PCBs design and production
Semester project spring 2022

Students

Baptiste Savioz: 284253

Blanche Brognart: 284193

Mathias Arnold: 299245

Supervisors

René Beuchat

Andrea Guerrieri

Robin Amacher

Veterans

Simon Dorthe

Valentin Karam

Swiss Solar Boat

EPFL

Swiss Solar Boat

EPFL

Spring 2022

Contents

Abbreviation

BMS	Battery Management System
CAN	Controller Area Network
CUS	Carbon Upper structure
DB	Distribution Board
IDE	Integrated development environment
IPA	Isopropyl alcohol
MCU	Microcontroller, in this report, it will specifically design the microcontroller as an integrated circuit.
MPPT	Maximum Power Point Tracking
SMD	surface mounted device
SOF	Start of frame
SOP	Single point of failure
SSB	Swiss Solar Boat
SSR	Solid state relay
PVT	Process, Voltage, Temperature

1 General Introduction

Swiss Solar Boat is an interdisciplinary association of the EPFL whose objective is to build and optimize its solar boat: the famous Papy Prao. This asymmetrical boat inspired by Polynesian pirogues is covered with solar panels and sails thanks to the sun or even flies over the water thanks to its foils.
The association is part of the MAKE project at EPFL and therefore aims to bring multidisciplinary skills with a practical approach, and to develop transversal knowledge. The objective for this year 2022 is simple: to win the Monaco Energy Boat Challenge. To achieve this, we need to optimise our boat, in particular with a new battery. And a new battery means new electronics to make it work. This is where this project gets its ambition: to design and produce the PCBs for the V2 battery. There will be three PCBs to produce: The relay block which controls the power supply to the various systems on the boat. The monitoring block which controls the fans to cool the battery and collects valuable information. Finally, there is the new microcontroller that handles communication with the rest of the boat. The new microcontroller is used for more than just the battery. It is intended to progressively replace all those present on the boat. It will therefore be the subject of a separate section in this report. To realize these PCBs, Swiss Solar Boat can count on a team of three students in credited projects: Baptiste Savioz, Blanche Brognart and Mathias Arnold.
Baptiste is a student in MA2 in robotics with a bachelor's degree in microengineering. As a semester project, he will design the new microcontroller. Blanche is a student finishing her master's degree in energy and who also did a bachelor's degree in microengineering. For her semester project, she is in charge of the new monitoring block. Furthermore, she will also be the second pilot of the Papy Prao. Finally, Mathias is a bachelor student in electrical and electronic engineering. His Bachelor project is to design the new relay block and to make the configuration for the new BMS.

A project in an association like SSB is not limited to this simple project and we are quickly overtaken by the responsibilities. It is an exciting job that involves us in a human way and not only for our technical skills. It is not possible to summarise all that we have done and learned in this report, nor is it the purpose of the latter. The long nights spent debugging the boat, preparing to test the battery. The countless kilometres of cable we saw pass by, to which we sometimes modified a connector, changed a sleeve or even completely replaced. The rules of the competition that we read and reread until we hurt our eyes to make sure that everything is in order. The countless coffees we drank at the local, rarely paying for them. The books we filled out with schematics and drawing to coordinate ourself between the different systems. All the helping hands given to colleagues out of passion for what we do. All this will go unnoticed in this report.

2 Microcontroller project

2.1 Introduction to the microcontroller project

The EPFL association Swiss Solar Boat is building a performance solar boat to participate in the MEB contest in Monaco. This boat is made up of an important part of electronics, vital to its good functioning. The electronics have been divided into several modules, depending on what they do and where they are located in the boat. In order to control these different modules and allow them to communicate with each other and with the user, we use a microcontroller: a kind of small embedded computer. The goal of this semester project is to design and build a new custom microcontroller board for the Swiss Solar Boat association to replace the version currently used on the boat.
Before going further in the explanations, it is time to talk about the structure of this sub-report in order to facilitate the comprehension and the research of information. The report is structured as follows:

Title	**Content**
Introduction	Context and motivation to the project
Specifications	Descriptions of all project requirements: temporal, functional, performance
Methodology	Descriptions of the working method, resources used and associated objectives
Solution analysis	openings on the different options and justifications for the design choices
Documentation	List of all technical details to use the microcontroller correctly

Now that this is done, we can answer the first question:

Why do we need a new custom microcontroller ?

First, some background. This project takes place in the spring of 2022, a few months after the start of the silicon crisis which began in October 2021. The price of silicon has jumped by 300%, due to several factors that are not the subject of this report. However, it has had many consequences for the electronics market. Many components are no longer available, delivery times are soaring, and some orders are being restricted by resellers. The consequences are direct for Swiss Solar Boat: the microcontroller used until now is no longer available and the stock of spare parts is gone. The slightest breakage or failure may ground the boat. It is therefore necessary to develop a new board with available parts, and why not make some improvements.

What are the goal and objectives of this project

Now that we understand what are the motivations to start this microcontroller project. Lets define its great lines. First of all, the communication between the modules of the boat is done through the CAN line, so this will be the essential part of the microcontroller. For this purpose a brand new MCU has been selected and ordered in fall 2021 during the semester project of Richard Ismer, that I will talk later on. It is the ATSAME51. It features two dedicated CAN line (one more than in the previous version of the custom microcontroller) as well as greater computational power, more GPIO and communication lines. The reason such a powerful microcontroller has been selected is because limitations of the previous one had been reached and it started to limit integration of new sensors for example. Therefore, this enhanced version had been dreamed to be more modular and adaptive thus, it will feature a shield extension. Moreover, during richard's project : *Remote update for Arduino applications on SAM board with bootloader through CAN bus*, a bootloader to reprogram this microcontroller through the CAN bus had been developped. It was based on the dev board *Feather M4 CAN* from *adafruit* which features the same microcontroller.
The clear objectives of this project are then : to design a PCB for the new microcontroller and then to produce and assemble a sufficient amount of them in order to replace the old version or to be use for new project at SSB.

What is really a microcontroller and what is its history in SSB ?

Firstly, the term microcontroller is used in this report in several senses. On the one hand, it is used literally to refer to the microcontroller, the integrated circuit. On the other hand, in a broader sense to designate the board on which it is mounted. Within the association, the term *Arduino* is often used to refer to custom microcontrollers. However, this will not be used in this report, as it is incorrect and misleading.
Then, a microcontroller is an integrated circuit with all the units needed to run a program, such as a processor and a non-volatile memory. In this sense it is similar to a microprocessor, as found in our computers. What distinguishes it, however, is that it also contains an input/output interface, allowing it to be integrated into an embedded system and to communicate with all necessary peripherals. To illustrate its operation by an example, a microcontroller could be the thinking unit of an actuation system. Its non-volatile memory allows it to store a program that will run as long as the board is powered. Thanks to its peripherals, it can communicate with other elements of the actuation system such as sensors or motors, make measurements and store them in its volatile memory. Furthermore, it can also receive instructions from other systems, again through its peripherals, such as the control system. Finally, thanks to the measurements, and the instructions received, it can process all this in an intelligent way thanks to its computer program in order to control the motors.
Custom microcontrollers have been on the boat since the beginning and several previous version have already been developed. To recap:

Version	Authors	MCU	date
1.1	Henry Papadatos	ATSAMD21	?
1.2	Valentin Karam	ATSAMD21	?
1.3	Yoann Lapijover	ATSAMD21	?
2.1	Baptiste Savioz	ATSAME51	March 2022
2.2	Baptiste Savioz	ATSAME51	May 2022

Table 1: History of the different version of the custom microcontroller board at SSB

Where is the microcontroller used on the boat?

The boat uses several different microcontrollers, not all of which are covered in this report, for example it also has a raspberry Pi or an Arduino MKR Zero. However, the microcontroller we are interested in is used in :

- The battery
- The cockpit
- The CUS
- The Palpeur
- sooner for the management of the new sensors.

2.2 Specifications

First of all, this project did not start from scratch, so there are certain constraints to be respected, on the one hand to allow a certain retrocompatibility (type of connectors, standards), and on the other hand, in the context of the silicon crisis, to use components already ordered and in stock at SSB. These constraints occasionally conflict with performance or good design practices, sometimes to the great displeasure of my supervisor[1]. It was therefore necessary to mention them as a preamble.

During the winter holidays of 2021-2022, between the two semesters, Simon Dorthe and I met to define the project more specifically. We started by drawing up the specifications, which I felt was essential before we could

[1]During the whole microcontroller part, personal pronoun will refer to Baptiste Savioz

start the design. The great line of the specifications will first be mentioned in a continuous text in order to articulate them and make meaningful link. Afterwards, they will then clearly be stated in a table in a way we can assess them more properly.

2.2.1 Project specifications

The specification starts by defining the functionality of the microcontroller. It must be able to be installed on the boat with the least possible integration effort and guarantee at least as much functionality as the version 1.3 it replaces. Thus, it must be powered by the 24[V] for which a buck converter and an LDO had already been chosen. Furthermore, the microcontroller board is based on a new MCU, which has already been chosen too and which integrates two CAN lines that must be accessible on the PCB with the standard connectors at SSB. In addition to these two CAN lines, the microcontroller must have an I2C line, an SD card port and a USB line to facilitate programming and debugging. Furthermore, because of the many problems on the CAN line and the noisy environment, the design must be as robust as possible. Finally, the PCB footprint should be limited and there should be the possibility to mount a shield on it to extend its functionality.

In view of the overall integration of the project within SSB, there are also some more general specifications. For example, Altium is to be used to design the PCB. This allows on the one hand to unify the different practices, and on the other hand allows other SSB members to easily consult/edit/modify the designs. In the same way to facilitate the integration of the project in SSB and to allow the largest number of people to use it, the microcontroller should be programmable with the Arduino environment. Large amount of code written for the previous version of the microcontroller could then be re-used.

Board requirements

Functionality	**Assessment**	**Flexibility**
The microcontroller must be powered with 24[V]	The 24[V] must come through a S04B-PASK-2(LF)(SN) connector with the CAN line	mandatory
The microcontroller should provide 5[V] for stability of the 3.3[V] supply as weel as for other application on the shield.	A buck converter TPS54233 should be used to convert the 24[V] into 5[V] and be able to provide stable 5[V] up to 1[A]	mandatory but there is some margin on the amount of power
A stable 3.3[V] power supply should power the MCU as well as other IC on the board or shield.	The 3.3[V] comes from the LDO AP7365 and must supply 500[mA]	mandatory but there is some margin on the amount of power
Two dedicated CAN line must be accessible on the board	The CAN transciever must be the TCAN337 and the connector is a S04B-PASK-2(LF)(SN)	One line is mandatory but there is some flexibility on the second one.
One I2C line must be accessible on the board for communication with peripherals and a second one available on the pinout of the shield.	The connector must be S04B-PASK-2(LF)(SN) . It should provide as well 3.3[V] (max 100[mA]) and GND.	One line is mandatory but there is some flexibility on the second one.
A SD port should be accesible to save some permanent data, as for example a logfile.	Standard microSD port	not mandatory
A USB line must be provided in order to have easier user interaction with the board	A micro USB connector with USB2.0	Important to some extend during the developing phase
The board must accept a shield in order to extend its capabilities	Standard female pin headers connectors (two times 1x25 with 2.54mm pitch) with 24[V], 5[V], 3.3[V] and GND accessible.	Mandatory but their is flexibility in the arrangement and type of connector.
The PCB should be made as small as possible	max 70mm x 70mm	Important to ensure integration on the boat
Reasonable price	arround CHF 50	No price constraint except that it must be reasonable
Production of the required PCB and spare.	10 fully assemble PCBs	5 fully working PCBs are mandatory at week 10, there is margin on the rest according to available supply

Table 2: Specification of the microcontroller

2.2.2 Timeline

In order to carry out the project correctly, the following timeline was drawn up when we defined the project. It take place during the sping semester 2022.

week 1	Integration into the associaton and getting started into the project
week 2	Find key issues
week 3	Solve key issues, start with design choices, draw the schematic
week 4	Complete drawing of the PCB and first feasible version
week 5	PCB fabrication and delivery
week 6	Assembly of V2.1
week 7	Test of V2.1
week 8	Learn from these experiments and set up a list of correction/improvement
week 8+	Draw corrected schematic and optimize routing, second feasible version
week 9	PCB fabrication and delivery
week 10	Assembly of V2.2
week 11	Integration
week 12	Integration
week 13	Integration
week 14	Integration

Table 3: Timeline for the microcontroller semester project

2.3 Methodology

In this part I will present the methodology I used throughout my semester project, as well as the different objectives that evolved as I progressed.

Two iteration for the microcontroller board

From the beginning it was clear that we would make two iteration of the microcontroller board, the first to ensure that the second would be flawless. Indeed, this new microcontroller board would have to be designed from scratch and couldn't benefit much from previous related project, since we have to use a new MCU and new components. It is a quite complex design for a student with low experience in the field, therefore two iterations seemed to be a bare minimum for a successful project. All physical implementation on the boat was to be completed by the end of week 10. So we had an extremely tight deadline to make these two iterations. I had to start quickly, and I couldn't afford to waste too much time on the design at the beginning. So I made a priority list for the first version, which hasn't the same level of requirements as the second.

The first version had to be a working prototype, available as soon as possible in order to test the design, the components and to eliminate as many errors as possible. Ideally, three weeks after the start I should have a first version that could be ordered. For this, I decided not to waste time with the optimization of the routing, nor to refine my schematics which would be read only by me. Similarly, the layout would be rudimentary but functional. However, none of this is true for the second version which is a final version. I have therefore taken all the necessary care to do each of the above details as it should be done.

Design procedure

Throughout this project, I always started from the specifications to write the technical documentation of the project, i.e. how things should work. Only once the technical documentation is written, I could start my design with clear and already fixed objectives. This avoids going back and forth between computerized PCB design tools, pencil and paper design of the various circuits, datasheets of the components, and taking advice from the various supervisors. Instead, the design process is smoother and more organised. One starts by reading the datasheets and following the design guidelines to write down the documentation. Then one can have the different choices evaluated by the supervisors and make the necessary corrections before proceeding to actual design with all the appropriate tools.
The more or less complete documentation was available since week 3. Due to this I was able to have a much

more clear view on the design choices. Moreover, the functionality of each element was stated, which made the testing part easier since, I know exactly what everything should do. Iterate from there was also simpler. If a mistake was spotted, I could start again by redefining what the element should do and how it should do it, then I could go to some supervisor to ensure that it was feasible and *wise*. Finally, I could implement the changes.

Git

In order to keep a clear vision of the progress of the project and to manage my files efficiently, I immediately decided to use Git. So I created a folder (generic-mcu-pcb) on the GitLab of Swiss Solar Boat. On the other hand, a great advantage of using Git is that I can have a nicely formatted documentation in the same place as the files needed for the project. Each person wishing to use this project will have to go through the GitLab folder and will see the documentation. Moreover, on Git, it is not fixed as in a report written at a date X, but it is dynamic and can be easily corrected or modified over time and the evolution of the project. Moreover, related projects can be added to the Git folder and update the documentation, as for example the cockpit shield which has recently been added.

Altium

As for the design of the PCB itself, several options were considered within SSB, but in the end Altium was chosen for the design of all future PCBs. So I had no choice but to train myself to use Altium. It's a good tool but very complex to use, which was not easy at the beginning of the project when time was short.

2.4 Design and Solution analysis

2.4.1 Power supply

The board is powered with a 24[V] DC line that goes all along the boat. It comes along with the CAN line of the boat and thus is mounted on a single 4 pole connector. The actual norm at SSB for CAN connection with 24[V] low power is to use a S04B-PASK-2(LF)(SN) with a dedicated color code and I decided to stick to it (section 2.4.3, figure 12). The power can be quite noisy and therefore we choose to gradually decreased the voltage from 24[V] to 5[V] through a high efficiency buck converter and then from 5[V] to 3.3[V] with a LDO IC to ensure stable and clean power.

Ground source

The first problem to solve when designing the power line is that of grounds. Indeed, it is necessary to guarantee that on the one hand, the ground of the card is identical on all the card, but especially that it is identical to those peripherals to which it is connected. Actually, between the USB line, the two I2C lines, and the two CAN lines, plus the possible connections through the shield, the different ground sources are numerous. For the grounds through the shield, it will be the designer of the shield to be careful. As for the I2C lines, it will also be necessary to be careful, but it can be assumed that the microcontroller will be used for applications as a master and will provide the power supply to the slave itself. So there shouldn't be any problems, neither with different grounds nor with ground loops. USB on the other hand is more problematic. A common ground between the devices is necessary and it is difficult to guarantee that this will be respected. For example, if the microcontroller is mounted on the boat and powered by the boat's 24[V]. A user then plugs into the USB port with their computer charging from the grid. There is no guarantee that the source of the grounds are identical. I have not been able to develop a satisfying solution to avoid this problem and it is up to the user to be careful. Finally, there is still the ground of the CAN lines. This is where the PCB gets its power and therefore its ground. I would like to remind you that the CAN protocol is formally defined as a 2-wire protocol. It does not require a common ground. The voltage difference between the grounds of the two systems is however

limited by the protcol (CANOPEN) to [-2, +7] Volt. It should be noted, however, that a typical CAN tranciever tolerates ±12[V] of difference between the grounds of the different systems connected, which is also the case for the TCAN337 CAN tranciever we use. Although a ground offset is tolerated, it is nevertheless problematic, and it is therefore recommended to ensure a common ground between the systems. Let's close the case and look at what this means for our application. In order to guarantee the proper functioning of the CAN lines of the boat, it is important that all systems with different power supplies connected to the CAN line have common grounds. This is the case at the time of writing this report. But is it up to the microcontroller to connect these grounds together? The answer I give is no. The different systems (High power, Low power 12V and 24V or PV) are already connected together via the negative terminal of the V2 battery. If we were to connect these grounds together through the microcontroller, we would fear a ground loop, which would inevitably create other problems. For these reasons I decided not to connect the grounds of the two CAN lines of the microcontroller together, instead I used a double pole double throw switch to choose where to take the power between the two CAN lines.

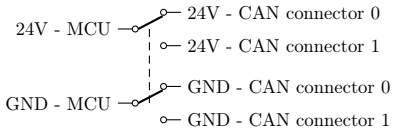


Figure 1: Double pole double Throw switch to select the power supply source of the microcontroller

This solution may seem a bit crude, but since the applications of this microcontroller are not yet fully defined and the electronics on the boat are often subject to change, it could not be guaranteed that there would never be an unfavourable situation. This solution has the advantage of avoiding breakages and accidents due to poor power supply management.

Buck converter and LDO

Now that we have solved the problem of the source of our power supply, we can move on to sizing it. The TPS54233QDRQ1 buck converter was chosen and ordered in the fall of 2021 and was therefore intended for. To design the electronic circuit of the buck converter, I used the tools provided by the manufacturer, namely *Webench* from *Texas Instrument*. After a little bit of adjustment on the Webench, especially because of the availability of some components, the following solution was found:

V_{in}	20 - 28	V
V_{out}	5	V
$I_{out,max}$	2	A

Table 4: Buck converter capability

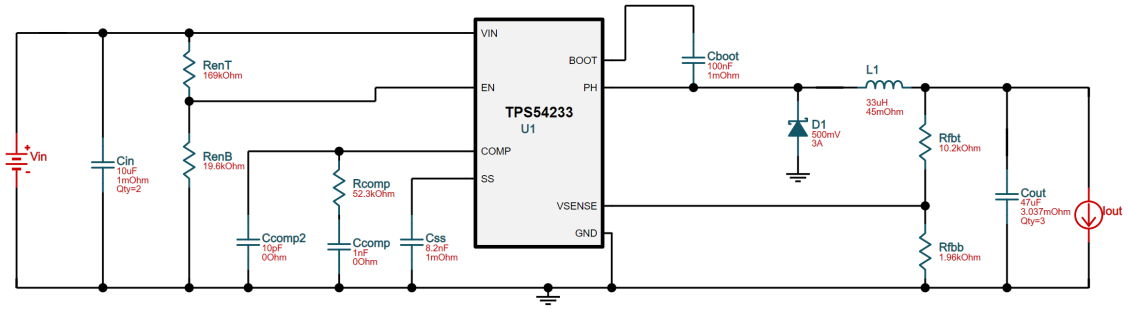
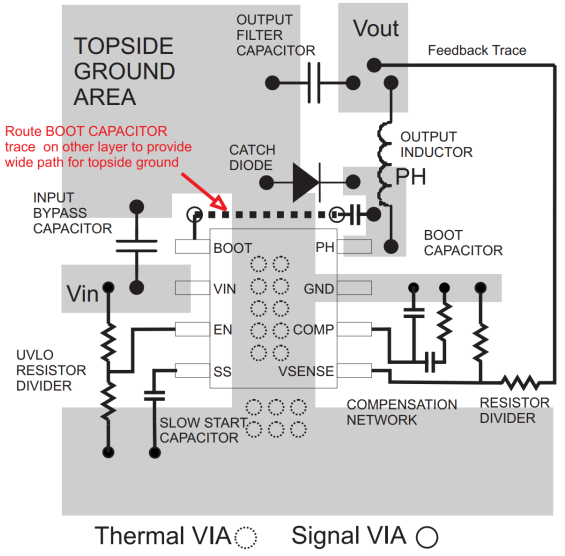
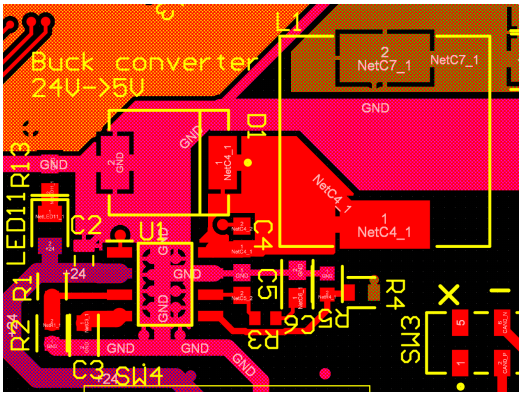


Figure 2: Buck converter circuit

I still had to outline this circuit on the PCB. Once again, I tried to respect the recommendations of the documentation in order to guarantee sufficient heat dissipation by having enough via dissipation.



(a) recommended Layout[1]



(b) V2.2 Layout

Figure 3: Buck converter layout

in order to protect the buck from possible power differences, for example with a slightly different 5[V] from the USB port, a Schottky diode has been added between the 5[V] output of the buck and the input of the LDO.

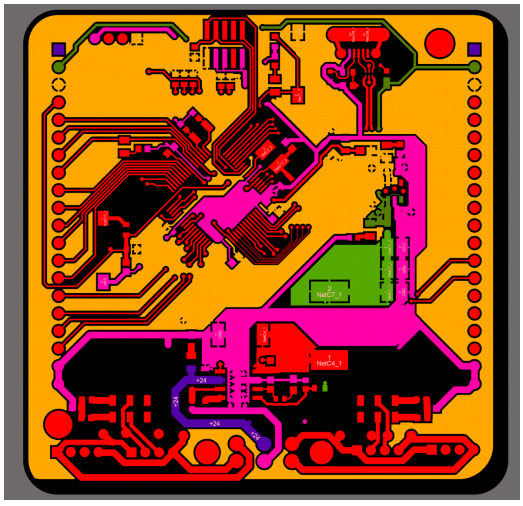
Once the buck designer was done, all that was left to do was to make the circuit for the LDO which reduces the voltage from 5[V] to 3.3[V]. The LDO AP7365 was already selected and ordered in fall 2021. I used that one for the PCB I produced however they are not recommended for new design and they will shortly fall out of supply. The newer version is the LDO AP7366 and is exactly equivalent in term of package and capability. Unfortunately I couldn't already use them since they weren't available at the moment of design but I strongly recommend to change to them, if new batches of PCBs have to be produced in the future. The circuit is very simple and I just followed the recommendations of the datasheet. In addition, I made sure that there was enough decoupling capacity all along the power line to the microcontroller. Indeed, it will have to operate in a very noisy environment.

V_{in}	2 - 6	V
V_{out}	3.3	V
$I_{out,max}$	600	mA

Table 5: LDO capability

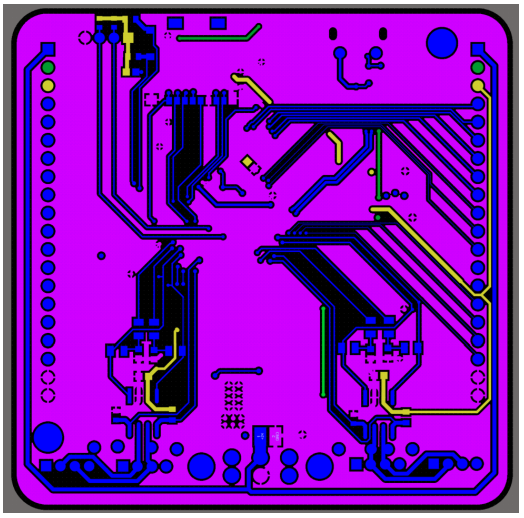
Power and Ground plane

For a robust PCB design, it is recommended to make a ground plane and a power plane in order to effectively protect the rest of the board from electromagnetic interference. For reasons of simplicity and time constraints, the first iteration was a 2-layer PCB without ground and power plane. As a general rule, a ground plane is made from a PCB with 2 or more layers and a power plane starting from 4 layers. The board we are producing is not too loaded for its dimensions (70x70mm) and the gain over the complexity of a 4 layer PCB did not seem huge. So my supervisor and I decided to stay with a two layer PCB for the second iteration as well, but this time with a ground plane for the bottom layer, and a somewhat original power plane for the top layer as it is not really isolated from the majority of the components and routes.



(a) Power plane, top face

Pink : GND
Orange : 3.3V
Green : 5V
Purple : GND



(a) Ground Plane, bottom face

Figure 5: Power and Ground planes of Microcontroller V2.2

2.4.2 Microcontroller

The first step in designing the circuitry accompanying the ATSAME51J20A-AF microcontroller IC is again to start with the power supply. The MCU has three power supplies, VDDIO for port A and internal systems, VDDIOB for port B, and VDDANA for the analog pins, ADCs and DAC. VDDCORE and VSW are used for the internal voltage regulator (LDO/Buck) for certain modes of operation. Moreover VDDBU, VSW and VSWOUT are internal power domain only.

The three power supplies are not independent and must share a common ground. The documentation provides a recommended circuit that I followed. I also added a ferrite bead to filter high frequency disturbances as recommended for the design in noisy environments. For the layout on the PCB, it's complicated to present it clearly because of all the tangles near the MCU, but I put each time the decoupling capacitors as close as possible trying to build a power line that follows a single flow. For the ground I adopted a star configuration, to avoid loops.

Once the power supply was connected to the microcontroller, we had to think about how to program it. The MCU only has the Serial Wire Debug (SWD) protocol to be programmed. This protocol shares the same

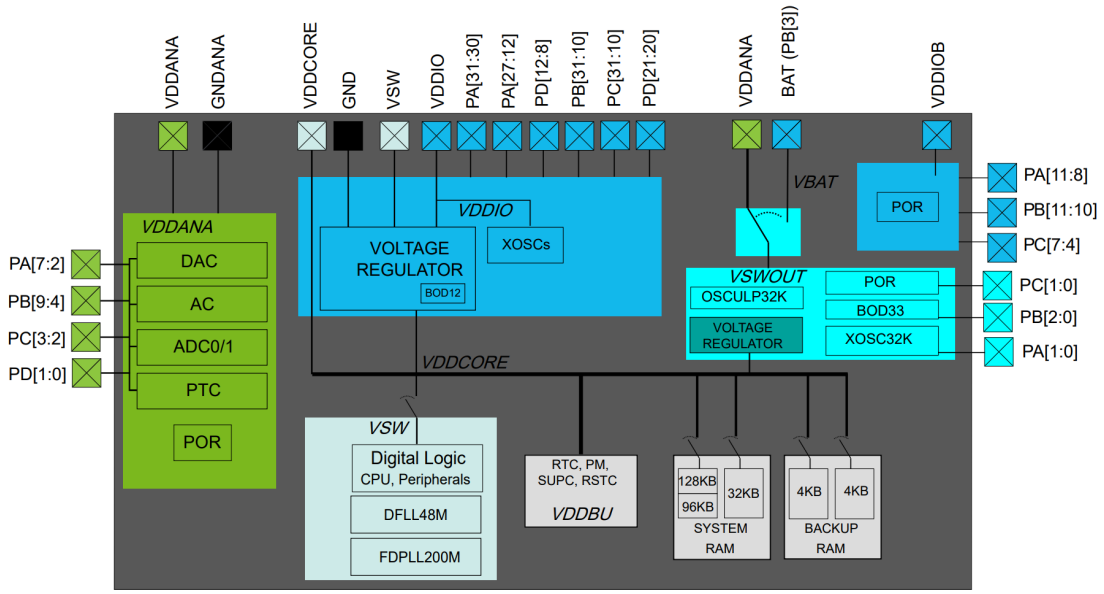
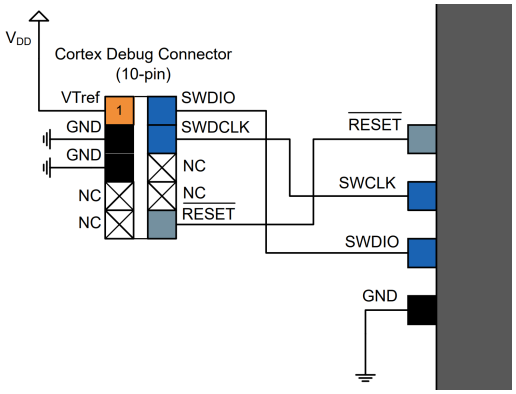


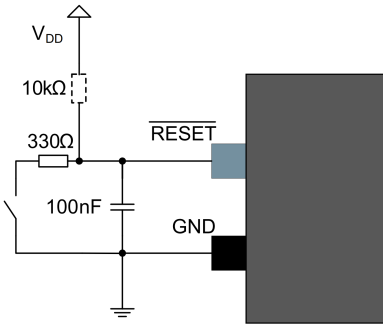
Figure 6: ATSAME5x power domain[2, p. 47]

connector as the JTAG (and they are often confused), a 10 position (5x2) connector with a pitch of 1.27[mm]. I chose the appropriate connector and made the circuit according to the recommendations (figure 8a). However, I made a mistake in the wiring during the first iteration which was corrected in the second. Furthermore, for a robust design, it is recommended to add a pull-up resistor of 1[kΩ] to the SWCLK pin, which was done.

