sequence labeled in Table 4.3.

Table 4.3: Testbench states

TB cycle	State	State No.
1	Idle	0
2	Charge	1
3	Discharge	2
4	Idle	0
5	Discharge	2
6	Charge	1
7	Charge error	3
8	Idle	0
9	Discharge	2
10	Heat error	4
11	Idle	0
12	Discharge	2
13	Shutdown	5

In Figure 4.22, the states, SOC, cell voltage and cell current are plotted. The complete figure of the testbench can be found in Appendix D. The states are cycling as expected. In the first six testbench cycles, the battery is operating under normal conditions. The testbench then forces the BMS into a charge error state by charging the battery above it's rated voltage. This can also be seen in voltage and current plots, where the voltage peaks above 4.25 V and as a result the current spikes to -2.5A. The BMS then enters the charge error state and the current immediately drops back to 0.025 A only supplying power to the BMS. The testbench then waits for the charger to be disconnected and returns to idle. Next, a heat error is forced, by modifying the measured temperature to 60 °C, which is higher than the 45 °C threshold. The BMS enters the heat error state and the power output is disabled, dropping the current back to 0.025 A. When the temperature drops below 30 °C the BMS returns to idle.

To have the BMS enter shutdown mode, the battery is completely discharged and the lower bound of the SOC is set to 0. This also shows the characteristic discharge curve of the OBPM where the submodules are supplied with power, with current peaks when the control system and UVC LEDs are enabled. When the battery is near completely emptied, the voltage dips below 3 V and the shutdown state is entered. Here the BMS itself is turned off and the currents drops to 0.0 A.

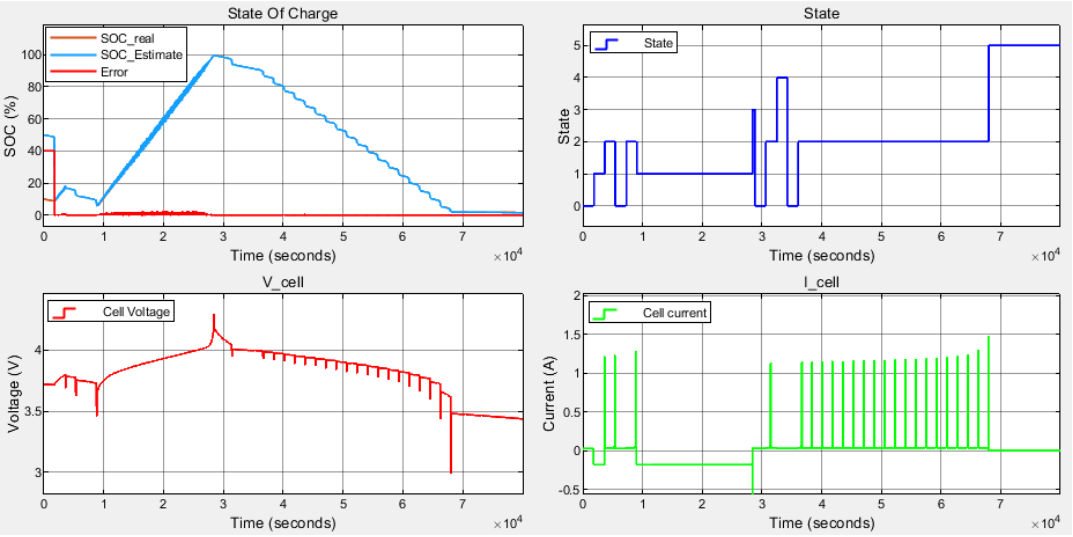


Figure 4.22: Testbench results

4.8.1. Discussion

With the results of the testbench, the Simulink model is concluded. The model gives insight into the feasibility of the design of this system. Although the system is functional in the simulator, some remarks need to be made.

Battery Model

The model is an approximation of what an actual battery would behave like. Not all parameters that the model required could be found using the datasheet of the battery, hence some typical values for lithium-ion batteries were entered. The full list of parameters can be found in Appendix A. Nevertheless, the model should be applicable for any type of battery, so although an actual battery would slightly differ from the model, the functionality of the BMS is still valid. Furthermore, key safety and control parameters, such as battery voltage and temperature, will prevent unsafe operation, regardless of deviations between practical and theoretical battery behaviour.

SOC estimation

Although the SOC estimation is approximate enough for the BMS to function, some improvements could still be made. It is assumed that the charge curve is the same for every charge cycle, however, ageing and temperature effects can alter this curve over time. Currently, the SOC estimation does not correct for this.

SOH estimation

The SOH estimation can only estimate the number of cycles measuring input and output current. Although this approach seems intuitive, this only takes into account charge and discharge. The battery model computes the cycles differently, namely in equivalent life cycles. This estimate relates the cycle life to the ambient temperature of the battery cell, charge rate and discharge rate. However, for the lifetime of the power module, this estimate comes close to the real capacity of the battery.

Charge Module

The module only uses an algorithm that implements the CCCV charging method. However, when the battery voltage has dropped under its minimal voltage, the battery should be charged differently until it has returned to its normal voltage range, for example by trickle charging. Since the battery model does not emulate this kind of behaviour, this feature was left out.

Output module

The buck and boost converters of the output module are modelled with ideal components. This means that when the implementation of these converters is realised, these components will more likely be selected based on compatibility with ICs and size.

5 Design Implementation

Finally, the Simulink design is adapted to use off-the-shelf components to design a PCB for the final product. In this process, the final components were chosen and they were either verified by implementing the chosen components back in Simulink where applicable, by using an Arduino simulator or they could not be verified. For components in the latter case, a test plan is written.

The chapter starts with choosing a battery in Section 5.1 and a battery gauge in Section 5.2. Next, in Section 5.3 a microcontroller is chosen and programmed. Subsequently, in Section 5.4 the protection measures are discussed. Furthermore, in Section 5.5 an IC for the charging circuit is selected and implemented. In Section 5.6, the ICs for the Buck and Boost converter are chosen and an estimation is made about the efficiency. Next, the implementation of the Human interaction is shown in Section 5.7. Lastly, in Section 5.8 the full system integration is shown and in Section 5.9 a test plan is made. Ultimately, in Section 5.10 a conclusion is drawn about how this process went and what could be improved.

5.1. Battery

The requirement that has the most impact on the design is the requirement that the battery should power the system for at least eight hours.

5.1.1. Capacity

In determining the capacity of the battery, the power consumption of the entire SPPE needs to be inventoried first. The total power consumption is displayed in Table 5.1.

Table 5.1: Current consumption broken down per submodule and in active or sleep mode

	Current (mA)	On-Time (minutes)	Period (hours)	Total consumption (mAh)
Sensing & Control submodule				
Active	25	2.75	0.5	18.2
Sleep	0.57	27.25	0.5	4.1
UVGI submodule				
Active at 15 V	263	1.75	0.5	552.9
Active at 3.3 V	4.8	1.75	0.5	2.2
Sleep	0.001	28.25	0.5	0.015
Power submodule				
Active	25	30	0.5	198.2
Total needed capacity			8	775.6

From this table it can be concluded that the minimal battery capacity should be 775.6 mAh for an operating time of 8 hours. Although, in constructing this table, for overview's sake, some small quiescent and inactive currents of small IC components have been neglected as well as some losses, so the minimal battery capacity should be chosen higher than this. However, these small currents are in the order of micro Ampere, therefore these can be neglected compared to the overall current flows.

5.1.2. (Dis)charge rate

Another important metric is the charge and discharge rate. This is expressed in 1 C, where 1 C expresses the current with which the battery would discharge its total capacity in one hour. This metric is not particularly constricting for charging the battery as it would normally not be charged with a high current as to increase the battery life since this is more important than charging the battery quickly.

However, the nature of this system is that once per half hour the LEDs start irradiating the filter. This means that compared to the baseline of the system there are now peak currents present. As the LEDs draw 263 mA at 15 V, this means that they draw 1.07 A at nominal battery voltage and 1.23 A when the battery is nearly empty (3.2 V). Therefore, the battery needs to be either 1,230 mAh with a 1 C rate or it can be smaller, but then with a 2 C quick discharging rate.

Taking the above into account and also that this project is realised with off-the-shelf components, the battery is selected to be 1050 mAh. The choice between off-the-shelf components was between an 800 mAh, 1050 mAh or 1500 mAh battery. Ultimately, the reason that the 1050 mAh was selected over the 800 mAh battery, was because it can ensure that the system always has 8 hours of charge, even when the battery aged and has lost some capacity. Secondly, this battery has the option to discharge with 10C (or less). Therefore, the battery of 1500 mAh, which would be twice the needed size, doesn't have to be used. The downside to this is that the 1050 mAh battery would degrade a little when it needs to discharge over 1 A, however, this is outweighed by the space saved on the PCB in the filter head and it is more comfortable for the user as it is lighter. The full specifications of the battery can be found in the datasheet [28].

5.2. Battery gauge

Battery gauging can be done in several ways. Solutions range from a simple voltage and current sensor to full Battery Management Systems. Ultimately, the BQ27426 was chosen for its ability to also sense the battery temperature, voltage and current. Also, this gauge is in a very small form factor, very energy efficient and can power directly from the battery itself. The other reason this gauge was chosen, was that it mimics the solution to estimate the SOC layout in Section 4.3.3.

The solutions mentioned in the previous chapter uses the OCV method at the charging stage as the initial conditions are not known, such that with coulomb counting an estimate of the SOC cannot be given. When discharging, the coulomb counting method was more reliable as the OCV method cannot interpolate accurately enough. The BQ27426 fuel gauge uses a patented technique from Texas Instruments called impedance-track fuel gauging [30]. Here Texas Instruments mentions that they measure the impedance of the battery's cell as a key input to estimate the remaining capacity. They do not mention how specifically they measure this. Where their system comes close to the previously proposed method is that the BQ27426 uses the OCV method when the system is off or in sleep, when coulomb counting does not work as the system is asleep/off, so continuously integrating is not possible. When the system is on, coulomb counting is used with the initial condition provided by the OCV method.

Furthermore, the gauge also has an automatic SOH estimator function. It derives this from a pre-programmed battery chemistry, combined with charge/discharge and temperature measurements. This enables the gauge to also compensate the SOC for the SOH. It also compensates the SOC for different temperature and charge rates, which it again derives from a pre-programmed battery chemistry.

5.3. Microcontroller

The microcontroller will function as the brain of the on-board power system. It will behave similarly to the BMS module mentioned in Chapter 4.3.

5.3.1. Selection

To select the right microcontroller, the signal the controller will be reading and writing needs to be known. An overview of the input and output signals of the BMS can be found in Table 5.2. From this table, the controller needs an I2C interface for communications and nine general input-output pins. Furthermore, the microcontroller should be low power, operable at 3.3 V and accessible for prototyping.

For this project, the ATmega328P was chosen. This is an 8-bit AVR microcontroller which offers high performance while keeping power consumption low. It also offers six sleep modes to reduce power consumption even further when the device does not need to be active [31]. The ATmega328P can be programmed using a syntax that is closely related to C/C++. Normally the ATmega chip is integrated on a more prototyping friendly board, the Arduino series. However, by making only use of the chip itself and adding to supporting components, power consumption can be reduced significantly. For the chip to function it has to be supplied with power (3.3V) and an oscillator to function as the system clock. According to the datasheet[31], the frequency of the oscillator cannot exceed 10 MHz when using 3.3V, hence an oscillator of 8 MHz was chosen. The wiring of these components can be found in Appendix E.

Table 5.2: Input and output signals of the BMS

Signal	I/O	PWM	Pin	Description
SDA	I/O	Yes	SDA0 (27)	I2C data port
SCL	I/O	Yes	SCL0 (28)	I2C clock
Enable_system	I	No	D2 (32)	Enables the UVC-leds and control system
Charger_detect	I	No	D3 (1)	Signal from CCCV module, HIGH if a charger is connected
CCCV_Enable	O	No	D4 (2)	Enables the CCCV module to charge the battery
Buck_Enable	O	No	D5 (9)	Enables the buck converter to output 3.3 V
Boost_Enable	O	No	D6 (10)	Enables the boost converter to output 15V
GPIO_Sens	I	No	D7 (11)	Signal from battery sensors, i.e. low SOC
LED_R	O	No	D8 (12)	Output for red indicator LED
LED_G	O	No	D9 (13)	Output for green indicator LED
Keep_Alive	O	No	D10(14)	Keeps system enabled

5.3.2. Programming the ATmega328P

The ATmega328P's programming consists mainly of two parts, a setup part in which the chip and other devices are initialized and a loop which runs continuously after the setup is finished. The complete code of the BMS can be found in Appendix H.

Setup

The setup defines the direction of the pins, enables the I2C protocol by using the *wire.h* library and configures the battery gauge with the right battery parameters.

Loop

The loop implements the Finite State Machine described in Section 4.5.1. It starts by updating all parameters needed for the FSM by requesting the SOC, Cell Voltage and Cell current from the battery gauge and by reading the digital input pins that detect the charger and system activation. Based on these inputs, it determines the next state of the FSM and writes the matching signals to the rest of the OBPM system. The last part of the loop set the indicator LEDs to the right mode to indicate the user about the state of the BMS. After this step, the microcontroller loops back to the starting point.

When FSM is in idle mode, it enters a low power mode. This reduces the power consumption of the ATmega328P chip, which increases the operating time of the BMS. The low power mode is exited every 8 seconds to update the BMS or when an interrupt pin signal is changed (pin 2 or 3), for example when the system is enabled or the charger is plugged in.

5.4. Battery protection

The battery protection is closely modelled after the battery protection in Section 4.5. As the microcontroller is still closely following the Finite State Diagram of Figure 4.15, most protection is done by the microcontroller. A difference in the charge error state would be that instead of going into this state by sensing that the battery voltage is too high, it takes the fault flag from the CCCV charger as input. However, referring to the documentation of the CCCV charger, it was unclear how to read out this flag, so in this implementation, still, the output voltage of the battery is used to determine if there is a charging fault.

One of the protection measures built-in is the shutdown state (state 5). This state is the only state that does not output the signal "Keep_Alive". The only way that the BMS can go into this state is if the battery voltage is below 2.75 V, which is a critical level. Therefore, the shutdown state is a last resort state. The "Keep_Alive" signal is, as can be seen in the schematic of the PCB design (Appendix E) under the LDO and cold start logic section, together with the "Charger_detect" signal necessary to power via "VCC" all the control logic of the on-board power management system, which is in turn necessary to be able to power the load of the other submodules.

Another external safety measure is the fuse at the output of the battery. The fuse is set at 2.5 A, as it is not expected nor desired to have higher currents than this. If this were to happen, this would be due to an internal battery error and the battery would need replacing. Therefore, having to also change a fuse at the same time outweighs having a push-button type fuse. In the latter case, the user has the potential to fumble with the fuse, leading to battery overheating, due to internal battery degradation, which in the worst-case scenario could lead to thermal runaway, with venting and a possible explosion as consequence.

5.5. Charging circuit

The charging circuit needs to be close to the same design as in Section 4.4.1. Therefore, the output needs to be able to reach 4.2 V and at least between 210 mA and 1 A. There should be a charge fault output that can be inputted in the microcontroller.

Taking the above and the requirement of a small form factor into account. The choice fell on the NCP1835 Integrated Li-Ion CCCV Charger by ON Semiconductor [32]. This charger can supply a 4.2 V CV stage and has a selectable CC stage between 300 mA and 1 A. This is a bit higher than the required 210 mA, which means it isn't possible to charge the battery at 0.2 C. However, charging it at 0.3 C is also still acceptable.

5.6. Output regulation

The power output of the OBPM system is implemented with two ICs. One buck converter IC to step down the battery voltage to 3.3V and one boost converter IC to step up to voltage to 15V.

5.6.1. Buck Converter

The buck converter is implemented by using the TPS62802 [33]. This IC functions as a 4 MHz switch that modulates its switching signal based on the set output voltage. The IC requires an external input and output capacitor, an inductor and resistor to complete the buck configuration. Values of these components were derived from the datasheet.

