

ELONMux

DOCUMENTATION & USER GUIDE

PCB revision: V1.0

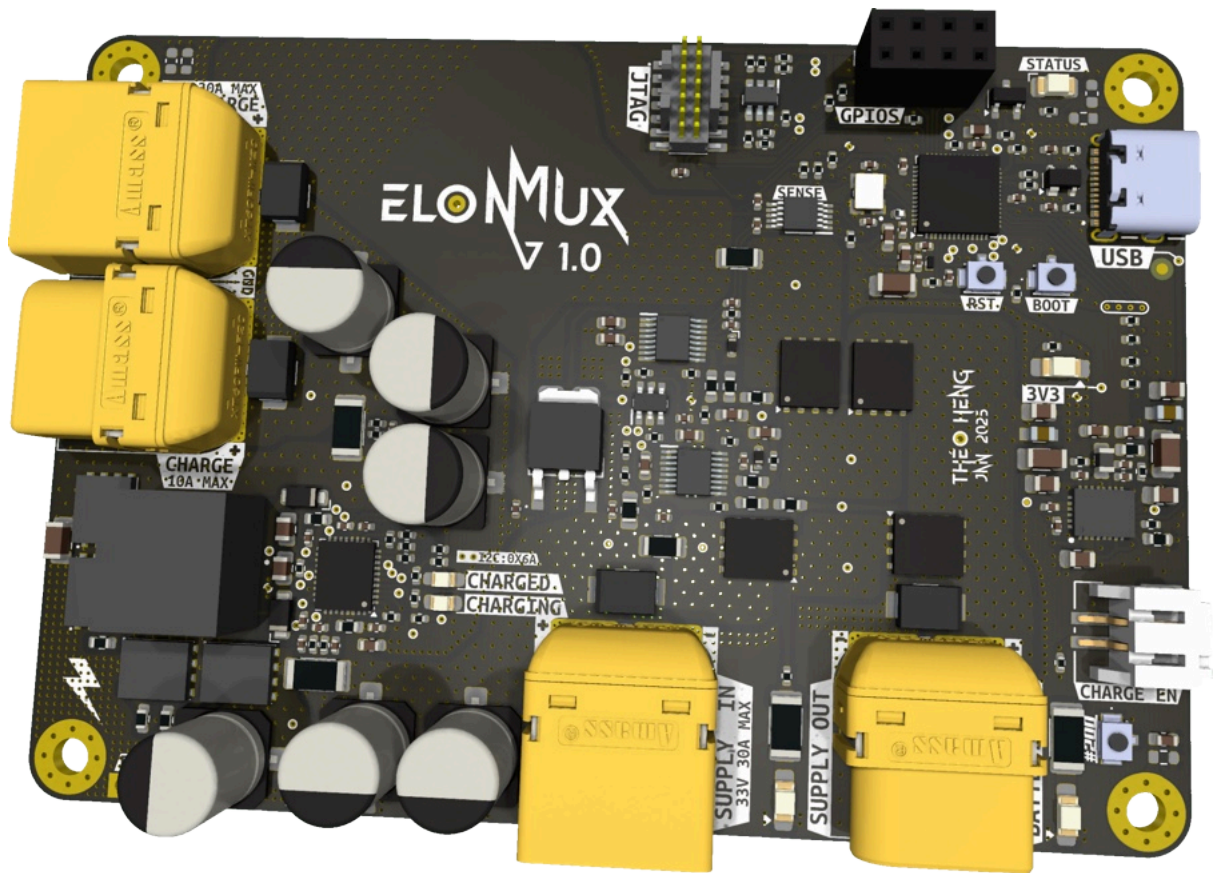
Documentation revision: R03

Author: Théo Heng

Contact: theoheng@icloud.com

[GitHub repository Hardware](#)

[GitHub repository Software](#)



Revision History

Author	PCB revision	Documentation revision	Date	Changes summary
Théo Heng	V1.0	R01	02/2025	Initial documentation
Théo Heng	V1.0	R02	02/2025	Added layout, pictures and test results
Théo Heng	V1.0	R03	03/2025	Added more test results, thermal images, software architecture and more

DISCLAIMER:

The ElonMux name is a wordplay between power multiplexer (MUX) and Elon Musk.
It is in no way related with the actual Elon Musk or recent events.
Anyone thinking something bad about me because of this name is a straight up IDIOT.