In addition, it was necessary to make the circuit for the reset pin. It is a simple RC filter with a time constant of 10[ms]. Specifically, the ATSAME5x recommends a discharge resistance of 330[Ω]. A push button is mounted for manual reset. From here the MCU should be functional and we can move on to configuring/implementing additional features.



(a) SWD connections [2, p. 2107]



(b) Reset switch circuit [2, p. 2102]

Figure 8

The MCU has several clocking options and can admit two external oscillators (32.768[kHz] for real time counting applications RTC and a higher frequency oscillator). J'ai décidé de monter qu'un crystal horlogoer pour les applications en temps réels. Le choix du crystal et le dimensionnement du circuit est assez difficile, car il y a beaucoup de détails à faire attention. Pour ce faire j'ai choisi un crystal avec une capacité très

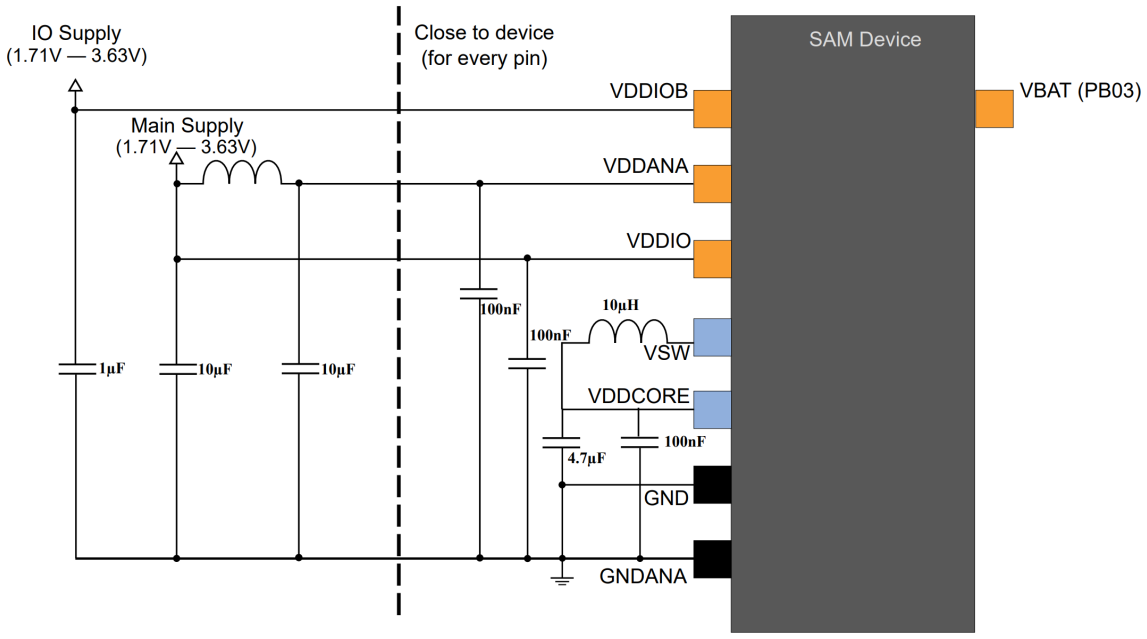
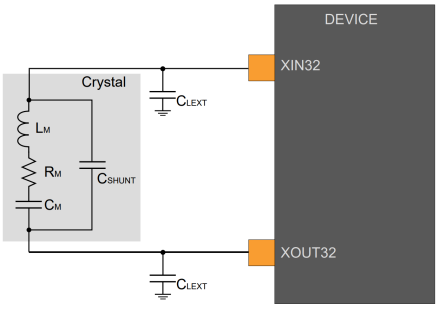


Figure 7: ATSAME5x power supply recommended connections [2, p. 2097]

faible de 7[pF], une resistance série équivalente (ESR) de 50[Ω], une shunt capacitance de 1.15[pF]. Selon la documentation, il pourrait très bien opéré sans plus de précautions (crystal avec capacité de charges inférieur à 12.5[pF]. Néanmoins, pour un desgin robuste il est recommandé de monter deux capacités de charges de 6[pF]. Le détail du calcul est donné ci-dessous.



	Selected	Required	unit
Load cap.	7	< 12.5	[pF]
Shunt cap.	1.15	< 1.7	[pF]
ESR	50	< 58	[kΩ]

$$C_{XIN32} = 3.1[pF], \quad C_{XOUT32} = 3.2[pF]$$
$$C_{para} = \frac{C_{XIN32}C_{XOUT32}}{C_{XIN32} + C_{XOUT32}} = 3.15[pF]$$
$$C_{LEXT} = 2(C_{Load} - C_{Para} - C_{Shunt} - C_{PCB})$$
$$C_{LEXT} \approx 6[pF]$$

Figure 9: Requirement for 32.768 [kHz] external oscillator [2, p. 2015]

USB

The ATSAME51 has one full speed (12Mbps) Universal Serial Bus 2.0 (USB) interface, that can handle host and device function. It must be powered between 3 and 3.6[V] to work and it has some conditions on the clock source and frequency, especially for host operations. We chose to use a micro-USB type B connector for the line, because it is compact, convenient, and highly available. USB connection is there to facilitate the development of software. In one hand because it can be easily reprogrammed as soon as there is a suitable bootloader flashed on the MCU. In the other hand because it can also power the board which eliminates the need of a lab bench power supply during development phase. Having two different power sources will cause problems thus in the first iteration, the VBUS (5[V]) of the USB line was left unconnected, ensuring that this will never happen. It was impossible neither to damage the board, the boat nor the computer connected. Nevertheless, it was inconvenient since when developing firmware, one always had to carry a lab bench power supply. We decided to fix that for the second iteration. The VBUS is thus connected to the rest of the board and supply 5[V]. As a safety feature, a schottky diode is installed between the VBUS and the entry of the LDO. This will protect both the board and the computer in almost every situation (expect the one of the grounds mentioned earlier. It is highly unlikely though and I don't think there exists a hardware solution to prevent it).

When it comes to the design of the USB line for the second iteration, I followed the recommendation for robust design in noisy environment. It embeds An USB transient protection as well as an RC filter for the shield. The USB data line is a differential pair line with an impedance of 90[Ω] properly configured and routed in *Altium*. Even though the design should be more robust and less sensitive to noise than in the first iteration with a simpler design, it comes with a negative. Indeed, USB transient protection are dedicated IC used for commercial applications, thus it is highly miniaturised. It is difficult to properly installed it on the board during assembly and untrained people are likely to fail. If USB isn't working, I would recommend getting rid of that protection and simply short the component. It is acceptable since this protection is in every case also present on the computer that will communicate with the board, and also because this is not a line that will be used when the boat is operated. Finally, the ID pin from the micro-USB B connector is by default left unconnected but a zero ohm resistor can me mounted on the board (by default not installed) if it needs to be grounded (in order to host USB communication).

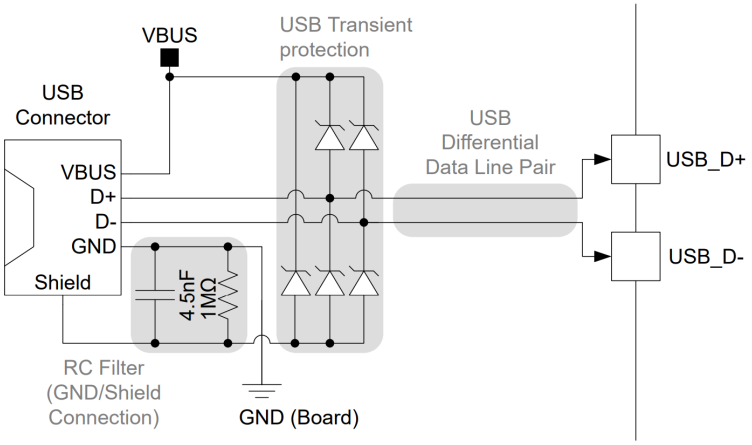


Figure 10: Robust USB Interface design recommendation [2, p. 2110]

SD

The microcontroller has two SD/MMC Host controllers, which means that it can be configured to use an SD card. The purpose of mounting an SD card on the board is to be able to save logs directly at the microcontroller level. In case of error in the CAN line, that could be a useful tool for debug. The SD1 line was in conflict

with the CAN lines so I chose to use the SD0 line. Then I followed the design recommendations given in the ATSAME5x datasheet.

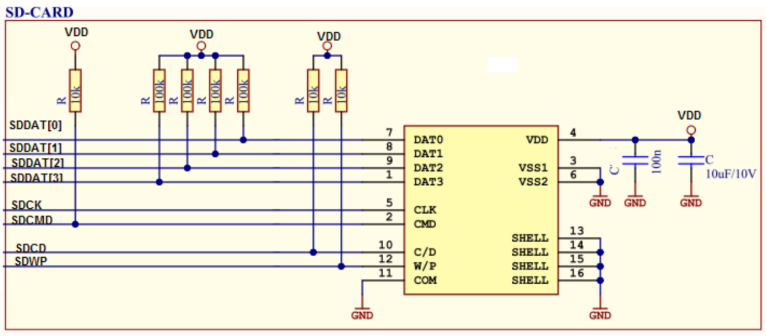


Figure 11: SD card typical circuit [2, p. 2111]

2.4.3 CAN

The ATSAME51 was chosen as the MCU for the new microcontroller board because it had two native CAN controllers. In addition, the CAN transceivers were chosen and ordered in the autumn of 2021. They are the TCAN337GDR and operate at 3.3[V] but are compatible with 5[V] logic. The data rate is flexible and goes up to 5 [MBPS], and they are CANopen compatible. The TCAN337GDR has two special pins, the S pin to activate the silent mode (high active), and the FAULT pin, to read if there is an error on the line (high active).

In addition, SSB has a norm for CAN connectors. There is a colour code, as well as a connector (S04B-PASK(LF)(SN)) and an order for the connections. I follwed this convention for my design.

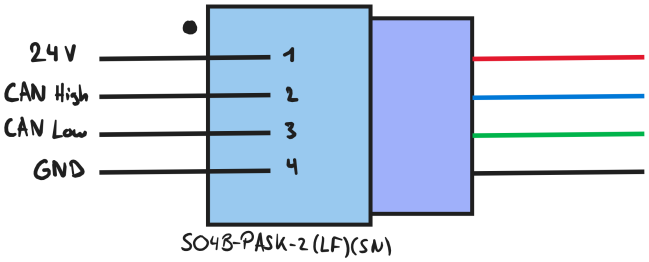


Figure 12: Swiss Solar Boat norm for CAN connectors

Again, I followed the design recommendations given by the datasheet. The CAN tranciever was placed as close as possible to the connector, so routing the line as a differential pair was neither necessary nor possible because of the short distance. As for the termination resistor, I opted for a flexible solution. I can choose to activate it or not via a DPDT switch. On the recommendation of my supervisor, it is not a 120[Ω] resistor but two 60[Ω] resistors in series with a grounded capacitor in the middle that were installed, in order to limit somewhat the disturbances.

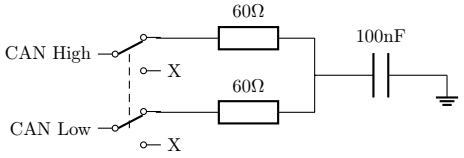


Figure 13: Double pole double Throw switch to activate or not CAN termination

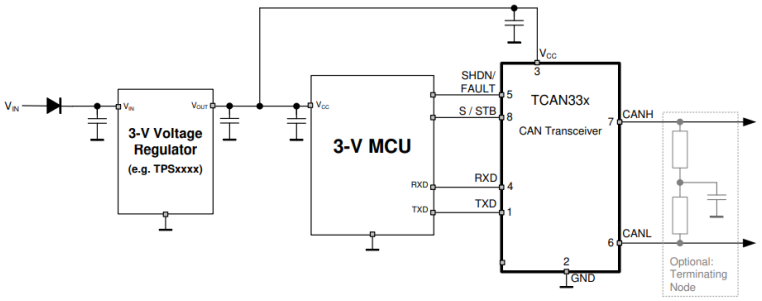


Figure 14: CAN typical circuit [3, p. 26]

Finally I was able to put everything together to wire properly the two CAN transceivers to the MCU. From the power selection with the DPDT switch, that comes through the CAN connectors, up to the proper wiring of the CAN transceivers. The Silent pin is High active. By default it is not meant to be use and a pull-down resistor is mounted. Nonetheless, if for any application One want to use it, it is still connected to the MCU and one can simply unsold the pull down resistor to have it properly functioning. Further more the FAULT pin needs an additional pull resistor, which is mounted. In order to facilitate debugging and to be able to install a logic analyser on the board when it is running, pin headers have been installed to analyse the CAN high and CAN low lines of both CAN lines.

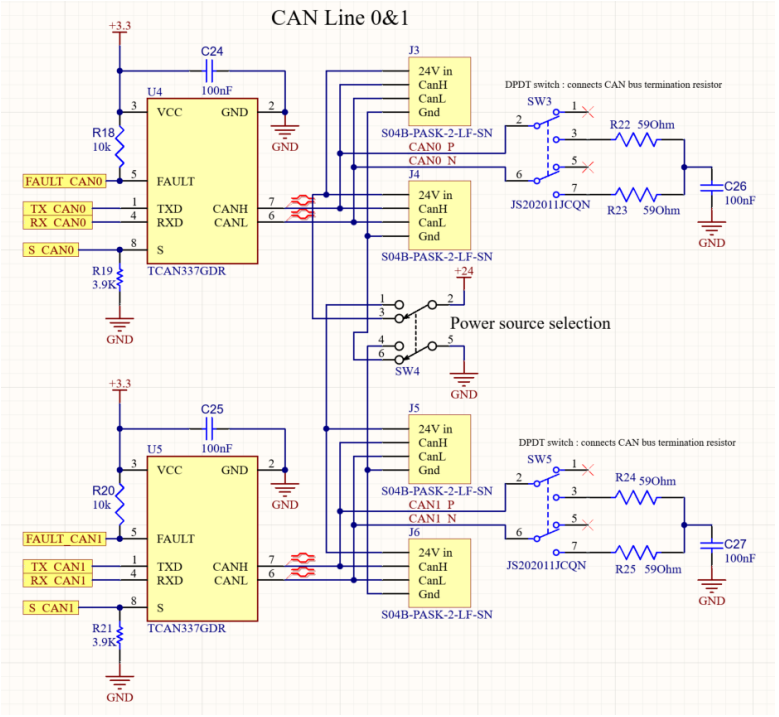
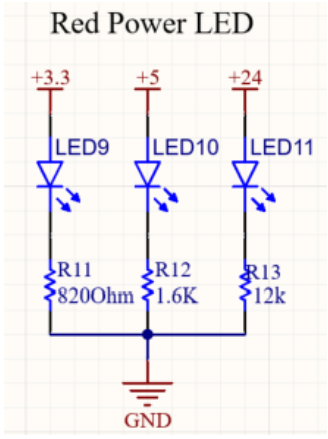


Figure 15: CAN transceivers schematic for microcontroller V2.2

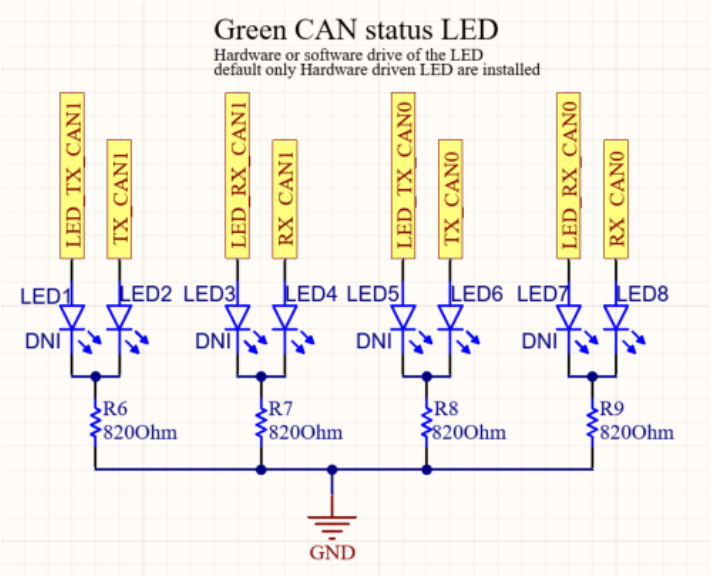
2.4.4 User interaction

With the first iteration we found that the card was not providing enough simple feedback to the user. So we decided to add a series of LED's to remedy this. Indeed, there is a red LED at each level of the power supply line (24[V], 5[V] and 3.3[V]), to facilitate debugging. At a glance, you can see if the power supply is

working as expected. Another interesting feedback is that of the CAN line. We wanted to get feedback when the card received or sent something on the CAN line. The simplest solution is to feed the LED directly with the communication line (RX/TX). However, you have to take into account the power supply capacity of the pin in question, and this distorts the signal. If this *hardware* solution did not prove satisfactory, we thought of a *software* solution. More complicated to implement, because it would require a code that manages the LED's at the same time as the messages, it has the advantage of not distorting the signal. I therefore implemented this hybrid solution by making the connections for both solutions on the PCB but installing only the *hardware* LED's by default.



(a) Power red LED's



(b) CAN green LED's

Figure 16: Feedback LED's installed on the Microcontroller V2.2

After providing these feedback LED's, we decided to add a switch with a parallel LED and a second additional LED for possible future application.

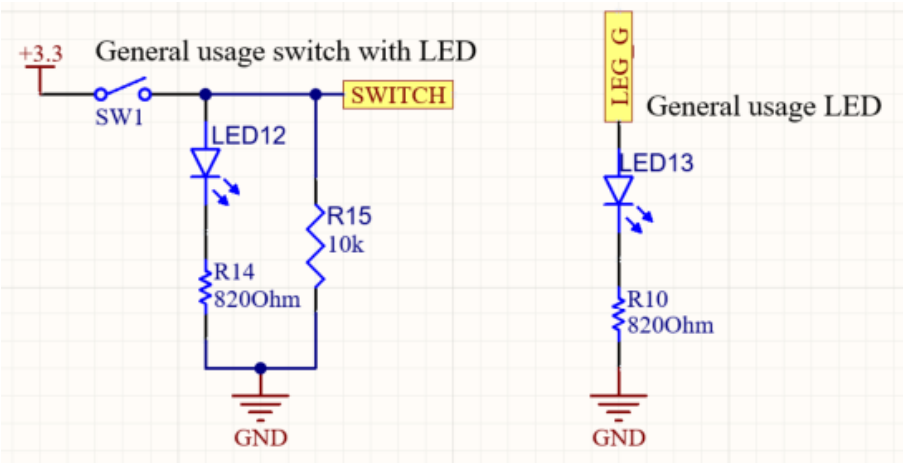
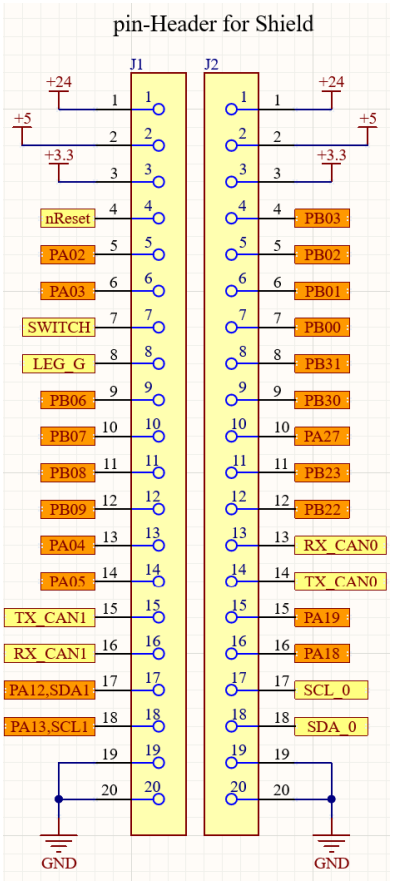


Figure 17: Additional Switch and LED's provided on the Microcontroller V2.2

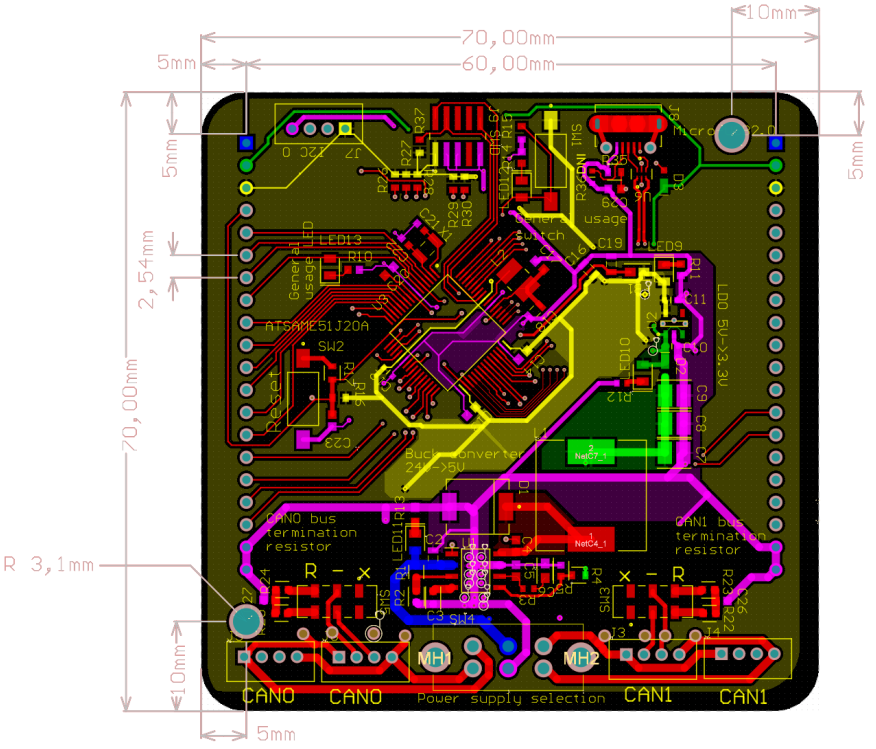
2.4.5 Shield

In order to modularise the design and allow a single microcontroller design to have as many applications as possible, we opted for the shield solution. By installing rows of pin headers, it is indeed possible to connect two PCBs together, in the case of the shield with additional circuitry for the microcontroller.

I opted for two rows of 20 standard female pin headers (2.54mm pitch, 1mm hole diameter), on each side of the board so that the shield would be stable and solid. Each row has a power supply, which would allow even half-shields to be made if necessary. The 40 available connections allow to put almost all the pins of the MCU on the shield. There are a few missing, but we were limited by the maximum size of the board (70x70mm), so we couldn't increase the size of the rows.



(a) Shield Pinout



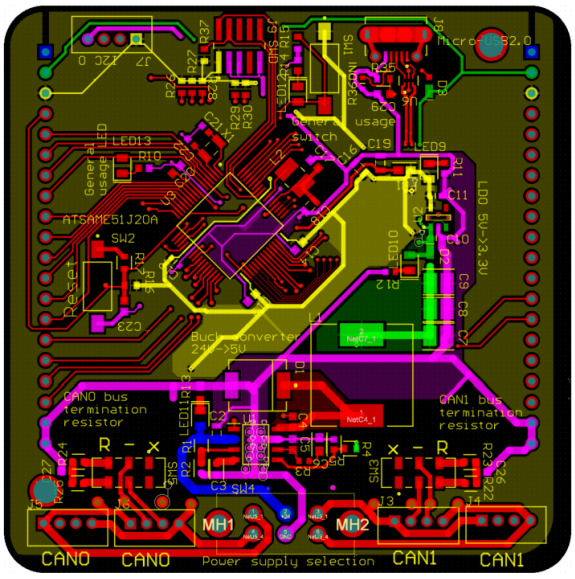
(b) Pin headers mechanical dimension

Figure 18: Pin headers connections for shields on the Microcontroller V2.2

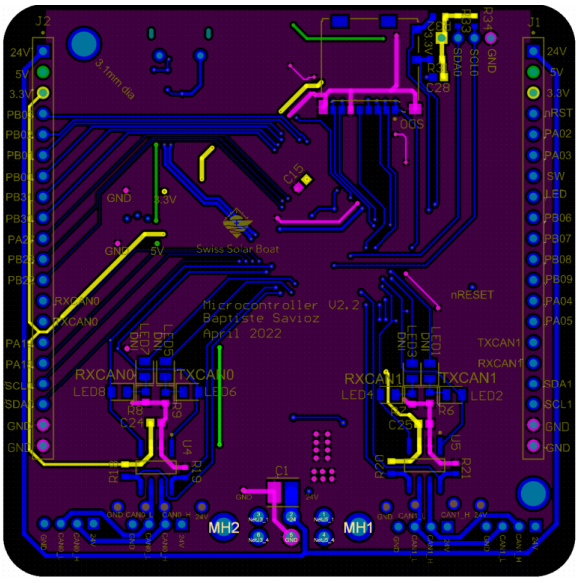
2.4.6 Routing

Now that all the circuits had been individually designed, all that remained was to make the complete schematic on *Altium* (section A), then to make the routing while respecting the manufacturer's manufacturing parameters. We chose to have the PCBs manufactured by *JLCPCB*, a Chinese PCB manufacturing company, because it was an extremely simple solution with low prices and very fast delivery and manufacturing times. The manufacturing capabilities of *JLCPB* were perfectly reasonable for what we wanted to do. *Altium* has a powerful tool for defining PCB design rules. Firstly, I defined the PCB stackup: the number of layers, the material, the thickness of each layer, in order to be able to calculate the impedance correctly (especially for the differential lines). Then, I configured the clearances of the tracks and pads, the size of the via according to the manufacturer's

manufacturing capacity. I then defined the different nets I had planned (24[V], 5[V], 3.3[V] and GND) with preferred track thicknesses appropriate to their use. Once the rules were well defined, I placed the components on the board, taking advantage of the experience of the first iteration to limit the number of overlaps and minimise the length of the tracks. The final result is as follows:

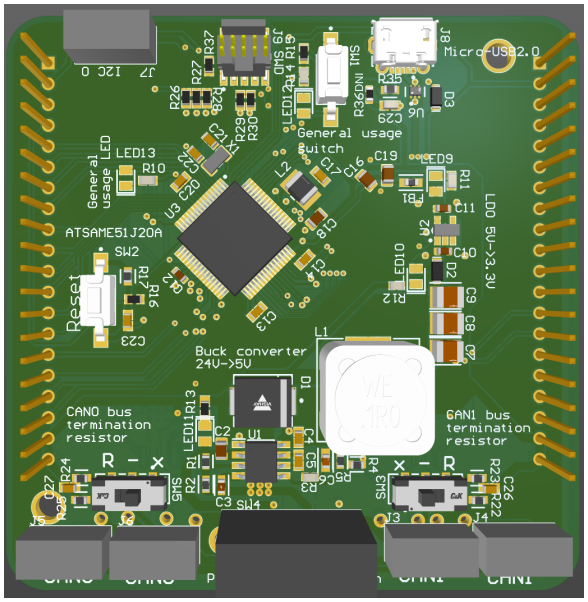


(a) Top view

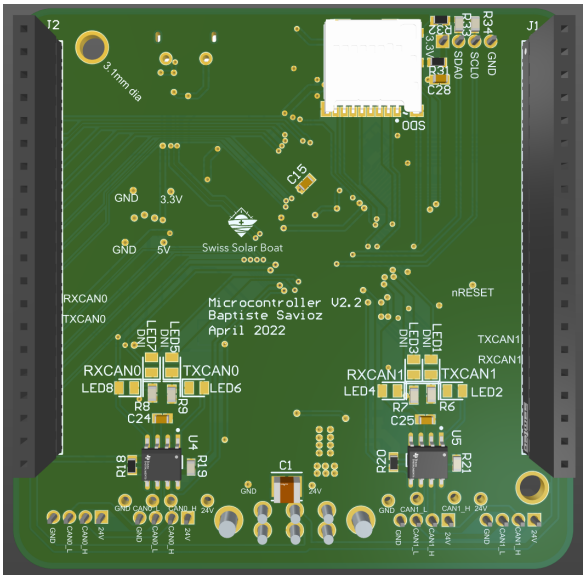


(b) Bottom view

Figure 19: View of routing of the microcontroller V2.2 in *altium*



(a) Top view



(b) Bottom view

Figure 20: 3D View of the microcontroller V2.2 in *altium*

2.5 Test

In order to validate the design, appropriate tests had to be conducted. Often overlooked, they are in fact just as important as the design itself. During this project, we conducted two phases of testing. The first in week 7 and 8 to validate the design of the first iteration and to draw up a list of improvements to be implemented for the second version. And finally the second in week 10 to confirm the design before starting the mass production of the 10 microcontrollers. Fortunately, we don't stop at these two test phases, but from then on, the damage is done.