An important reason for choosing a switchmode buck converter was power conversion efficiency. This IC automatically enters a power save mode in which high efficiencies can be achieved. In Figure 5.1 the efficiency curve of the buck converter is shown at an output voltage of 3.3V. The buck converter supplies power to the SaC submodule, which draws approximately 25 mA of current. For the nominal voltage of the battery (3.7V), the efficiency will be around 93%.

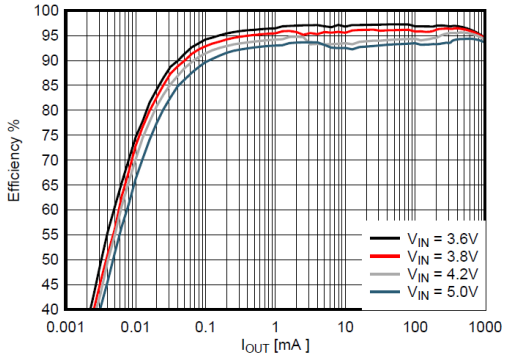


Figure 5.1: Efficiency of the buck converter ($V_{out} = 3.3V$)

5.6.2. Boost Converter

The implementation of the boost converter is realised with the TPS55330 IC [34]. The IC works similarly as the buck converter, but with a switching frequency of 350 kHz. The IC also requires external components to properly function:

1. Input & output capacitor, inductor and diode for boost configuration
2. Two resistors for a feedback loop that determines the output voltage
3. Miscellaneous resistors and capacitors to set IC parameters

Just as with the buck converter, the efficiency of the boost converter is important to analyse. For boost converter configured to boost to 15V, Figure 5.2 shows the efficiency versus output current. The boost converter powers the UVC LEDs which draw 263 mA. This means that the boost converter will have an efficiency of 93 to 94 %.

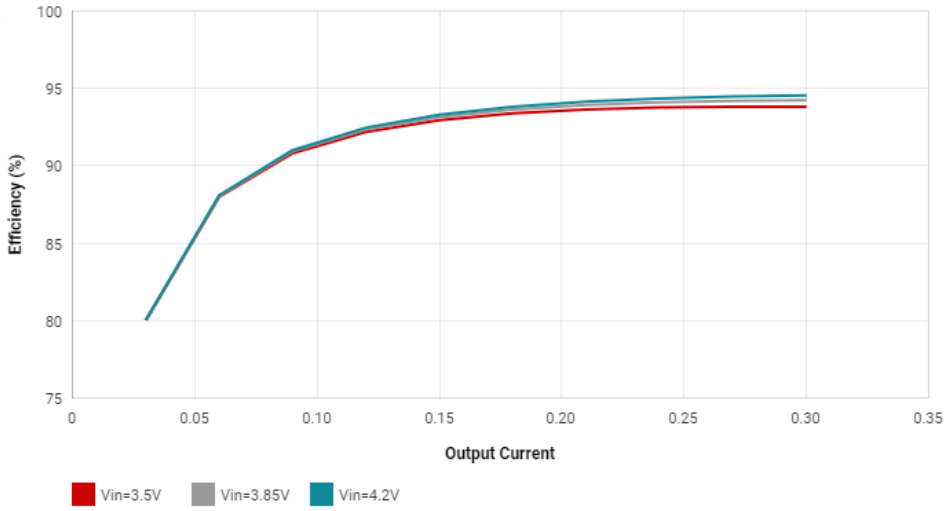


Figure 5.2: Efficiency of the boost converter ($V_{out} = 15V$)

The complete circuit and component values can be found in Appendix E.

5.7. Human Interaction

The interaction options would normally go on the outside of the device. However, designing a full face mask falls out of the scope of the thesis. Therefore, the LEDs and on/off switch are placed on the PCB together with all the other subsystems. The schematics can be found in Appendix E.

5.7.1. LEDs

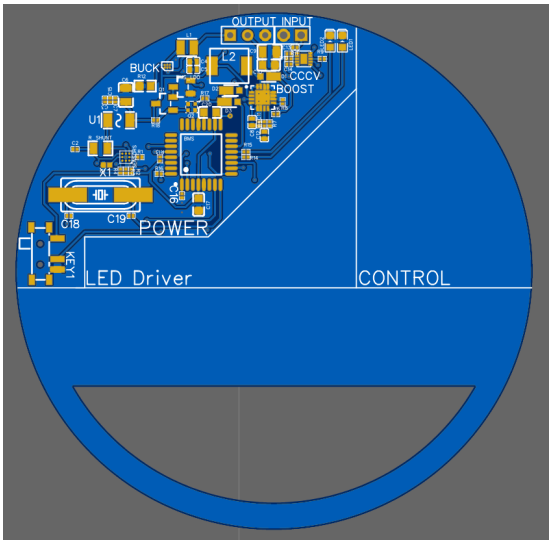
The working of the LED closely follows the same design as presented in Subsection 4.7. In the actual application they are driven by the microcontroller (see Appendix H). At the input of the LEDs, there are resistors to set the current needed by the diodes to emit light.

5.7.2. On/off Switch

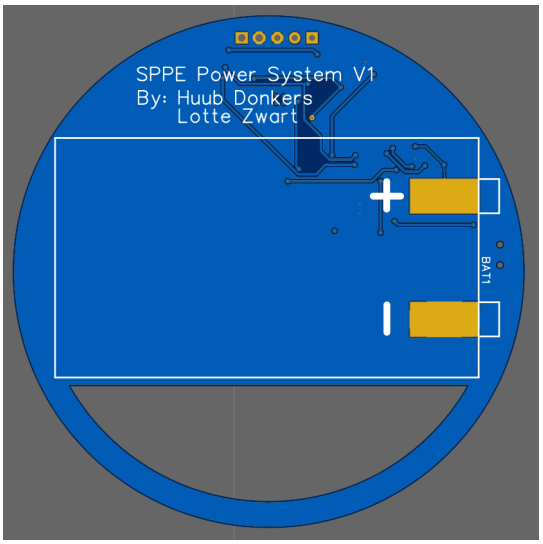
This switch sends the signal "System_enable" to the BMS. Without this signal, the BMS would still be on and measuring, permitted that the battery is full enough, but it has the load decoupled. This can also be seen in the Finite State Diagram of Figure 4.15. Therefore, if the system is turned off by the user it is never really off until the BMS goes into shutdown state. This is, to ensure that the battery protection keeps working. If the user switches the device on, but the LED is not doing anything, the user knows the battery is fully drained and he knows to charge it.

5.8. Full system integration

The complete schematic of the on-board power system can be found in Appendix E. This schematic was realised on a circular PCB as seen in Figure 5.3. The circular form was chosen so that the PCB would fit inside the SPPE package. The battery cell limited the small size of the PCB, so a diameter of 7.5cm was needed to fit the battery on the board and still leave space for an airflow hole on the bottom of the PCB. The PCB also leaves room for the other two submodules to be integrated on the board.



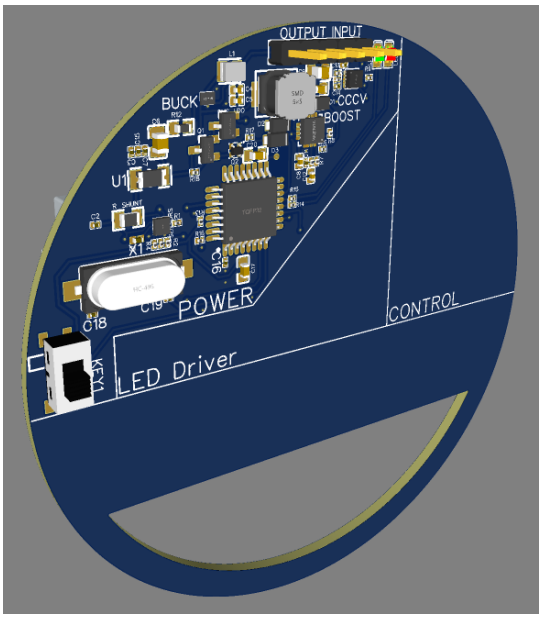
(a) Front view



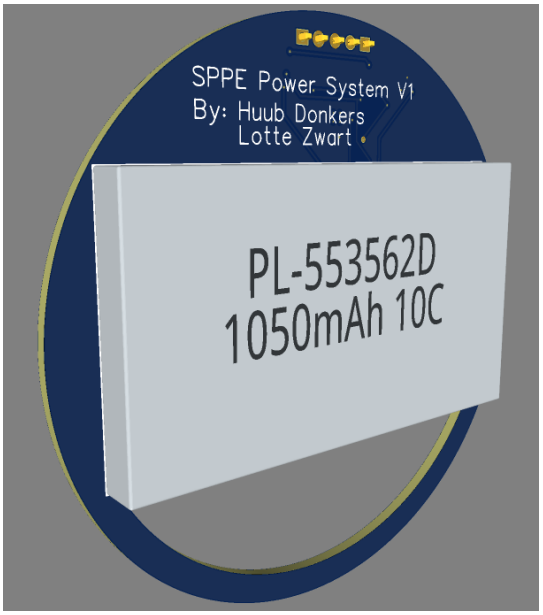
(b) Back view

Figure 5.3: PCB design of the OBPM.

To get a better feel for the dimensions of the PCB design, a 3D model was created. From Figure 5.4 it can be seen that the battery cell will largely be responsible for the thickness of the whole SSPE package.



(a) Front view



(b) Back view

Figure 5.4: 3D PCB design of the OBPM.

5.9. Test plan

Due to the restrictions of the BAP-project, the PCB design of the OBPM system could not be prototyped and tested. In this subsection, a test plan is formulated to verify the functionality of the implementation design.

First, all implementations must be verified individually. This would preferably be done by having each IC mounted on a break-out board for easy prototyping. This way all components can be connected via a breadboard and replacement can be done rapidly.

5.9.1. Battery Gauge

The I2C communication with the battery gauge needs to be verified. Although code has been written to set up and read the gauge, this is not tested. If the code functions as intended, the battery gauge will be set up for a lithium-ion battery of 1050 mAh with a maximum voltage of 4.2V and outputs the SOC, cell voltage and cell temperature when requested. Since the battery gauge acts as a slave device, the master device can be realised by any microcontroller with an I2C interface.

5.9.2. Charger

The charger IC needs to be tested to see if it outputs the right constant current. In this case, the expected output current would be 300mA. The charger IC can be connected to a lithium-ion battery and measured with a scope to check if the CCCV algorithm is functional. It should also be checked if 2.8 V is measured on the V2P8 pin to indicate that a charger is connected.

5.9.3. Buck converter

For the buck converter, it needs to be verified that it outputs 3.3 V and that the voltage ripple is within the designed range. This needs to be done for the full output range of the battery, from 2.75 V up to 4.2 V. Even though the battery will never be able to have an output when it is at 2.75 V, due to the protection circuitry of the microcontroller, it is still interesting to test the behaviour of the buck converter when the input voltage is lower than the output voltage. This is because it is realistic that the input will at times be lower than 3.3 V.

5.9.4. Boost converter

First of all, it is important to see that the boost converter indeed outputs 15 V and that the voltage ripple is within designed ranges. While verifying this, special attention should be paid to the start-up phase. From Figure 4.20 it can be seen that the converter draws a lot of oscillating current on the input. However, Simulink and other simulation programs neglect real-life losses and parasitic capacitances, therefore it is important to see if in real life it also performs as in the simulation. If this is the case then further steps need to be taken to prevent this. The fuse will definitively blow if the system would be connected as it is only rated up until 2.5 A. Luckily, the selected battery can handle current rates up to 10 C, but this is not a desired situation.

5.9.5. Microcontroller

When all ICs have been verified, they can be connected to the microcontroller. First, it should be confirmed that the microcontroller reads all its inputs as expected and that the signals are connected to the right pins. This can be done with a serial monitor for example. When all inputs are read as expected, they can be inputted to the BMS FSM and the output signals can be measured with a scope or multimeter.
If the microcontroller functions as intended, the low power mode can be tested. If it behaves as intended, the device should wake up every 8 seconds or when an interrupt pin is changed.

5.9.6. Protection measures

With a verified microcontroller and ICs, the system can be completely built and the protective measures can be tested. For this, a similar testbench would need to be realised, as was used to test the Simulink model in Section 4.8. This means that the BMS should cycle through all its states and end in a shutdown mode. To do this safely, a battery cell should not be used, as over- and undercharging can be damaging to the cell. Instead, a variable power supply and series resistor can be used to model the battery voltage, keeping it between the upper and lower bound of the battery gauge input. If this system works as intended, a similar response as seen in Figure 4.22 should be acquired.

5.9.7. Human interaction

For the LEDs, should be checked to behave according to Table 4.2. To force all states a controlled voltage source can be used as well as a blow dryer on the microcontroller to simulate an overheating event.

5.10. Discussion

Although a final PCB is made and a testplan is construed, there are a few points that are suboptimal in the current design.

Battery selection

In selecting the battery size it is assumed that the microcontroller always needs to be on. However, as it turned out, a low-power option could be implemented, such that instead of drawing 25 mA continuously, it draws only 0.57 mA, which could have reduced the battery sizing and with that the size.

Choice of ICs

Due to time constraint in this project, several ICs have been considered, but the choice was between 10 ICs per component. Therefore, with the exception of the CCCV charger, all the ICs are within our specifications, but better performing or low power ICs could have been found.

For the CCCV charger, the requirement was to charge it with 0.2 C as this charge rate prolongs battery life. From Subsection 4.4.1 it became apparent that with low charge rates a CCCV charger was preferred over a constant power constant voltage charger. However, for a larger charge rate (1 C), the reverse was true. For the charge IC, therefore a requirement was to be able to charge it with 2 C. However, the charge range of this IC is between 300 mA and 1 A, which is above 210 mA. This was fine at the point there was still a larger battery due to projected larger loads (battery sizes that were considered where 2665 and 1500 mA). However, with a change in the final week a new charge IC was not implemented and the battery is now charging with 0.3 C, which is functional, but suboptimal.

Trace thickness

Another important aspect of the design is the trace width. The PCB was designed with, where possible, traces with a size of 10 mil or 0.254 mm. These are rated for currents up to 2.35. This was assumed to be a maximum trace width. However, this is not the case and for the final design this will not restrict the output, however, there is a fuse that is rated until 2.5 A. There may be currents higher than 2.35. When this is the case, the traces will start heating up. However, the only case this will happen is when a short circuit is occurring, so the current will rapidly increase, leaving not a lot of time for the traces to heat up as the fuse will be quickly tripped.

PCB Lay-out

Lastly, in hindsight, the layout could have been routed better. At the current moment, as can be seen in Figures 5.3 & 5.4, the battery is put at the top, close to the edge, where the bottom still has space. A better layout could have been in the battery in the centre. Here, the PCB could have been made smaller. Also, the components (except for the non-SMD components) could have been above the battery, this would make integrating it in a filter head like the artist impressions of Figure 1.2, easier as there is room to clamp it down in the design.

6 Discussion

6.1. Assessment of requirements

In this section, the requirements specified in Chapter 2 will be assessed by comparing them to the results of the OBPM design.

6.1.1. General requirements

Table 6.1 lists all the general SPPE requirements. It can be seen that most requirements have been met. The operation time of the battery was based on the power consumption in Table 5.1 and the lifetime of the components was based on the cycle life of the battery. This is at least 500 cycles, but since the battery cell is not completely discharged and charged, the actual cycles expected is estimated to be 750. By assuming the device is used for 300 days every year, this results in a lifetime of 2.5 years. This is, however, a rough estimate and further testing of the battery is necessary to acquire a more accurate number.
The replaceability and airtightness of the SPPE were not considered for this PCB design, so these requirements do not apply to the OBPM design. The last requirement could either be a pass or a fail. For this to be certain, more research needs to be done to the total weight and maximal dimension of the SPPE package and the type of mask that it will be fitted to.