Acknowledgments

This project was my first ever PCB.

I started working on this board in late September 2024, just after joining the electrical team of EPFL Xplore. At that time I had never done anything related to PCB design. This is why I would like to thank multiple people, without whom I wouldn't have managed to finish this rather large project in only a few months.

Eliot Abramo - EL SE 24-25:

Eliot accepted me into its team without meeting me beforehand, and assigned me this project directly, despite knowing that I had no experience in this kind of work. Thank you for your trust, your support all along, your shenanigans for getting me the budget, your reviews, your unstoppable passion for what we do.

Federico Bise - former EL TL, EL Advisor 24-25:

Federico, with its extensive PCB design experience, was my sort of mentor for this project. Thank you for answering all my questions, taking the time to review my early schematics, and most of all the fun we had each time when talking PCB like 2 big nerds. If I managed to get the confidence to finish this project it was mostly because of you.

Vincent Nguyen - former EL TL:

Thank you Vincent for your clean KiCad template, and your Amulet controller documentation that greatly helped me to learn a lot from a more advanced engineer.

Pedro Conde - EL TL 24-25:

Thank you Pedro for your trust, reviews and support.

Arion Zimmermann - EPFL Xplore Founder:

The inspiration for this reference document comes from the *Power System reference* document made by Arion for its famous swan song Pollux III. Thank you also for your interest in this project despite your retirement, and the geographical separation. Thanks for the zoom review and discussion.

Table of contents

Revision History.....	2
Acknowledgments.....	3
Table of contents.....	4
1.Introduction & Motivations.....	6
2 System Requirements.....	7
2.1 Power Requirements.....	7
2.2 Features Requirements (to be tested).....	7
2.3 Additional features implemented (to be tested).....	7
3 General Architecture.....	8
3.1 MCU & Firmware.....	8
3.1.1 Programming the ESP32-S3.....	8
3.1.2 MCU Firmware.....	9
3.2 BQ25756E Battery Charger.....	11
3.2.1 Charger specifications.....	11
3.2.2 Status indications and external control.....	11
3.2.3 I2C control.....	12
3.2.4 Thermal considerations.....	12
3.3 Power Path.....	12
3.3.1 Topology.....	13
3.3.3 Delay estimations.....	13
3.2.4 Power Connectors.....	15
4 PCB Layout.....	16
4.1 Stackup.....	16
4.2 Power Routing.....	17
5 Mechanical considerations and dimensions.....	19
6 Orderables Specifications and Price History.....	20
6.1 PCB options to select for order.....	20
7 Testing and Results.....	21
7.1 Assembly.....	21
7.2 Charger.....	21
7.3 Power Path.....	21
7.4 Additional features.....	21
7.5 Thermal images.....	22
8 How-to-Use.....	24
8.1 Wiring diagram.....	24
8.2 Charging procedure.....	24
9 Discussion.....	25
9.1 Suggestions for V1.1.....	25
10 References.....	26
11 Photos.....	27

1.Introduction & Motivations

ElonMux is a standalone power PCB in EPFL Xplore's rovers.

Its two main functions are:

- **Integrated Battery Charger:**

The 2024-2025 Xplore's rover uses a 7S8P custom Li-Ion battery. Due to its limited capacity, it has to be regularly removed from the rover to be charged with an external charger. This process is time consuming, annoying for the software teams that needs a permanent connection to the rover without power interruption, and most importantly risky for the rover, as with its tight interior and numerous wires and cables running everywhere, it was not uncommon to disconnect wires, or breaks parts or connectors when removing and inserting the battery multiple times. Therefore ElonMux aims to relocate the external battery charger on a custom and integrated PCB. To start a battery charge, one just need to connect an external power supply into the rover's chassis (eg. 24V). This eliminates the need to extract the discharged batteries to charge them. This project was imagined a long time ago, even already a couple of years back by Arion.