2.5.1 First iteration testing

In order to verify the design of the first iteration, I first started with a visual inspection of my assembly, before testing with a multimeter that I had no shorts in critical areas. Once that was done, I could start by testing the power supply line. Using a lab power supply, I connected to the 24[V] input and checked with a lab multimeter that I had the correct voltage levels all the way down the line to the 3.3[V] output of the LDO. Secondly, after checking that everything was working as expected, I connected a small power resistor that we have at the SSB facility to the 5[V] power supply terminal. With this simple setup, I was able to test the buck under a resistive load up to 1[A] (5[W]) for about 5 minutes. In order to test it more rigorously, I should also have tested it under other types of loads (inductive, capacitive, time varying). I did not consider this to be necessary since the constraint on the power output of the buck converter (5[V] 1[A]) is far too high for the application. Moreover, the theoretical design should guarantee double, so I judged that we had a comfortable margin. I did not test the LDO too much. Indeed, there was no electronic circuit to design for this component (saving me the mistakes I could have made). They are reliable and their datasheet guarantees 600[mA] of output current, more than what is needed according to the specifications.

Once the power supply line was tested, it was time to test if I could program the microcontroller. As described above, it is programmed using the Serial Wire Debug (SWD) protocol. This requires a dedicated programmer. At SSB we have two JLINK EDU Mini's which are SWD programmers. I also used the Microchip Studio software which provides a higher level interface to the programmer. After unsuccessful attempts, I had my design reviewed and it was Professor Reuné Beuchat who discovered the first error in the PCB: I had reversed the GNDDETECT and nRESET pin on the SWD port, compared to what is defined by the protocol... I corrected this error with my soldering iron by doing a manual routing, and one faulty cable later, I finally managed to program the microcontroller. For this, I used the Adafruit UF2 bootloader[2], which is the same one used on the Feather M4 CAN development board. This makes it possible to program the microcontroller via the USB port and to use simpler tools, such as the Arduino environment, to facilitate further testing.

Once the bootloader was installed, I could easily program the microcontroller and test some of its features. First of all, I tested the CAN1 line directly on the CAN of the boat, from the CUS, when it was on land. The microcontroller read the messages without error and was able to send messages that were also read without error by the Raspberry Pi. Once again these tests were concluded under ideal conditions. I would have liked to have been able to test the CAN line more thoroughly, a critical element, but unfortunately this was unavoidable. There are many unsuccessful tests this semester and the association could not afford to miss one more. Moreover, the time was very limited for this project and the effort in terms of programming and connectivity to integrate a microcontroller on the boat at this stage of development was not feasible.

Moreover, I also tested the I2C 1 line of the microcontroller with a current sensor INA219 from *adafruit*. With a simple assembly: a laboratory power supply, a potentiometer, an LED and this current sensor, I could make sure that the I2C 1 line was working properly. It worked perfectly, nevertheless after having read the datasheet of the microcontroller on the advice of my supervisor, I noticed that the pull-up resistors defined by the I2C protocol were not installed and that it was necessary to do it. So I corrected this for the second

[2]A bootloader is a piece of code that execute when the MCU is powered. It is usually not modify when re-programming the device and may enhance some functionnality. The most common is to use simpler way to reprogram the device.

iteration. Again for reasons of time limitations, I could not develop the software necessary to test the I2C 0 line.

Finally, I tested some GPIOs with a basic *blink* and a small assembly as well as some analog pins with a lab power supply. These tests were conclusive.

Conclusion about the first testing session

This first series of tests allowed me to verify that my design was mostly correct. Despite a few small problems, I had a working prototype that could perform all the tasks in the specification decently. Furthermore, although I could not test the CAN0 or I2C0 line, it was reasonable to assume that this would not cause any problems. Indeed, the electronic design being strictly identical between the two lines 0 and 1, if one works, it is reasonable to assume that the second one will too. On the other hand, a big negative point that I became aware of was the difficulty of producing the software for this new custom microcontroller. I was able to produce code very quickly with the Arduino environment and the configuration provided by *Adafruit* for its *feather M4 CAN* development board. However, when I had to use things that were not in this configuration, it quickly turned into a nightmare. I drew attention to this in week 7 and the second design review. However, although I was aware of the problem, I didn't realise how important it was and how difficult it would be to solve and my big mistake was not to insist on it so that we could take the problem further upstream. Eventhough this wasn't strictly part of my semester project at Swiss Solar Boat, it greatly compromise the integration of the project and I should have put more effort on that.

	Test conducted
Basic insepection	Visual insepection and with multimeter to check for shorts
Power line testing	Measurement with multimeter and load test (5[V],1[A] for 5 minutes)
Programing	programming using SWD protocol and JLINK and flashing a bootloader
CAN 0	Not tested
CAN 1	Tested on the CAN line of the boat
I2C 0	Not tested
I2C 1	Tested with a current sensor INA219
SD slot	Not tested
GPIO	Some GPIOs tested with a blink code and a simple resistor + LED circuit
Analog Pins	some Pins tested with a lab power supply.

Table 6: Test conducted for the first iteration

2.5.2 Second iteration testing

As the design of the second iteration was mainly similar to the first, the testing conducted prior to the production of the entire PCB series was not very extensive. The main changes were in the optimization of the PCB layout and routing, which should only slightly affect the functionality. In addition, the timing was again extremely tight. So I started by assembling a complete board, before conducting the same tests as for the first iteration described in the table 6. This time, I had no errors and everything worked as expected, despite the same limitations as before. So I was confident enough to go into mass production.

2.6 Programation

This section concerns the integration of the V2.2 microcontroller and the different solutions explored to program it. Strictly speaking it is outside the scope of this semester project and would deserve another semester project on its own. Nevertheless, I will try to present the work done until the date of writing this report: June 13, 2022.

First of all, the ATSAME51 is programmed using the SWD protocol. SSB has two *JLINK EDU mini* from *SEGGER* that allow this to be done. It is a relatively complicated tool to use (command in the terminal), so I also used *Microchip Studio* from *Atmel* in order to manage the *JLINK* and to have a graphical interface facilitating the use. With this base, I will present you the two solutions to program the microcontroller V2.2 that SSB explored.

2.6.1 Arduino

Swiss Solar Boat is an association that wants to be accessible to students of all sections and levels. For this reason, it is willing to use the *Arduino* environment to program microcontrollers. It is a sufficiently simple tool so that, for example, a mechanical engineering student wishing to integrate strain gauges into a part can do so. Furthermore, all the code written for the boat so far has been done with *Arduino* and changing the environment now would make it difficult. With the preamble out of the way, let's move on to how *Arduino* works.
Arduino is a brand that offers all the necessary tools to create an embedded system, from the hardware with the boards and sensors to the software with the programming environment allowing to program them : *Arduino IDE*. The objective is to make our microcontroller V2.2 *Arduino* compatible so that it can be programmed with it. The first step is the bootloader allowing to program the board using a USB connection. The second is to have a definition of the board so that *Arduino* can compile code for it.

The *adafruit* M4 CAN feather on which Richard Ismer did his semester project is based on the same microcontroller: the ATSAME51. Therefore, *adafruit* has already done some of the work necessary to integrate this board into the *Arduino* environment. The UF2 bootloader from *adafruit* (free of rights) can be downloaded and flashed on the board with the *JLINK* so that it can be programmed by USB afterwards. Then, we can take the definitions of the board made by *adafruit* for the *Arduino* environment. They are presented in the form of a package that can be directly installed from the *Arduino IDE* board manager. These definitions include a hardware definition of the board: on which pin is available what. For example, pin PA03 is an analogue pin that uses channel 4 of the ADC0, and also allows the generation of the external interrupt 6 and is part of the SERCOM3 is an example of an hardware definition. Then there are also tools in the package to compile the code, that is to say in our case a compiler for ARM machine that makes the link between the code we write, the hardware definitions of the board and the machine.

The problem with using the *adafruit* package and the Feather M4 CAN board is that you can only do what *adafruit* has designed for this board, which reduces the functionality of our V2.2 microcontroller by half... So we decided to develop our own board definition. Fortunately, I was able to use what *adafruit* had done as a basis (free of rights). To make a new board definition, we start by modifying the variant.h and variant.cpp files of the board package we start with. It is in these files that we make the link between the hardware and the software. I was able to define new pins for the *arduino* environment and adapt their functionality (PWM, analog, SERCOM, Interrupt, timer, etc.). Once this was done, we could start to modify the board.txt and plateform.txt files to create our own package. This is where we link the variants we have created with the necessary compilation tools (ARM machine). Moreover, there is a variety of useful details for the package we are going to release. Then, you still have to create a JSON file, indicating where and how to download the package. In addition, I have put the JSON file and the package I have written on my personal github in public access. Finally, by providing the URL to download the JSON file to *arduino IDE*, one can download the package containing the definitions for the V2.2 microcontroller directly from the *arduino IDE* board manager and start using the board with the newly defined features. This is a simple and effective solution to allow anyone to program the microcontroller. The installation guide is available on the Gitlab documentation as well as in the SSB Notion.

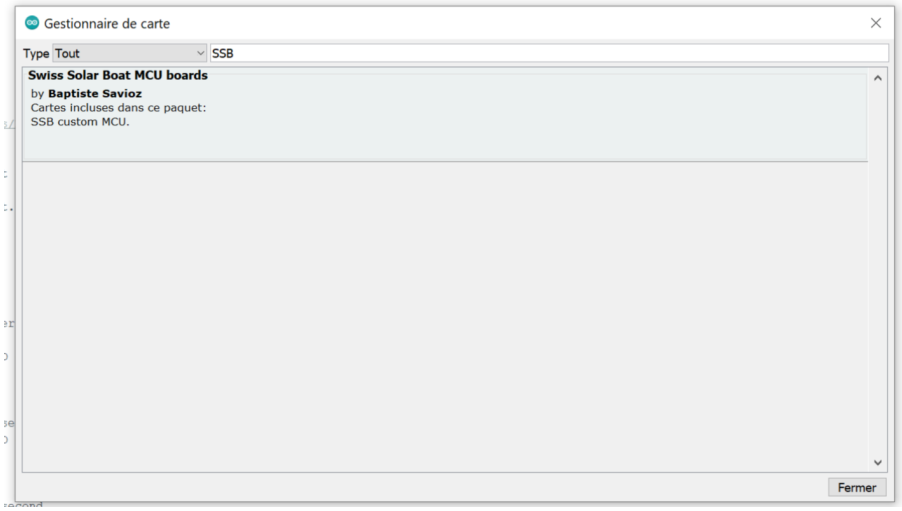


Figure 21: Swiss Solar Boat *Arduino* package to install directly through the board manager

Nevertheless, there were several limitations that I did not manage to overcome. I did not manage to use the ADC1, only the ADC0. Indeed the command line to use the ADC1 in a variant is, to my knowledge, not defined. It would be necessary to make modifications one level lower, thus at the level of the tools of compilation and that exceeds by far my knowledge and the time at disposal. This is the first limitation, you can only use half of the analogue pins. Secondly, the UF2 bootloader, which is required to program the microcontroller via the USB line, has configured it in such a way that a start of frame (SOF) bit at 1kHz is sent to pin PA23. However, pin PA23 is also the RX pin of the CAN0 line. Therefore, we could not get the CAN0 line to work. This time again, modifications would have to be made that are far beyond my knowledge. This second limitation is really compromising. Finally, I have not implemented all the functions for all the pins. Some interrupts, some timers, some SERCOM ports still need to be added and we would surely discover further limitations.

2.6.2 MPLAB

The second solution that Swiss Solar Boat considered for programming the microcontrollers is *MPLAB*. This is the software provided by *ATMEL*, the manufacturer of the ATSAME51. This solution has the advantage that it is sure to have no limitations, but it comes with a huge negative point: it is extremely complex to use. This would limit the people within the association capable of writing software or debugging it to a handful of people. In parallel, it is Sébastien Delsad and Richard Ismer who have been working on writing code that would allow the microcontroller to be used with *MPLAB*. At the time of writing this solution is still under development. They have managed to get both CAN lines working, but are still struggling to use them simultaneously.

2.7 Conclusion to the microcontroller project

The 10 microcontrollers have been assembled and comply with the specifications to the fullest. Although there are some minor errors, such as a small mistake on the markings on the surface of the board, for example, I consider the project as in itself a success. However, there is still a long way to go before it is successfully integrated into SSB. At the time of writing this report two microcontrollers (cockpit and battery) are almost implemented, but the solution especially concerning the programming of the latter is still far from being satisfactory and this is a job for the coming weeks and who knows, for a next semester project.

3 Design of the battery's PCBs

3.1 General introduction

3.1.1 Why do we need a new battery?

This semester the choice to design a new battery was made. The aim was to optimise the V1 battery. On the one hand it was reduced in weight by using fewer busbars. The busbars are heavy because they are made of copper. On the other hand the battery has been optimized by limiting its consumption. In order to do this, the fans have been replaced, going from a maximum consumption of 8W to 3.6W per unit. Finally the BMS was replaced by the n-BMS by Lithium Balance, as the tinyBMS used in the battery V1 was not always reliable.

Therefore, the aim of this semester project was to design the PCBs for the new battery. The PCB' s to design are the following:

- Relay block
- Monitoring block
- Micro-controller

3.1.2 Specifications of the battery

In order to implement the PCBs in the new battery there are requirements related to the choice of components for the battery that they have to fulfill.

BMS The n-BMS by Lithium Balance is a very powerful and reliable BMS that can be used to manage batteries much bigger than the one we are building. It consists of a master unit (MCU), which handles all the interactions with the control electronics and a slave unit (CMU), which handles all the cells. The BMS has a multitude of GPIOs which can be used to control it and to drive relays, as well as a CAN interface to communicate with the other systems on the boat. The user can configure all parameters via the BMS Creator software provided by Lithium Balance.

Fans The fans chosen to cool the V2 battery are NF-F12 industrialPPC [4] (Protected Performance Cooling). They are optimal because they ensure moderate power consumption compared to comparable high-speed fans while providing high airflow and pressure capacity and limiting noise levels. The fans are powered by a three-phase DC motor supplied with 12V and regulated by a PWM signal. The fan characteristics are detailed in table 7.

Fan specification		**Comments**
Size	120x120x25 mm	
Max. input power	3,6 W	
Max. input current	0,3 A	For 5 fans the output current is therefore 1,5 A
Operating voltage	12 V	
Max. PWM current	3.6 μ A	The current required for the PWM signal is very low, thus a power circuit is not needed.
PWM frequency	$>$ 20 kHz	Driving the coils at rates greater than 20 kHz moves the noise outside of the audible range
Connector	4 pins	
PWM low level	0-0.4V	
PWM high level	5V	

Table 7: Characteristics of NF-F12 industrialPPC

3.1.3 Battery wiring diagram

Figure 22 shows the complete Battery wiring diagram.

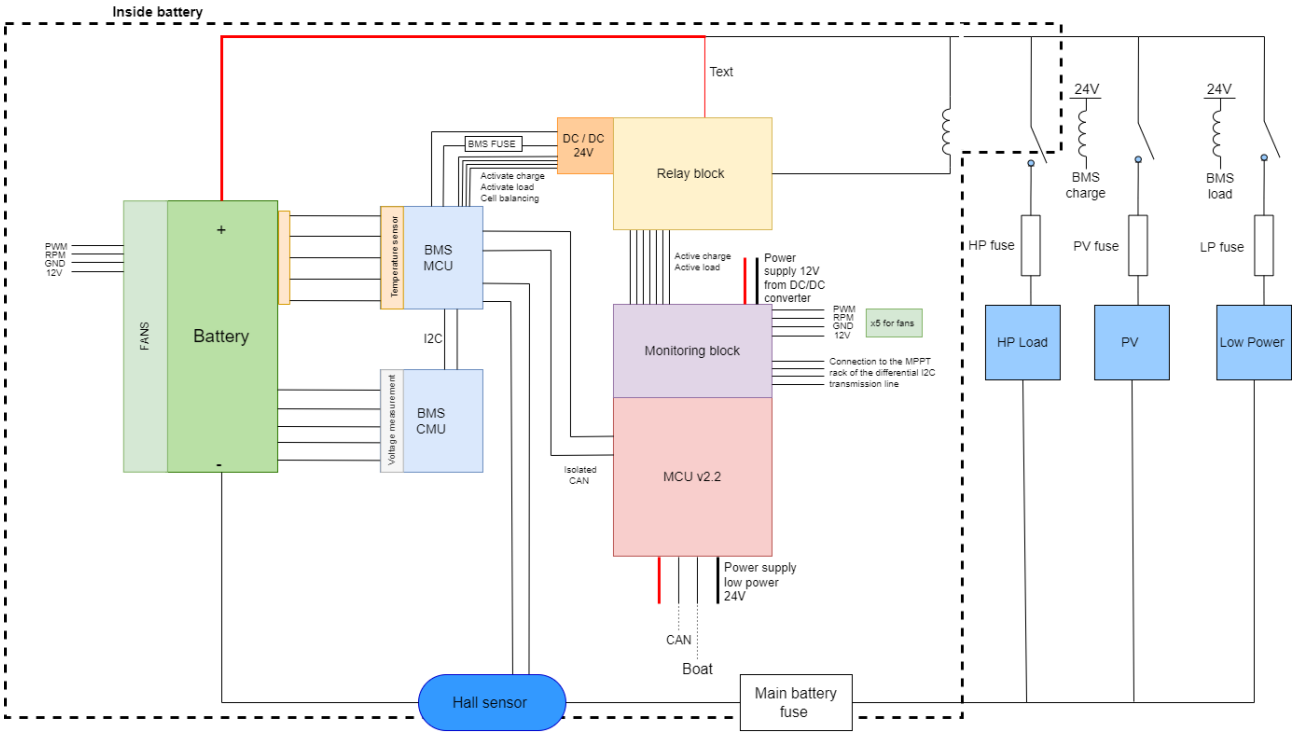


Figure 22: Electrical circuit diagram of the V2 battery

3.2 Monitoring board project

3.2.1 Specifications

The monitoring block has two functions:

- Drive the PWM signal for the 5 fans . In fact, as seen in section 3.1.3, the monitoring block receives directly from the MCU the PWM signal indicating the rotation speed of the fans. The signal is then sent to the five fans.
- Convert a differential I2C communication signal to normal I2C. Indeed the communication between the MPPT and the MCU is done by I2C communication. To limit the noise on this signal, the I2C is used in differential mode. The monitoring block is therefore used to convert I2C from differential to normal mode in order to limit the size of the MCU.

Functional requirements In order to fulfil the above functions, various constraints must be respected with regard to the entire boat. The features to be respected are listed in table 8.

Functionality	Comments	Assessment
PCB size	PCB should be small as possible	70mm x 70 mm
Shield of MCU	The monitoring block being a shield of the microcontroller its size and its connections must be adapted to the pcb of the microcontroller.	Standard male pin headers connectors
Power supply 12V	The fans are powered by a 12 Volt power supply.	12V must come through a S04B-PASK-2(LF)(SN) (2 lines 12V, 2 lines GND)
Connection to a PWM pin of the microcontroller	Choose a microcontroller pin able to send a PWM signal	Connected to a TCC pin of MCU
Isolation between the PWM signal of the microcontroller and the fans	The grounds are different for the microcontroller and the fans. Thus it is important to isolate the PWM signal.	Use of an optocoupler
I2C differential to normal converter		Use converter PCA9615
2 connectors for the relay block		A connector S05B-PASK-2(LF)(SN) should be use
5 connector for fans		A connector 470531000 should be use
Connector for MPPT		A connector S04B-PASK-2(LF)(SN) should be use
Reasonable price		Around 30 CHF

Table 8: Summary of the functional requirements of the monitoring block. All the requirements are mandatory.

3.2.2 Timeline and methodology

Timeline In order to carry out the implementation of the monitoring block in the battery, the timeline presented in table 9 was followed. The goal was to have the physical implementations done week 12.

Two iteration for the monitoring block For the monitoring block, I [3] opted to do 2 iterations of the PCB before week 12. Indeed it was the first time I designed a PCB and therefore 2 iterations allowed me to correct possible errors made on the first version.

In the first version the vast majority of the circuit was done. In order to have a circuit that works the first time, I tested a part of the circuit on a breadboard. This allowed me to correct ground errors that I made. Finally in the first version the I2C differential to I2C converter, the PWM signal control and the fan power supply worked correctly. However a second version was necessary to modify or add the following things:

- Connections between the relay block and the MCU
- Modification of the position of the pins headers with the MCU
- Adding a pull-up resistor to the PWM signal input from the MCU.
- Optimisation of the routing

Design procedure Designing a PCB requires a great degree of rigour and seriousness in order to avoid omitting details that might not make it work. Indeed, multiple iterations due to mistakes can seriously delay the achievement of the project, and waste time and money. Therefore I followed the following procedure to designate my PCB.

[3] In this part, the pronoun I refer to Blanche Brognart

Week 1	Understanding of the V1 battery PCB Discussion of the requirements of the new monitoring block
Week 2	Circuit of the monitoring block
Week 3	Choice of components Monitoring block circuit Ordering the components
Week 4	Testing the circuit on a breadboard
Week 5	Change of the circuit PCB design on altium
Week 6	Ordering the components Ordering the PCB V1
Week 7	Documentation
Week 8	Welding the components onto the PCB Test
Week 9	Modification of the circuit and PCB
Week 10	Ordering the V2 version
Week 11	Welding and test
Week 12	Implementation
Week 13	Implementation
Week 14	Implementation

Table 9: Timeline for the monitoring block semester project

1. Creating the schematic in line with the requirements
   A very important part of the project is the schematic of the circuit. For the schematic I started from the requirements that the PCB has to fulfil. Then once all the objectives were clear, I started to design the circuit, paying close attention to the recommendations in the datasheets. To make sure that I had designed my circuit correctly I tested it on a breadboard to draw any possible design errors.

2. Use Altium to create PCB layout
   The challenge was to get to know this software for making PCBs.

3. Design your PCB stackup
   In the design it was also important to think about the stackup. Indeed in PCB design stackup has a role on impedance, which refers to how much and how fast electricity can move along a trace.

4. Define design rules
   This step is largely dictated by the PCB manufacturing company's standards. For our part the PCBs were ordered from JLC PCB and thus it was important to respect their standards.

5. Place your components and drill holes
   The components have been placed in an optimal way in terms of routing but also according to the requirements (especially for connectors).

6. Route the traces
   The routing was done in such a way as to optimise it. Furthermore I was careful to minimise the impedance of the traces and to pay attention to the size of the traces according to the current that had to flow through it.

7. Add labels and identifiers

3.2.3 Solution analysis

Fans driver The 4 pins of the 5 battery fans are connected to a 12V power supply, the ground, a PWM and an RPM signal which provides feedback on the actual PWM signal of the fan.

1. Power supply

The 12V power supply and fan ground are provided by an isolated stepdown DC DC converter module located outside the battery. It ensures the transition from 50V at the battery output to 12V. The 12V is used to power the CUS, the sidewall and the fans of the battery. At maximum the DC DC converter should provide 3A of current which is largely achievable as it can provide a maximum of 5A (60W power).

2. PWM signal

The PWM signal comes out of the PM10 pin of the micro-controller. However the PWM cannot be connected directly to the fans for two reasons :

- As mentioned in section 3.2.1, the ground of the micro-controller and the 12V power supply is different. Thus it is necessary to isolate the two signals.
- Secondly, according to NF-F12 fan specifications table 7, the PWM high voltage needs to be at 5V to ensure that the fan reads the received PWM.

In order to galvanically isolate the two signals, an optocoupler was inserted to separate the two signals. The *H11L1M* [5] is chosen to perform this task. It has the particularity of having a high speed integrated circuit detector integrated with a transmitter diode allowing a high data rate of up to 1MHz, well above what is required, i.e. 20kHz. The output incorporates a Schmitt trigger which provides hysteresis for noise immunity and pulse shaping.

Let's have a look at the characteristics of the H11L1M to check if they are consistent with our requirements and the elements to design around. Figure 23 shows the circuit diagram of the H11L1M.

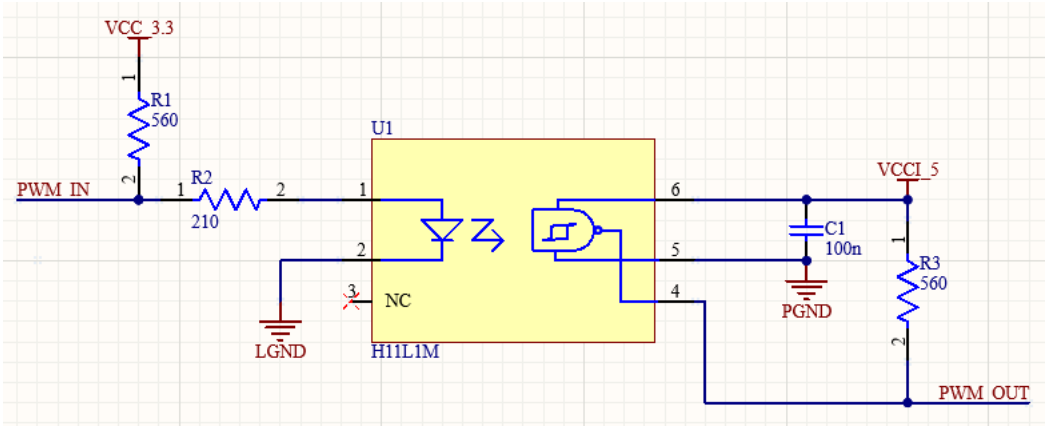


Figure 23: Schematic of the monitoring block optocoupler. PWM_IN corresponds to the PWM coming from the microcontroller and PWM_OUT corresponds to the PWM sent to the fans.

- The output operating voltage range is 3 to 15V. Therefore, an output of 5V is possible.
- The second thing to pay attention to is the maximum forward current of the diode (I_F max). The H11L1M has a maximum forward diode current of 30 mA and a typical current of 10 mA. The input resistance (R2 on the schematic Figure 23 is therefore designed via equation (1) with typical I_F.

$$R_2 = \frac{VCC_{PWM_{IN}} - \text{Forward voltage diode}}{\text{Typical } I_F} \tag{1}$$

$$R_2 = \frac{3.3V - 1.2V}{10mA} = 210\Omega$$

- The Schmitt trigger of H11L1M is inverter, therefore the output of the optocoupler is inverted with respect to the input. This aspect has been taken into account in the micro-controller code. When the output is set to 1 with a duty cycle of 100% the fans are switched off and when the output is set to 0 they run at their maximum speed. The truth table of H11L1M is presented table 10.

Input	**Output**	**Fans**
High	Low	Slow rotation speed
Low	High	High rotation speed

Table 10: Truth table of H11L1M

- R1 is a pull-up resistor preventing the fans from running when the micro-controller is switched off. In fact, when the micro-controller is switched off, the battery is expected to be off and therefore the fans are required to be stopped and not at their maximum speed. The resistor R1 has been designated as follows. As large as possible to limit the current in the PWM signal when the logic level is low and small enough that the current in the emitter diode of the optocoupler is greater than the threshold current (I_F). Therefore, the value of 560 Ω was chosen as a compromise between the two factors. (The value of the resistor could have been taken slightly larger but for logistical reasons and simplicity I opted for a value of 560 Ohm, as most of the resistors in the circuit are equal to 560 Ω).

- C1 is placed between the 5V and ground output. This capacitor acts as a decoupling capacitor. It is a local source of power for the optocoupler as it stabilises the output voltage when the current changes rapidly. Typical values for high frequency noise decoupling capacitor should be between 0.01 µF to 0.1 µF. In our case the value of C1 was taken at 0.1 µF.

- Finally the optocoupler has an open collector at the output. Therefore, R3 acts as a pull-up at the output (passive rising edge). A large pull-up resistor allows the output transistor to be better saturated and consume less current, and a small resistor allows a higher output frequency. To balance these two considerations, the R3 resistor is set at 560 Ω.

An important point to look at is the origin of the isolated 5V voltage of the optocoupler. In order to have an isolated voltage from the micro-controller, the 12V of the fan supply is converted to 5V. To do this we use a buck converter, the *LT1117-5* [6]. A buck converter was chosen over the switching regulator for the following reasons. The switching regulator has a better efficiency and therefore less joule losses but it is more expensive, takes more space and requires more components. It is useful when the voltage difference between input and output is large and the current is high. In our case the voltage difference is important since it goes from 12V to 5V but the current is low. The maximum current that the optocoupler will require is when the output PWM is at 0, i.e. I_{MAX} = 5V / 560 Ω = 9 mA. Therefore the maximum power that can be lost in the buck converter is calculated equation (2).

$$P_{MAX} = (V_{IN} - V_{OUT}) \cdot I_{MAX} \tag{2}$$

$$P_{MAX} = (12 - 5) \cdot 0,009 = 62,5 \text{ mW}$$

Thus, losses are quite low and do not justify the use of a switching regulator.

This regulator is suitable for our requirements because it can take an input voltage of between 6.5V and 15V. In our case, the input voltage is 12V. The output voltage for this converter is fixed at 5V with a margin of error of 0.1V. The output current can be up to 800 mA, which is far above what is needed because I_{MAX} is equal to 0,9 mA. Figure 24 shows the integration of the DC-DC converter into the monitoring block circuit. Let's look at the components and connections around the integrated circuit LT1117-5.

- The LT117 offers the option of modulating the output voltage below 5V. To do this, a resistor is placed between VOUT and OUT. In our case, we are not interested in modulating the output voltage, so VOUT and Out are connected to the same potential.