Table 6.1: Requirement assessment of the general requirements

General requirements	**Result**	**Pass**
The SPPE uses UVGI to sterilize the filter material	N/a	N/a
The SPPE's mechanical filter is replaceable	N/a	N/a
The SPPE's electronics are reusable after the mechanical filter is replaced	Integrated	Yes
The SPPE's battery operation time is at least eight hours	~ 10h	Yes
The components' lifetime is at least two years	~ 2.5y	Yes
The SPPE is designed preferably using off-the-shelf components	Standard ICs	Yes
The SPPE is designed circularly	Partly reusable	Yes
The SPPE should communicate internal conditions to the user	Indicator LEDs	Yes
The SPPE filter module should be airtight, apart from the airflow channel	N/a	N/a
The SPPE should be small enough to fit the side of a PPE	7.5cm diameter	Yes/No

6.1.2. Functional requirements

Table 6.2 shows all functional requirements of the OBPM system, which have all been met. It should be noted, however, that most of these results were obtained from the Simulink simulation and are therefore only an estimate of what results a real system would give. For example, the power efficiency of the output modules is only a characteristic of the datasheets. An actual measurement would have to be performed to obtain a more accurate result.
The battery cell current limits are usually limited by the design of the other submodules. However, they are protected to operate out of bounds by the implementation of a fuse.
The SOC estimation is generally around 2% when a noisy voltage is measured, as seen in Figure 4.22. This is only tested in Simulink however and the SOC that is estimated by the implemented battery gauge could have different characteristics. This is also the case for the power efficiency and ripple voltages of the buck converters found in Section 4.6.

Table 6.2: Requirement assessment of the functional requirements

Functional requirements	Result	Pass
The operating time of the power system is at least eight hours	~ 10h	Yes
The power system must inform the user about the state of the battery	Indicator LEDs	Yes
The battery cell must operate within its specified cell voltage range	BMS limits	Yes
The cell charge current cannot exceed the maximum rated charge current	Fuse protection	Yes
The cell discharge current cannot exceed the maximum rated discharge current	Fuse protection	Yes
The battery cell must operate within its specified temperature range	BMS limits	Yes
The State-of-Charge deviation must not exceed 5%	Max. 3%	Yes
The power efficiency must be at least 90 %	Average 93%	Yes
The ripple voltage of the output terminals should preferably be lower than 0.1%		
3.3V output ripple	25 mVpp (0.083%)	Yes
15V output ripple	0.6 mVpp (0.009%)	Yes

6.1.3. Non-functional requirements

The final requirements to check are listed in Table 6.3. These have all been fulfilled, but some can be debatable. The battery management system has got protection circuitry, but these are, apart from the fuse, implemented digitally. This still should protect the battery cell, but are not so much a 'circuit'. The medical safety standards are also arbitrary. Although all components individually were chosen to work within the parameters of the design, this, of course, does not guarantee that the complete system adheres to medical standards as well. The only way to be certain of this is by applying for a medical CE mark and having the design tested to EU medical standards.
The OBPM system has a separate controller from the rest of the system. However, it should be noted that it is still possible to combine this SaC group to make the complete system more space-efficient.

Table 6.3: Requirement assessment of the non-functional requirements

Non-functional requirements	Result	Pass
The battery management system should have protection circuits	Integrated	Yes
The battery system should adhere to medical safety standards	Component limits	Yes/No
The battery should be charged via a 5V micro USB connection	Vcc CCCV 2.8 - 6.5V	Yes
The output of the battery system shall contain a 15 volt and 3.3 volt terminal	15 V and 3.3 V out	Yes
The battery controller has a separate (micro)controller	ATmega382P	Yes
The PCB design should preferably make use of surface mounted devices	Mostly	Yes
The PCB design should not exceed 8 cm in diameter	7.5cm	Yes

6.2. Differences Simulink model and implementation

The implementation of OBPM system aimed to stay as close to the Simulink model as possible. Although it is functionally similar, some alterations to the implementation were made.

The SOC and SOH are not estimated by the algorithms as described in Chapter 4. The battery gauge IC already estimated these, so it was practical to use these in the implementation. There would have been a good reason to use our own algorithms to estimate the SOC and SOH if they were more accurate. However, since this cannot be confirmed and the battery gauge IC is produced by the well-known tech company TexasInstruments, the SOC from the battery gauge was chosen instead.

Furthermore, the buck and boost converter operate in a slightly different way than a regular buck and boost converter. They rely on a feedback input that tries to minimize the error between the actual output voltage and the desired output voltage.

The implementation also features a cold start circuit that is not part of the Simulink model. This will enable the ATmega328 chip to power up even when it doesn't keep itself powered. Instead, the charger enables the power input for the microcontroller. This system was not yet tested.

7

Conclusion & Recommendations

The goal of this project was to design a complete system with a functional battery management system, that supplies required power to the rest of the system, ensures safe battery operation and aims to maximize battery life. In this, the project has succeeded as the module has fulfilled almost all requirements. The design has been worked out as much as possible within the boundaries that were set. The final result is a 3D PCB design which is now ready to be tested.

Simulink offered a very intuitive way of modelling the OBPM system. It is very flexible and many parameters could be easily adjusted to reflect real-life components more closely. There is some learning curve in setting up all of the different components, but when this stage is passed, running multiple different simulations can be done with little alterations to the complete system.

What makes this design unique is the approach to estimate the SOC in the Simulink model. In Chapter 5, it became apparent that the battery gauge uses some of the same principles. However, the design is not the same and future research could be done to see which implementation would work best. Furthermore, this project implemented as a low-power, small form factor battery management system. Most Battery Management Systems are designed for electric vehicles, so in this context, this project is quite unheard of and could benefit other research that needs a BMS in which they have a low average load with incidental large current peaks.

7.1. Recommendations

The first thing that should be done to continue this project is to test all assumptions made in Chapter 5 and follow the test plan stated in Section 5.9. Once this is finished and the system is adjusted where needed, it can be integrated with the other submodules to form the electrical core of the SPPE.

One of the things worth looking into is how the PCB can be made smaller by changing the battery. As a person needs to wear this on his/her face for an entire day, comfort is a factor and making the filter heads smaller is making it lighter, which should be more comfortable. Also, making the filter heads smaller is less intimidating to patients when a health care professional is wearing it.
There are two ways to accomplish this. Either, two smaller batteries are selected, which together have the same capacity as the original battery and if they are put in series they can output the required output current at around 1C per battery. Or a flexible lithium-ion battery is used. The latter kind is up and coming [5], but unfortunately not a lot of research has yet been done on this topic. The option of two batteries would require cell-balancing, which would also take a bit of extra energy from the battery and space on the PCB. Both options are worth investigating.

Furthermore, it is worth looking into changing the buck converter to an LDO. If this doesn't negatively impact the output efficiency, then it would probably save components and space.

Lastly, the ATmega328P that this submodule uses is the same model and type as the Sensing and Control module. In the interest of power-saving, it might be interesting to look into whether both systems could be partially redesigned as to use the same microcontroller. This would save 25 mA at peak performance and 0.57 mA continuously.

Bibliography

[1] M.-W. Wang, M.-Y. Zhou, G.-H. Ji, L. Ye, Y.-R. Cheng, Z.-H. Feng, J. Chen, "Mask crisis during the COVID-19 outbreak," *European Review for Medical and Pharmacological Sciences*, vol. 24, no. 6, pp. 3397–3399, 2020.

[2] D. Wang, et al., "Epidemiological characteristics and transmission model of Corona Virus Disease 2019 in China," *Journal of infection*, vol. 80, no. 5, pp. 25–27, 2020.

[3] W. Kowalsky, *Ultraviolet Germicidal Irradiation Handbook: UVGI for Air and Surface Disinfection*. Berlin, Germany: Springer, 2009.

[4] M. A. Hannan, M. M. Hoque, A. Hussain, Y. Yusof, and P. J. Ker, "State-of-the-art and energy management system of lithium-ion batteries in electric vehicle applications: Issues and recommendations," *IEEE Access*, vol. 6, pp. 19 362–19 378, 2018.

[5] K. Clemens, "Flexible lithium ion battery breakthrough," May 2019, accessed Jun. 9, 2020. [Online]. Available: https://www.designnews.com/electronics-test/flexible-lithium-ion-battery-breakthrough/176830709560875

[6] R. Zhang, B. Xia, B. Li, L. Cao, Y. Lai, W. Zheng, H. Wang, and W. Wang, "State of the art of lithium-ion battery soc estimation for electrical vehicles," *Energies*, vol. 11, p. 1820, 07 2018.

[7] M. Morris and S. Tosunoglu, "Comparison of rechargeable battery technologies," *ASME Early Career Technical Journal*, November 2012.

[8] H. Kiehne, *Battery Technology Handbook*, 2nd ed. Florida, USA: CRC Press, 2003.

[9] Battery University, "Bu-107: Comparison table of secondary batteries," July 2019, accessed April. 21, 2020. [Online]. Available: https://batteryuniversity.com/index.php/learn/article/secondary_batteries

[10] Z. Zhang and S. Zhang, *Rechargeable Batteries*, 1st ed. Berlin, Germany: Springer, 2015.

[11] G. W. A. Bhatt, R. Whithers, "Lithium-ion batteries," March 2016, accessed June 19th, 2020. [Online]. Available: https://www.science.org.au/curious/technology-future/lithium-ion-batteries

[12] S. S. Zhang, "The effect of the charging protocol on the cycle life of a li-ion battery," *Journal of Power Sources*, vol. 161, no. 2, pp. 1385 – 1391, 2006.

[13] S. M. Rezvanizaniani, Z. Liu, Y. Chen, and J. Lee, "Review and recent advances in battery health monitoring and prognostics technologies for electric vehicle (ev) safety and mobility," *Journal of Power Sources*, vol. 256, pp. 110 – 124, 2014.

[14] L. Buccolini, A. Ricci, C. Scavongelli, G. DeMaso-Gentile, S. Orcioni, and M. Conti, "Battery management system (bms) simulation environment for electric vehicles," *2016 IEEE 16th International Conference on Environment and Electrical Engineering (EEEIC)*, pp. 1–6, 2016.

[15] M. Ichimura, "The safety characteristics of lithium-ion batteries for mobile phones and the nail penetration test," in *INTELEC 07 - 29th International Telecommunications Energy Conference*, 2007, pp. 687–692.

[16] M. Chen, F. Bai, S. Lin, W. Song, Y. Li and Z. Feng, "Performance and safety protection of internal short circuit in lithium-ion battery based on a multilayer electro-thermal coupling model," *Applied Thermal Engineering*, vol. 146, pp. 775 – 784, 2019.

[17] 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 Adv.*, vol. 8, pp. 33 414–33 424, 2018.

[18] T. S. Bryden, A. Holland, G. Hilton, B. Dimitrov, C. P. de León Albarrán, and A. Cruden, "Lithium-ion degradation at varying discharge rates," *Energy Procedia*, vol. 151, pp. 194 – 198, 2018, 3rd Annual

Conference in Energy Storage and Its Applications, 3rd CDT-ESA-AC, 11–12 September 2018, The University of Sheffield, UK.

[19] E. Rogers, *Understanding Boost Power Stages in Switchmode Power Supplies*, Texas Instruments, March 1999, accessed Jun. 18, 2020. [Online]. Available: https://www.ti.com/lit/an/slva061/slva061.pdf?ts=1592421405523

[20] ——, *Understanding Buck Power Stages in Switchmode Power Supplies*, Texas Instruments, March 1999, accessed Jun. 18, 2020. [Online]. Available: https://www.ti.com/lit/an/slva057/slva057.pdf?ts=1592422319540

[21] J. Kühnel, "Voltage regulators for power management," *Analog Dialogue*, vol. 30, no. 4, 1996.

[22] International Electrotechnical Commission (IEC), "Iec 60601-1 medical electrical equipment - part 1: General requirements for basic safety and essential performance," 2005, accessed Jun. 18, 2020. [Online]. Available: https://www.ele.uri.edu/courses/bme484/iec60601-1ed3.0_parts.pdf

[23] J. Brandstetter et al., *Successful Practices for Battery Powered Medical Devices*, Advanced Medical Technology Association (AdvaMed), 2017, accessed Jun. 18, 2020. [Online]. Available: https://www.advamed.org/sites/default/files/resource/advamed_successful_practices_for_battery_powered_devices_final_final_updated_toc.pdf

[24] European Parliament, "Directive 2007/47/ec of the european parliament and of the council of 5 september 2007 amending council directive 90/385/eec on the approximation of the laws of the member states relating to active implantable medical devices, council directive 93/42/eec concerning medical devices and directive 98/8/ec concerning the placing of biocidal products on the market (text with eea relevance)," 2007, accessed Jun. 18, 2020. [Online]. Available: https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX:32007L0047

[25] *What requirements should your medical and laboratory devices comply with to be allowed on the European market?*, Ministry of Foreign Affairs of the Netherlands, 2017, accessed Apr. 24, 2020. [Online]. Available: https://www.cbi.eu/market-information/medical-laboratory-devices/what-requirements-should-you-comply/

[26] Mathworks, "Simscape block libraries," 2020, accessed Apr. 28, 2020. [Online]. Available: https://nl.mathworks.com/help/physmod/simscape/ug/introducing-the-simscape-block-libraries.html

[27] MathWorks, "Battery," Mar 2008, accessed Apr. 28, 2020. [Online]. Available: https://nl.mathworks.com/help/physmod/sps/powersys/ref/battery.html

[28] *Polymer Lithium Ion PL-553562D 1050 mAh Battery Data Sheet*, AA Portable Power Corp, 2005, accessed Jun. 17, 2020. [Online]. Available: https://www.batteryspace.com/prod-specs/2158.pdf

[29] Woodbank Communications Ltd, "Battery life (and death)," 2005, accessed Jun. 19, 2020. [Online]. Available: https://www.mpoweruk.com/life.htm

[30] P. Fundaro, *Impedance Track™ Based Fuel Gauging*, Texas Instruments, September 2007, accessed Jun. 16, 2020. [Online]. Available: https://www.ti.com/lit/wp/slpy002/slpy002.pdf?ts=1592298006259&ref_url=https%253A%252F%252Fwww.google.com%252F

[31] *ATmega328P: 8-bit AVR Microcontroller with 32K Bytes In-System Programmable Flash*, Atmel Corporation, January 2015, accessed Jun. 16, 2020. [Online]. Available: http://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P_Datasheet.pdf

[32] *NCP1835 Integrated Li-Ion Charger*, 5th ed., ON Semiconductor, February 2009, accessed Jun. 2, 2020. [Online]. Available: https://www.onsemi.com/pub/Collateral/NCP1835-D.PDF

[33] *TPS6280x 1.8-V to 5.5-V, 0.6A / 1-A, 2.3-µA IQ Step Down Converter*, 2nd ed., Texas Instruments, January 2019, accessed Jun. 16, 2020. [Online]. Available: https://www.ti.com/lit/ds/symlink/tps62807.pdf?ts=1592394799185&ref_url=https%253A%252F%252Fwww.google.com%252F

[34] *TPS55330 Integrated 5-A, 24-V Boost/SEPIC/Flyback DC-DC Regulator*, 2nd ed., Texas Instruments, January 2019, accessed Jun. 16, 2020. [Online]. Available: https://www.ti.com/lit/ds/symlink/tps55330.pdf?ts=1592396145984&ref_url=https%253A%252F%252Fwww.google.com%252F

Appendix

A

Battery model parameters

Table A.1 list all parameters that were used in the Simulink model of the lithium battery cell [27].

Table A.1: All battery parameters of the Simscape battery model used.

Parameter	**Value**	**Unit**
Nominal voltage	3.7	V
Rated capacity	1.00	Ah
Initial SOC	10	%
Battery Response Time	30	s
Discharge		
Maximum capacity	1.05	Ah
Cut-off Voltage	2.75	V
Fully charged voltage	4.23	V
Nominal discharge current	0.21	A
Internal resistance	0.040	Ohm
Nominal capacity	1.00	Ah
Exponential zone	[3.9, 0.05]	[V, Ah]
Temperature		
Initial cell temperature	21	°C
Nominal ambient temperature T1	25	°C
Second ambient temperature T2	45	°C
Maximum capacity at T2	1.05*0.96	Ah
Initial discharge voltage	4.0	V
Voltage at 90% maximum capacity	3.9	V
Exponential zone	[4.0, 0.1]	[V, Ah]
Thermal resistance	0.6	°C/W
Thermal time constant	2000	s
Heat loss difference	0	W
Aging		
Initial battery age	0	Cycles
Ambient temperature Ta1	25	°C
Capacity at EOL	1.05*0.8	Ah
Internal resistance at EOL	0.04*1.2	Ohm
Charge current (nominal, maximum)	[0.210, 1.05]	A
Discharge current (nominal, maximum)	[0.210, 10.5]	A
Cycle life at 100% DOD, Ic and Id	500	Cycles
Cycle life at 25% DOD, Ic and Id	1800	Cycles
Cycle life at 100% DOD, Ic and Idmax	51	Cycles
Cycle life at 100% DOD, Icmax and Id	450	Cycles
Cycle life at 100% DOD, Ic and Id at T2	450	Cycles

B

LED Truth Table

Table B.1 list all possible inputs and corresponding output signals for the LED driver. Equations 4.6, 4.7 and 4.8 were derived based on this table.