- **2-Channels, 0-Down-Time Power Path:**

To be able to charge the batteries efficiently with the integrated charger while being able to use the rest of the rover features, it was important to implement a robust power path. **The goal is to power the rover with the external power supply while the battery is charging, and then immediately switch to battery power when the supply is disconnected, ensuring a 0-down-time charging procedure.**

2 System Requirements

Please tick working features once tested.

2.1 Power Requirements

Continuous charge current	At least 5A - tested up to 6A. Better to have active cooling above 3A of charge current	<input checked="" type="checkbox"/>
Continuous discharge current	30A - tested at 20A for 7 min, 30A for 1 min with no active cooling	<input checked="" type="checkbox"/>
Peak discharge current $\leq 1s$	40A - tested at 40A for 2s 45A for 1s with no active cooling	<input checked="" type="checkbox"/>
Input voltage	Needs to work with 24V external power supply - ok	<input checked="" type="checkbox"/>

2.2 Features Requirements (to be tested)

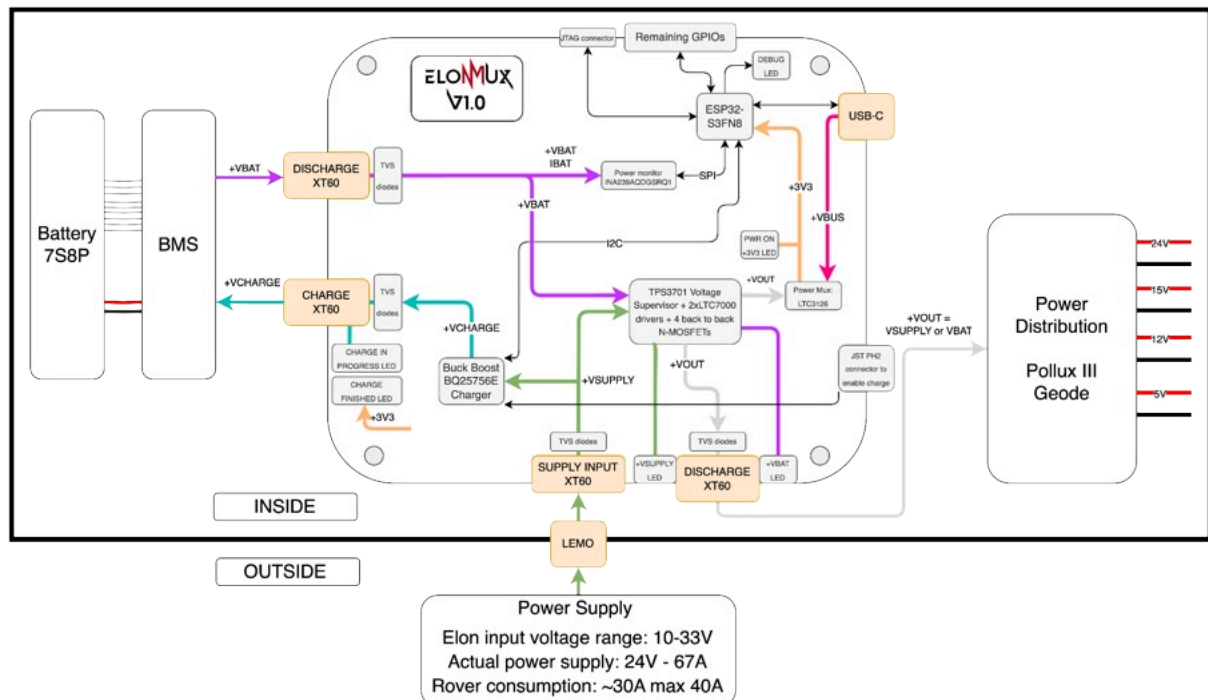
Safety	Power path circuitry independent from MCU - ok	<input checked="" type="checkbox"/>
Custom charge profiles	Lower current at beginning - Implemented, not tested yet	<input type="checkbox"/>
Charge status indicator	2 LEDs, charged/charging - schematic issue	<input type="checkbox"/>
USB-C programming	For easier quick code changes - ok	<input checked="" type="checkbox"/>
External charge control	Can turn ON-OFF the charge with external button - ok	<input checked="" type="checkbox"/>
Power OUT indicator	LED indicating which power source is powering the rover - ok	<input checked="" type="checkbox"/>