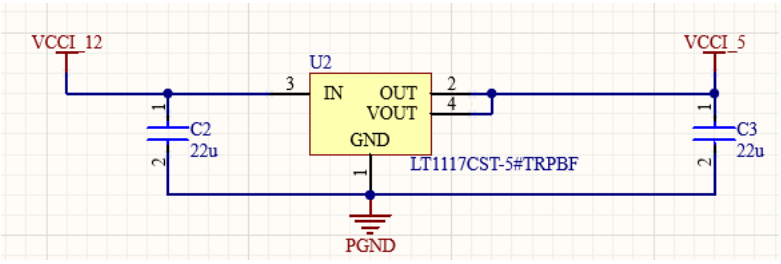


Figure 24: Schematic of the monitoring block DC-DC converter

- C2 and C3 are two decoupling capacitors. It is recommended by the datasheet to take decoupling capacitors higher than 10 µF to ensure a good transient response of the load with large changes in load current. Therefore the values of these capacitors are taken at 22 µF.

MPPT to MCU interface For the conversion from differential I2C to normal I2C the following integrated circuit is used: *PCA9615*. Figure 25 shows the schematic of the differential to normal I2C converter circuit. The differential inputs are DSCLM, DSCLP, DSDAP and DSDAM and the output is the clock signal (SCL) and the data signal (SDA). The SDA and SCL outputs are then connected to the pins of the MCU reading I2C.

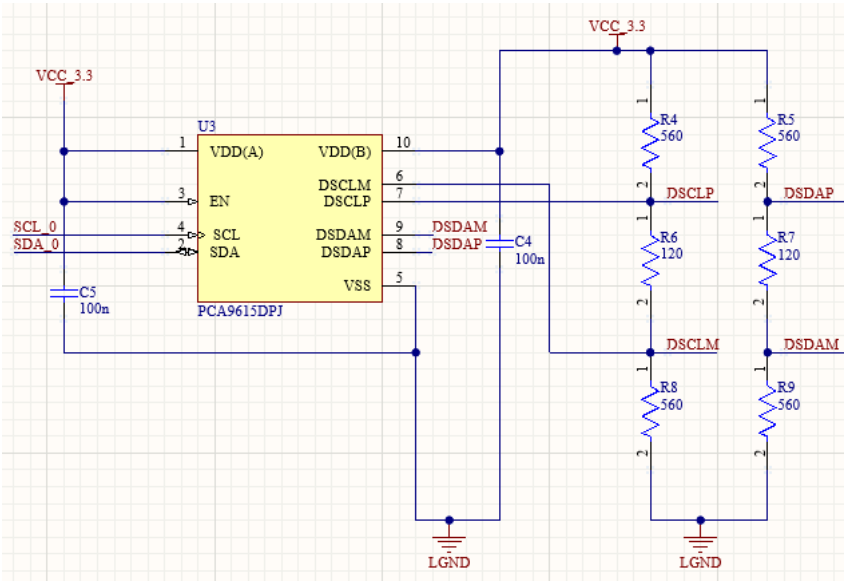


Figure 25: Schematic of the monitoring block I2C differential to I2C converter

- According to the PCA9615 datasheet [7], the polarisation of the differential lines is done by a pull up and pull down resistor. This arrangement is suitable for long distances of more than one metre (1m), and against problems of ringing and reflections on the unterminated bus. In addition the differential I2C side of the PCA9615 is powered by the VDD(B) power pin at 3.3V in our case. This allows the differential I2C bus to also be biased in the idle state (D+ more positive than D-) to be compatible with the I2C bus, when not transmitting data. In our case the pull up (R4 and R5) and pull down (R8 and R9) resistors have been set to 560 Ω.
- Between the pull-up and pull-down resistors, a third resistor is also placed. The three resistors complete the transmission line. The cable between the MPPTs and the monitoring unit is connected via an *ETHERLINE*

FD P Cat [8]. Therefore, the transmission line ends with the characteristic impedance of the cable, i.e. 100 Ω defined by the three resistors. Equation 3 shows that the three parallel resistors give 100 Ω.

$$\mathrm{R4//R6//R8} = \frac{560 \cdot 560 \cdot 120}{560 + 560 + 120} = 98,8\Omega \tag{3}$$

- C2 and C3 are two decoupling capacitors. Their values are set to 0.1 µF. They allow to stabilise the input and output voltage when the current changes rapidly.

Connectors The monitoring unit provide the following connections:

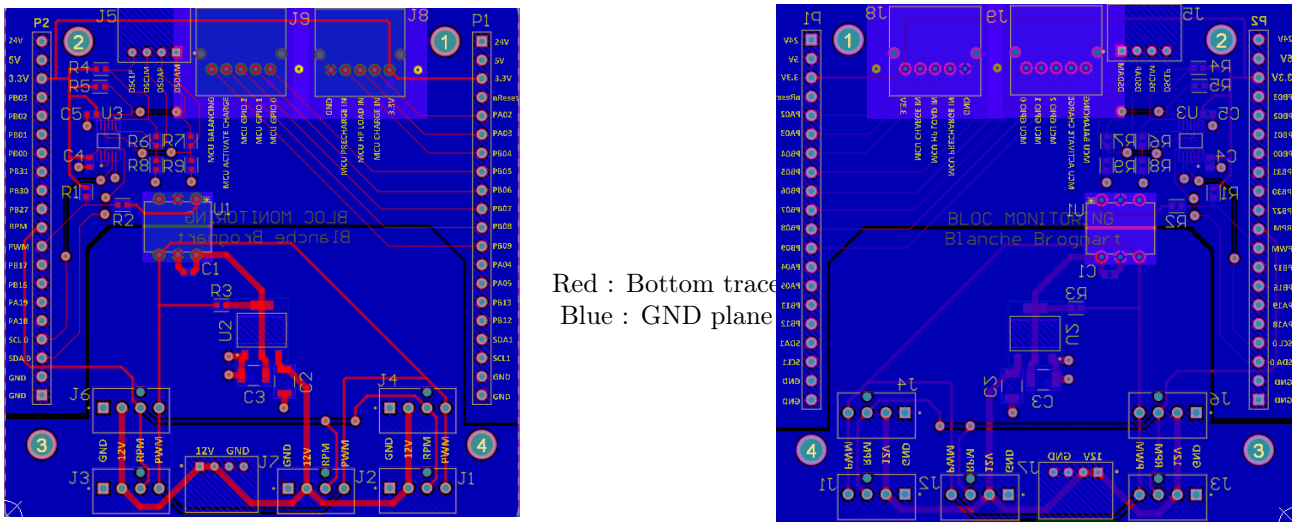
- connections to the micro-controller
- connections to the relay block
- connections to the 5 fans
- connections to the I2C transmission line from the MPPTs
- connections to the 12V power supply of the fans from the DC DC converter from 50V to 12V.

The choice of connectors was made based on the requirements of the current connectors on the boat and to what they are connected. Table 11 summarises the types of connectors and the number of pins.

Functionality	**Type**	**Number of pins**	**Numbers of connectors**
Connection to MCU	Pin header male	20 pos	2
Connection to relay bloc	S5B-PH-K-S (LF)(SN)	5 pos	2
Connection to 5 fans	470531000	4 pos	5
Connection to MPPT	S4B-PH-K-S (LF)(SN)	4 pos	1
Connection to 12V power supply	S4B-PH-K-S (LF)(SN)	4 pos	1

Table 11: Details of the monitoring block connector types

PCB Design After designing the schematic of the monitoring block, the next step was to design the PCB layout. The PCB has the same dimensions as the micro-controller, i.e. 70mm by 70mm, and a thickness of 1,6mm. The PCB could have been smaller but we opted for a shield that has the same dimensions as the MCU. The PCB has two layers, top and bottom. All components were placed on the top layer, leaving the bottom layer free for ground plane. The 2D design of the monitoring block is shown in figure 27.



(a) Top face

(a) Bottom face

Figure 27: 2D face of the monitoring block

For the design of the PCB, it is important to take care of different elements such as current in the traces, current path, interference and others. Thus when designing the PCB I took care of these different elements. The choices made to optimise the monitoring block are detailed below.

1. Ground plane
   In order to reduce electrical noise and interference from ground loops, two ground planes were placed on the bottom layer. A ground plane for the micro-controller ground and a ground plane for the ground of fan power supply. Figure 27.b shows the distribution of the two ground planes on the bottom side of the PCB.

2. Trace size
   The size of the traces was assessed according to the current flowing through them. For digital signals the traces are set to 0.254mm, for power supplies coming from the MCU the traces are 0.5mm thick and the fan power supply traces are 1mm. The trace sizes have been checked using a trace thickness calculator based on the current flowing through it and the thickness of the PCB.

Figure 28 shows the finale assembled monitoring block.

3.2.4 Test

Two sets of tests were carried out to check the correct operation of the monitoring block, one on V1 and another on V2. The tests to perform are detailed Table 2.

In the first and second test iterations all the tests detailed in table 1 were performed. All tests performed as expected. Figure 29.b shows the results on V2 of the monitoring block when testing the PWM signal control for the fans. The blue signal is the output PWM and the yellow signal is the input. It is clear that the output signal is 180° dephased from the input signal. Moreover, the output signal is between 0 and 5V with a frequency of 20kHz. We have the expected results.
The test shows spikes when the PWM signal changes from level 0 to 1. This can be explained by the parasitic inductance in the trace at the optocoupler. The parasitic inductance forms an LC circuit with the output capacitance of the optocoupler transistor. This circuit is activated when the transistor is switched on, resulting in a sudden current jump.

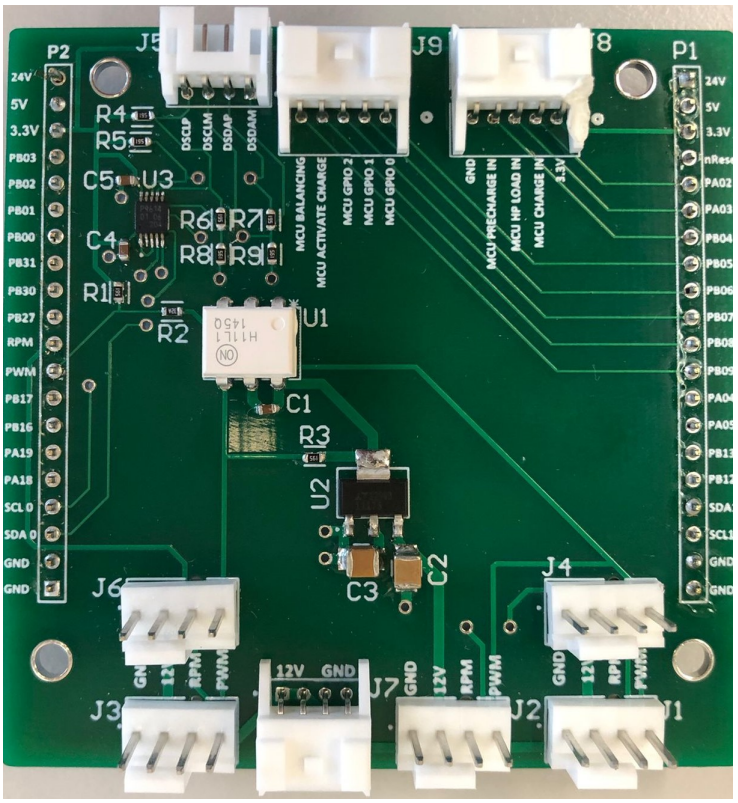
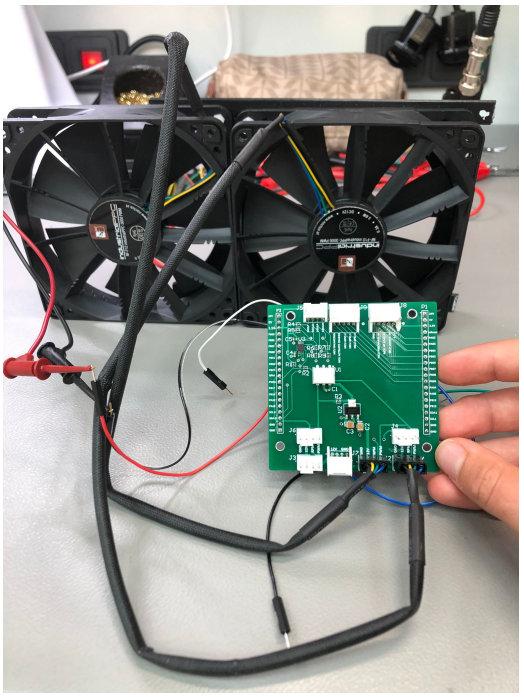


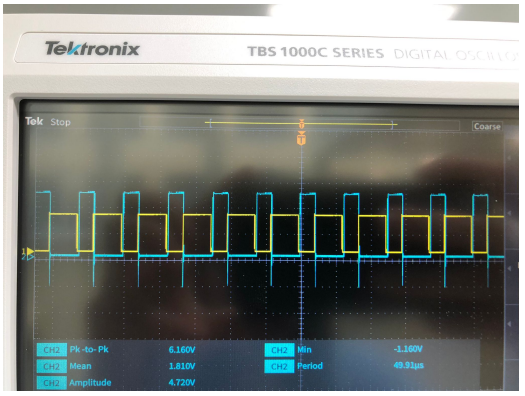
Figure 28: Top face of the monitoring block

Part to test	**Details of the test**
Checking connections	A multi-meter is used to check that the elements connected to the same potential are well connected.
PWM signal control for fans	The circuit is connected to an alternative voltage source in PWM mode between 0 and 3.3V and to a DC voltage source at 12V. Thus with an oscilloscope it is verified that the output PWM signal sent to the fans is a PWM signal between 0 and 5V and 180° dephased compared to the input PWM signal.
Variation of the fan speed according to the duty cycle of the PWM signal.	The fans are connected to the suitable connectors. We check that according to the variation of the duty cycle of the input PWM signal the speed of rotation of the fans changes. We note that in our case, the greater the duty cycle is, the lower is the speed of rotation of the fans.
I2C differential to I2C converter	The PCA9615 works in both directions. To test it we send an I2C signal via a micro-controller and we observe via an oscilloscope that we have a differential signal at the output corresponding to the input signal. The output signal we observe is the signal between the two differential potentials (e.g. between DSCLP and DSCLM and between DSDAM and DSDAP). However, testing in this way does not provide 100% verification of our circuit.

Table 12: Details of the tests performed to verify the functioning of the monitoring unit



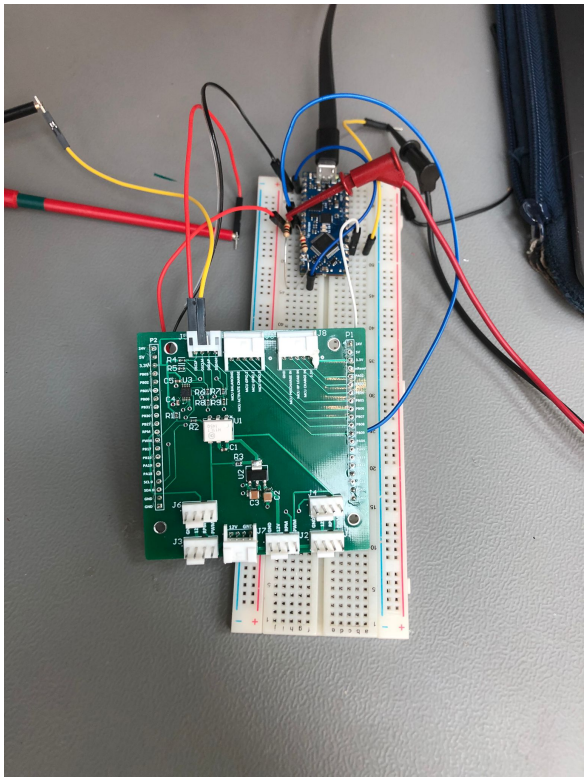
(a) Test set up



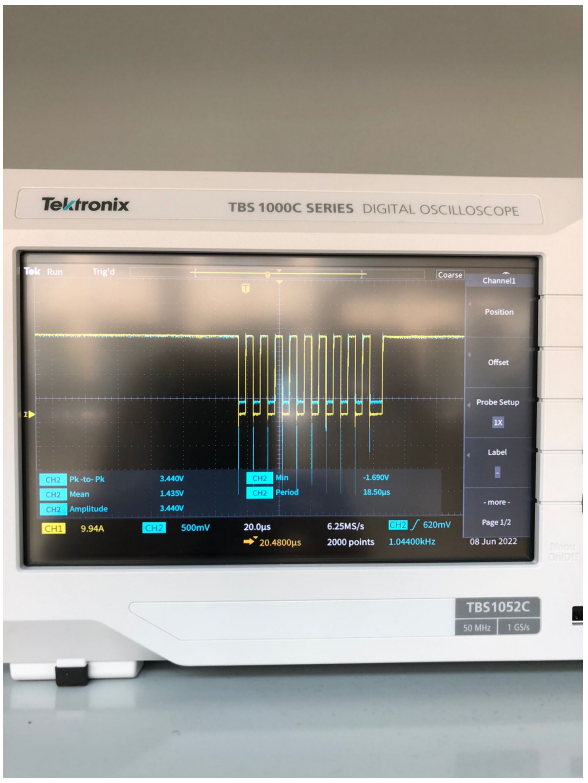
(b) Test results. The blue signal is the output signal and the yellow signal is the input signal.

Figure 29: Pictures of the test performed to verify the PWM signal sent to the fans. The fans are connected to the PCBs, a PWM signal is also connected as input between 0 and 3.3V with a frequency of 20kHz and a constant voltage source of 12V.

Figure 30.b shows the results of the V2 monitoring block on the differential to normal I2C converter test. The results obtained are satisfactory since the input signal corresponds to the output signal. However, we observe many spirkes on the output signal. This is due to the fact that the oscilloscope cable does not have the same impedance as the I2C signal transmission cable on the boat. Thus we have an impedance cut of 100 ohm in the oscilloscope cable.



(a) Test set up



(b) Test results. The blue signal is the output signal and the yellow signal is the input signal.

Figure 30: Pictures of the test performed to verify the I2C differential to normal converter circuit. The SDA and SCL inputs are connected to an arduino and the differential outputs are checked with an oscilloscope.

3.3 Relay Block project

3.3.1 Introduction

The relay block provides the interface between the BMS, the relays that control the different subsystems of the boats (HP load, LP load, PVs/MPPTs (charge) and MCU of the battery). It has to provide power to the BMS, let the BMS control the load and the charge relays, connect the boats ignition switch, as well as the emergency switch. The relay block also routes the control signals it needs to receive from the MCU through optocouplers, as the MCU is galvanically isolated from the BMS. In order to achieve this, all the relevant rules of the MEBC have to be respected.

In order to fulfil the above functionalities, various constraints must be respected with regard to the entire boat. The features to be respected are listed in table 13.

3.3.2 Timeline

Timeline In order to carry out the implementation of the relay block in the battery, the timeline presented in table 14 was followed. The goal was to have the physical implementations done in week 12.

Functionality	Assessment	Flexibility
Control all relays (LP load relay, HP load relay, Precharge relay, charge relay).	The BMS has to be able to disconnect all loads and the PVs at all times to protect the battery.	Precharge and HP load relays can be controlled by the MCU, as the BMS still has indirect control through the LP load relay
Embed all "small" relays.	LP load relay and precharge relays can be directly embedded on the relay block PCB, the charge and HP load relays are too big.	-
Embed 24V DC/DC converter.	This is needed to power some of the relays as well as the BMS.	Embedded on a small, seperate PCB, so that it can quickly be replaced in case of failure.
Connect the ignition switch.	Allows the pilot to wake the BMS from sleep mode in order to start the boat.	-
Connect the emergency switch.	The emergency switch has to be able to cut power to the HP load and the PV/MPPTs under full load (Rule 7.16 MEBC).	-
Embed fuses	The BMS and all loads, as well as the PV/MPPTs have to be fused (Rule 7.17 MEBC)	-
Connections between MCU and BMS	The MCU is galvanically isolated from the BMS, thus the connections need to be optocoupled	-

Table 13: Summary of the functional requirements of the relay block

Week 1	Understanding of the V1 battery PCB Discussion of the requirements of the new relay block
Week 2	Understanding of the V1 battery PCB Discussion of the requirements of the new relay block
Week 3	Understanding the new BMS and how it needs to be interacted with
Week 4	Understanding the new BMS and how it needs to be interacted with
Week 5	Choice of components & ordering
Week 6	Design in Altium
Week 7	Design in Altium & ordering
Week 8	Assembly of the PCB
Week 9	Testing of the relay block itself
Week 10	Testing of the relay block interacting with the BMS
Week 11	Testing of the relay block interacting with the BMS and MCU
Week 12	Implementation
Week 13	Implementation
Week 14	Implementation

Table 14: Timeline for the relay block semester project

3.3.3 Design procedure

As the relay block is heavily dependent on how the BMS works and which control signals it needs, understanding the BMS is the first step in the design procedure.

BMS requirements The n-BMS from Lithium Balances is designed as a state machine, according to figure 31. There are 2 control signals, BMS_activate_charge and BMS_activate_load. On start-up, the BMS performs a preliminary error check, and if no errors are present, it will go into READY mode. Once it is in READY mode, the BMS_activate_charge and BMS_activate_load signals can be used to tell the BMS to activate the charge or the load respectively. The BMS will perform an error check once again, and if no erros are present, activate the desired subsystems.

This state machine is designed to be used with the contactor arrangement in figure 32. According to its user manual, all of these contactors are optional, and the ones that are not used will be skipped in the activation process.

The n-BMS has 3 more relevant control signals:

1. BMS_activate_balancing: This signal allows the BMS to enable the balancing of the battery cells, where all cells will be brought to the same voltage by discharging them through a resistor.
2. BMS_activate_sleep: This signal will force the BMS to enter sleep mode. In sleep mode, all contactors are open.
3. BMS_wake: This signal will wake the BMS up from sleep mode (BMS_activate_sleep has to be inactive for this to have an effect)

Uniting the requirements of the boat with the requirements of the BMS Unfortunately, the Swiss Solar Boat is not structured according to figure 32. In the boat, there are the HP loads (ESCs), the LP loads (electronics, fans, pumps) and the PV/MPPTs. Therefore, some changes had to be made in order to get a working system.

First of all, the integrated pre-charge functionality of the BMS cannot be used, as it is designed to pre-charge the entire system (meaning both the load and the charge), and that is not what is required for our application. In fact, every time the load or the charge is activated in the state-machine, the BMS automatically goes through the pre-charge contactor, which in our case would waste power whenever the PV/MPPTs are connected. In the boat, only the HP load need to be precharged, due to the very large decoupling capacitances of the ESCs (2 mF) that need to be charged up on start-up.

In the boat, we also differentiate between the HP and LP load, which need to able to be controlled separately. In order to do this, it was decided that the BMS only directly controls the LP load, and the HP load, as well as the pre-charge, will be controlled by the MCU of the battery.

In order to save on weight, I[4] also decided not to include the main contactor (LOAD POSITIVE on figure 32), as it serves only redundancy purposes, while adding a SOP to the system. On top of that, it would have to be placed in the main battery loop, and therefore be able to handle up to 360 A of current, so a heavy and expensive relay is necessary. Finally, after discussing this with professor André Hodder, we decided to connect all relays on the high voltage side of the battery, in order to assure a common ground between all subsystems at all times.

[4] in this section, the pronoun I will refer to Mathias Arnold

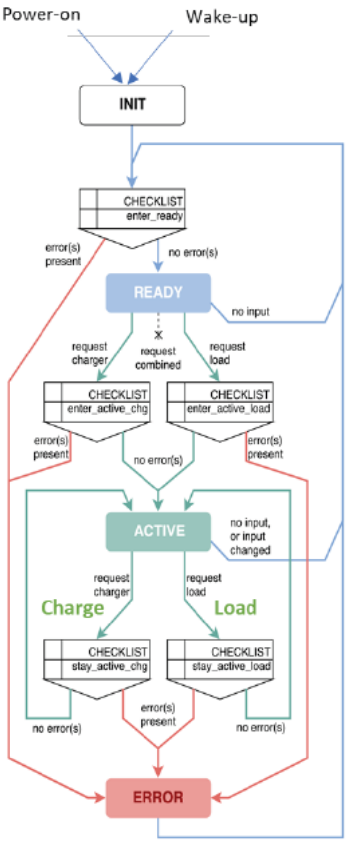


Figure 31: State machine of n-BMS

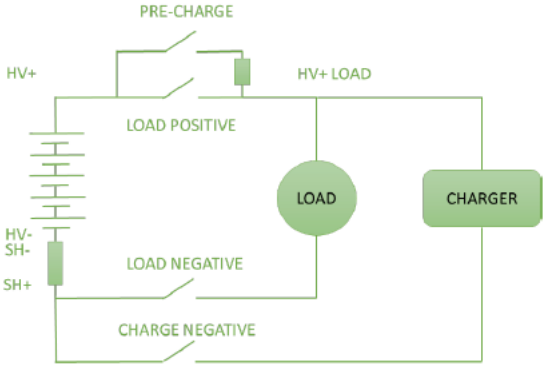


Figure 32: Desired contactor arrangement of n-BMS

Electrical circuit diagram of battery V2 With all of these changes in mind, the finalized high level electrical circuit diagram of the battery (figure 33) was created, where the interactions between the BMS, the MCU and the relays are defined.

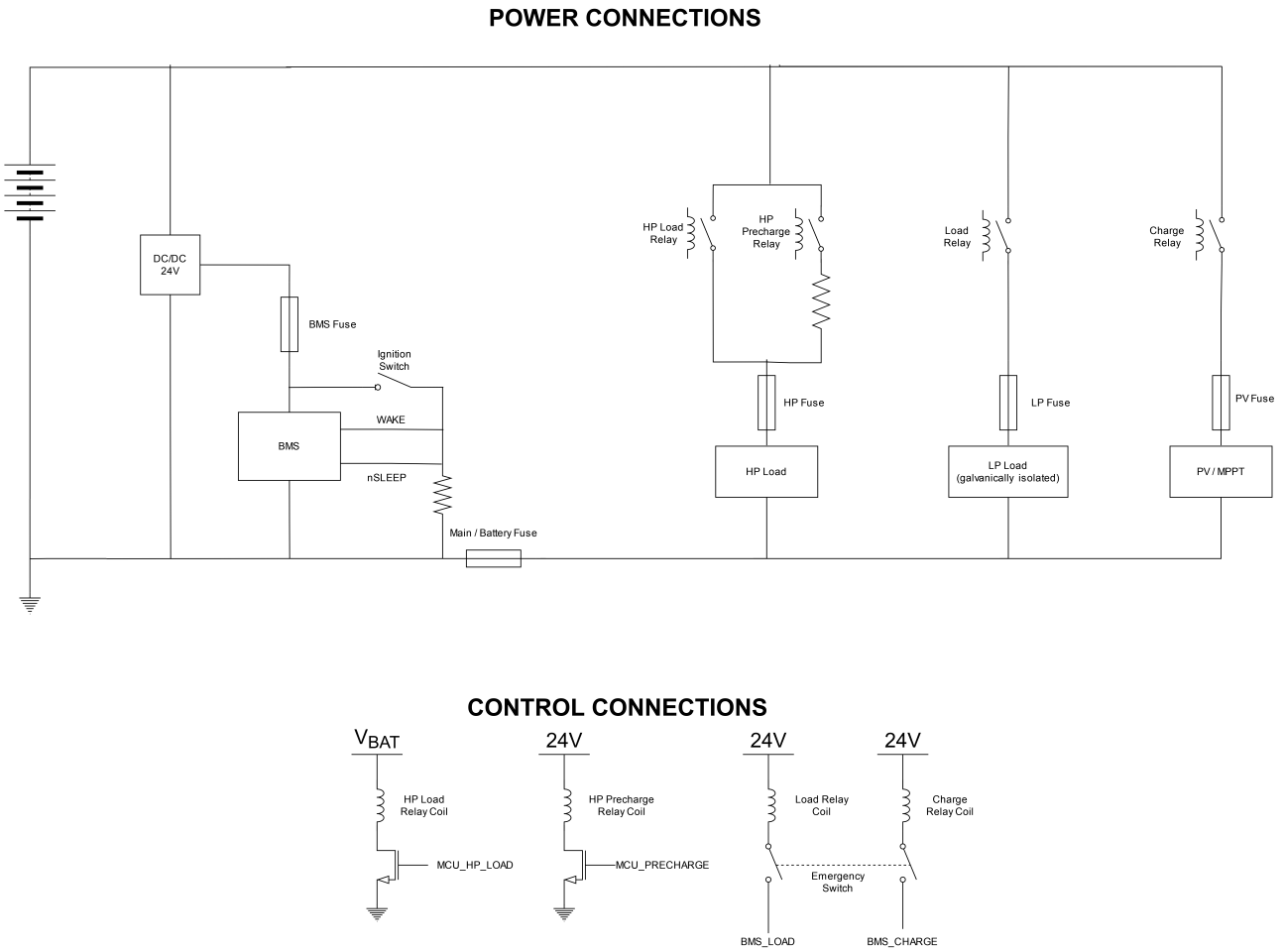


Figure 33: High level electrical circuit diagram of battery V2

The HP load relay and the pre-charge relay are controlled by the MCU through a transistor, the LP load and charge relays are controlled by the BMS. Doing it this way implies that the BMS does not have direct control over the HP load. However, if the BMS deactivates the LP load, the MCU will no longer be powered and the HP load relay will automatically disconnect. Therefore, the BMS does have control over the entire system, albeit indirect.

Additional design choices

- the 24V DC/DC converter used to power the BMS and some of the relays is implemented on a seperate PCB that can be plugged into the relay block. This is done so that the converter can quickly be exchanged in case of failure.
- As the BMS_load signal only controls the LP loads, which should always be on when the boat is turned on via the ignition switch, the BMS_activate_load signal is hardwired to ground (it is low active). This ensures that the BMS will turn on the LP loads on start-up.