Table B.1: Truth table for the inputs and outputs of the LED driver logic

Signals					**Output**			**Description**	
active	**error**	**charger**	**SOC_Full**	**SOC_Low**	**Y1**	**Y2**	**Y3**	**State**	**LED**
0	0	0	0	0	0	0	0	Device off	off
0	0	0	0	1	0	0	0	Device off	off
0	0	0	1	0	0	0	0	Device off	off
0	0	0	1	1	0	0	0	Device off	off
0	0	1	0	0	0	0	1	Charging	green blink
0	0	1	0	1	0	0	1	Charging	green blink
0	0	1	1	0	0	1	0	Fully charged/On	green still
0	0	1	1	1	0	1	0	Fully charged/On	green still
0	1	0	0	0	0	0	0	Device off	off
0	1	0	0	1	0	0	0	Device off	off
0	1	0	1	0	0	0	0	Device off	off
0	1	0	1	1	0	0	0	Device off	off
0	1	1	0	0	0	1	1	Heat or charge error	red blink
0	1	1	0	1	0	1	1	Heat or charge error	red blink
0	1	1	1	0	0	1	1	Heat or charge error	red blink
0	1	1	1	1	0	1	1	Heat or charge error	red blink
1	0	0	0	0	0	1	0	Fully charged/On	green still
1	0	0	0	1	1	0	0	Battery Low	red still
1	0	0	1	0	0	1	0	Fully charged/On	green still
1	0	0	1	1	0	1	0	Fully charged/On	green still
1	0	1	0	0	0	0	1	Charging	green blink
1	0	1	0	1	0	0	1	Charging	green blink
1	0	1	1	0	0	1	0	Fully charged/On	green still
1	0	1	1	1	0	1	0	Fully charged/On	green still
1	1	0	0	0	0	1	1	Heat or charge error	red blink
1	1	0	0	1	0	1	1	Heat or charge error	red blink
1	1	0	1	0	0	1	1	Heat or charge error	red blink
1	1	0	1	1	0	1	1	Heat or charge error	red blink
1	1	1	0	0	0	1	1	Heat or charge error	red blink
1	1	1	0	1	0	1	1	Heat or charge error	red blink
1	1	1	1	0	0	1	1	Heat or charge error	red blink
1	1	1	1	1	0	1	1	Heat or charge error	red blink

C

Load model results

Figure C.1 illustrates the load behaviour of the SaC and UVGI modules. These are characterised by current peaks that occur every 30 minutes. The UVC LEDs require 263 mA at 15V every 30 minutes for 1.75 minutes and the control logic requires a continuous current of 0.57 mA and peak currents of 25mA every 30 minutes for 2.75 minutes at 3.3 V.

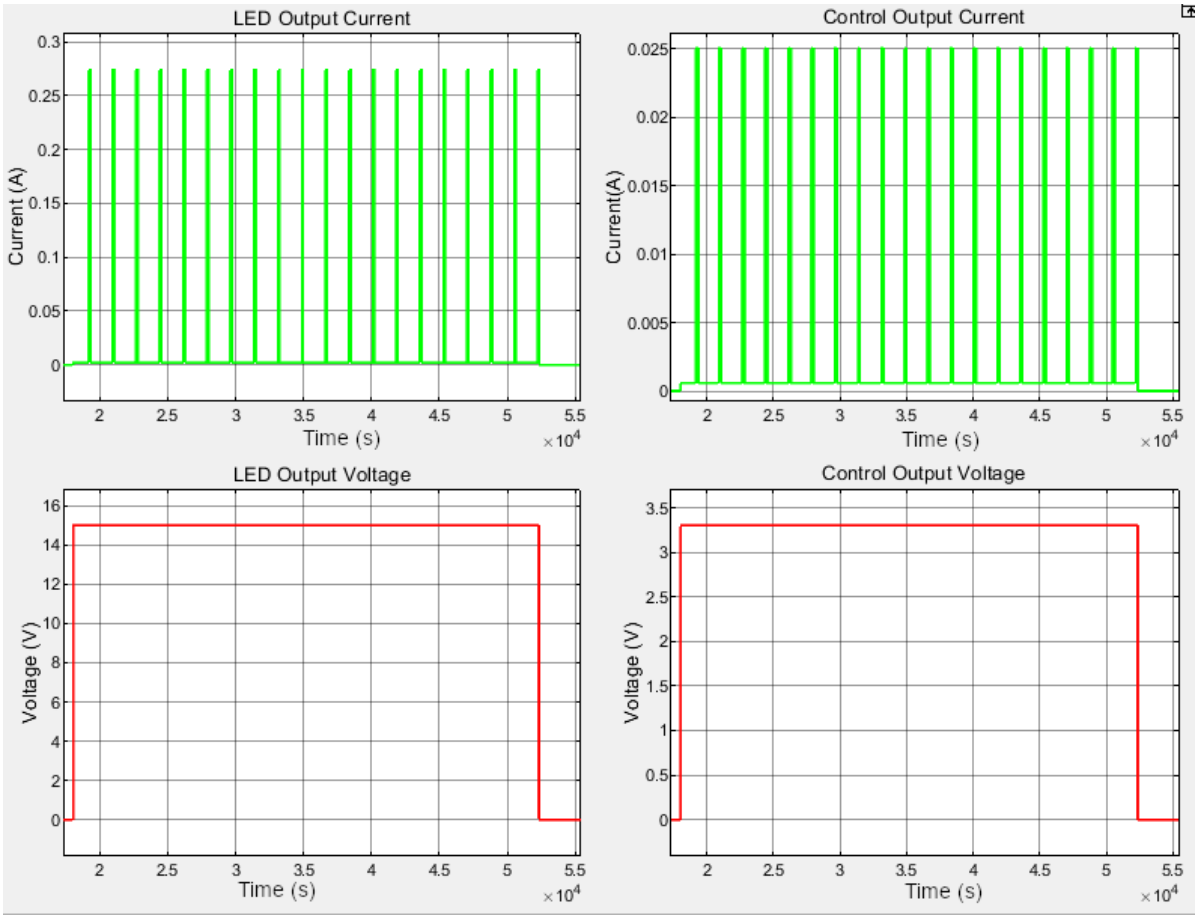


Figure C.1: Model of the subsystem loads

D

Testbench results

Results of the testbench as described in Section 4.8.

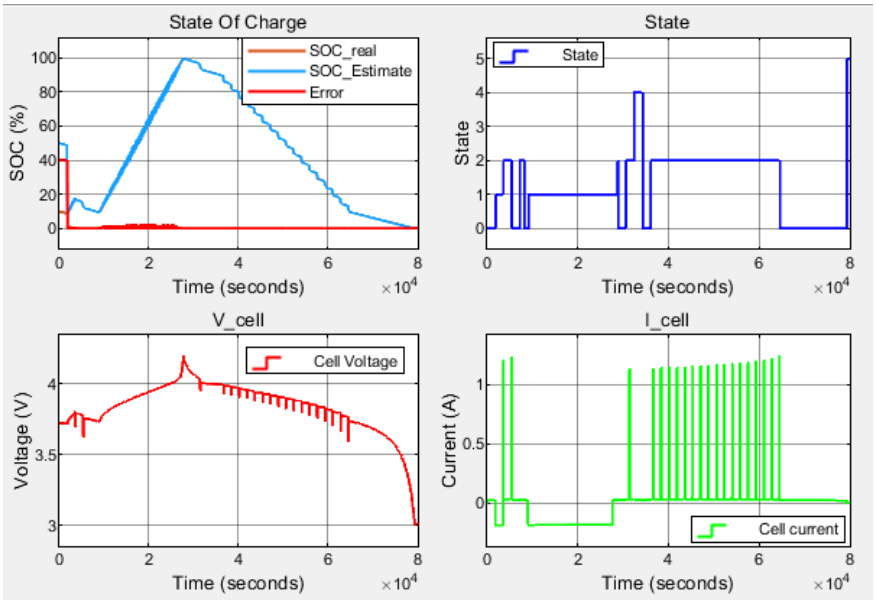


Figure D.1: Simulink Testbench results (SOC, States, Cell Voltage and Cell Current)

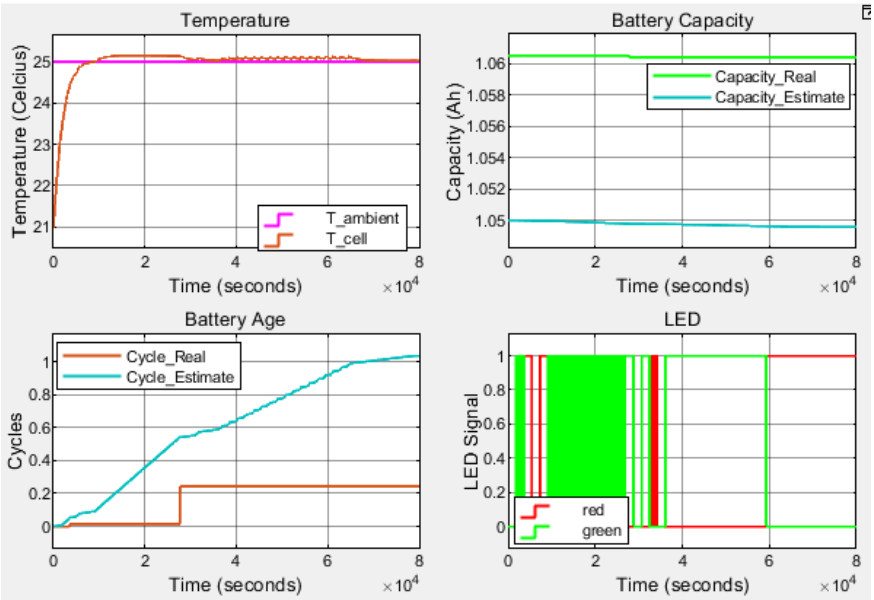


Figure D.2: Simulink Testbench results (Cell Temperature, Capacity, Age and LED signals)

E

Schematic of the circuit design

Final design of the OBPM circuit schematic. This design was created in EasyEda, a web-based EDA-toolsuite. The design is split up in sections for a more clear overview and resembles the design layout of the Simulink model.

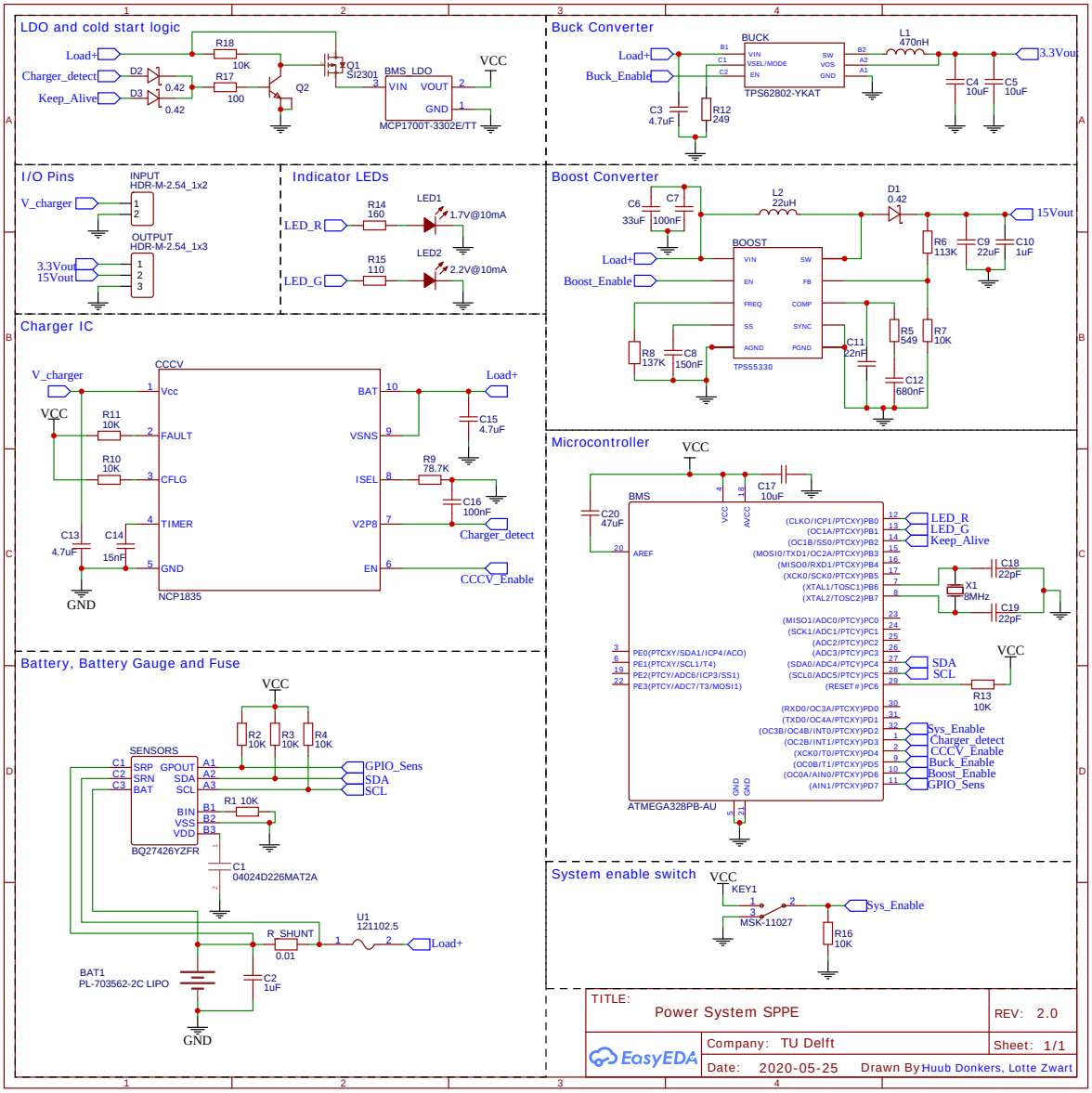


Figure E.1: Circuit design of the OBPM subsystem

F

Simulink model

This chapter illustrates the complete Simulink model. All blocks, modules and connections are visualised in Figure F.1 to F.24.