2.3 Additional features implemented (to be tested)

UART programming	Alternative way of programming - should work, not tested yet	<input type="checkbox"/>
JTAG programing	Alternative way of programming - should work, not tested yet	<input type="checkbox"/>
Status indication 1	Remaining GPIOs can be used to monitor a I2C OLED screen to display status information like charging current and so on - ok	<input checked="" type="checkbox"/>
Status indication 2	Status LED controlled by MCU - ok	<input checked="" type="checkbox"/>
Active cooling	ElonMux has 2 solder holes to power a 24V fan - ok	<input checked="" type="checkbox"/>
Quick change of charge profiles with external button	The remaining GPIOs can be used to select a different charge profile depending on the need (fast charging, slow charging...)- ok	<input checked="" type="checkbox"/>

3 General Architecture

Below is presented the block diagram of ElonMux V1.0.



3.1 MCU & Firmware

ElonMux features an ESP32-S3-FN8 SoC, a dual-core microprocessor and with 8MB of in-package flash. For the intended job, this chip is clearly overpowered in terms of computational power and flash memory, but it has a lot of GPIOs, two i2C controllers, and supports natively USB programming. Plus it is always good to have extra flash. An external 40MHz crystal is required for this SoC.

The MCU communicates via **I2C** to the battery charger. (functional)

The MCU communicates via **SPI** to the current and voltage monitor. (not implemented yet)

3.1.1 Programming the ESP32-S3

The MCU can be programmed either by **USB**, **JTAG** or **UART**.

To program over USB or UART, solder the USB jumper (GPIO3 to +3V3).


To program over JTAG with the JTAG connector, solder the JTAG jumper (GPIO3 to GND).

DO NOT LEAVE FLOATING - DO NOT SOLDER BOTH JUMPER AT THE SAME TIME

To put the chip in bootloader mode for UART or USB, hold the RST and BOOT button simultaneously, then release the RST button, then release the BOOT button.

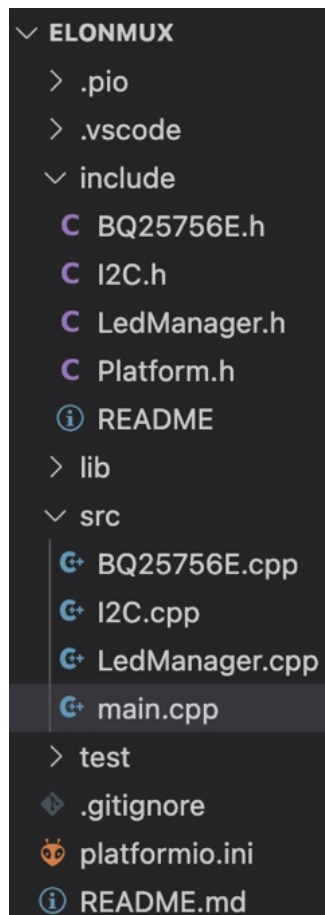
The recommended way to flash ElonMux is through Platform.io in VSCode using the USB-C port.