- The ignition switch is connected to the BMS_wake and nBMS_sleep signals, in order to wake up the BMS when it is activated. A pull down resistor has to be used to pull both signals to ground when the ignition switch is off.
- BMS_balancing is controlled through the MCU via an inverting optocoupler. This ensures that if the MCU is off, BMS_balancing is high (it is low active) and no balancing takes place unless explicitly requested by the operator.
- 3 additional signal are taken from the MCU and sent to GPIOs on the BMS through optocouplers that can be used for additional control, should it be necessary in the future.

3.3.4 Solution analysis and component choices

Connections All connections to the relay block and their explanations can be found in table 15.

The connectors used are from the JST PA 2.00mm-pitch family for low power (MCU and BMS signals) and the MOLEX Micro-fit 3.0mm pitch family for high power connectors (HP Load relay, battery leads, HP load, LP loads, charge relays, etc).

Relays

- Relays that are not embedded on the relay block PCB, but only controlled by it
  - HP load relay
    - The same relay as in the battery V1 is used, which is the EV200ADANA. It is powered directly by the battery voltage.
  - Charge relay
    - The same relays as in the battery V1 are used, as they are on the MPPT block and were not subject to a redesign for this project. The relays used are 2 G7J-4A-T DC24 and 1 G7J-2A2B-T DC24. They are powered by 24 VDC.
- Relays that are embedded on the relay block PCB
  - I decided to use SSRs for all relays that are embedded on the PCB itself, as the advantages over traditional relays are numerous. SSRs do not have physically moving parts and are therefore less susceptible to shocks and vibrations. They also have a longer lifetime, are less susceptible to electrical noise, switch faster and consume less power than traditional relays. However, there is a small leakage current that has to be considered (on the order of a few µA), as the switching is done by a transistor.
  - LP load relay
    - The LP loads consume a maximum of $P_{max} = 150\,\mathrm{W}$. They consist of all the electronics of the boat, the fans of the battery and the pumps. At the minimum battery voltage, $V_{bat,min} = 30\,\mathrm{V}$, this yields $I_{max} = \frac{P_{max}}{V_{bat,min}} = 5\,\mathrm{A}$. The CN100D24 SSR can handle up to 6 A, which is dangerously close to the maximum current. Therefore, 2 CN100D24 SSRs are used with the same distribution as in the battery V1. One of them handles the electronics and pumps, which consume a maximum of 90 W, and therefore $I_{max} = 3\,\mathrm{A}$, the other one handles the fans, which consume a maximum of 60 W, and therefore $I_{max} = 2\,\mathrm{A}$. This way, there is a big enough security margin.
  - Pre-charge relay
    - The maximum current going through the pre-charge relay is fixed by the pre-charge resistor and is given by $I_{max} = \frac{V_{bat,max}}{R_{precharge}} = 3.3\,\mathrm{A}$, where $R_{precharge} = 15\,\Omega$. This value has been kept from the relay block V1. Therefore, the CN100D24 can also be used for this.

Connector	Name	# pins	Description	Functionality
J1	Battery leads	2	-	Provides V_{bat} and battery ground to the relay block.
J2	Power supply	6	2 pins for V_{bat} 2 pins for 24 V 2 pins for GND	Connector to the separate 24V DC/DC converter PCB. It provides V_{bat} to the converter and receives 24 V.
J3	BMS Power	3	1 pin for 24 V 2 pins for GND	Provides power to the BMS. 2 GND connections are used to sink the relatively high inrush currents that relays can generate from the BMS. This is done in accordance with the n-BMS user manual.
J4	BMS1	5	nBMS_GPIO_IN_0 nBMS_GPIO_IN_1 nBMS_GPIO_IN_2 nBMS_BALANCING nBMS_ACTIVATE_LOAD	nBMS_ACTIVATE_LOAD is hardcoded to GND, as the LP loads should always be active when the BMS is not in sleep mode. nBMS_BALANCING activates the balancing of the battery cells and is controlled by the MCU. The remaining signals are connected to GPIOs on the BMS configured as input and can be used by the MCU for additional control.
J5	BMS2	5	nBMS_ACTIVATE_CHARGE BMS_LOAD BMS_CHARGE nBMS_ACTIVATE_SLEEP BMS_WAKE	nBMS_ACTIVATE_CHARGE allows the BMS to activate the charge relay and is controlled by the MCU. BMS_LOAD drives the LP load relay. BMS_CHARGE drives the charge relay. nBMS_ACTIVATE_SLEEP and BMS_WAKE are connected to the ignition switch. When the ignition switch is closed, the signals are connected to 24V and the BMS will leave sleep mode. When it is opened, both signals are pulled to ground through a pull-down resistor and the BMS enters sleep mode.
J6	Ignition Switch	2	-	Connects to the ignition switch in the cockpit.
J7	HP Load Relay	2	-	Connects to the coil of the HP load relay.
J8	Charge Relays	2	-	Connects to the coils of the charge relays.
J9	HP Load	2	both pins are connected together	Connects to the high potential side of the HP Load through a power resistor in order to precharge the capacitances of the ESCs right after start-up of the boat.
J10	Emergency Switch	4	2 pins for load 2 pins for charge	Connects to the Mushroom type emergency switch button on the outside of the boat. Both the LP load relay (and thus indirectly the HP load relay) and the charge relay are immediately deactivated when the emergency switch is pressed.
J11	LP Load	4	2 pins for LP + Pumps 2 pins for Fans	Provides power and ground to both LP loads, taken after the LP load relay.
J12	MCU1	5	MCU_3V3 MCU_CHARGE_IN MCU_HP_LOAD_IN MCU_PRECHARGE_IN MCU_GND	MCU_CHARGE_IN is optocoupled to 24V and can directly control the charge relay (in parallel with the BMS) if the coressponding jumper on the PCB is set, if this is deemed necessary. MCU_HP_LOAD_IN is optocoupled to 24V and directly controls the HP load relay through a transistor. MCU_PRECHARGE_IN is optocoupled to 24V and directly controls the pre-charge relay through a transistor. 3V3 and GND are provided to control the optocouplers.
J13	MCU2	5	nMCU_GPIO_OUT_0 nMCU_GPIO_OUT_1 nMCU_GPIO_OUT_2 nMCU_ACTIVATE_CHARGE MCU_BALANCING	nMCU_GPIO_OUT_x is optocoupled to 24V and then sent to BMS to control additional GPIOs as BMS input. nMCU_ACTIVATE_CHARGE is optocoupled to 24V and then sent to BMS to let the BMS activate the charge relay. MCU_BALANCING is optocoupled to 24V through an inverting optocoupler and then sent to the BMS to activate the balancing of the cells. The reasoning behind the inverting optocoupler is that the balancing does not take place when the MCU is off.

Table 15: All connections of the relay block

Transistors to drive relays The HP load relay, the pre-charge relay and the charge relay (albeit it only if the jumper is connected) are driven by n-channel MOSFETs. The control signals for these MOSFETS all come from the MCU and are optocoupled to 24V, therefore the MOSFETS need to have V_{GS} ratings that take this into account.

- HP load relay
  - The SIHU2N80AE-GE3 is used for the HP load relay. It can handle a V_{GS} up to 30 V, has a low $V_{GS_{th}}$ and a low $R_{DS,on}$, and can withstand a $V_{DS,max} = 800\,\text{V}$. This is important, as the HP load relay can induce very large voltage spikes when it is turned off, which is explained more in detail in the flyback diode section.
- Pre-charge and charge relay
  - The PMV213SN,215 is used for these two relays. It can handle a V_{GS} up to 30 V, has a low $V_{GS_{th}}$ and a low $R_{DS,on}$, and can withstand a $V_{DS,max} = 100\,\text{V}$. A transistor with similar ratings was already used in the relay block V1, and it worked as intended.

Optocouplers I chose optocouplers that have a totem pole output, as we want the output to either be connected to 24 V or GND. This is because they either drive a MOSFET or are connected to a GPIO of the BMS. For the normal (non-inverting) optocoupler, the TLP5705H(D4-TP,E was chosen. It allows an output peak current of up to 5 A, which is largely sufficient to drive MOSFET gates or GPIOs of the BMS. However, it requires an input current on the diode side of 6.5 mA to 10 mA. The MCU pins can provide a maximum of 8 mA, which is too close for comfort. To solve this problem, the anode on the input side is connected to MCU_3V3 through a resistor and the cathode is connected to an N-channel MOSFET, the gate of which is connected to the actual MCU pin, as illustrated in figure 34.

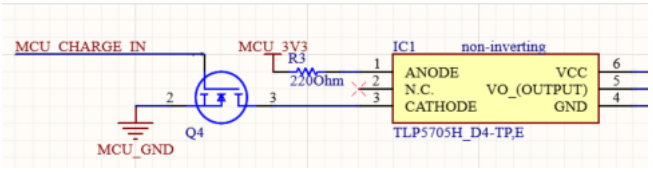


Figure 34: Example of optocoupler connection

The MOSFET used for this purpose is the SSM3K36TU,LF. It has a low $R_{DS,on}$ and a low $V_{GS_{th}}$ which is necessary for this application, as it is only driven by 3.3 V on the gate.

The purpose of the resistor is to limit the current to the operating range of the optocoupler. From the datasheet of the TLP5705H(D4-TP,E, the forward voltage of the diode is typically $V_F = 1.55\,\text{V}$ and the current has to be between 6.5 mA and 10 mA. In order to leave some room for the PVT dependance of the diode forward voltage, the current in the middle of this range is chosen as the target current, $I_{target} = 8.25\,\text{mA}$ Thus, the resistor can be calculated as follows:

$$R_{opto} = \frac{V_{cc,MCU} - V_F}{I_{target}} = 212\,\Omega$$

A standard 220 Ω resistor is chosen for this.

For the inverting optocoupler (used for MCU_Balancing as explained previously), the ACPL-M483-500E is chosen. It was already used in the relay block V1 for this purpose. It once again requires an input current that is to close to the maximum that MCU can provide on its pins, between 4 mA and 7 mA, so the same transistor configuration as in figure 34 is used. However, the resistor value is different. From the datasheet

of the ACPL-M483-500E, the forward voltage of the diode is typically $V_F = 1.5\,\text{V}$ and the current has to be between 4 mA and 7 mA. In order to leave some room for the PVT dependance of the diode forward voltage, the current in the middle of this range is chosen as the target current, $I_{target} = 5.5\,\text{mA}$ Thus, the resistor can be calculated as follows:

$$R_{opto} = \frac{V_{cc,MCU} - V_F}{I_{target}} = 327\,\Omega$$

A standard $360\,\Omega$ resistor is chosen for this.

Both optocouplers used recommend putting a 0.1 µF capacitor between the output V_{DD} and GND to smooth out the output, which I did for all of them.

Flyback diodes

- Traditional relays
  - To understand the necessity of flyback diodes for traditional relays, it is important to understand how they work. A relay is a switch that is controlled by a magnetic field which comes from an inductor. When there is a current flowing through the inductor, the magnetic field created in it pushes the switch into the closed position and current can flow (in case of normally open relay, which is what is used at SSB). However, the inductor is governed by the equation $v = L\frac{di}{dt}$, where v is the voltage across the inductor, i the current through it and L its inductance. Once the relay is turned off (set to open), the current abruptly falls to 0, which therefore causes a large voltage spike (large $|\frac{di}{dt}|$). This needs to be adressed, as otherwise components surrounding the relay can break. In order to do this, a flyback diode is used from the negative side of the relay coil to the positive. The voltage spike can pass through the diode and dissipate through the resistance of the coil.
  - HP load relay
    - The STTH6010WY diode is used to protect the HP Load relay. It can handle a high current and high temperatures, which is why it was an ideal choice.
  - Charge relay
    - The BAS16,215 diode is used to protect the Charge relays. It can handle a reverse voltage of up to 100 V, which largely sufficient for the 24 V across the relay coil.
- Solid state relays
  - SSRs are driven by a diode on the input and the ones chosen in this design already have an internal flyback diode. However, according to figure 35 taken from the datasheet of the CN100D24, there has to be a flyback diode across the load of the SSR.
  - For all 3 SSRs (2 LP load relay and pre-charge relay) the STTH6010WY has been chosen, at it is very robust and powerful, with a low leakage current.

24V DC/DC converter As explained previously the 24V DC/DC converter is placed on a separate, small PCB for quick and easy replacement in case of failure.

In total, the DC/DC Converter needs to be able to supply

$$I_{tot} = I_{BMS} + I_{Charge\,relay} + I_{LP\,load\,relay} + I_{Pre-charge\,relay} + I_{optocouplers} + I_{Pull\,down\,ignition}$$

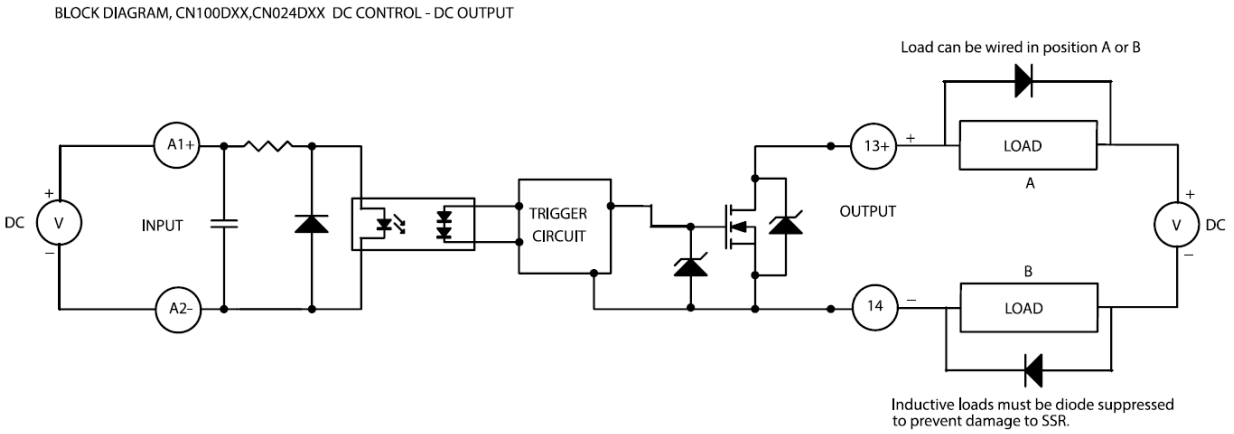


Figure 35: Example of optocoupler connection

Using the datasheets of these components, we have that

$$
\begin{aligned}
I_{BMS} &= \frac{P_{max,BMS}}{V_{BMS}} = \frac{3.8\,\mathrm{W}}{24\,\mathrm{V}} = 160\,\mathrm{mA} \\
I_{Charge\,relay} &= 2I_{\text{G7J-4A-T DC24}} + I_{\text{G7J-2A2B-T DC24}} = 2 \cdot 83\,\mathrm{mA} + 83\,\mathrm{mA} = 249\,\mathrm{mA} \\
I_{LP\,load\,relay} &= 2I_{CN100D24} = 16\,\mathrm{mA} \\
I_{Pre-charge\,relay} &= I_{CN100D24} = 8\,\mathrm{mA} \\
I_{optocouplers} &= 7I_{\text{TLP5705H(D4-TP,E}} + I_{\text{ACPL-M483-500E}} = 7 \cdot 3\,\mathrm{mA} + 3\,\mathrm{mA} \qquad = 24\,\mathrm{mA}
\end{aligned}
$$

and thus,

$$I_{tot} = 457\,\mathrm{mA}$$

.
Due to the lack of availability of automotive grade buck converter regulator ICs because of the ongoing semiconductor shortage, it has been decided to use a premade chip, the RPMH24-1.5 made by RECOM Power. It can provide up to 1.5 A at 24 V for an input voltage range of 26 V to 60 V at an efficiency of up to 92%, which is ideal for this application.

Several passive components were used alongside the IC in order to smooth out the input and output voltages, according to the recommendations in the "EMC filtering suggestion according to EN55032" section on page 8 of the datasheet.

Fuses The relay block and 24V DC/DC converter PCBs embed several fuses in order to prevent short-circuits.

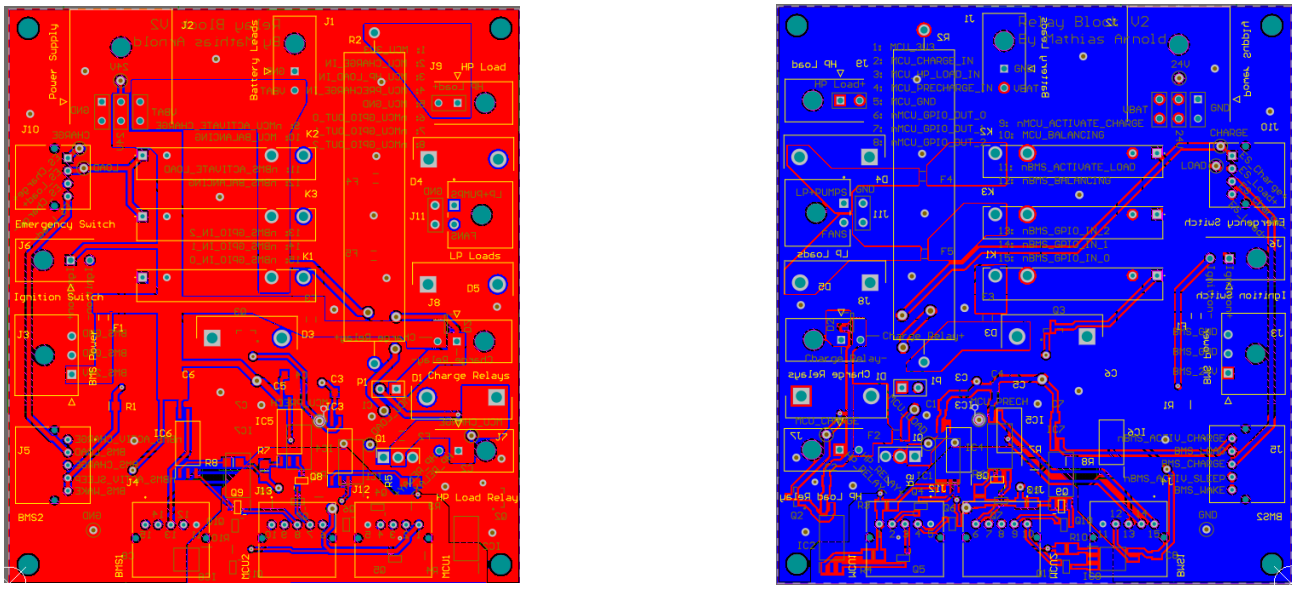
- 24V DC/DC converter PCB
  - A 2.5 A fast acting fuse was put on the output in order to protect the IC.
- relay block PCB
  - BMS Fuse
    - As discussed previously, the BMS uses 160 mA of current, so a 250 mA fuse was put on the 24V connection to BMS to protect against short circuits. However, this proved to be too small during testing, which is most likely due to high inrush currents during start-up to charge some capacitors on the BMS. It was replaced with a 1.5 A fuse.

- HP load relay Fuse
  - The HP load relay is the only component on the PCB that is directly connected to V_{bat} (besides the DC/DC converter, of course), therefore it also needs to be fused. According to its datasheet, it has an inrush current of up to 1.3 A, so a 2 A fuse was chosen.
- LP load fuses
  - As the LP load remains identically connected as on the relay block V1, the same fuses are used. There is a 4 A for the electronics and pumps and 3 A fuse for the fans.
- Pre-charge fuse
  - The maximum current through the pre-charge line is 3.3 A. Therefore, a 5 A fuse was used.

Final schematics of the relay block 24V DC/DC converter Please refer to appendix C for the full schematic of both circuits.

3.3.5 PCB Design

After designing the schematic of the relay block and 24V DC/DC converter PCB, the next step was to design the PCB layout. The relay block PCB has dimensions of 80mm by 90mm, and a thickness of 1,6mm. The PCB could have been smaller but the routing for all the optocouplers would have been much more complicated, and size was not a limiting factor in the design. All through hole components were mounted on one side, the surface mount components are distributed across both sides. The 2D design of the relay block PCB is shown in figure 36.

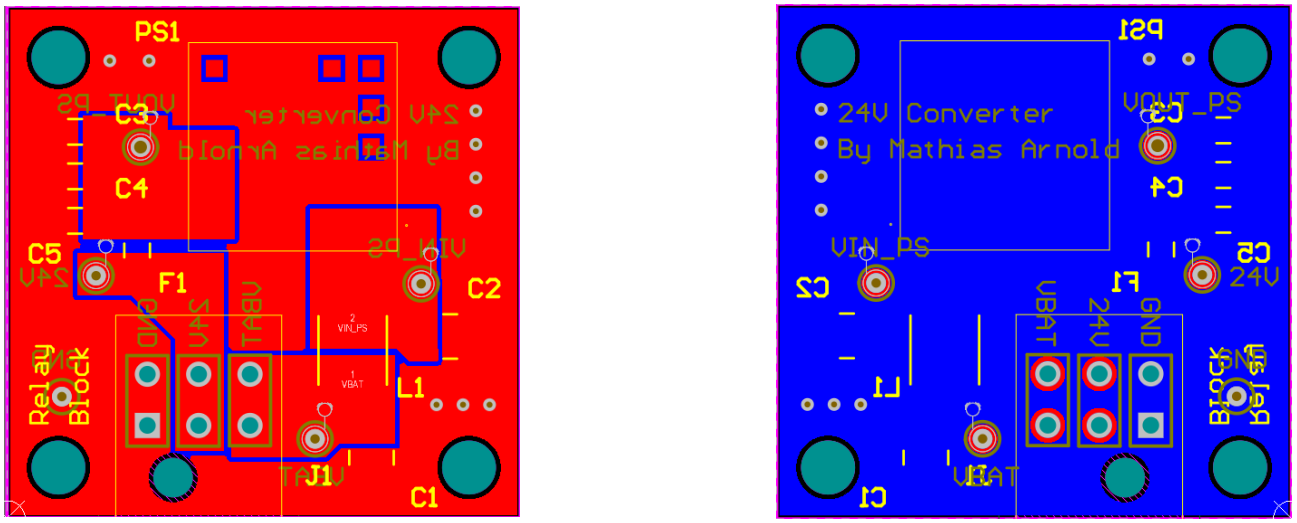


(a) Top face

(b) Bottom face

Figure 36: 2D face of the relay block

The 24V DC/DC converter PCB has dimensions 30mm x 30mm. All components are mounted on the top side. It connects directly to the relay block PCB. The 2D design of the 24V DC/DC converter PCB is shown in figure 37.



(a) Top face

(b) Bottom face

Figure 37: 2D face of the relay block

For the design of the PCB, it is important to take care of different elements such as current in the traces, current path, interference and others. Thus when designing the PCB I took care of these different elements. The choices made to optimise the relay block and 24V DC/DC converter are detailed below.

1. Ground plane
   In order to reduce electrical noise and interference from ground loops, two ground planes were placed on both layers. A ground plane for the micro-controller ground and a ground plane for the ground of the battery. Figure 36.b shows the distribution of the two ground planes on the bottom side of the PCB.

2. Trace size
   The size of the traces was assessed according to the current flowing through them. For digital signals the traces are set to 0.254mm, for power supplies coming from the MCU the traces are at least 0.8mm thick, and most of the time thicker polygon pours were used. The trace sizes have been checked using a trace thickness calculator based on the current flowing through it and the thickness of the PCB.

Figure 38 shows the final assembled relay block. Figure 39 shows the final assembled 24V DC/DC converter.

3.3.6 Test

Tests were carried out to check the correct operation of the relay block and 24V DC/DC converter. They are summarized in table 16.

3.3.7 Errata

During the schematic design of the relay block PCB, I accidentally switched up the MCU_CHARGE and MCU_LOAD signals when connecting them from the MCU_Connection block to the Relays block in the top level schematic (see figure 40).

This means that the pinout noted on the PCB is slightly wrong. In fact, the pin called MCU_CHARGE_IN controls the HP load relay, and the pin called MCU_HP_LOAD_IN controls the charge relay (if the corresponding jumper is connected). As this can easily be fixed in software when programming the MCU, I decided not to reorder a PCB for this sole reason. Sebastien Delsad, the person who is programming the battery MCU has been informed. The mistake was also noted explicitly on the back of the relay block PCB.

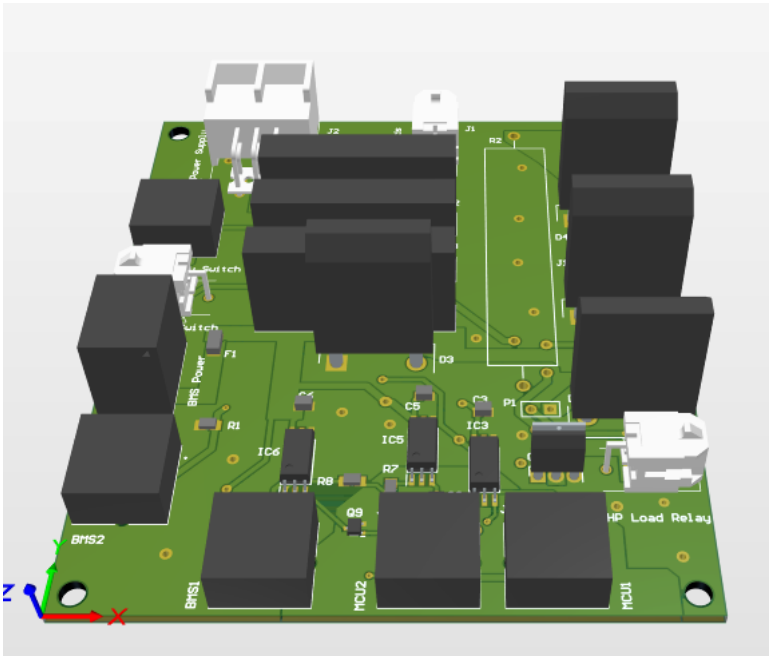


Figure 38: Top face of the relay block

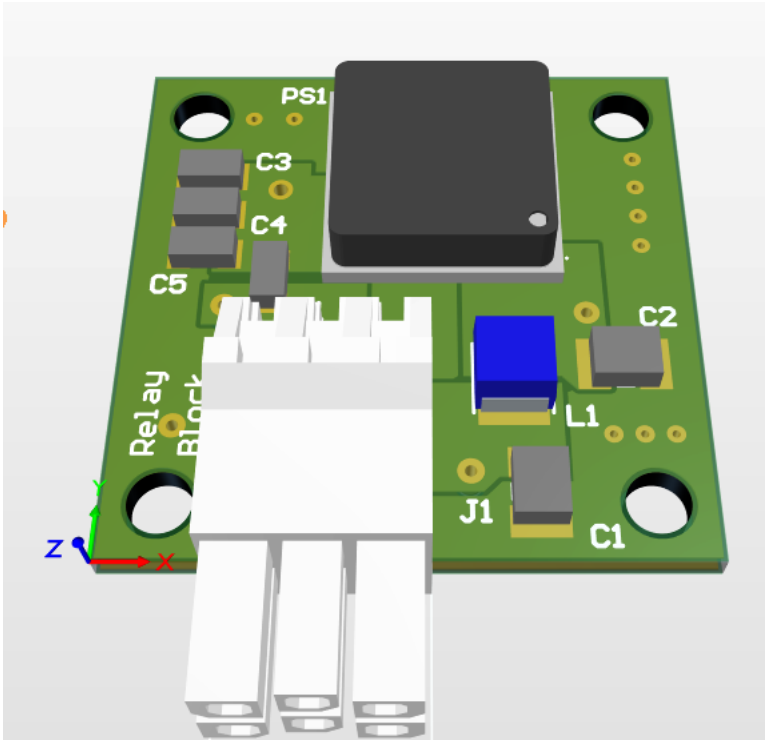


Figure 39: Top face of the 24V DC/DC converter

Part to test	Details of the test
Checking connections	A multi-meter is used to check that the elements connected to the same potential are well connected.
24V DC/DC converter	The battery leads on the relay block are connected to a 30V DC lab power supply. Verifying with a multimeter showed that the DC/DC converter correctly outputs 24V
Optocoupler outputs	Using a microcontroller to connect 3.3V and Ground to the corresponding pins on the MCU1 connector and then manually connecting all input signals to either 3.3V or GND showed that all optocouplers work as intended and the output signals are at 24 V or GND respectively.
Relay tests	Using the MCU_HP_LOAD_IN, BMS_LOAD and BMS_CHARGE, I tested that the relays are in their correct state and lit up a small LED in series with a resistor across it to verify this. Everything worked as intended, except for one mistake i did during the schematic design phase, which is explained below.

Table 16: Details of the tests performed to verify the functioning of the relay block and 24V DC/DC converter

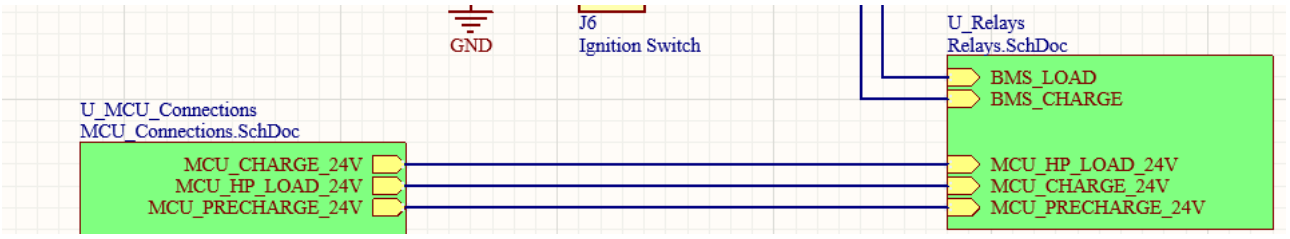


Figure 40: Error in schematic design

3.4 Microcontroller

The V2.2 microcontroller also has a role to play in the V2 battery. Indeed, the n-BMS chosen for the latter communicates on a CAN line and we did not want it to do so directly on the CAN line of the boat. This is where the microcontroller with its two separate CAN lines comes into play. On one line it reads the data sent by the BMS, i.e. the different battery logs we want to retrieve, and on the other line it handles the communication between the battery and the rest of the boat. Moreover, the BMS cannot control all the relays that we have planned in our state machine. It is therefore up to the microcontroller via the relay block to control the precharge of the High Power Load, as well as to activate the contactor of the High Power Load, when requested on the CAN line of the boat. In addition, all the functions of the monitoring block go through the microcontroller. It controls the modulation of the battery cooling fans. Thanks to the differential I2C line installed on the monitoring block, it can read the values of the MPPT rack current sensors.