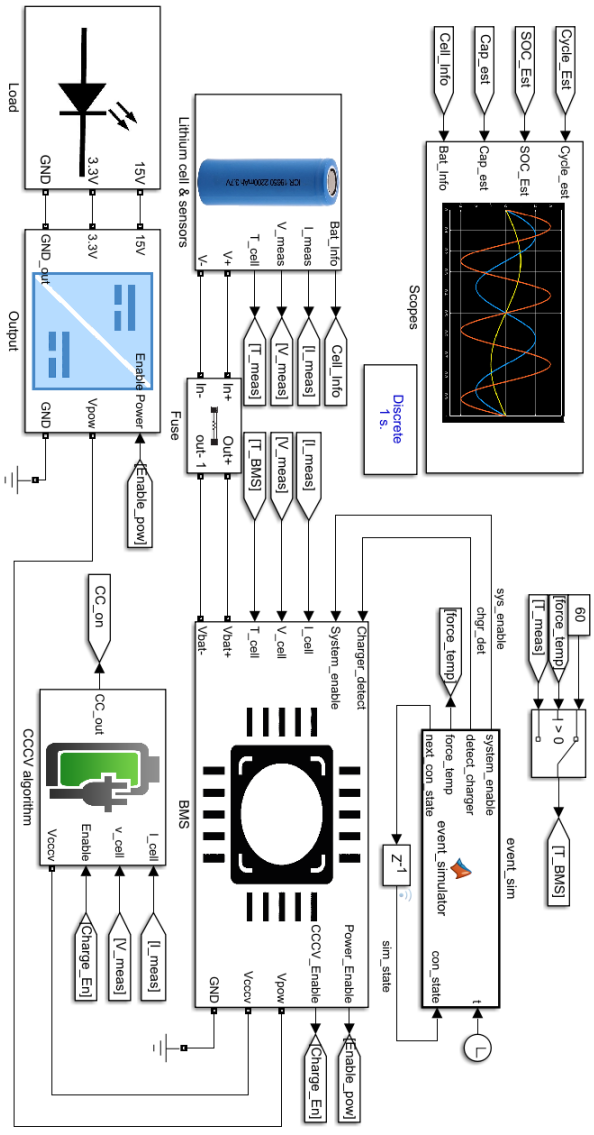


Figure F.1: Simulink model of the power system

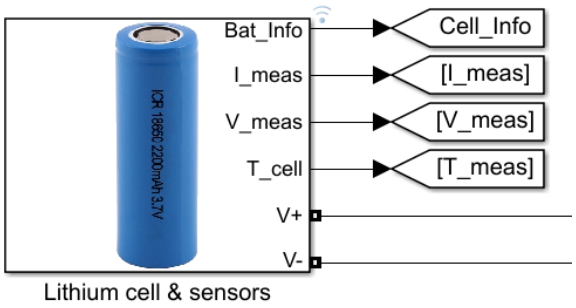


Figure F.2: Simulink block of the battery cell

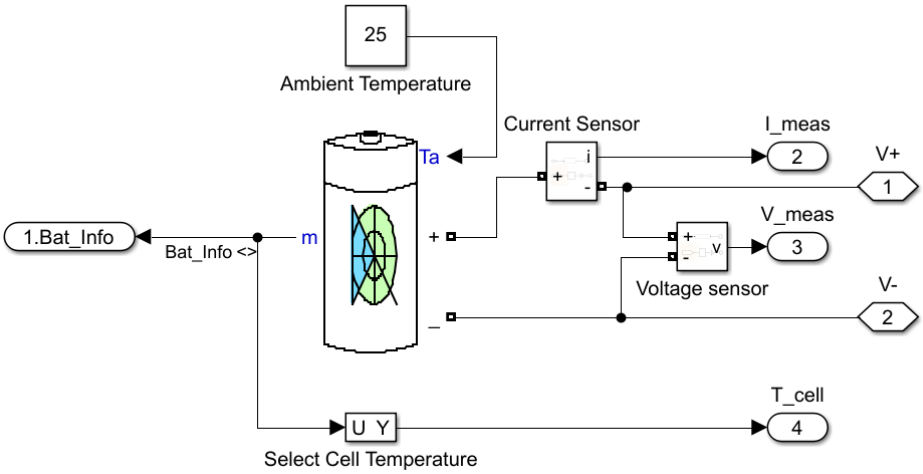


Figure F.3: Internal structure of the battery cell

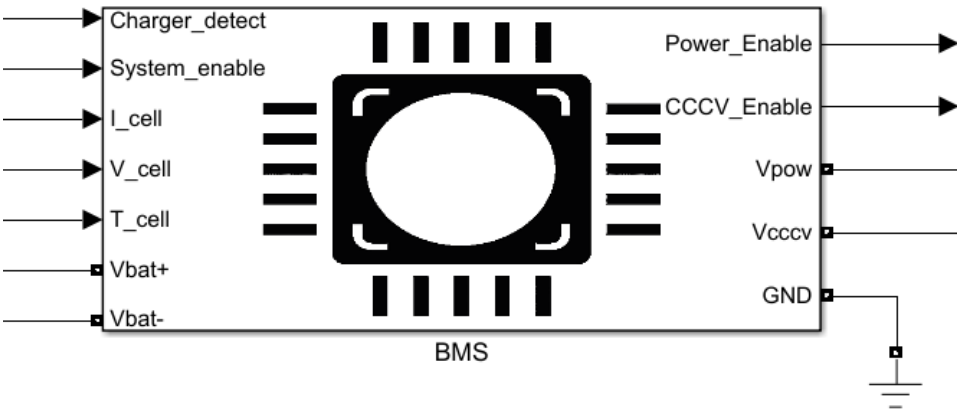


Figure F.4: Simulink block of the BMS

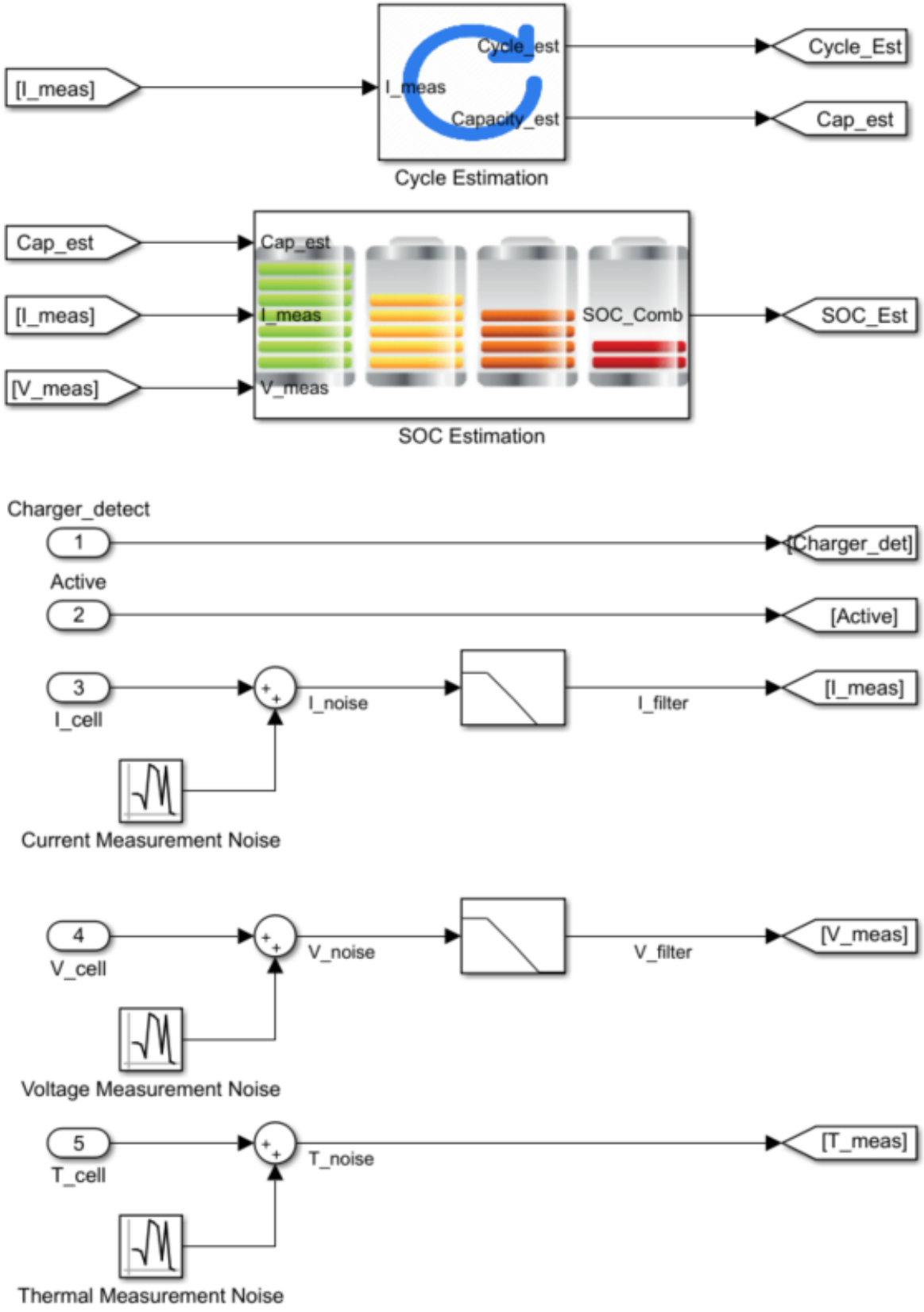


Figure F.5: Internal structure of the BMS (SOH estimation, SOC estimation, input signals with noise and subsequent noise filtering).

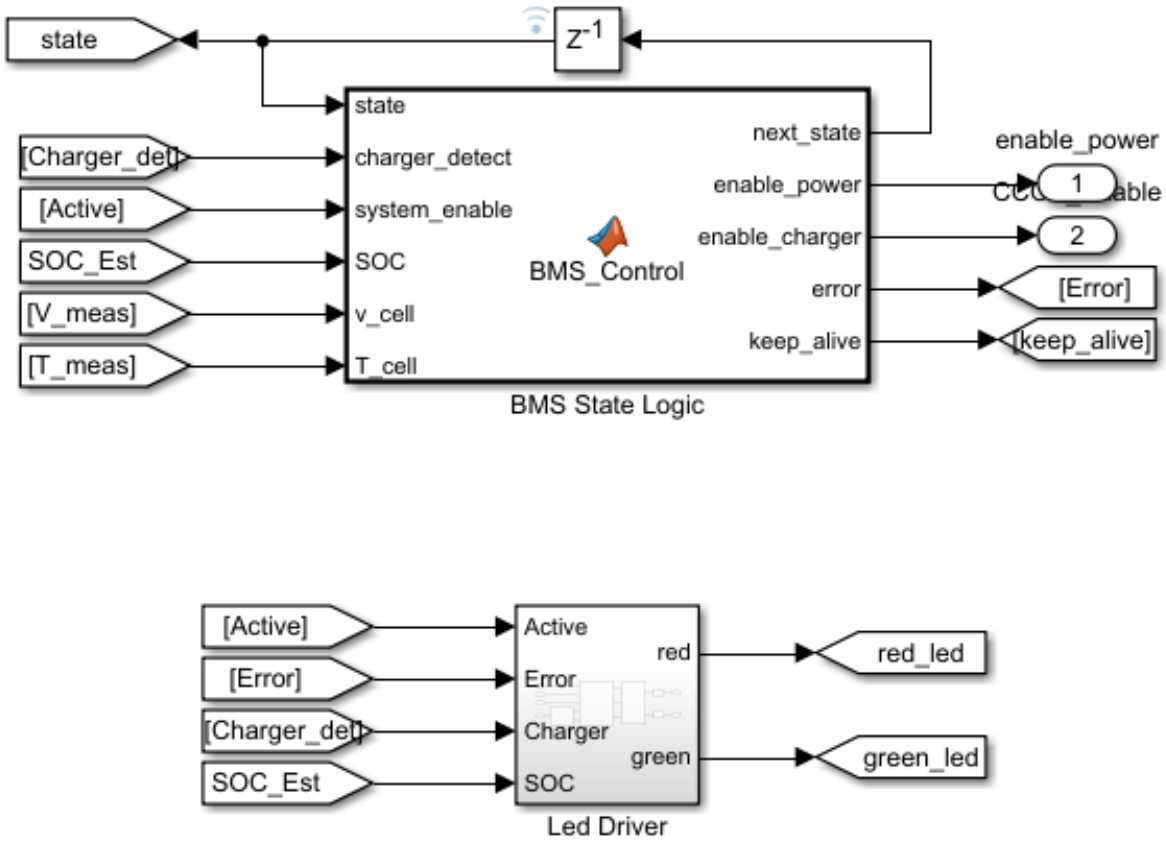


Figure F.6: Internal structure of the BMS (FSM, LED driver, Power lines)