Follow those steps to program ElonMux:

1. Clone the [ERC_POWER_SW](#) repository.
2. Install Platform.io in VSCode
3. Open the ElonMux folder in VScode. **FOLLOWING STEPS WON'T WORK IF THE PROJECT IS NOT OPENED FROM A PARENT DIRECTORY.**
4. Flash the code using the  button in the bottom bar. Keep in mind that the charger needs to be powered by an external power supply from the Supply in connector in order to be programmed.
5. That's it!

3.1.2 MCU Firmware

The firmware has the following structure:



main.cpp:

- The main.cpp file contains the core logic for the ElonMux project, managing the charging process and system status. It initializes the BQ25756E charger, configures GPIO pins, and sets up the OLED display for user feedback. The main loop continuously monitors the charger state, reads current measurements, handles user input via buttons to adjust charging parameters, and updates the display with relevant information such as battery voltage, charge current, and elapsed charging time. It also manages LED indicators to reflect the current charging state and handles fault conditions.

i2c.cpp & i2c.h:

- The I2C.cpp and I2C.h files implement a cross-platform I2C communication interface for the ElonMux project. The header file provides platform-specific definitions for both Arduino and STM32 environments, allowing for flexible deployment. The implementation provides essential I2C functions including reading and writing 8-bit and 16-bit values to specific registers, as well as utilities for modifying individual register bits. These files abstract the low-level I2C communication details, making it easier to interact with the BQ25756E battery charger and other I2C peripherals in the system. The implementation handles the complexities of I2C transactions including proper start/stop conditions and error checking, ensuring reliable communication with connected devices regardless of the underlying hardware platform.

BQ25756E.cpp & BQ25756E.h:

- The BQ25756E.cpp and BQ25756E.h files implement a comprehensive driver for the BQ25756E battery charger IC. The header file defines all register addresses, bit masks, and configuration structures, while providing a cross-platform class interface that works on both Arduino and STM32 systems. The implementation file contains methods to initialize the charger, configure charging parameters (voltage/current limits, safety timers, termination conditions), read ADC measurements (battery voltage, charge current), and monitor system status. The driver provides complete access to all chip features, including power path management, temperature monitoring, fault handling, and advanced charging modes. These files abstract the complexity of direct register manipulation behind an intuitive API, making it easy to integrate and control the charger in the ElonMux project while ensuring safe and efficient battery charging across different operating conditions.

3.2 BQ25756E Battery Charger

BQ25756E is a I2C-controlled, 1 to 7 cell, battery charger controller. It was partially chosen because it is from Texas Instruments, which means they provide a complete Excel sheet for component dimensioning, a development board for layout and component choice guidelines, clear documentation, high reliability and so on.

We need a buck boost charger, as the current 24V supply is not sufficient to charge the battery to 28.7 V.

The Excel sheet with the computations can be found in the computation folder.

This charger can be easily controlled over I2C by the MCU to allow for custom charge profiles, with a controlled charge current. This means we can decide to use a very low current when the battery is low to preserve its lifetime.

3.2.1 Charger specifications

Below is a table presenting the implemented charger specifications:

Input voltage	4.2V - 33V
Nominal input voltage	optimized for 24V
Maximum input current	20A
Switching frequency	600kHz
Regulation target (full charge)	28,32V - currently real voltage is 27.9V after completed charge cycle
Maximum charge current	10A - not tested above 6A

3.2.2 Status indications and external control

STATUS LED ARE NOT FUNCTIONAL IN V1.0 - WRONG CONNECTION

The charger controls 2 LEDs with the following functions:

CHARGING STATE	STAT1 ORANGE LED	STAT2 GREEN LED
Charge in progress (including recharge)	ON	OFF
Charge done	OFF	ON
Charging fault detected (TS out of range, safety timer fault, etc.)	ON	ON
Charge disabled (EN_CHG = 0, or CE pin high)	OFF	OFF

To enable the charge the EN_CHG bit must be 1 and the CE pin must be low.

A JST PH2.0 connector is placed on the PCB to be able to connect an external switch to the CE pin. The default charger state is **ENABLED**. A jumper can be soldered to disable the charge permanently for preliminary testing.