To summarize the tasks of the microcontroller in the V2 battery are :

- Communicate with the boat through the CAN line
- Retrieve logs from the BMS via the BMS I-CAN
- Control the activation of the charge relay by the BMS
- Control the activation of the cell balancing by the BMS
- Control the high power load relays (precharge relay and main contactor)

- Give the modulation for the control of the colling fans
- Read the values of the MPPT rack current sensors through the differential I2C line.

The programming of the battery microcontroller has been entrusted to Sebastien Delsad and is at the time of writing still in progress.

4 PCBs production

Once we had designed our PCBs and manufactured them, we were able to move on to the production phase. We were able to take advantage of the new DLL and dedicated space for MAKE projects as well as their brand new electronics fabrication room. A room equipped with a variety of brand new soldering irons, magnifying glasses, binoculars, etc. In short, everything we needed. There is also all the necessary machinery for PCB assembly by solder reflow, including a vapour phase furnace and a manual pick and place. We preferred this solution for most of the assembly because it is much faster, especially for mass production, in particular of the 10 V2.2 microcontroller PCBs, but also because the quality of the assembly is significantly better.

The first step in the manufacture of PCBs is to clean them thoroughly with Isopropyl Alcohol (IPA). This removes all dirt and grease. This makes the rest of the assembly easier and prevents short circuits.

Figure 41: Cleaning of a PCB with IPA

When soldering in a vapour phase furnace, it is necessary to have placed the components to be soldered on solder paste in the appropriate place. The first step is to place the solder paste only where there is a footprint for a component. The solder paste is then screen-printed with a stencil, in exactly the same way as for textile serigraphy. The stencil is supplied by the PCB manufacturer and is made from the PCB design file. It is important to be precise when laying down the paste. The stencil must be well aligned with the PCB. The DLL is equipped with a machine for this purpose. Furthermore, the thickness of the paste layer must be consistent and finely adjusted. This parameter is chosen by selecting the thickness of the stencil. Despite the precision this step requires, there is room for error, depending on the type of paste used, the oven cycle and the footprints

of the components. As a rule of thumb, it is estimated that up to 20% error in the solder paste deposit is not a problem.

Figure 42: Solder paste application with Stencil

Once the solder paste has been applied to the PCB, the components must now be placed. Of course, you can only do one side at a time. For PCBs with components on both sides, we start with the side with the lightest components, which we will secure with kapton tape when the second side goes into the oven.

The pick and place is the dedicated tool. It allows to quickly and precisely place a component. An important step in order not to waste too much time and not to make mistakes is to prepare your components in advance, to label them well and to follow as much as possible a logical order when placing them. This step is quite time consuming and thankfully we were able to count on the help of many SSB members.

Figure 43: Component placement with a manual pick and place

The vapour phase furnace and the pick and place are suitable for production in serie. For the microcontroller V2.2 assembly, we proceeded in batches of 5.

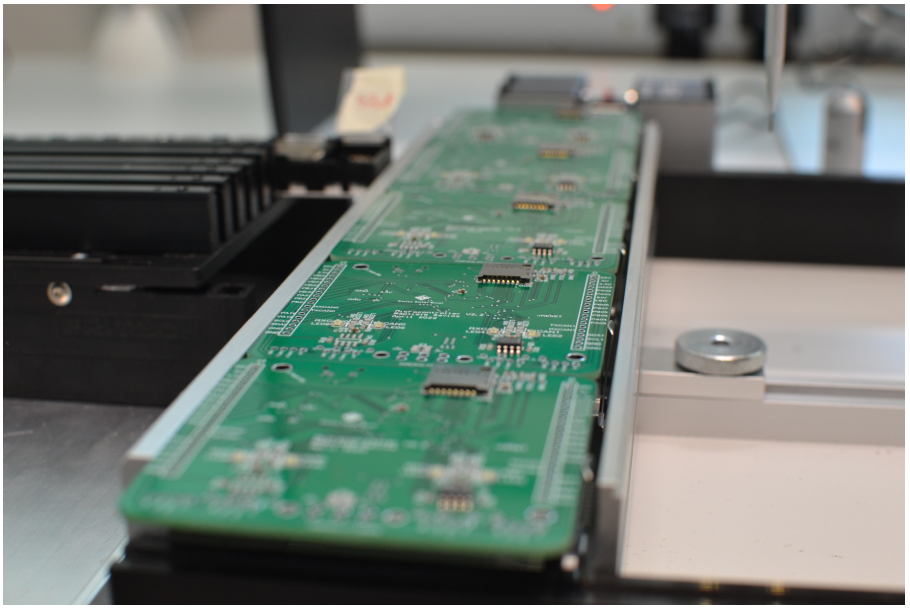


Figure 44: Microcontroller production by batches

Once the components have been placed on the paste, they still need to be soldered in the vapour phase oven. The PCBs are placed on a plate which the oven will pick up before starting its soldering cycle. The cycles can be more or less elaborate, but in our case, the basic cycle with a temperature of 243 degrees was perfectly suitable.

Figure 45: reflow oven used to solder most of the components

The vapour phase furnace can unfortunately only solder SMD components, so there are some components that we still have to do by hand, like all the through-hole components for example. We were well equipped and accompanied to the DLL electronics workshop to do this.

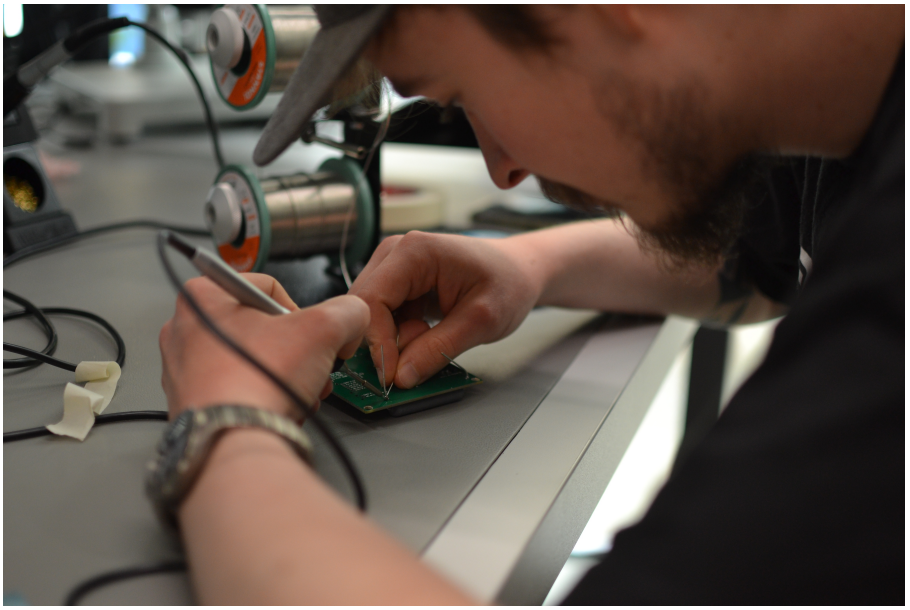


Figure 46: hand soldering of the relay block

This results in the 3 PCBs shown in figure 47, 48 and 49 perfectly soldered. We can now proceed to the tests phase of these boards.

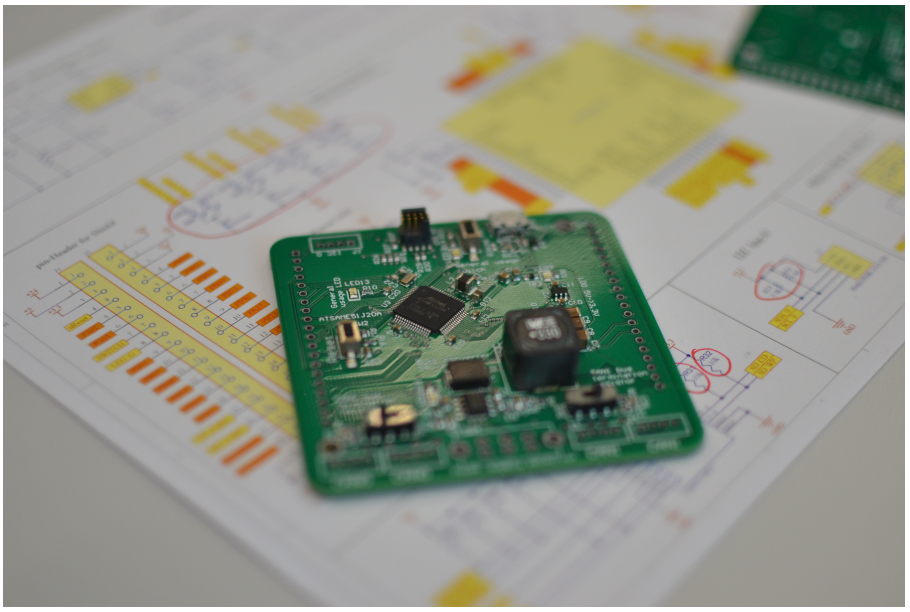


Figure 47: Picture of the assembled MCU

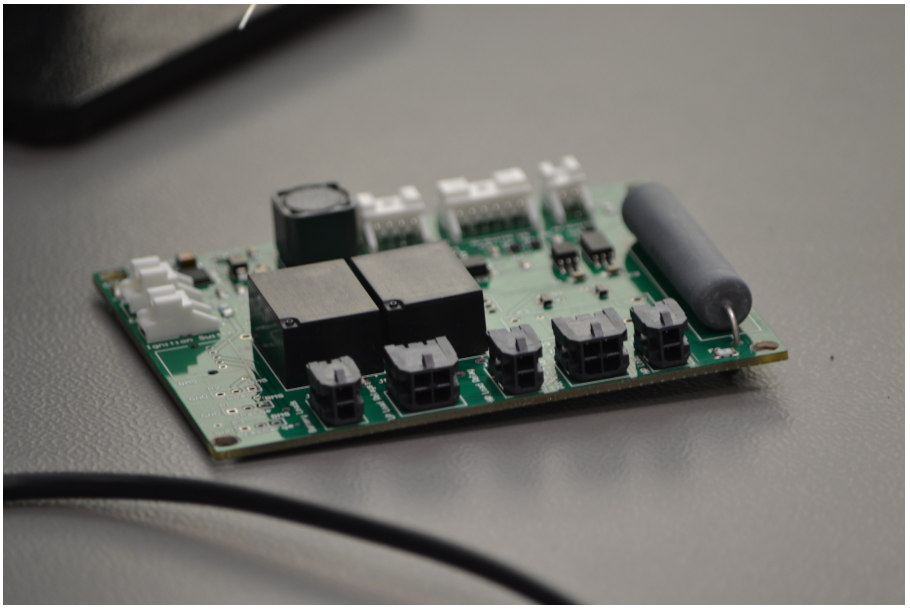


Figure 48: Picture of the assembled Relay block

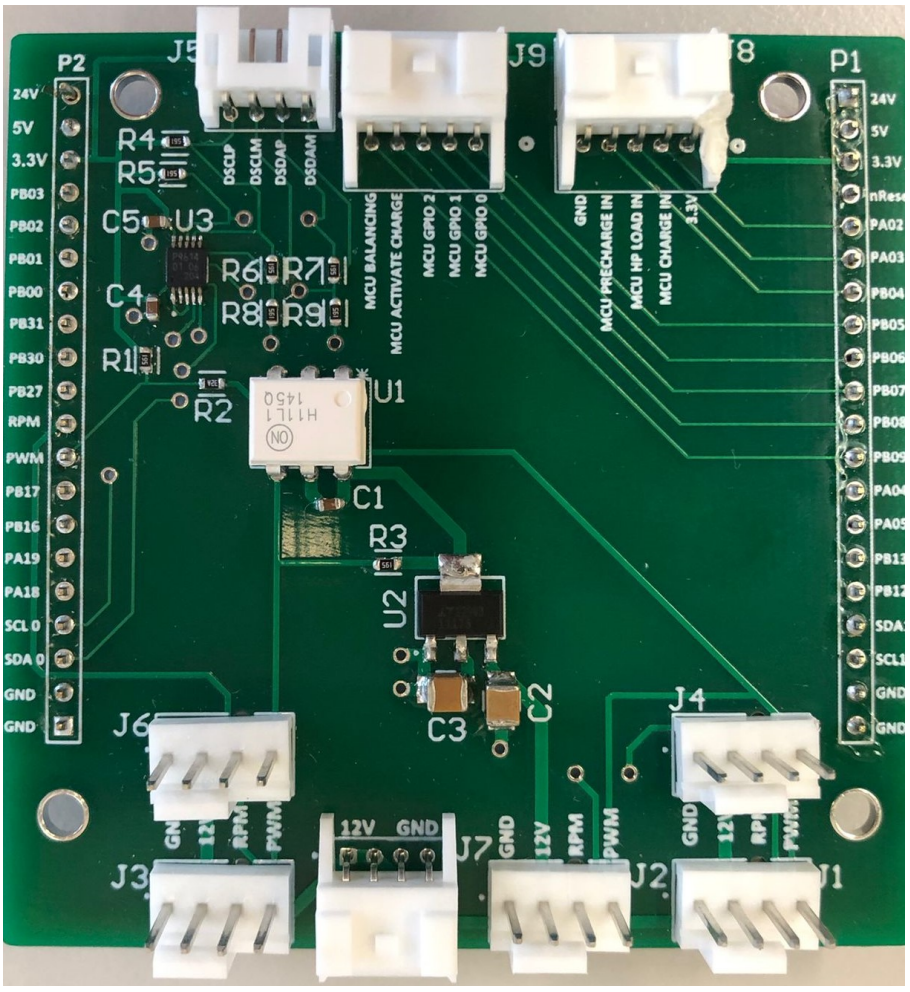


Figure 49: Picture of the assembled Bloc monitoring

5 Conclusion

To conclude, the PCBs design and production project, all the PCBs have been produced and are working. Sometimes not exactly as planned but they are working and meeting the requirements. We'd consider this as a first success. The V2 battery is well on its way to being implemented. Electrical tests need to be conducted in order to thoroughly test to the electronic in every possible configuration. In terms of timeline, high power discharge tests will be conducted in the following week and we expect to have everything ready and functioning two week before the Monaco Energy Boat Challenge 2022.

Moreover the project was not only about our work on the PCBs but also about our involvement in the association. Indeed, we quickly formed a team and we participated in the debugging of the boat, the tests on the lake and the development of the software. Blanche passed her boat licence and will be the second pilot of the Dahu for the competition. Baptiste also has been selected to participate in the competition and will be in charge of the battery.

Finally, both from a technical and human point of view, we consider this project a success. There are mistakes in our design. Fortunately, they do not affect the final results, but more importantly, they are a lesson to be learned. We are now looking forward to the future and the first results of the V2 battery. In the same way we want to see the applications of the microcontroller develop and why not see a next project that will put an end to the programming difficulty.

The next step is a victory in Monaco!

Acknowledgement

Finally, the project experience at SSB is not only a technical challenge but also a human experience of teamwork. We would like first like to thank the entire SSB community for their welcome and support and without whom this project would not have been the same. Secondly, we would like to thank Simon Dorthe in particular for his constant commitment and for the passion he brings to his work. René Beuchat, our supervisor, who pushed us to go beyond our limits and who always gave us good advice. Alix Louvet, Elsa Musy and Thomas Bonnaud from the battery team with whom we immediately got on well and without whom the hours of testing and planning would have been very long. Felix Schmeding for his help with the CAN line tests and the general functioning of the boat as well as for his good mood. André Hodder for his help, never in office hours, with the V2 battery diagram. Benoît Prudhomme-Lacroix, Gaspard Besacier, Mehdi Kirchen and Nour Lachat for coming to help us with the PCB assembly. Sylvain Hauser for his precious advice and experience and for turning a blind eye to our extensive use of the facilities. Sébastien Delsad who didn't count his hours and who gave 200% to the battery code. Richard Ismer for his help with MPLAB. Simon Tychyj for his reassuring presence. Baptiste Ranglaret from the Racing Team for his help with the BMS configuration. Robin Amacher for his supportive supervision. One team, One Boat, One Goal

Appendix A Microcontrolleur documentation

Table 17: Exhaustive list of components for Microcontrloller V2.2

	Name	Component	package	Description
1	C1, C7, C8, C9	capacitor 47uF	1210	
2	C2, C16, C19	Capacitor 10µF	0805	
3	C3	Capacitor 8.2nF	0603	
4	C4, C13-15, C17, C20, C23-28	Capacitor 100nF	0603	
5	C5	Capacitor 10pF	0603	
6	C6	Capacitor 1nF	0603	
7	C10, C11, C12	Capacitor 1µF	0603	
8	C18	Capacitor 4.7µF	0805	
9	C21, C22	Capacitor 18 pF	0603	
10	C29	Capacitor 4.7nF	0603	
11	D1	DIODE SCHOTTKY 40V 3A	DO214AB	
12	D2, D3	Schottky Diodes 1.0 Amp 40 Volt		MSS1P4-M3/89A
13	FB1	Chip Ferrite Bead 220Ω @ 100MHz	0603	BLM18SG221TN1D
14	J1, J2	Shield Connector 20pos		ESQ-120-14-G-S
15	J3, J4, J5, J6, J7	Can connector	S04B-PASK-2-LF-SN	
17	J8	Micro-USB B connector		105017-0001
18	J9	SWD connector		HDR 10 POS 1.27mm SMD
19	L1	Power Inductor 33µH	WE-PD	7447709330
20	L2	SMD Inductors 10uH	1210	CBC3225T100MR
21	LED1-8, LED12-13	Green LED	0805	If 2mA, Vf 1.9V
22	LED9-11	Red LED	0805	If 2mA, Vf 1.75V
23	R1	Resistor 169kΩ	0603	1%
24	R2	Resistor 19.6kΩ	0603	1%
25	R3	Resistor 52.3kΩ	0603	1%

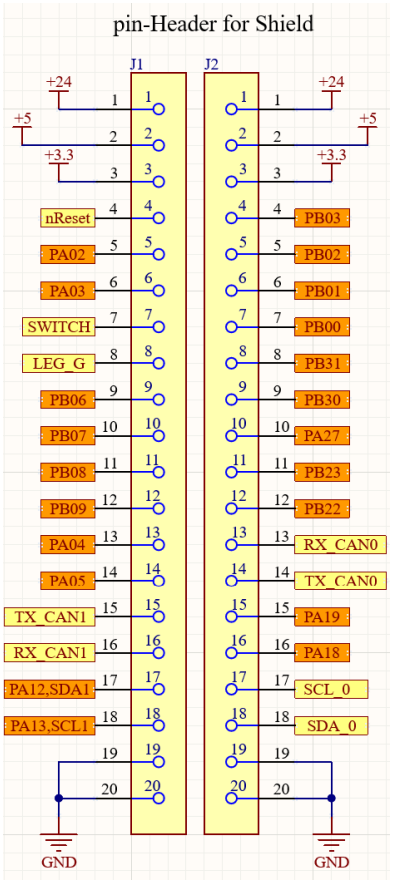
Continued on next page

Table 17: Exhaustive list of components for Microcontrloller V2.2 (Continued)

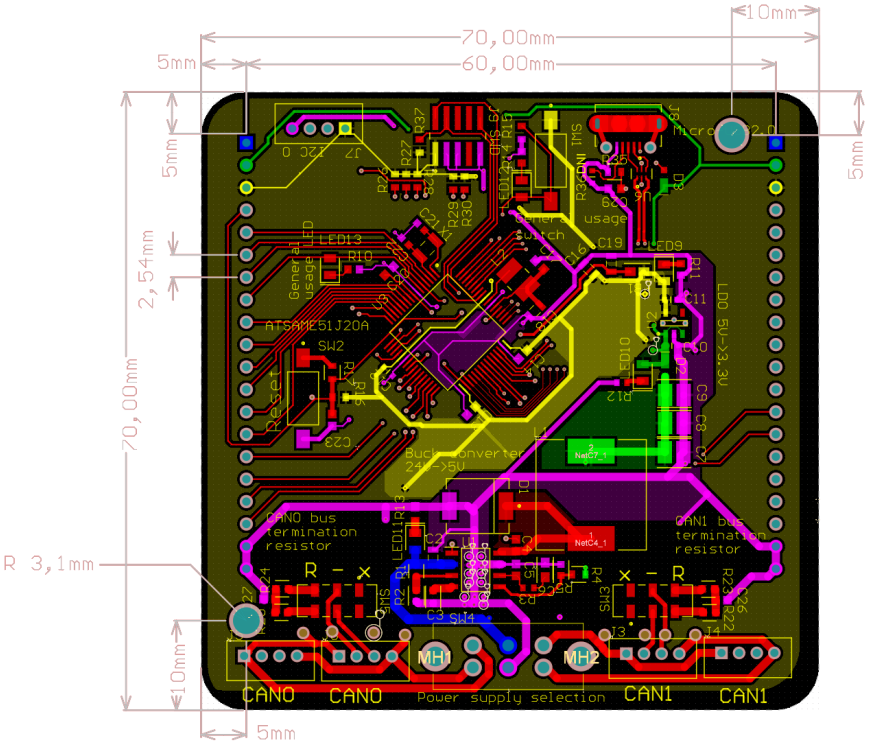
	Name	Component	package	Description
26	R4	Resistor 10.2kΩ	0603	1%
27	R5	Resistor 1.96kΩ	0603	1%
28	R6-11, R14	Resistor 820Ω	0603	
29	R12	Resistor 1.6kΩ	0603	
30	R13	Resistor 12kΩ	0603	
31	R15-16, R18, R20, R26-32	Resistor 10k	0603	
32	R17	Resistor 330Ω	0603	
33	R19, R21, R33-34	Resistor 3.9k	0603	
34	R22-25	Resistor 59Ω	0603	1%
35	R35	Resistor 1MΩ	0603	
36	R36	Resistor 0Ω	0603	Do Not Install
37	R37	Resistor 1k	0603	
38	SD0	Micro SD Card holder		Molex 104031-0811
39	SW1, SW2	Push Button Switch		506-FSMSM
40	SW3, sw5	Switch DPDT		JS202011JCQN, 300MA 6V
41	SW4	Switch DPDT		MFS201N-9-Z, 1A 125VAC 30VDC
42	U1	Buck converter	SOIC8	TPS54233QDRQ1
43	U2	LDO 3.3V 0.6A	SOT25	AP7365-33WG-7
44	U3	Microcontroller	64-Pin TQFP	ATSAME51J20A-AF 32-Bit ARM Cortex-M4F RISC 1MB Flash
45	U4-5	CAN tranciever	8-SOIC	TCAN337GDR
46	U6	USB transient protection	6UQFN	USBULC6-2M6
47	X1	Crystal 32.768 kHz 7pF		ABS07-166-32.768KHZ-T

name	**Functionality**	**Type**	**Number of pins**	**Numbers of connectors**
J1-2	Shield connection	Pin header female (2.54mm pitch)	20 pos	2
J3-6	CAN connection	S04B-PASK-2(LF)(SN)	4 pos	4
J7	I2C connection	S04B-PASK-2(LF)(SN)	4 pos	1
J8	USB2.0 connection	Molex 105017-0001	5 pos (+ 6 shield)	1
J9	SWD connection	SMTC-FTSH-105-01-L-DV-K_V	5x2 pos	1

Table 18: Complete list of Connectors



(a) Shield Pinout



(b) Pin headers mechanical dimension

Figure 50: Pin headers connections for shields on the Microcontroller V2.2

Table 19: Pinout for the Microcontroller V2.2

	MCU Pin	Arduino Pin Name	comment
1	PA00		XIN32
2	PA01		XOUT32
3	PA02	PA02 / A0 / DAC0	on the shield
4	PA03	PA03 / A5 / AREF	reference voltage for ADC can be choosed from that pin, but never higher than 3.3V, on the shield
5	PB04	PB04 / SWITCH_BUILTIN	general usage Switch and LED connected to the pin, on the shield
6	PB05	PB05 / LED_BUILTIN	General usage LED connected to the pin, on the shield
9	PB06	PB06	on the shield
10	PB07	PB07	On the shield

Continued on next page

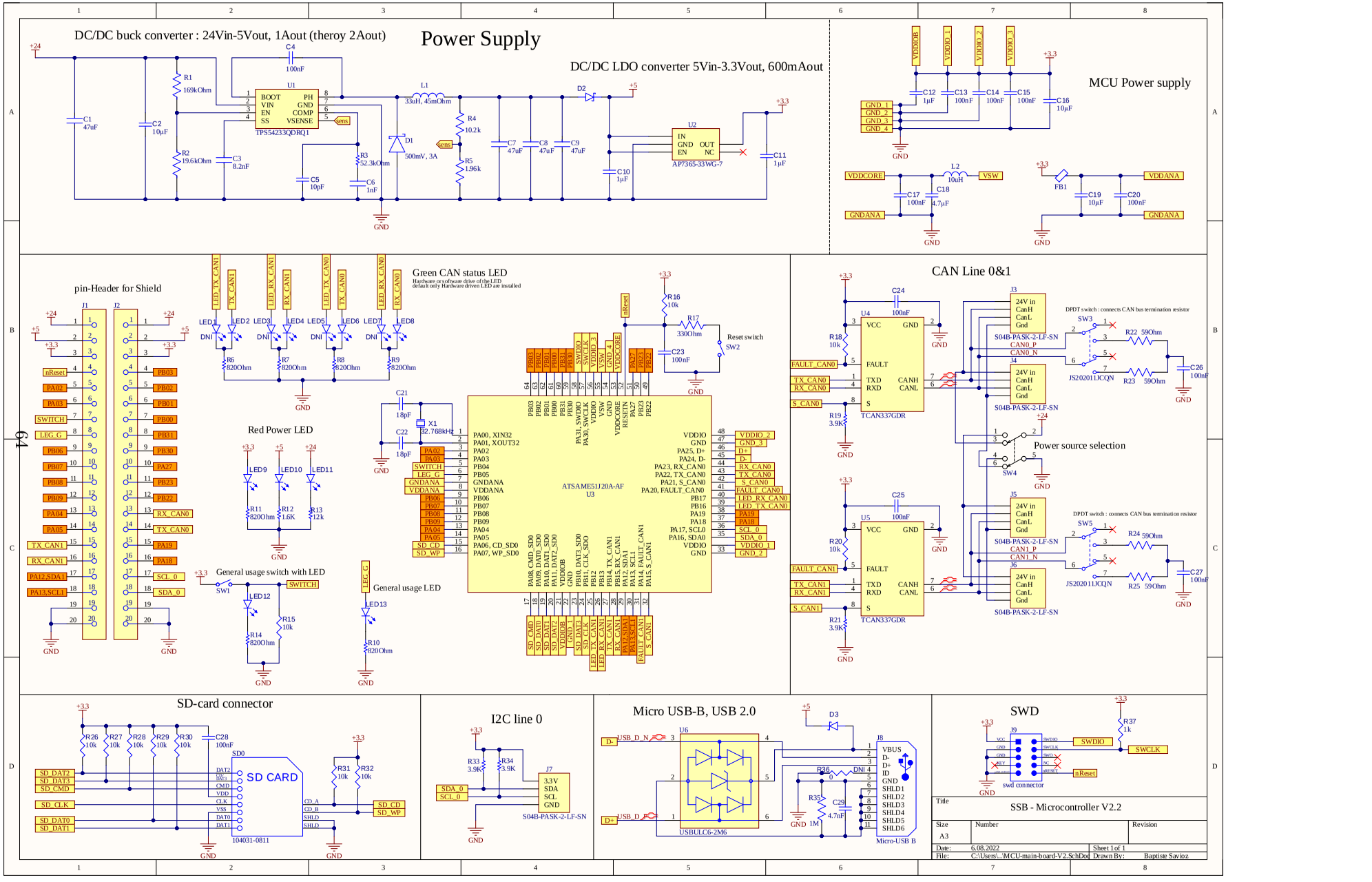
Table 19: Pinout for the Microcontroller V2.2 (Continued)