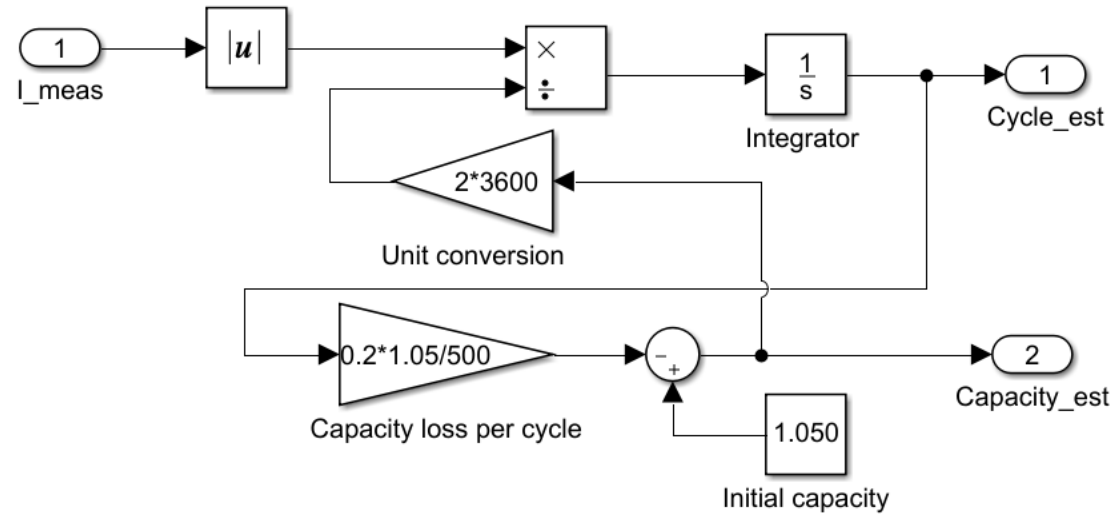


Figure F.7: Internal structure of the SOH estimation

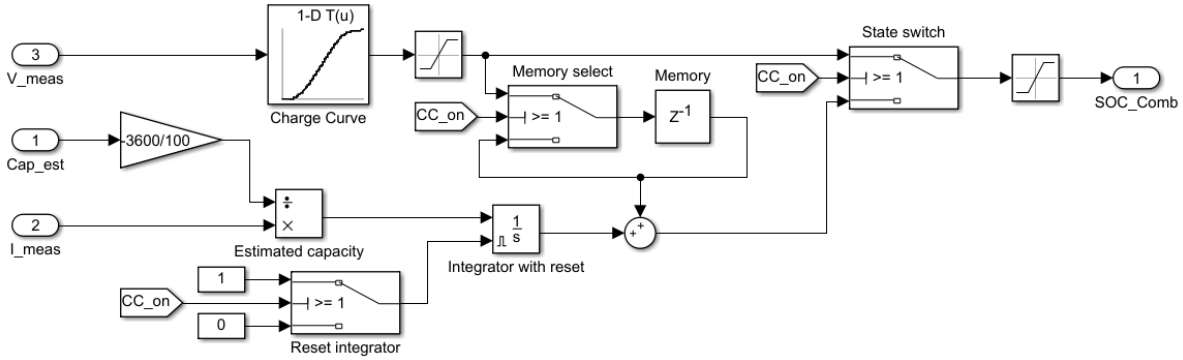


Figure F.8: Internal structure of the SOC estimation

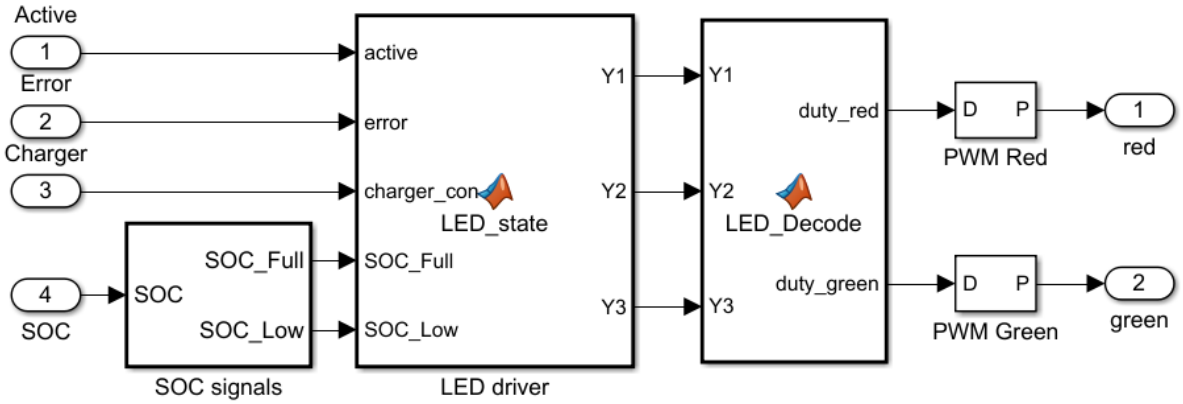


Figure F.9: Internal structure of the LED driver

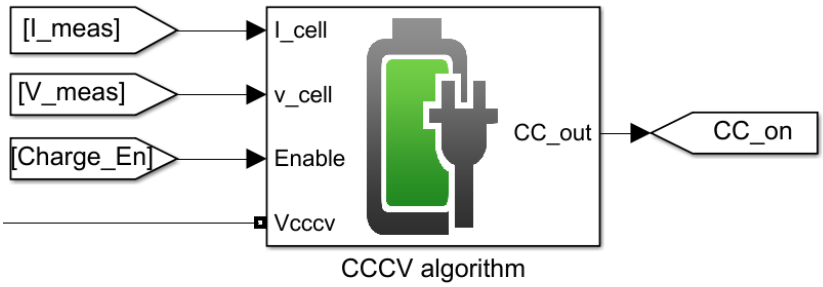


Figure F.10: Simulink block of the CCCV charger

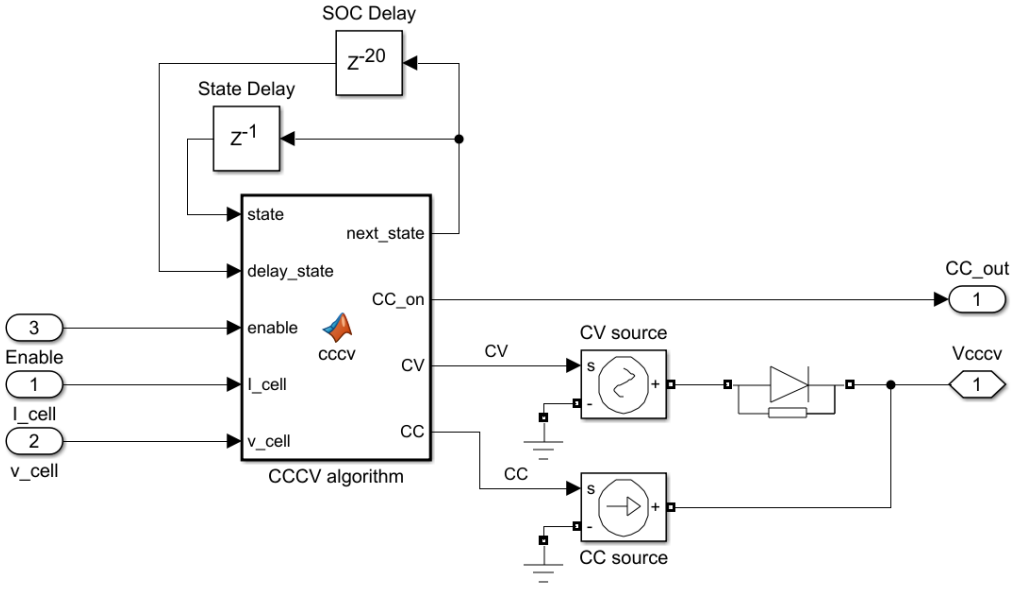


Figure F.11: Internal structure of the CCCV charger

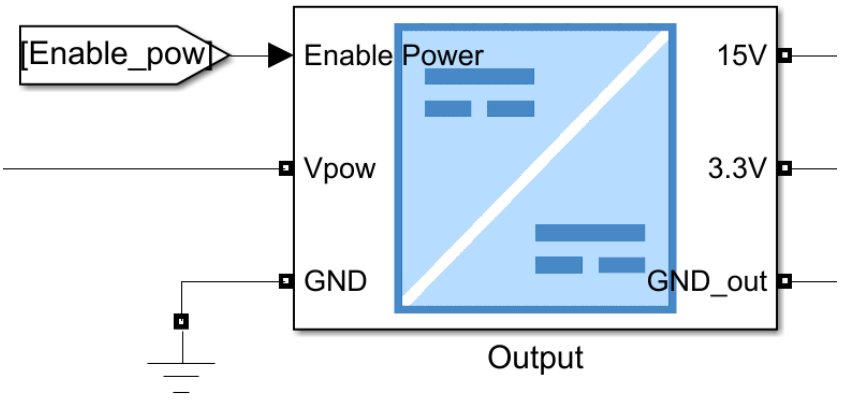


Figure F.12: Simulink block of the power output

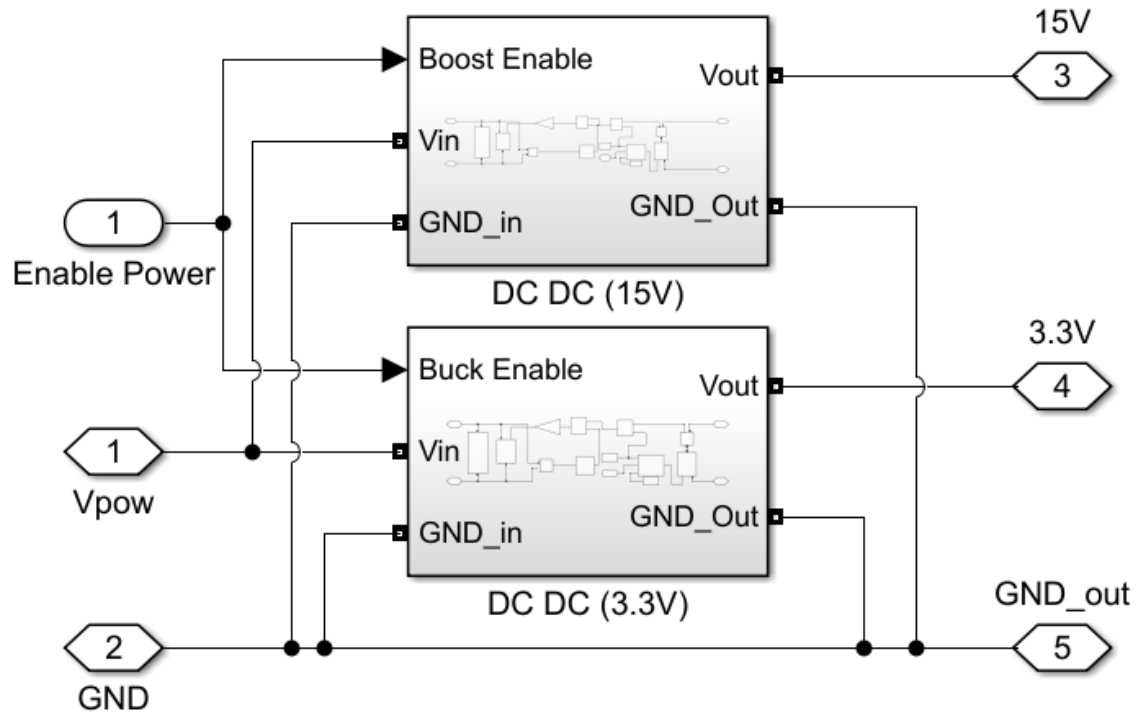


Figure F.13: Internal structure of the power output

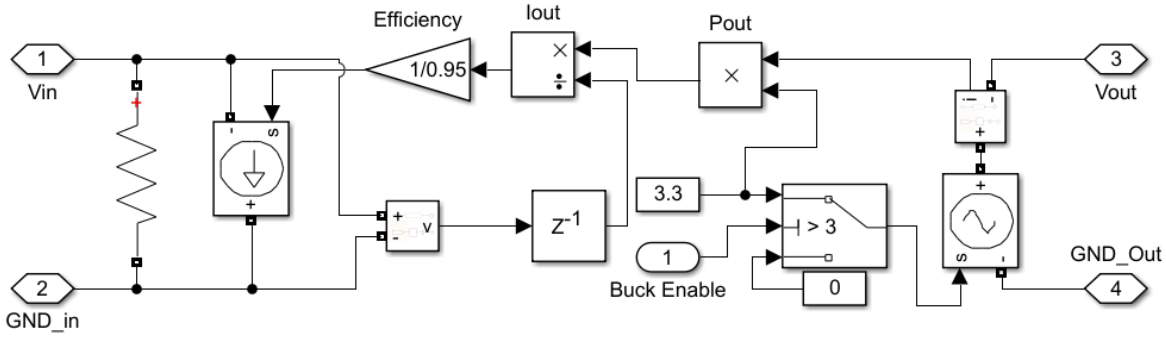


Figure F.14: Internal structure of the buck converter model

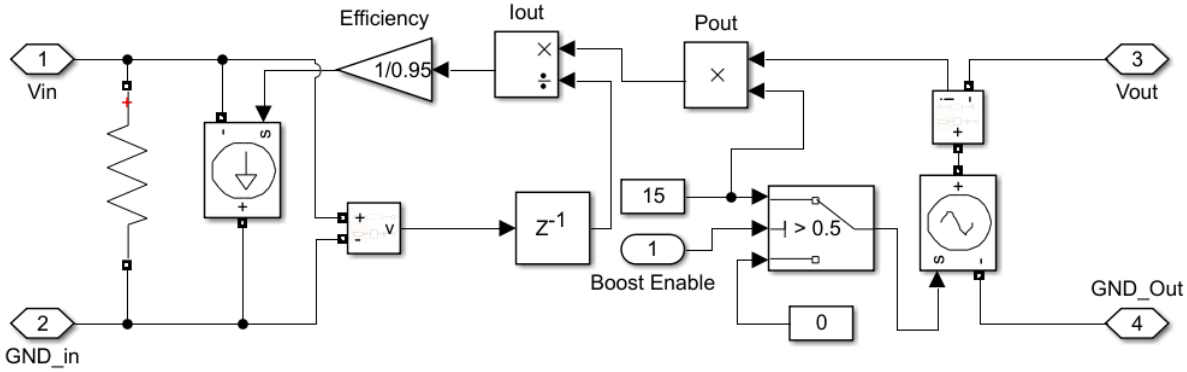


Figure F.15: Internal structure of the boost converter model

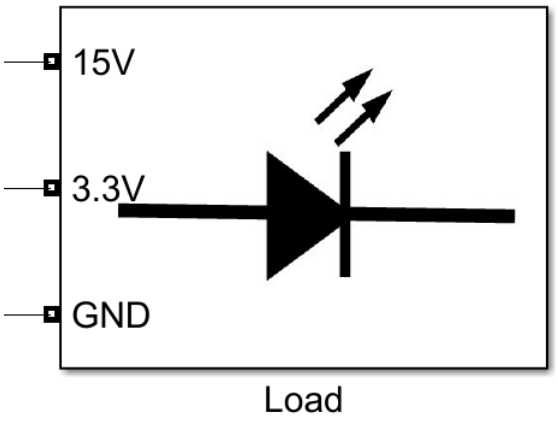


Figure F.16: Simulink block of the load

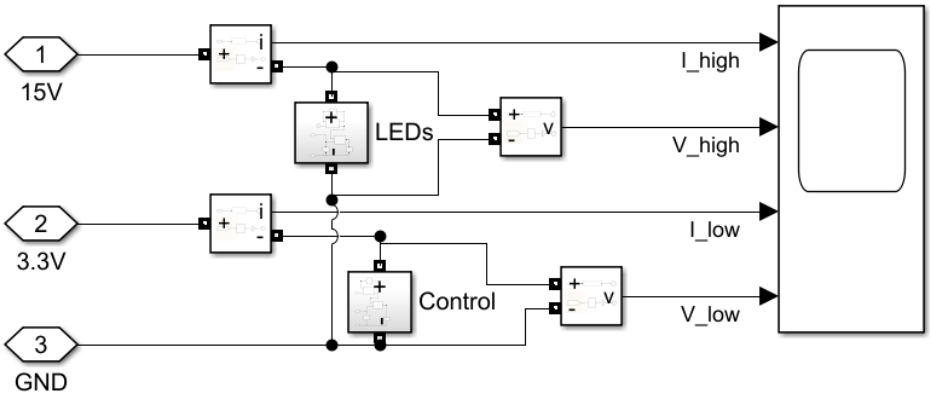


Figure F.17: Internal structure of the load

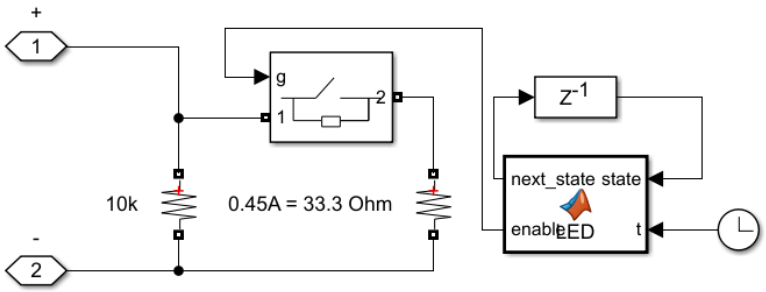


Figure F.18: Simulation of the LED load

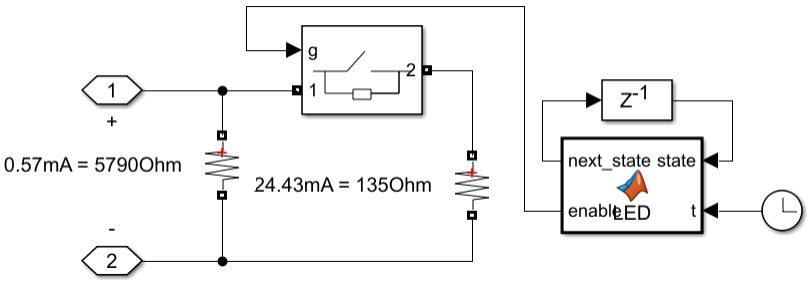


Figure F.19: Simulation of the control load

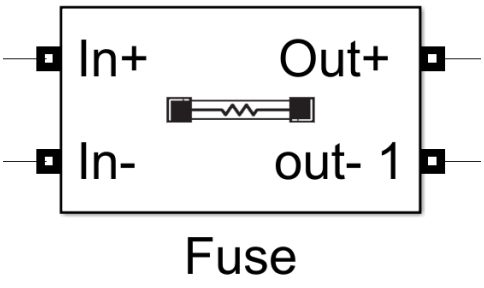


Figure F.20: Simulink block of the fuse

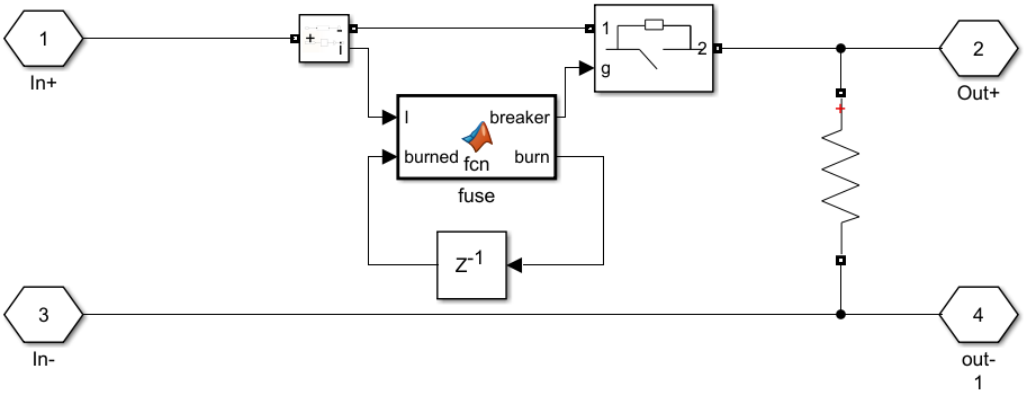


Figure F.21: Internal structure of the fuse

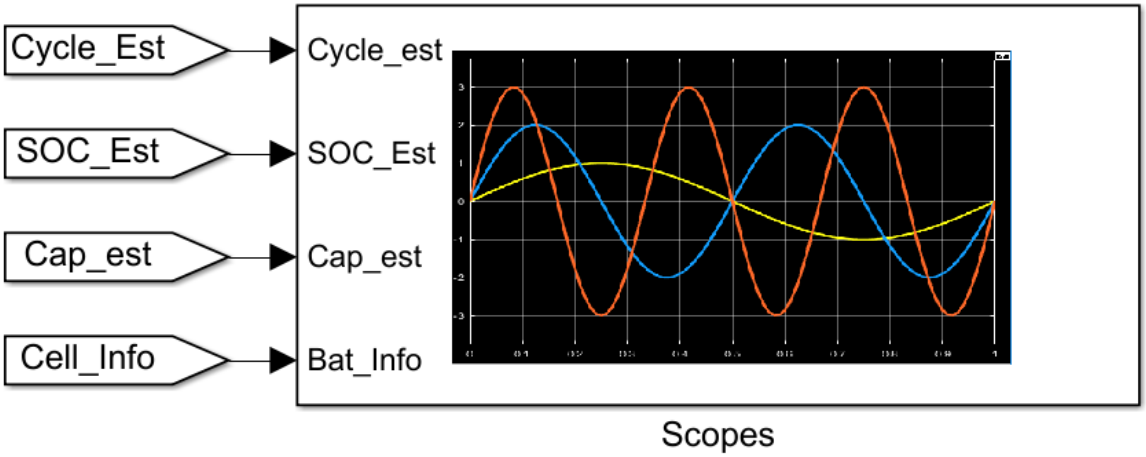


Figure F.22: Simulink block of the scopes

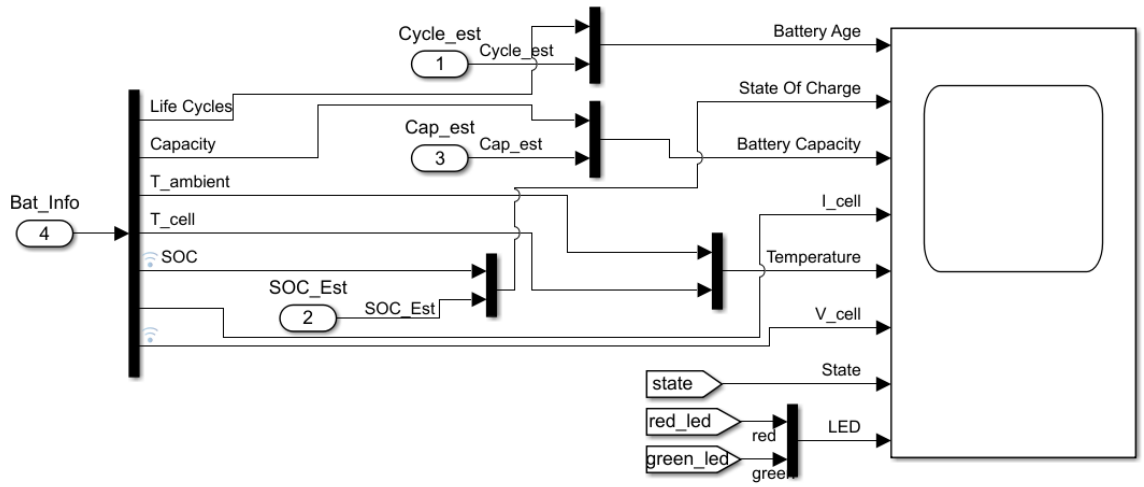


Figure F.23: Internal structure of the scopes

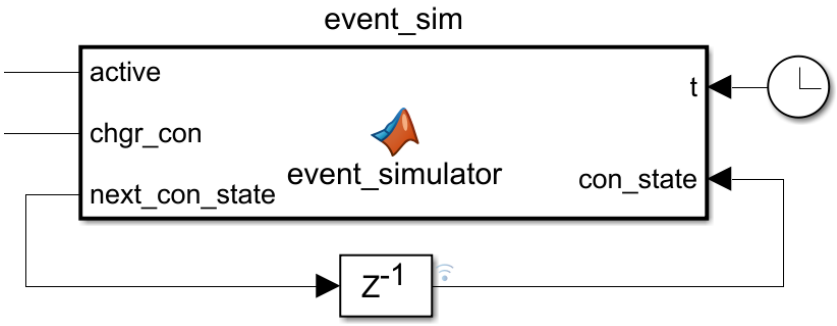


Figure F.24: Simulink block of the event simulator

G

Simulink Functions

BMS FMS

```
function [next_state, enable_power, enable_charger, error, keep_alive] =
BMS_Control(state, charger_detect, system_enable, SOC, v_cell,T_cell)
next_state = state;
SOC_max = 95;
SOC_min = 10;
v_overcharge = 4.25;
v_cutoff = 3.0;
T_max = 45;
T_min = 30;


switch state
    case 0 %Idle state
        enable_power = 0;    %Disable power output
        enable_charger = 0;  %No charging
        error = 0;           %No error
        keep_alive = 1;      %BMS operational

        if v_cell < v_cutoff
            next_state = 5;  %To shutdown
        elseif charger_detect == 1 && SOC < SOC_max
            next_state = 1;  %To charge state
        elseif charger_detect == 0 && SOC > SOC_min && system_enable == 1
            next_state = 2;  %To discharge state
        end

    case 1 %Charge state
        enable_power = 0;   %Disable power output
        enable_charger = 1; %Charging
        error = 0;          %No error
        keep_alive = 1;     %BMS operational

        if v_cell > v_overcharge
            next_state = 3; %To charge error state
        elseif T_cell > T_max
            next_state = 4; %To heat error state
        elseif charger_detect == 0
            next_state = 2; %To discharge state
        end

    case 2 %Discharge state
        enable_power = 1;   %Discharge battery on load
        enable_charger = 0; %No charging
        error = 0;          %No error
```

```
45          keep_alive = 1;     %BMS operational
46
47          if v_cell < v_cutoff
48              next_state = 5;  %To shutdown
49          elseif T_cell > T_max
50              next_state = 4; %To heat error state
51          elseif charger_detect == 1
52              next_state = 1; %To charge state
53          elseif SOC <= SOC_min || system_enable == 0
54              next_state = 0; %To idle state
55          end
56
57      case 3 %Charge error state
58          enable_power = 0;   %Battery is disconnected from load
59          enable_charger = 0; %No charging
60          error = 1;          %Error
61          keep_alive = 1;     %BMS operational
62
63          if charger_detect == 0
64              next_state = 0; %To idle state
65          end
66
67      case 4 %Heat error state
68          enable_power = 0;   %Battery is disconnected from load
69          enable_charger = 0; %No charging
70          error = 1;          %Error
71          keep_alive = 1;     %BMS operational
72
73          if T_cell < T_min
74              next_state = 0; %To idle state
75          end
76
77      case 5 %Shutdown
78          enable_power = 0;   %Battery is disconnected from load
79          enable_charger = 0; %No charging
80          error = 0;          %Error
81          keep_alive = 0;     %Shutdown BMS
82
83      otherwise
84          enable_power = 0;   %Source is connected to load
85          enable_charger = 0; %No charging
86          error = 0;          %No error
87          keep_alive = 1;     %BMS operational
88
89          next_state = 0;     %Idle default
90  end
```

CCCV FSM

```
function [next_state, CC_on, CV, CC] = cccv(state, delay_state, enable, I_cell, v_cell)

 next_state = state;

 switch state

     case 0 %OFF
         CC = 0;
         CV = 0;
         CC_on = 0;
          if enable == 1
              next_state = 4;
          end

     case 4 %SOC buffer
         CC = 1.5;
         CV = 0;
         CC_on = 0;

          if delay_state == 4
          next_state = 1;
          end

     case 1 %CC
         CC = 1.5;
         CV = 0;
         CC_on = 1;
          if enable == 0
              next_state = 0;
          elseif v_cell >= 4.199
              next_state = 2;
          end

     case 2 %CV
         CC = 0;
         CV = 5.0;
         CC_on = 0;
           if enable == 0
              next_state = 0;
          elseif I_cell > -0.075
              next_state = 3;
          end

     case 3 %Full
         CC = 0;
         CV = 0;
         CC_on = 0;
           if enable == 0
              next_state = 0;
           end

     otherwise
         CC = 0;
         CV = 0;
```

```
55          CC_on = 0;
56          next_state = 0;
57
58  end
```

Fuse

```
function [breaker, burn]= fcn(I,burned)

if I > 2.5 || burned == 1
    breaker = 0;
    burn = 1;
else
   breaker = 1;
   burn = 0;
end
```

LED Driver

G.0.1. LED state function

```
function [Y1, Y2, Y3] = LED_state(active, error, charger_con, SOC_Full, SOC_Low)

A = active;
B = error;
C = charger_con;
D = SOC_Full;
E = SOC_Low;

Y1 = A*~B*~C*~D*E;
Y2 = A*~C*~E || C*D || B*C || A*D || A*B;
Y3 = C*~D || B*C || A*B;
```

G.0.2. LED decode function

```
function [duty_red, duty_green] = LED_Decode(Y1, Y2, Y3)

if      ~Y1*~Y2*~Y3 == 1
    duty_red = 0;
    duty_green = 0;
elseif  ~Y1*~Y2*Y3 == 1
    duty_red = 0;
    duty_green = 0.5;

elseif  ~Y1*Y2*~Y3 == 1
    duty_red = 0;
    duty_green = 1;

elseif  ~Y1*Y2*Y3 == 1
    duty_red = 0.5;
    duty_green = 0;

elseif  Y1*~Y2*~Y3 == 1
    duty_red = 1;
    duty_green = 0;

else
    duty_red = 0;
    duty_green = 0;
```

```
end
```

Event Simulator

```
function [active, chgr_con, next_con_state]= event_simulator(t,con_state)

t = t/3600;

if mod(t, 2) == 0
   next_con_state = con_state + 1;
    if next_con_state >= 5
        next_con_state = 0;
    end
else
   next_con_state = con_state;
end

switch con_state
    case 0  %Charge
        chgr_con = 1;
        active = 0;
    case 1 %Discharge
        chgr_con = 0;
        active = 1;
    case 2 %Discharge
        chgr_con = 0;
        active = 1;
    case 3 %Discharge
        chgr_con = 0;
        active = 1;
    case 4 %Discharge
        chgr_con = 0;
        active = 1;
    otherwise
        chgr_con = 1;
        active = 1;
        next_con_state = 0;
end
```

Load Simulator

```
function [next_state, enable] = LED(state,t)
next_state = state;

if mod(t, 29*60) == 60*0.5
    next_state = 1;
end

if mod(t, 29*60) == 60*2.25
    next_state = 0;
end

switch state
    case 0
        enable = 0;
    case 1
        enable = 1;
    otherwise
        enable = 0;
        next_state = 0;
end
```

ATmega328P Code

Testbench

```
#include <Wire.h> //Library used for I2C communication

//Define global variables