3.2.3 I2C control

The charger is controlled by the MCU over I2C at the following address: 0x6A

The library used to configure and access its register is a copy of the open source version available on my GitHub here: [bq25756e_multiplatform](https://github.com/bq25756e_multiplatform).

The code was copied instead of added as a git submodule to make easier future repository handling by new Xplore members.

3.2.4 Thermal considerations

As the PCB is expected to produce significant heat during the charging process and high discharge, it is recommended to install heatsink on the mosfets, and add a small fan.

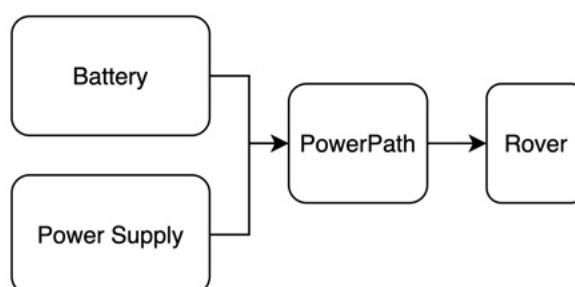
- Charging: Above 3A of charging current, it is recommended to use the fan
- PowerPath: Above 15A of discharge current, it is recommended to use the fan.

3.3 Power Path

The power path is a very critical part of ElonMux. If not designed carefully, 2 things can happen because of the delay between the moment a power source is disconnected and the other one takes over.

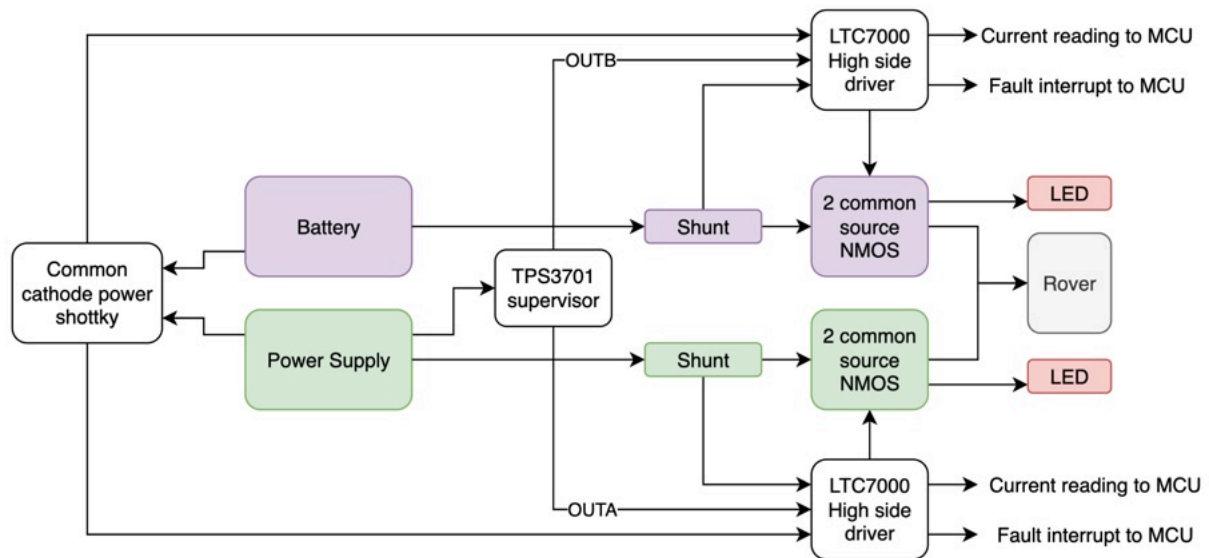
If this delay is:

- 1) **too long**, then it means that the rover will be left without power, and all the downstream components and computers will shut down.
- 2) **too short**, then the 2 power sources will be temporarily shorted together, risking damage to the battery or the power supply, and potentially ElonMux itself.