	MCU Pin	Arduino Pin Name	comment
11	PB08	PB08 / A2	On the shield
12	PB09	PB09 / A3	On the shield
13	PA04	PA04 / A4	On the shield
14	PA05	PA05 / A1 / DAC1	On the shield
15	PA06		CD SD0
16	PA07		WP SD0
17	PA08		CMD SD0
18	PA09		DAT0 SD0
19	PA10		DAT1 SD0
20	PA11		DAT2 SD0
23	PB10		DAT3 SD0
24	PB11		CLK SD0
25	PB12	PB12 / LED_TX_CAN1	by default LED not mounted on the board
26	PB13	PB13 / LED_RX_CAN1	by default LED not mounted on the board
27	PB14	TX_CAN1 / TX_CAN	TX CAN 1 pin, On the shield
28	PB15	RX_CAN1 / RX_CAN	RX CAN 1 pin, On the shield
29	PA12	PA12 / SDA1	SDA I2C 1 pin, on the shield, no pull up resistor mounted
30	PA13	PA13 / SCL1	SCL I2C 1 pin, on the shield, no pull up resistor mounted
31	PA14	PA14 / FAULT_CAN1	Fault pin for CAN1, input active high
32	PA15	PA15 / S_CAN1	Silent mode pin for CAN1, output active high
35	PA16	PA16 / SDA0	SDA I2C 0 pin, on the shield
36	PA17	PA17 / SCL0	SCL I2C 0 pin, on the shield
37	PA18	PA18	On the shield
38	PA19	PA19	On the shield
39	PB16	PB16 / LED_TX_CAN0	by default LED not mounted on the board
40	PB17	PB17 / LED_RX_CAN0	by default LED not mounted on the board
41	PA20	PA20 / FAULT_CAN0	Fault pin for CAN1, input active high
42	PA21	PA21 / S_CAN0	Silent mode pin for CAN1, output active high
43	PA22	TX_CAN0	TX CAN 0 pin, On the shield
44	PA23	RX_CAN0	RX CAN 0 pin, On the shield
45	PA24		D-
46	PA25		D+
49	PB22	PB22	On the shield

Continued on next page

Table 19: Pinout for the Microcontroller V2.2 (Continued)

	MCU Pin	Arduino Pin Name	comment
50	PB23	PB23	On the shield
51	PA27	PA12	On the shield
57	PA30		SWD CLK
58	PA31		SWD IO
59	PB30	PB30	On the shield
60	PB31	PB31	On the shield
61	PB00	PB00 / A6	On the shield
62	PB01	PB01 / A9	On the shield
63	PB02	PB02	On the shield
64	PB03	PB03 / A7	On the shield


Power Supply
DC/DC buck converter : 24Vin-5Vout, 1Aout (theroy 2Aout)
DC/DC LDO converter 5Vin-3.3Vout, 600mAout
TPS54233QDRQ1
AP7365-33WG-7
MCU Power supply
pin-Header for Shield
Green CAN status LED
Hardware or software drive of the LED
default only Hardware driven LED are installed
Red Power LED
General usage switch with LED
General usage LED
Reset switch
ATSAME51J20A-AF
CAN Line 0&1
TCAN337GDR
DPDT switch : connects CAN bus termination resistor
Power source selection
SD-card connector
SD CARD
I2C line 0
Micro USB-B, USB 2.0
USBULC6-2M6
SWD
swd connector
SSB - Microcontroller V2.2
Date: 6.08.2022
Sheet 1 of 1
File: C:\Users\..\MCU-main-board-V2.SchDoc
Drawn By: Baptiste Savioz

Appendix B Block Monitoring documentation

Table 20: Exhaustive list of components for the block monitoring V2

	Name	Component	package	Description
1	C1, C4, C5	capacitor 100nF	0603	
2	C2, C3	Capacitor 22µF	0805	
3	R1, R3, R4, R5, R8, R9	Resistor 560 Ω	0603	
4	R2	Resistor 210 Ω	0603	
5	R6, R7	Resistor 120 Ω	0603	
5	P1, P2	Pin header male 20 pos		
6	J1, J2, J3, J4, J6	Connector fans 4 pos		470531000
7	J5	Connector MPPT 4 pos		S4B-PH-K-S (LF)(SN)
8	J7	Connector power supply for fans 4 pos		S4B-PH-K-S (LF)(SN)
9	J8, J9	Connector relay block 5 pos		S5B-PH-K-S (LF)(SN)
10	U1	Optocoupler		H11L1M
11	U2	Buck converter		LT1117CST-5TRPBF
12	U3	Converter I2C differential to I2C		PCA9615DPJ

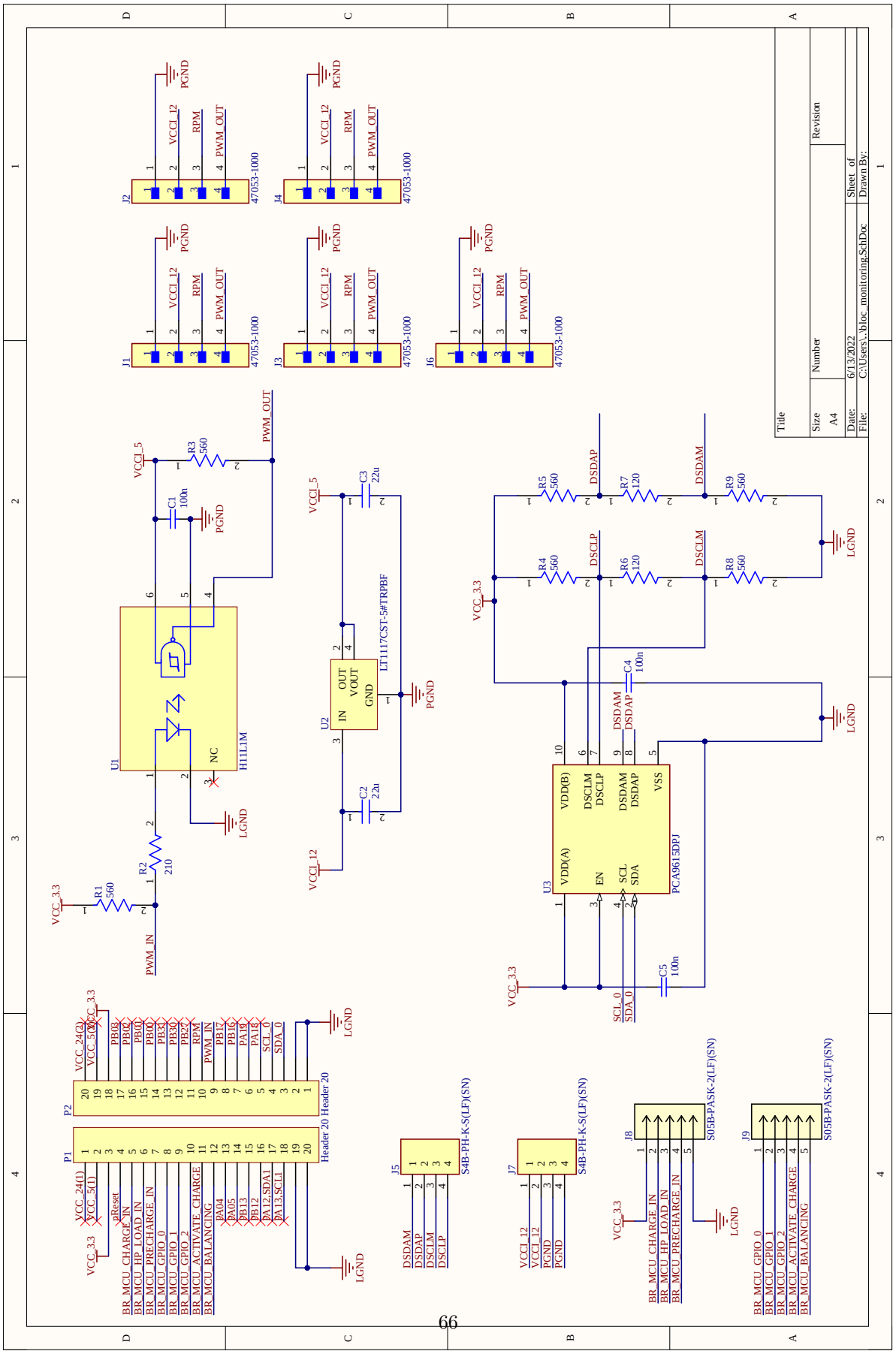


Figure 51: Full schematic of the block monitoring

Appendix C Relay Block and 24V DC/DC Converter documentation

Table 21: Exhaustive list of components for the relay Block

	Designator	Component
1	C1, C2, C3, C4, C5, C6, C7, C8	C0805C104M5RACTU
2	D1, D3, D4, D5	STTH6010WY
3	D2	BAS16215
4	F1	CC12H250MA-TR
5	F2	0458002.DR
6	F3	0685F5000-01
7	F4	0685P4000-01
8	F5	C1Q3
9	IC1, IC2, IC3, IC4, IC5, IC6, IC7	TLP5705H(D4-TP,E
10	IC8	ACPL-M483-500E
11	J1	43045-0200
12	J2	44428-0602
13	J3	43650-0301
14	J4	S05B-PASK-2(LF)(SN)
15	J5	S05B-PASK-2(LF)(SN)
16	J6	43045-0200
17	J7	43045-0200
18	J8	43045-0200
19	J9	43045-0200
20	J10	S04B-PASK-2(LF)(SN)
21	J11	43045-0400
22	J12	S05B-PASK-2(LF)(SN)
23	J13	S05B-PASK-2(LF)(SN)
24	K1, K2, K3	CN100D24
25	P1	1-826629-2
26	Q1	SIHU2N80E-GE3

Continued on next page

Table 21: Exhaustive list of components for the relay Block (Continued)

	Designator	Component
27	Q2, Q3	PMV213SN215
28	Q4, Q5, Q6, Q7, Q8, Q9, Q10, Q11	SSM3K36TU,LF
29	R1	RC0805JR-0710KL
30	R2	UB15-15RF1
31	R3, R4, R5, R6, R7, R8, R9	RR1220P-221-D
32	R10	ERJ6RBD3600V

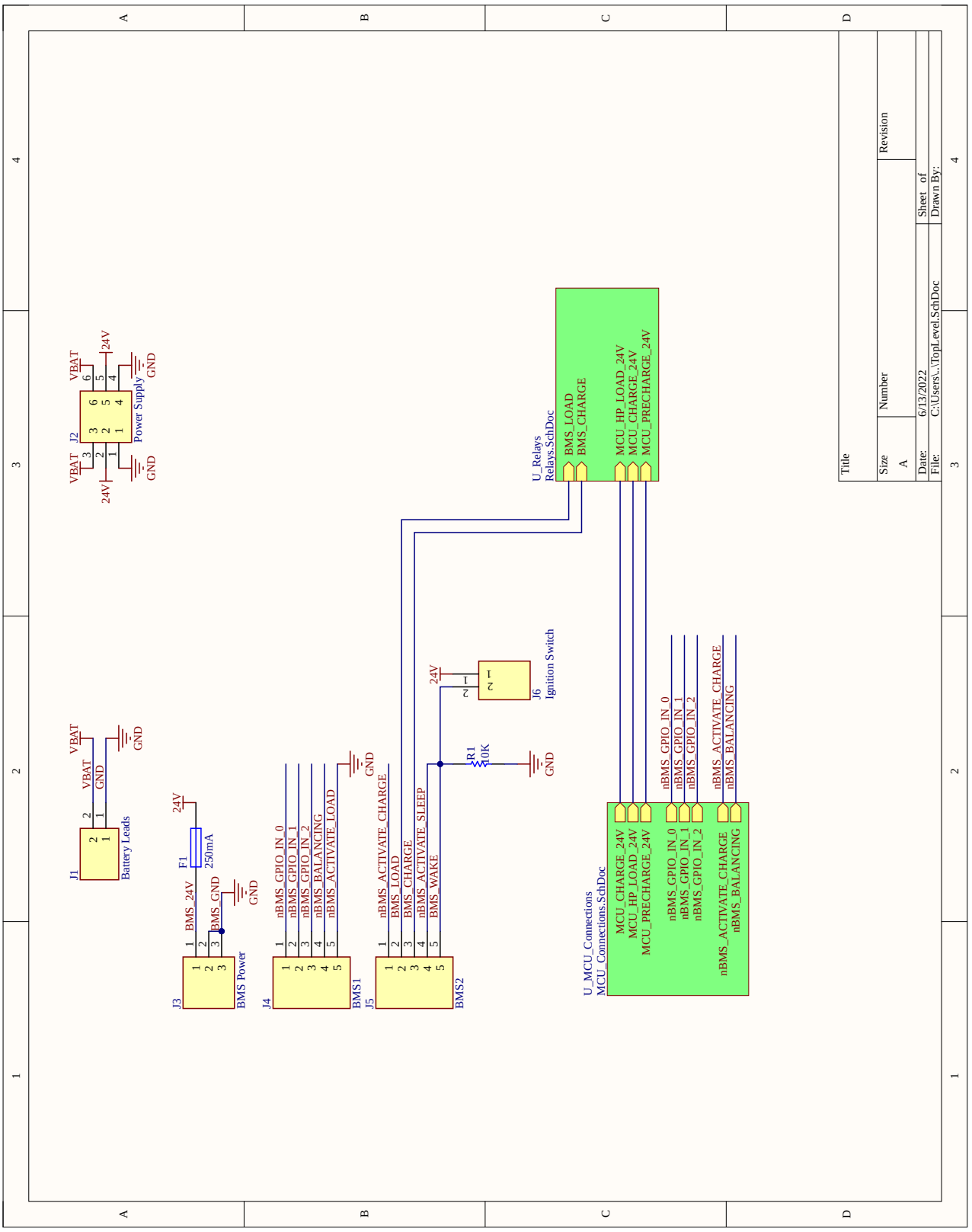


Figure 52: Full schematics of the Relay Block PCB - Top Level

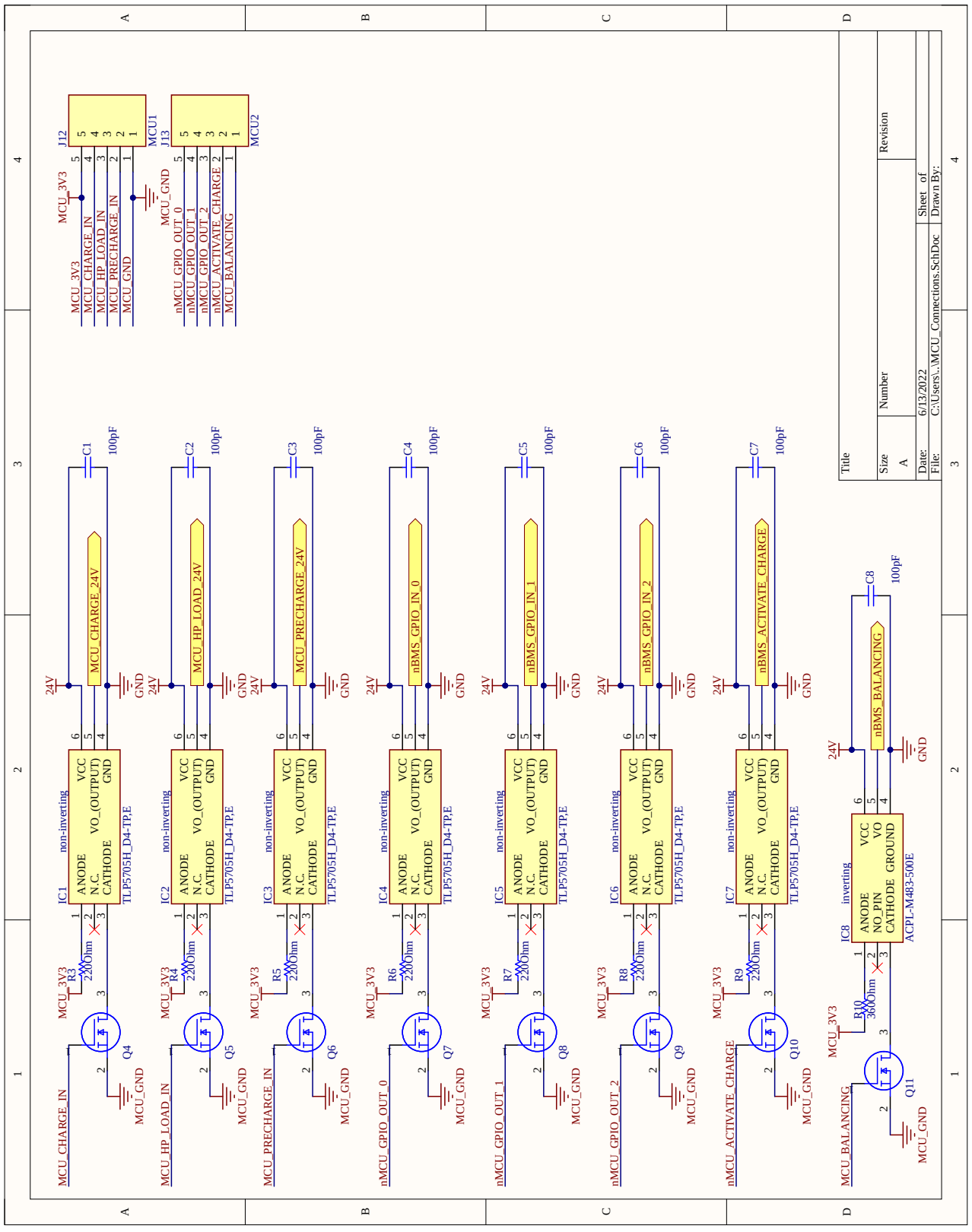


Figure 53: Full schematics of the Relay Block PCB - MCU - Connections

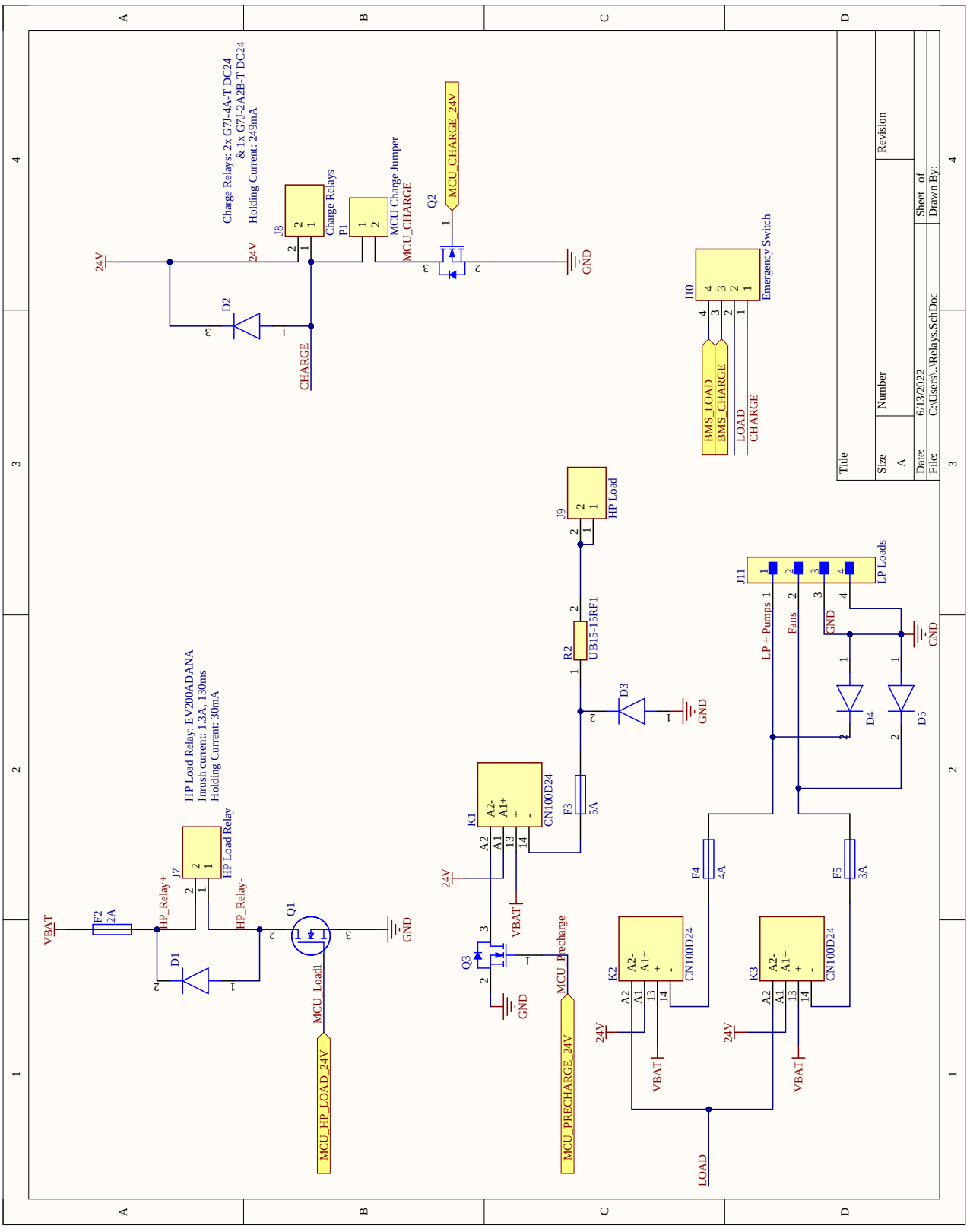


Figure 54: Full schematics of the Relay Block PCB - Relays

Table 22: Exhaustive list of components for the 24V DC/DC converter

	Designator	Component
1	C1	HMK325C7475KM-PE
2	C2	C3225X7S2A335K200AB
3	C3, C4, C5	CL31B106KBHNNNE
4	F1	045802.5DR
5	J1	44764-0602
6	L1	ASPI-4030S-100M-T
7	PS1	RPMH24-1.5

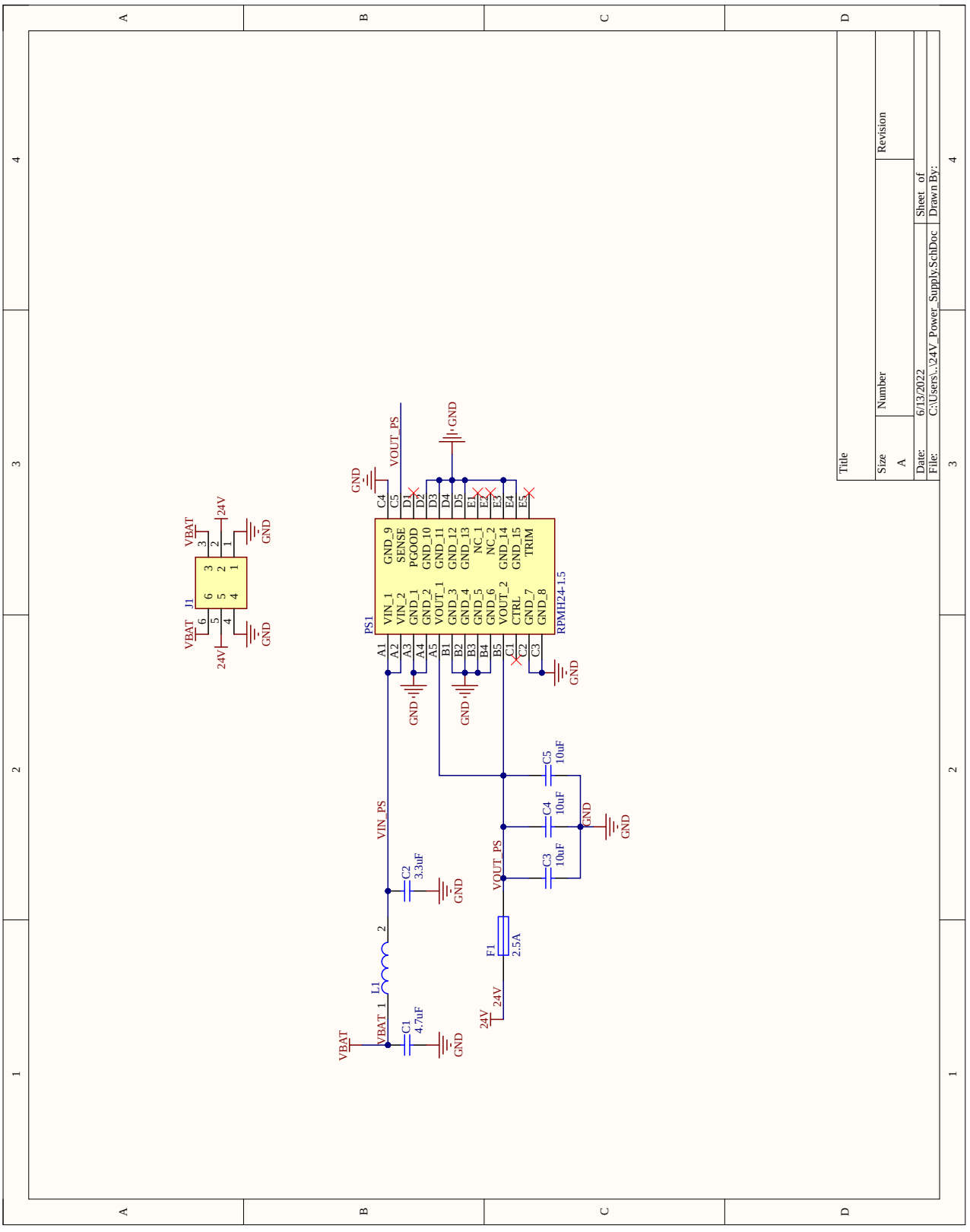


Figure 55: Full schematics of the 24V DC/DC Converter PCB

Appendix D Cockpit Shield for the microcontroller

D.1 Introduction and Motivation

The design of the microcontroller is modular in order to provide all the basic and redundant elements between the systems, such as a CAN and I2C line or the two power supplies 5[V] and 3.3[V] for example. On the other hand, with its two rows of pin headers and the possibility of putting a shield on it, it leaves a lot of flexibility to develop special applications. That is, rather than having to adapt or develop a new solution for a specific application from scratch, one can start directly from the solid base of this microcontroller to quickly design a shield that will embody the specific features of the application. A very concrete example is the cockpit microcontroller. The function of the microcontroller is very simple: it must read all the buttons, levers and knobs on the cockpit and send the measured values to the CAN line. In addition, it has to power some LEDs to give feedback to the pilot. In theory, this is very simple, but in practice it quickly became a cable nightmare... That's why we decided to replace the V1.3 microcontroller and put the new V2.2 instead with a dedicated shield. This will make it much easier to assemble and disassemble the cockpit, make the assembly more reliable with connectors designed for it, as well as reduce the risk of human error.

D.2 Specifications

Previously, it was the V1.3 microcontroller that took care of the CAN communication and the reading of most of the sensors. As there were no dedicated connectors for this application, a lot of wire was soldered to the pads, which often got torn off due to wear and tear, creating a nice frankenstein of cable and junk. Furthermore, the cockpit joystick[9] works in 5[V] and therefore the V1.3 microcontroller working in 3.3[V] could not directly make the measurement. The solution previously developed was to add an arduino Nano operating in 5[V] which could measure the values of the joystick, then communicate them in I2C through an IC level shifter to the microcontroller. It is quite easy to see that debugging all this quickly became an ordeal...

So we decided to simplify and have a dedicated connector for each sensor. This means fewer wires, less risk of errors or failures, and we save time when assembling. Moreover, we decided to do without the arduino Nano and the level shifter solution and to use instead a voltage divider to make the measurement directly with the microcontroller.

In addition, we also had significant time constraints to meet, as this project started in week 11 of the semester. The design had to be done as quickly as possible, so that between design time, manufacturing and delivery time, and then assembly, testing and programming, it would be ready and implemented before we left for Monaco. That's why we had to use only connectors in stock at SSB, i.e. S0xB-PASK-2(LF)(SN) , in the sizes available.

Shield requirements

Functionality	**Assessment**	**Flexibility**
a connector for some general usage LED	Low power LED will be use and they should be directly power by the MCU (a few mA). A connector S0xB-PASK-2(LF)(SN) should be use	Not mandatory but as much LED port as possible should be installed
The second I2C line should be available	dedicated Pull-up resistor will be mounted on the line. A S04B-PASK-2(LF)(SN) must be use, and provide as well power with 3.3[V].	Not very important
A connector for High Power Switch and its dedicated LED	A S04B-PASK-2(LF)(SN) connector and a proper electrical circuit design must be use	Mandatory
A connector for the dead man Switch	A S02B-PASK-2(LF)(SN) connector and a proper electrical circuit design must be use	Mandatory
A connector for the arming Switch	A S02B-PASK-2(LF)(SN) connector and a proper electrical circuit design must be use	Mandatory
Connectors for two potentiometers (knob)	2 S03B-PASK-2(LF)(SN) connectors must be use	Mandatory
A connector for the selector	A S09B-PASK-2(LF)(SN) connector must be use	6 channel are Mandatory but there is margin for the 4 left.
A connector for the joystick	A S08B-PASK-2(LF)(SN) connector and a proper electrical circuit design must be use	Mandatory but there is some flexibility on the selected solution

Table 23: Specification of the microcontroller

D.3 Design and Solution analysis

We, Felix Schmeding, Valentine Houlier and Baptiste Savioz, first started by writing down the specifications of the shield and clearly define its functionalities : what should it do and how. Felix Shmeding and Valentine Houlier were responsible for the cockpit and I[5] was there to understand their need and design the shield.

The first thing I noticed is that until now, no precautions were taken from an electronic point of view when using the buttons or the LEDs. The buttons were driven directly by the weak pull resistor of a pin of the V1.3 microcontroller. I decided to add an RC filter circuit in order to have a more robust use. For the sizing, I chose standard values of resistors and capacitors, respectively 10 [kΩ] and 100 [nF], which gives us a time response of $\tau = RC = 10[ms]$, perfectly adapted for our application. Moreover, there was some LED to be driven. For example, the high power switch connector has a built-in LED. The reference for this switch was lost and I couldn't have any information about it.

As a precaution, a current-limiting resistor has been provided, although its dimensioning is still uncertain. The value of 820 [ω] was chosen because being a standard value for driving an LED consuming 2[mA] with a 3.3[V] supply, we were taking little risk. This value was then confirmed by tests.