// Input pins
const int chargerDetectPin = A0, sysEnablePin = A1, tCellPin = A2, vCellPin = A3, SOCPin = A4;
//Output pins
const int chargerEnablePin = 4, buckEnablePin = 6, boostEnablePin = 5, keepAlivePin = 2;
const int errorPin = 3, greenLedPin = 12, redLedPin = 13;

//Define BMS thresholds
int maxSOC = 95;
int minSOC = 15;
double vOvercharge = 4.25;
double vCutoff = 3.0;
int tMax = 45;
int tMin = 30;

void setup() {
  //Inputs
  pinMode(chargerDetectPin, INPUT);
  pinMode(sysEnablePin, INPUT);
  pinMode(tCellPin, INPUT);
  pinMode(vCellPin, INPUT);
  pinMode(SOCPin, INPUT);
  //Outputs
  pinMode(chargerEnablePin, OUTPUT);
  pinMode(buckEnablePin, OUTPUT);
  pinMode(boostEnablePin, OUTPUT);
  pinMode(redLedPin, OUTPUT);
  pinMode(greenLedPin, OUTPUT);
  pinMode(keepAlivePin, OUTPUT);
  pinMode(errorPin, OUTPUT);

  //Setup serial connection
  Serial.begin(9600);

}

//Define loop variables
int state, nextState = 0;       //State variables
double vCell, SOC, tCell;       //Input longs
bool  chargerDetect, sysEnable; //sensorError, Input booleans
bool  enablePower = 0, enableCharger = 0, error = 0, keepAlive = 1; //Output booleans

```

```
45 //Define variables for LED driver function
46 unsigned long previousMillis = 0;
47 const long blinkInterval = 1000; //Blink once every two seconds
48 bool blinkState = 0;
49
50 void loop() {
51
52   //Update BMS parameters
53   getBatInfo();  //Requests the SOC, cell voltage and cell temperature from the battery sensor
54   chargerDetect = digitalRead(chargerDetectPin);
55   sysEnable = digitalRead(sysEnablePin);
56
57   //Print inputs
58   Serial.print("SOC:\tV_cell:\tT_cell:\tcharge:\tenable:\n");
59   Serial.print(SOC); Serial.print("\t");
60   Serial.print(vCell); Serial.print("\t");
61   Serial.print(tCell); Serial.print("\t");
62   Serial.print(chargerDetect); Serial.print("\t");
63   Serial.print(sysEnable); Serial.print("\n\n");
64
65   state = nextState;
66
67   switch(state){
68     case 0: //Idle
69       //Signals
70       enablePower = 0;     //Disable power output
71       enableCharger = 0;   //No charging
72       error = 0;           //No error
73       keepAlive = 1;       //BMS operational
74
75       //Switch statement
76        if(vCell < vCutoff){
77          nextState = 5; //To shutdown state
78       }
79         else if(chargerDetect == 1 && SOC < maxSOC){
80          nextState = 1; //To charge state
81         }
82         else if(chargerDetect == 0 && SOC > minSOC && sysEnable == 1){
83          nextState = 2; //To discharge state
84         }
85       break;
86
87     case 1: //Charge
88       //Signals
89       enablePower = 0;     //Disable power output
90       enableCharger = 1;   //Charging
91       error = 0;           //No error
92       keepAlive = 1;       //BMS operational
93
94       //Switch statement
95         if(vCell > vOvercharge){
96          nextState = 3; //To charge error state
97         }
98         else if(tCell > tMax){
99          nextState = 4; //To heat error state
100        }
```

```
101         else if(chargerDetect == 0){
102          nextState = 2; //To discharge state
103         }
104       break;
105
106     case 2: //Discharge
107       //Signals
108       enablePower = 1;     //Enable power output
109       enableCharger = 0;   //No charging
110       error = 0;           //No error
111       keepAlive = 1;       //BMS operational
112
113       //Switch statement
114       if(vCell < vCutoff){
115        nextState = 5; //To shutdown
116         }
117         else if(tCell > tMax){
118          nextState = 4; //To heat error state
119         }
120         else if(chargerDetect == 1){
121          nextState = 1; //To charge state
122         }
123         else if(SOC <= minSOC || sysEnable == 0){
124          nextState = 0; //To idle state
125         }
126       break;
127
128     case 3: //Charge Error
129       //Signals
130       enablePower = 0;     //Disable power output
131       enableCharger = 0;   //No charging
132       error = 1;           //Error
133       keepAlive = 1;       //BMS operational
134
135       //Switch statement
136         if(chargerDetect == 0){
137          nextState = 2; //To discharge state
138         }
139       break;
140
141     case 4: //Heat Error
142       //Signals
143       enablePower = 0;     //Disable power output
144       enableCharger = 0;   //No charging
145       error = 1;           //No error
146       keepAlive = 1;       //BMS operational
147
148       //Switch statement
149         if(tCell < tMin){
150          nextState = 0; //To idle state
151         }
152       break;
153
154     case 5: //Shutdown
155       //Signals
156       enablePower = 0;     //Disable power output
```

```
157         enableCharger = 0;   //No charging
158         error = 0;           //No error
159         keepAlive = 0;       //Shutdown BMS
160
161       break;
162
163       default:
164         //Signals
165         enablePower = 0;     //Disable power output
166         enableCharger = 0;   //No charging
167         error = 0;           //No error
168         keepAlive = 1;       //BMS operational
169         nextState = 0;       //Default to idle
170       }
171
172       //Write signals to pins
173       digitalWrite(buckEnablePin, enablePower);
174       digitalWrite(boostEnablePin, enablePower);
175       digitalWrite(chargerEnablePin, enableCharger);
176       digitalWrite(errorPin, error);
177       digitalWrite(keepAlivePin, keepAlive);
178
179       setLEDs(sysEnable, error, chargerDetect, SOC);
180
181       //Delay to improve monitor readability
182       delay(200);
183   }
184
185   void setLEDs(bool A, bool B, bool C, int SOCstate){
186
187       //Declare variables to use for LED driver
188       bool D = 0, E = 0;
189       bool y1, y2, y3;
190       unsigned long currentMillis = millis();
191
192       //Set parameters for full and low battery
193       if(SOCstate > 95){
194         D = 1;
195         }
196       else if(SOCstate < 25){
197           E = 1;
198           }
199
200       //Define state of battery from truth table
201       y1 = A*!B*!C*!D*E;
202       y2 = A*!C*!E || C*D || B*C || A*D || A*B;
203       y3 = C*!D || B*C || A*B;
204
205       //Switches state to blink led when neccesary
206       if(currentMillis - previousMillis >= blinkInterval){
207         previousMillis = currentMillis;
208         blinkState = !blinkState;
209         }
210
211       //Select how to turn on LEDs
212       if(!y1*!y2*!y3 == 1){ //Off state, all leds are disabled
```

```
213     digitalWrite(redLedPin, LOW);
214     digitalWrite(greenLedPin, LOW);
215     }
216     else if(!y1*!y2*y3 == 1){ //Charging, green led is blinking
217     digitalWrite(redLedPin, LOW);           //Disable red led
218     digitalWrite(greenLedPin, blinkState);  //Blink green led
219     }
220     else if(!y1*y2*!y3 == 1){ //Full charge/device enabled
221     digitalWrite(redLedPin, LOW);           //Disable red led
222     digitalWrite(greenLedPin, HIGH);        //Enable green led
223     }
224     else if(!y1*y2*y3 == 1){ //Error signal
225     digitalWrite(redLedPin, blinkState);    //Blink red led
226     digitalWrite(greenLedPin, LOW);         //Disable green led
227     }
228     else if(y1*!y2*!y3 == 1){ //Battery level low
229     digitalWrite(redLedPin, HIGH);          //Enable red led
230     digitalWrite(greenLedPin, LOW);         //Disable green led
231     }
232     else{ //Default
233     digitalWrite(redLedPin, LOW);
234     digitalWrite(greenLedPin, LOW);
235     }
236   }
237
238 int getBatInfo(){
239
240  SOC = analogRead(SOCPin);
241  SOC = SOC/675*100;        //0 to 100
242
243  vCell = analogRead(vCellPin);
244  vCell = 2.5 + vCell/675*2;   //2.5 to 4.5
245
246  tCell = analogRead(tCellPin);
247  tCell = tCell/675*80;       //0 to 80
248 }
```