The circuit presented in figure 56 are used for :

- the arming switch : RC filter

[5]in this appendice, the pronoun I will refer to Baptiste Savioz

- the dead man switch : RC filter
- the high power switch : RC filter and LED driver
- the additionnal LED (not implemented yet) : LED driver
- the joystick buttons : RC filter

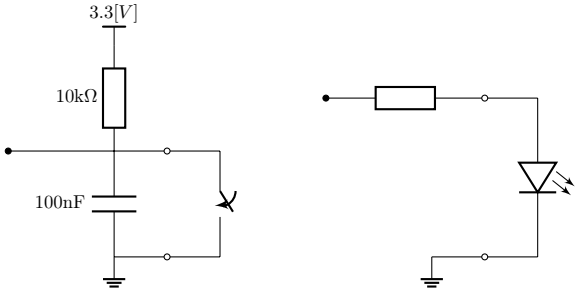


Figure 56: simple RC filtering circuit for switches and a LED driver circuit

One of the big negatives of the solution that was used until now was the use of an arduino Nano to read the joystick values. We had two microcontrollers to program, twice as many cables and twice as much chance of something going wrong. The improvement we wanted to make was to eliminate the arduino Nano and do all the measurements directly with the V2.2 microcontroller. To do this we had to convert the voltage levels correctly so that we never exceeded the level that the microcontroller could accept (3.3[V]) and so that we did not distort the signal while doing so. The simplest and most elegant solution was to use a voltage divider. According to the documentation of the joystick, it outputs a current of up to 8[mA] with an output impedance of 2[Ω]. Furthermore, in order to guarantee the integrity of the output signal, a load impedance of at least 1[kΩ] must be guaranteed. As a precaution we have taken an order of magnitude margin with this value. This gives a current output of the joystick of about 500[μA] Then, we had to make sure that on the measurement side, we don't distort the signal either. This is because the microcontroller is measuring in the middle of the voltage divider and so the leakage current from the pin and the analog to digital converter (ADC) will affect the voltage division. According to the documentation, the leakage current of a pin configured as an input is typically 15[nA], and therefore several orders of magnitude lower than the current drawn by the voltage divider and can therefore be neglected. The total resistance of our resistive voltage divider is therefore 10[kΩ]. I chose the resistance values among the available ones, which gives: R1 = 3.3[kΩ] and R2 = 6.2[kΩ].

$$U_{in} = 5[V]$$
$$U_{out} = U_{in}\frac{R_2}{R_1 + R_2} = 5[V]\frac{6.2[k\Omega]}{3.3[k\Omega] + 6.2[k\Omega]}$$
$$U_{out} = 3.26[V]$$

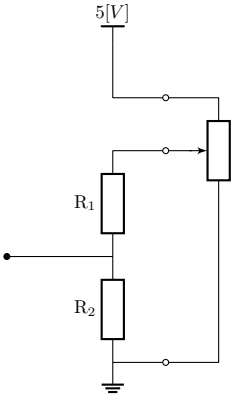
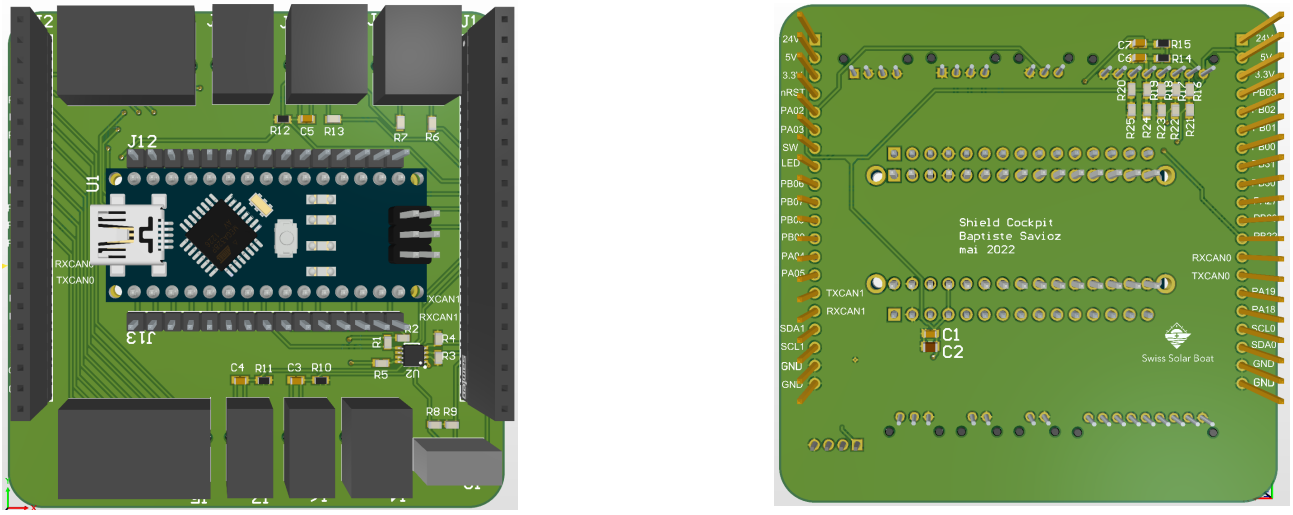


Figure 57: Voltage divider circuit and dimensioning

The number of measurements to be made on the cockpit is consequent and it is not possible to make them all with the V2.2 microcontroller which has a limited number of GPIOs still available. So we had to make some choices:

- The 10 channel selector only uses 6 channels and we chose to measure 8 of them. Although we can never use the full functionality of the selector, this still leaves us 2 channels free for future applications, a comfortable margin.
- tth 2 pins have been dedicated to general purpose LEDs. They are not yet in use, but future applications are planned to give feedback to the driver.

Finally, we thought of a way to increase the number of GPIOs available and still be able to read the joystick values if the voltage divider solution did not work. We decided to integrate the possibility to mount an arduino Nano and a level shifter directly on the shield. Normally, if everything goes according to plan, they will never be used or even installed on the shield. However, if we were to extend the functionality of the cockpit, we could do so quickly, with little effort.
After, the first test, we are able to confirm that we won't have to mount any arduino on the shield since the voltage divider is working just as expected.



(a) Top view

(b) Bottom view

Figure 58: View of the cockpit shield in Altium

D.4 Conclusion

As I write this conclusion, the shield has been assembled, the cockpit buttons rewired and the code adapted. Felix has been able to test that everything works as expected and that's already very good news. The last step now is to test it in real conditions on the boat, but for that we have to wait until it is rewired.

Shield Cockpit V1.1 for the microcontroller V2.2

24V Low power
5V : 600mA
3.3V 300mA

pin-Header for Shield

J1 J2

analogPot1, PA02
analogPot2, PA03
generalLed0 ,SWITCH
HP led, LEG_G
analogX, PB06
analogY, PB07
analogz, PB08
HPswitch, PB09
nArmingSwitchPA04
nDeadManSwitch, PA05
PA12,SDA1
PA13,SCL1

selecteur0, PB03
selecteur1, PB02
selecteur2, PB01
selecteur3, PB00
selecteur4, PB31
selecteur5, PB30
selecteur6, PA27
selecteur7, PB23
PB22, generalLed1
nMannetteSwitchA, PA19
nMannetteSwitchB, PA18

Arduino Nano : Not to be installed

J12 22-28-4153
U1 Arduino Nano
MH1 MH2 MH3 MH4
D1/TX D0/RX RESET_1 GND_1 D2 D3 D4 D5 D6 D7 D8 D9 D10 D11 D12
VIN GND_2 RESET_2 +5V A7 A6 A5,SCL A4, SDA A3 A2 A1 A0 AREF 3V3 D13
J13 22-28-4153
C1 100nF C2 10µF

Level shifter IC : Not to be installed

U2 PCA9306DCTR
SCL2 SDA2 SCL1 SDA1 EN VREF1 VREF2 GND
R1 2.2k R2 2.2k R3 2.2k R4 2.2k R5 200k
PA13,SCL1
PA12,SDA1

Cockpit's knob connectors

J3 S03B-PASK-2(LF)(SN)
analogPot1, PA02
J4 S03B-PASK-2(LF)(SN)
analogPot2, PA03

Selector connector

J5 S09B-PASK-2(LF)(SN)
selecteur0, PB03
selecteur1, PB02
selecteur2, PB01
selecteur3, PB00
selecteur4, PB31
selecteur5, PB30
selecteur6, PA27
selecteur7, PB23

General usage LED

resistor to drive LED
R6 820Ohm
R7 820Ohm
J8 S04B-PASK-2
Led0
Gnd
Led1
Gnd
generalLed0 ,SWITCH
PB22, generalLed1

I2C 1 connector

Pull-up resistor for I2C
R8 3.9K
R9 3.9K
J9 S04B-PASK-2-LF-SN
3.3V
SDA
SCL
GND
PA12,SDA1
PA13,SCL1

Arming Switch connector

R10 10k
C3 100nF
J6 S02B-PASK-2(LF)(SN)
nArmingSwitchPA04

Dead Man Switch connector

R11 10k
C4 100nF
J7 S02B-PASK-2(LF)(SN)
nDeadManSwitch, PA05

Joystick connector

Voltage divider 5V-3.3V
R14 10k
R15 10k
R16 3.3k
R17 3.3k
R18 3.3k
R19 3.3k
R20 3.3k
R21 6.2k
R22 6.2k
R23 6.2k
R24 6.2k
R25 6.2k
C6 100nF
C7 100nF
J10 S08B-PASK-2(LF)(SN)
analogX, PB06
analogY, PB07
analogz, PB08
nMannetteSwitchA, PA19
nMannetteSwitchB, PA18

High Power Switch connector

R12 10k
C5 100nF
R13 820Ohm
J11 S04B-PASK-2
switchP
switchN
ledP
ledN
HPswitch, PB09
HP led, LEG_G

Title	SSB- Shield Cockpit V1.1	
Size A4	Number	Revision
Date: 6.08.2022	Sheet 1of 1	
File: C:\Users\..\MCU-shield-cockpit-V1_1.SchDoc	Drawn By: Baptiste Savioz	

Appendix E Routing PCB between the connectors of the back distribution board and the interior of the high power compartment.

E.1 Introduction and motivation

A PCB was useful to arrange the connections between the DB back and the high power part of the boat. This PCB has no function and just facilitates the connections between these two parts of the boat.

For this PCB, two iterations were made. A first iteration was done as quickly as possible to implement it quickly. However, the routing was not optimised in this first version. Therefore a second version was realised.

E.2 Specifications

The challenge of this PCB is to make a routing PCB that is compatible with the V1 and V2 battery and the V1 and V2 DB back. The aim is to make the connections between the different elements cleaner while not creating any EMC problems. Figure 59 shows the different connections that the PCB must have: the DB back, the high power part, the DC / DC converters and the emergency switch. All PCB specifications are detailed table 24.

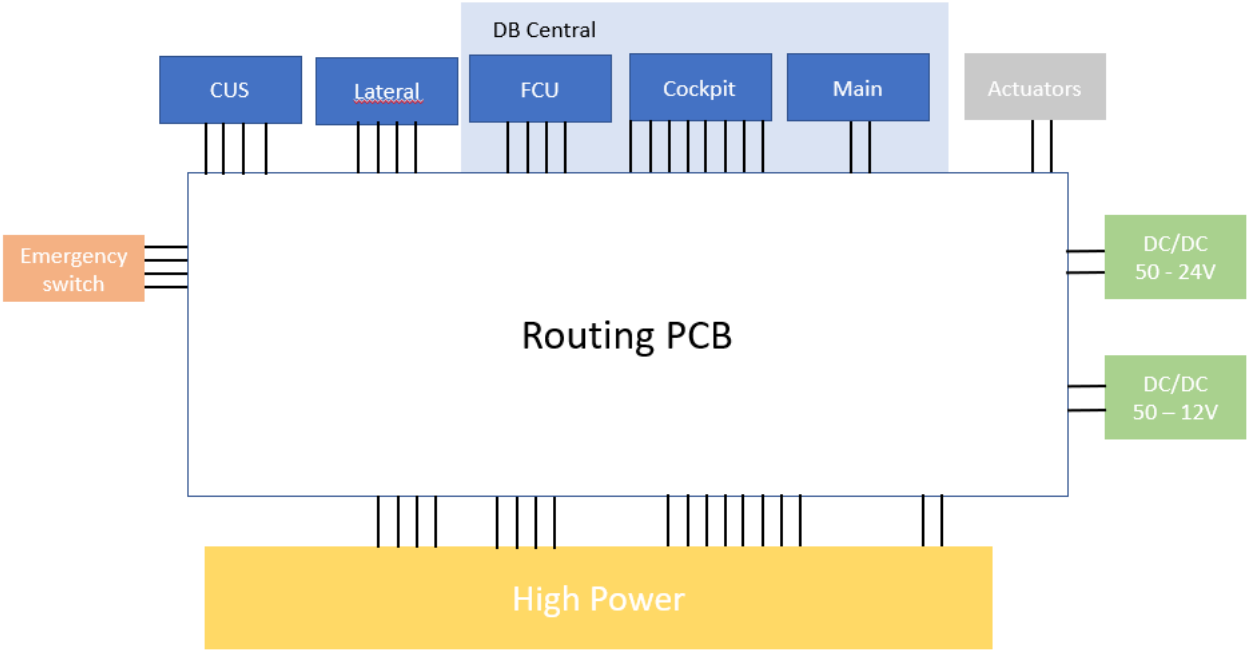


Figure 59: Schematic of the different elements to which the PCB is connected

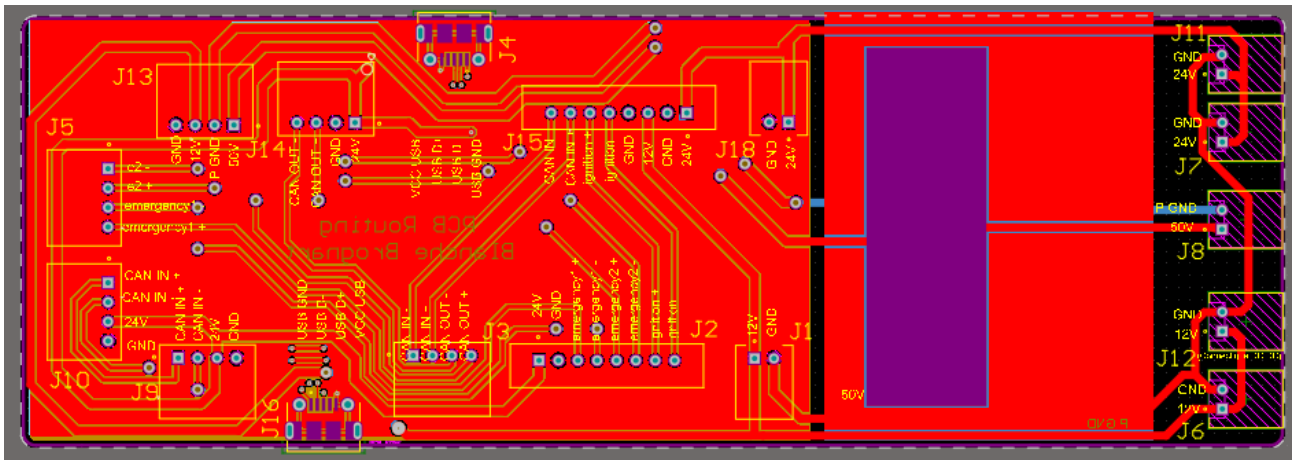
PCB Routing requirements

Functionality	**Assessment**	**Flexibility**
2 USB port	A connector Micro USB-B 1050170001 should be use	Not mandatory to use this type of connector but it allows to match the impedance of USB transmission
A connector for the CUS	A connector S04B-PASK-2(LF)(SN) should be use	Mandatory
A connector for the Lateral	A connector S04B-PASK-2(LF)(SN) should be use	Mandatory
A connector for the emergency switch	A connector S04B-PASK-2(LF)(SN) should be use	Mandatory
A connector for the FCU	A connector S04B-PASK-2(LF)(SN) should be use	Mandatory
A connector for the Cockpit	A connector S08B-PASK-2(LF)(SN) should be use	Mandatory
A connector for the Main	A connector S02B-PASK-2(LF)(SN) should be use	Mandatory
A connector for the actuators	No connectors are used but the wire is soldered directly to the exposed trace	Mandatory
A connector for the DC/DC 12V	A connector S02B-PASK-2(LF)(SN) should be use	Mandatory
A connector for the DC/DC 24V	A connector S02B-PASK-2(LF)(SN) should be use	Mandatory
Connectors for the high power	Connectors S02B-PASK-2(LF)(SN), S08B-PASK-2(LF)(SN), S04B-PASK-2(LF)(SN) should be use	Mandatory
2 CAN test connectors	Connectors S04B-PASK-2(LF)(SN) should be use	Not mandatory

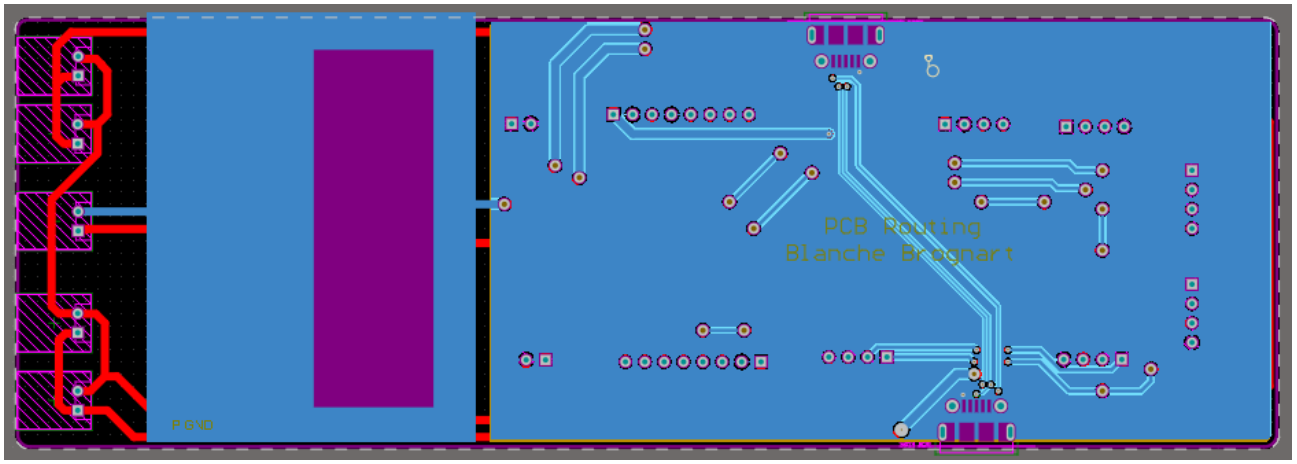
Table 24: Specification of the Routing PCB

E.3 Design and Solution analysis

The routing PCB is a rectangular PCB of 130mm x 44mm and 1.6mm thick. The PCB has 4 layers. All components are on the top layer, the flat grounds are on the middle and bottom layers. To define the thickness of the different layers, the recommendations of JLC PCB are used [10]. Figure 60 shows the top and bottom 2D view of the PCB in Altium. The challenge of this PCB was not in the circuit as the PCB is composed of connectors only but in the design of the PCB. The following list presents the important elements I paid attention to in the design of the PCB.



(a) Top face



(b) Bottom face

Figure 60: 2D view of the PCB routing in Altium

- Size track
  The size of the trace has been chosen to match the current flowing through it. All the lines carrying the digital signals are 0.5mm thick. All the power supplies except the 50V battery supply to the actuators are 1mm thick because the current flowing through them is of the order of 3A. Finally, for the 50V trace, the trace is directly exposed in order to add tin. Indeed maximum 22A circulates in this trace and according to the trace thickness calculator it would be necessary to have 50 mm wide trace internally to carry as much current. Therefore, the choice was made to expose the traces to add track thickness afterwards.

- USB traces
  For USB traces, extra care must be taken because the USB connector has an impedance equivalent to 90 ohms. Therefore it is of fundamental importance that the USB tracks are spaced appropriately, in order to keep the 90 Ohm impedance. A very common, but also effective, rule of thumb is to use a minimum spacing of "2W" where "W" is the width of the track subject to impedance control. At the connector the impedance of 90 ohm is also guaranteed by the Micro USB-B connectors 1050170001.

- Ground plane
  The ground planes are located on all layers of the PCB. There are two flat grounds on all 4 layers, the 50V battery ground and the DC-DC converter ground. The ground planes help to limit interference.

Additionally having a ground on the top layer allows the SMD pads to have a ground. Finally the reason for choosing 4 layers instead of 2 is to have a complete ground layer for the USB signals.

E.4 Conclusion

The first iteration of the routing PCB has been implemented on the boat. Figure 61 shows a picture of this PCB in the boat. The PCB works fine except the USB does not respect the 90 Ohm differential impedance. However, the USB is only used very rarely and therefore this PCB is suitable until the second version is implemented. The second version has not yet been implemented but will be the week of June 20-26.

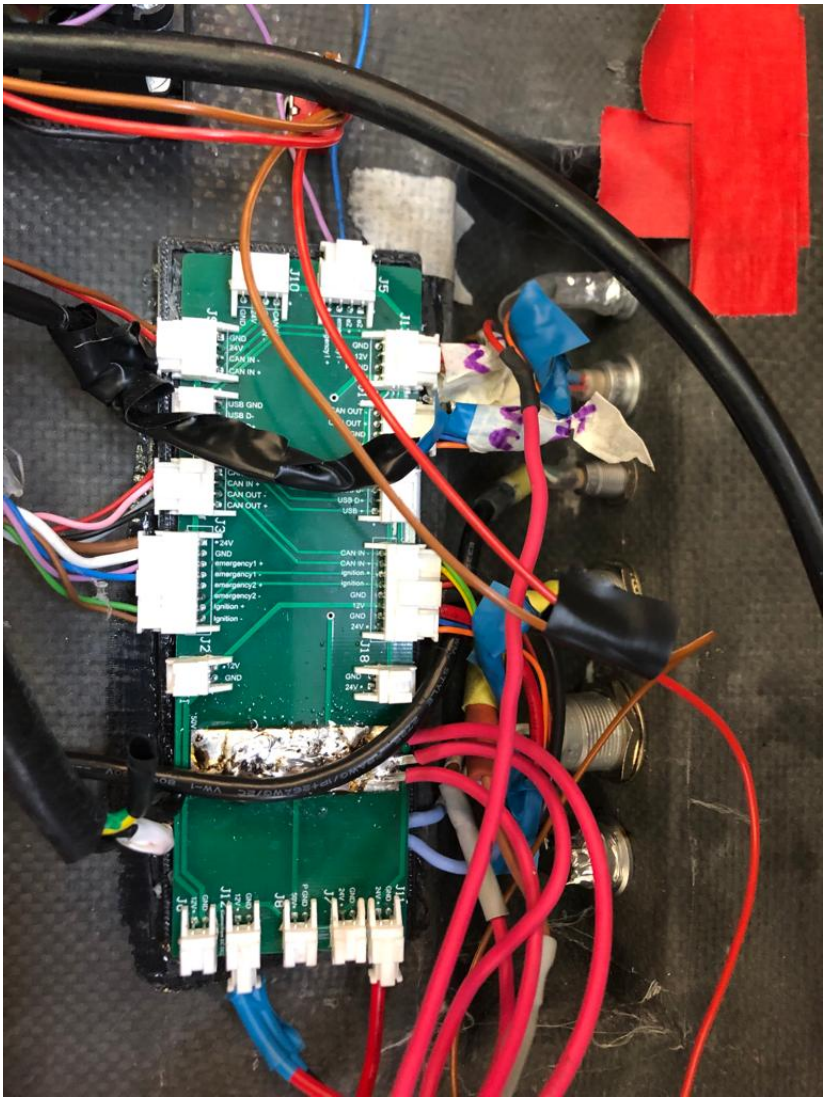


Figure 61: Pictures of the implementation of the V1 routing PCB in the boat

E.5 Documentation

Table 25: Exhaustive list of components for the PCB routing

	Name	Component	Description
1	1, J6, J7, J8, J11, J12, J18	Connector 2 pins	S2B-PH-K-S(LF)(SN)
2	J2, J15	Connector 8 pins	B8B-PH-K-S(LF)(SN
3	J3, J5, J9, J10, J13, J14	Connector 4 pins	S4B-PH-K-S(LF)(SN)
4	J4, J16	Connector USB	1050170001

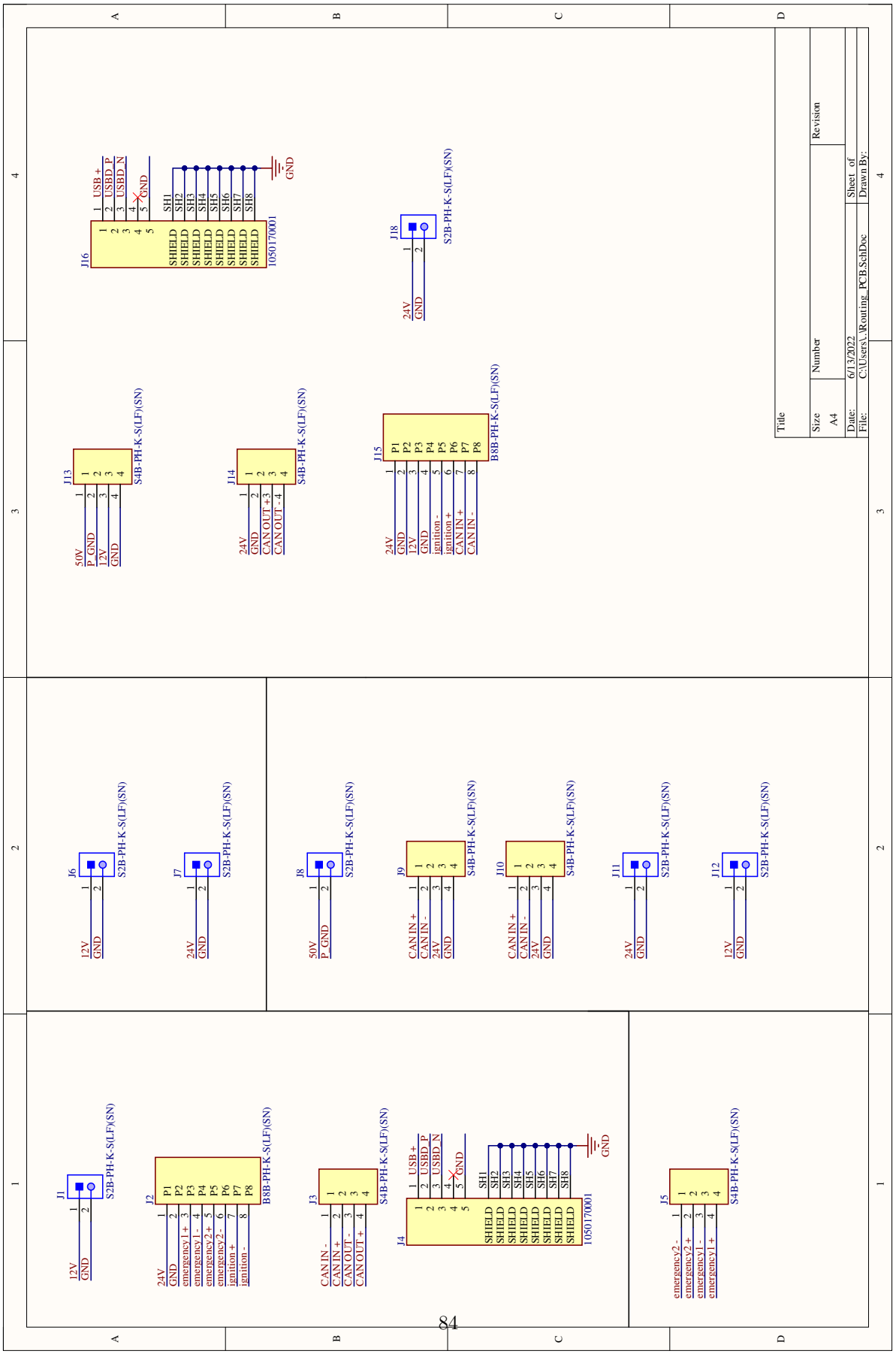


Figure 62: Full schematic of the PCB routing

References

[1] *2-A, 28-V INPUT, STEP-DOWN DC/DC CONVERTER WITH ECO-MODE™*. TPS54233-Q1. Revised. Microchip. Mar. 2012.

[2] *SAM D5x/E5x Family DataSheet*. ATSAME51J20A. Rev. A. Microchip. July 2017.

[3] *TCAN33xx 3.3-V CANTransceivers with CAN FD (Flexible Data Rate)*. TCAN337GDR. Revised. Microchip. Dec. 2019.

[4] *NF-F12 industrialPPC-2000*. URL: https://noctua.at/en/nf-f12-industrialppc-2000.html.

[5] *H11L1M: 6-Pin DIP Schmitt Trigger Output Optocoupler*. URL: https://www.onsemi.com/products/interfaces/high-performance-optocouplers/high-speed-logic-gate-optocouplers/h11l1m.

[6] *LT1117 Datasheet and Product Info | Analog Devices*. URL: https://www.analog.com/en/products/lt1117.html.

[7] *PCA9615DPJ NXP USA Inc. | Circuits intégrés | DigiKey*. URL: https://www.digikey.fr/fr/products/detail/nxp-usa-inc/PCA9615DPJ/4843290.

[8] *Datasheet ETHERLINE FD P FC Cat. 5 2x2x22/7 AWG*. URL: https://t3.lappcdn.com/fileadmin/documents/technische_doku/datenblaetter/ETHERLINE/Etherline_Leitungen/DB2170894EN.pdf.

[9] *Proportional multi-axis fingertip controllers - non-contacting Hall effect technology*. HF45S10. APEM.

[10] *PCB Prototype & PCB Fabrication Manufacturer - JLCPCB*. URL: https://jlcpcb.com/.