Main

```
1  #include <Wire.h>        //Library used for I2C communication
2  #include <LowPower.h>    //Library to control low power modes
3
4  //Define global variables
5  //Input pins
6  const int sensorPin = 7, chargerDetectPin = 3, sysEnablePin = 2;
7  //Output pins
8  const int chargerEnablePin = 4, buckEnablePin = 5, boostEnablePin = 6, keepAlivePin = 10;
9  const int redLedPin = 8, greenLedPin = 9;
10
11 //Define BMS thresholds
12 int maxSOC = 95;
13 int minSOC = 15;
14 double vOvercharge = 4.25;
15 double vCutoff = 3.0;
16 int tMax = 45;
17 int tMin = 30;
18
```

```
19 void setup() {
20   //Set inputs
21   pinMode(sensorPin, INPUT);
22   pinMode(chargerDetectPin, INPUT);
23   pinMode(sysEnablePin, INPUT);
24
25   //Set outputs
26   pinMode(chargerEnablePin, OUTPUT);
27   pinMode(buckEnablePin, OUTPUT);
28   pinMode(boostEnablePin, OUTPUT);
29   pinMode(keepAlivePin, OUTPUT);
30   pinMode(redLedPin, OUTPUT);
31   pinMode(greenLedPin, OUTPUT);
32
33   //Initiate I2C
34   Wire.begin();
35
36   //Initialize the battery sensor
37   sensorSetup();
38 }
39
40 //Define loop variables
41 int state, nextState = 0;       //State variables
42 long SOC, vCell, tCell;         //Input doubles
43 bool chargerDetect, sysEnable; //Input booleans
44 bool enablePower = 0, enableCharger = 0, error = 0, keepAlive = 1; //Output boolean
45
46 //Define variables for LED driver function
47 unsigned long previousMillis = 0;
48 const long blinkInterval = 1000; //Blink once every two seconds
49 bool blinkState = 0;
50
51 void loop() {
52
53   //Update BMS parameters
54   getBatInfo();  //Requests the SOC, cell voltage and cell temperature from the battery sensor
55   chargerDetect = digitalRead(chargerDetectPin);
56   sysEnable = digitalRead(sysEnablePin);
57
58   //Finite State Machine of the BMS
59   state = nextState;
60
61   switch(state){
62     case 0: //Idle
63       //Signals
64       enablePower = 0;     //Disable power output
65       enableCharger = 0;   //No charging
66       error = 0;           //No error
67       keepAlive = 1;       //BMS operational
68
69       //Switch statement
70        if(vCell < vCutoff){
71          nextState = 5; //To shutdown state
72       }
73         else if(chargerDetect == 1 && SOC < maxSOC){
74          nextState = 1; //To charge state
```

```
75          }
76          else if(chargerDetect == 0 && SOC > minSOC && sysEnable == 1){
77           nextState = 2; //To discharge state
78          }
79
80        //Enter low power mode
81        lowPowerMode();       //Wakes up on interrupt pins or every 16 seconds
82        break;
83
84      case 1: //Charge
85        //Signals
86        enablePower = 0;     //Disable power output
87        enableCharger = 1;   //Charging
88        error = 0;           //No error
89        keepAlive = 1;       //BMS operational
90
91        //Switch statement
92          if(vCell > vOvercharge){
93           nextState = 3; //To charge error state
94          }
95          else if(tCell > tMax){
96           nextState = 4; //To heat error state
97          }
98          else if(chargerDetect == 0){
99           nextState = 2; //To discharge state
100         }
101      break;
102
103     case 2: //Discharge
104       //Signals
105       enablePower = 1;     //Enable power output
106       enableCharger = 0;   //No charging
107       error = 0;           //No error
108       keepAlive = 1;       //BMS operational
109
110       //Switch statement
111       if(vCell < vCutoff){
112        nextState = 5; //To shutdown
113         }
114         else if(tCell > tMax){
115          nextState = 4; //To heat error state
116         }
117         else if(chargerDetect == 1){
118          nextState = 1; //To charge state
119         }
120         else if(SOC <= minSOC || sysEnable == 0){
121          nextState = 0; //To idle state
122         }
123       break;
124
125     case 3: //Charge Error
126       //Signals
127       enablePower = 0;     //Disable power output
128       enableCharger = 0;   //No charging
129       error = 1;           //Error
130       keepAlive = 1;       //BMS operational
```

```
131
132         //Switch statement
133           if(chargerDetect == 0){
134            nextState = 2; //To discharge state
135           }
136         break;
137
138       case 4: //Heat Error
139         //Signals
140         enablePower = 0;     //Disable power output
141         enableCharger = 0;   //No charging
142         error = 1;           //No error
143         keepAlive = 1;       //BMS operational
144
145         //Switch statement
146           if(tCell < tMin){
147            nextState = 0; //To idle state
148           }
149         break;
150
151       case 5: //Shutdown
152         //Signals
153         enablePower = 0;     //Disable power output
154         enableCharger = 0;   //No charging
155         error = 0;           //No error
156         keepAlive = 0;       //Shutdown BMS
157
158       break;
159
160       default:
161         //Signals
162         enablePower = 0;     //Disable power output
163         enableCharger = 0;   //No charging
164         error = 0;           //No error
165         keepAlive = 1;       //BMS operational
166         nextState = 0;       //Default to idle
167       }
168
169       //Write signals to pins
170       digitalWrite(buckEnablePin, enablePower);
171       digitalWrite(boostEnablePin, enablePower);
172       digitalWrite(chargerEnablePin, enableCharger);
173       digitalWrite(keepAlivePin, keepAlive);
174
175       //Set LEDs to indicate state
176       setLEDs(sysEnable, error, chargerDetect, SOC);
177   }
```

Sensor Setup

```
1   void sensorSetup(){
2
3     int sensorAddress = B1010101;
4     long dataLong;
5     byte newCSum, oldCSum, oldCapMSB, oldCapLSB;
6     byte newCapMSB = 0x04, newCapLSB = 0x1A;
7
```

```
//-Chem ID Setup-----------------------------------------------------------------------------
  //Read chem ID
  Wire.beginTransmission(sensorAddress);
  //Execute control command 0x00 and 0x01
  Wire.write(0x00);
  Wire.write(0x01);
  //Execute Chem_ID Command 0x0008
  Wire.write(0x00);
  Wire.write(0x08);
  Wire.endTransmission();

  //Request Chem ID bytes
  Wire.requestFrom(sensorAddress, 2);

  if (2 <= Wire.available()) {    // if two bytes were received
  dataLong = Wire.read();         // receive high byte (overwrites previous reading)
  dataLong = dataLong << 8;       // shift high byte to be high 8 bits
  dataLong |= Wire.read();        // receive low byte as lower 8 bits
    }

  if(dataLong != 1202){  //Set chem ID to right mode
  //Set chem ID (4.2V)
  Wire.beginTransmission(sensorAddress);
  //Execute control command 0x00 and 0x01
  Wire.write(0x00);
  Wire.write(0x01);
  //Execute chem B change command 0x0031
  Wire.write(0x00);
  Wire.write(0x31);
  Wire.endTransmission();
  }

//-Configure battery capacity ----------------------------------------------------------------
  Wire.beginTransmission(sensorAddress);
  //Execute control command 0x00 and 0x01
  Wire.write(0x00);
  Wire.write(0x01);
  //Execute SET_CFGUPDATE Command 0x0013
  Wire.write(0x00);
  Wire.write(0x13);
  //Wait for sensor to update
  delay(1000);
  //Execute block data memory control 0x61 and 0x00
  Wire.write(0x61);
  Wire.write(0x00);
  //Exexcute data block class to acces Design capacity parameter (0x3E and 0x52)
  Wire.write(0x3E);
  Wire.write(0x52);
  //Write block offset using datablock command (0x3F 0x00)
  Wire.write(0x3F);
  Wire.write(0x00);

  //Read checksum register
  Wire.write(0x60);
  Wire.endTransmission();
  Wire.requestFrom(sensorAddress, 1);
```

```
64
65    if (1 <= Wire.available()) {     // if a byte was received
66    oldCSum = Wire.read();          // receive data byte
67      }
68
69    //Read old capacity bytes
70    Wire.beginTransmission(sensorAddress);
71    Wire.write(0x4A);
72    Wire.endTransmission();
73    Wire.requestFrom(sensorAddress, 1);
74
75    if (1 <= Wire.available()) {     // if a byte was received
76    oldCapMSB = Wire.read();          // receive data byte
77      }
78
79    Wire.beginTransmission(sensorAddress);
80    Wire.write(0x4B);
81    Wire.endTransmission();
82    Wire.requestFrom(sensorAddress, 1);
83
84    if (1 <= Wire.available()) {     // if a byte was received
85    oldCapLSB = Wire.read();          // receive data byte
86      }
87
88    //Write new capcity bytes 1050 mAh = 0x041A
89    Wire.beginTransmission(sensorAddress);
90    Wire.write(0x4A);       //MSB offset
91    Wire.write(newCapMSB); //Set MSB
92    Wire.write(0x4B);       //LSB offset
93    Wire.write(newCapLSB); //Set LSB
94
95    //Compute new checksum
96    newCSum = (255- oldCSum - oldCapMSB - oldCapLSB) % 256;
97    newCSum = 255 - ((newCSum +  newCapMSB + newCapLSB) % 256);
98
99    //Write new checksum
100   Wire.write(0x60);
101   Wire.write(newCSum);
102
103   //Soft reset
104   //Execute control command 0x00 and 0x01
105   Wire.write(0x00);
106   Wire.write(0x01);
107   //Execute SOFT_RESET Command 0x0042
108   Wire.write(0x00);
109   Wire.write(0x42);
110   Wire.endTransmission();
111
112   //Wait to change state
113   delay(1000);
114
115 //-------------------------------------------------------------------------------------------------------------
116
117   }
```

Get battery information

```
int getBatInfo(){
  int sensorAddress = B1010101;

  vCell = receiveData(sensorAddress, 0x04, 0x05); //Store received data in output variable
  vCell = vCell/1000;                             //Convert to V from mV

  SOC = receiveData(sensorAddress, 0x1C, 0x1D);   //Store received data in output variable

  tCell = receiveData(sensorAddress, 0x02, 0x03); //Store received data in output variable
  tCell = tCell/10 - 273.13;                      //Convert from 0.1K to C
 }
```

Receive I2C data

```
long receiveData(int address, byte byte0, byte byte1){
  //Function variable to store received data
  long dataLong;

  //Write to selected registers
  Wire.beginTransmission(address);
  Wire.write(byte0);
  Wire.write(byte1);
  Wire.endTransmission();

  //Request voltage bytes
  Wire.requestFrom(address, 2);

  if (2 <= Wire.available()) {    // if two bytes were received
  dataLong = Wire.read();         // receive high byte (overwrites previous reading)
  dataLong = dataLong << 8;       // shift high byte to be high 8 bits
  dataLong |= Wire.read();        // receive low byte as lower 8 bits
    }

  return dataLong;
  }
```

Low Power Mode

```
void lowPowerMode(){
  attachInterrupt(0, activate, CHANGE);   //Set interrupt 0 (pin 2) to trigger when changed
  attachInterrupt(1, activate, CHANGE);   //Set interrupt 1 (pin 3) to trigger when changed

  LowPower.powerDown(SLEEP_8S, ADC_OFF, BOD_OFF); //Enter sleep mode, function exits when woken up
}
```

Activate

```
void activate(){
  detachInterrupt(0); //Removes interrupt from pin 2
  detachInterrupt(1); //Removes interrupt from pin 3
  }
```

Set indicator LEDs

```
void setLEDs(bool A, bool B, bool C, int SOCstate){

    //Declare variables to use for LED driver
    bool D = 0, E = 0;
```

```
5       bool y1, y2, y3;
6       unsigned long currentMillis = millis();
7
8       //Set parameters for full and low battery
9       if(SOCstate > 95){
10        D = 1;
11        }
12      else if(SOCstate < 25){
13          E = 1;
14          }
15
16      //Define state of battery from truth table
17      y1 = A*!B*!C*!D*E;
18      y2 = A*!C*!E || C*D || B*C || A*D || A*B;
19      y3 = C*!D || B*C || A*B;
20
21      //Switches state to blink led when neccesary
22      if(currentMillis - previousMillis >= blinkInterval){
23        previousMillis = currentMillis;
24        blinkState = !blinkState;
25        }
26
27      //Select how to turn on LEDs
28      if(!y1*!y2*!y3 == 1){ //Off state, all leds are disabled
29      digitalWrite(redLedPin, LOW);
30      digitalWrite(greenLedPin, LOW);
31      }
32      else if(!y1*!y2*y3 == 1){ //Charging, green led is blinking
33      digitalWrite(redLedPin, LOW);           //Disable red led
34      digitalWrite(greenLedPin, blinkState);  //Blink green led
35      }
36      else if(!y1*y2*!y3 == 1){ //Full charge/device enabled
37      digitalWrite(redLedPin, LOW);           //Disable red led
38      digitalWrite(greenLedPin, HIGH);        //Enable green led
39      }
40      else if(!y1*y2*y3 == 1){ //Error signal
41      digitalWrite(redLedPin, blinkState);    //Blink red led
42      digitalWrite(greenLedPin, LOW);         //Disable green led
43      }
44      else if(y1*!y2*!y3 == 1){ //Battery level low
45      digitalWrite(redLedPin, HIGH);          //Enable red led
46      digitalWrite(greenLedPin, LOW);         //Disable green led
47      }
48      else{ //Default
49      digitalWrite(redLedPin, LOW);
50      digitalWrite(greenLedPin, LOW);
51      }
52    }
```