3.3.1 Topology

Below is a high level block diagram of the power path topology:



Working principle:

The TPS3701 is a window voltage detector with internal reference for over and undervoltage monitoring. But if you connect it in a specific way, you can use it as a power supply detector. Basically, if the power supply is connected, one of its outputs will be high and the other one low. If it is not connected, the output will be reversed. That way we ensure that the 2 outputs are never high at the same time.

Input INA & INB	OUTA	OUTB	Rover powered by
~0V	LOW	HIGH	Battery
VSUPPLY	HIGH	LOW	Power Supply

OUTA controls the LTC7000 driver for the power supply path MOSFETs.

OUTB controls the LTC7000 driver for the battery path MOSFETs.

The two LTC7000 provide analog feedback to the MCU on the current being drawn by the rover. They can also monitor faults, and send an interrupt to the MCU. Otherwise they are fully independent from the rest of the PCB. This was done to completely eliminate the risk of a software or boot up issue causing the power path to fail, which means a risk of damage to the PCB or battery/supply.

3.3.3 Delay estimations

The load being large, complex and capacitive, inrush current must be limited, and the turn on time needs to be slower than the turn off. The pull-up gate drive to the power MOSFET from TGUP is passed through an RC delay network, RG and CG, which greatly reduces the turn-on ramp rate of the MOSFET. Since the MOSFET source voltage follows the gate

voltage, the load is powered smoothly from ground. This dramatically reduces the inrush current from the source supply and reduces the transient ramp rate of the load allowing for slower activation. The turn-off of the MOSFET is not affected by the RC delay network as the pull-down for the MOSFET gate is directly from the TGDN pin.

By doing this we ensure rapid turn OFF, and slow power ON of the power source taking over. With the current values for RG and CG, we should expect around 0.1V/us of slew rate, meaning it will take around 240us for the supply rail to reach 24V when a 24V power supply is plugged. See detailed calculation on the PCB schematic, section powerpath.

In summary:

- The battery is plugged and powers the rover.
- We plug the power supply, the battery is disconnected in around 30us.
- The power supply slowly takes over and reaches full amplitude in around 240us.
- -> We have around 100 us of power outage. This should be fine given the amount of capacitance on the bus down stream.

3.4 Connectors

3.2.4 Power Connectors

The power connectors are XT-6, with female for the power outputs, and male for the power inputs. They can handle 60A.

A TVS diode is placed close to each power connector, with **working voltage of 33V**, and **clamping voltage of 53.3V**.

No reverse polarity protection on those connectors, please follow generally accepted standards!

3.2.4 Other Connectors

ElonMux also features a standard USB-C connector that can be used for programming and powering the MCU with 5V and up to 3A.

A FTSH-107-01-F-DV-K is used for the JTAG interface.

A standard 2x4, 2.54mm pitch pin socket is used for UART and the OLED screen.

JST PH2.0 is used to connect the charging enable button.

4 PCB Layout

4.1 Stackup

I went with a 4 layer design because 6 layers seemed daunting as my first ever PCB. However, as you might know, it is no harder to go with 6 layers instead of 4 but anyway.

The stackup is the following, it's the standard for 2oz OUT, 1oz IN.

Layers: 4 PCB Thickness: 1.6 Inner Copper Weight: 1oz Outer Copper Weight: 2oz Unit: mm

Impedance Configure + New Impedance Duplicate Impedance Calculate

Impedance (Ω)	Type	Signal Layer	Top Ref	Bottom Ref	Trace Spacing (mm)	Impedance trace to copper (mm)
90	Differential Pair (Non coplanar)	L1	/	L2	0.2	/

JLC041621-3313(Finished thickness1.61mm±10%) JLC041621-7628(Standard/Finished thickness1.64mm±10%) JLC041621-7628A(Special/Finished thicki

Impedance (Ω)	Type	Signal Layer	Top Ref	Bottom Ref	Trace Width	Trace Spacing	Impedance trace to copper
90	Differential Pair (Non coplanar)	L1	/	L2	0.2764	0.2000	/

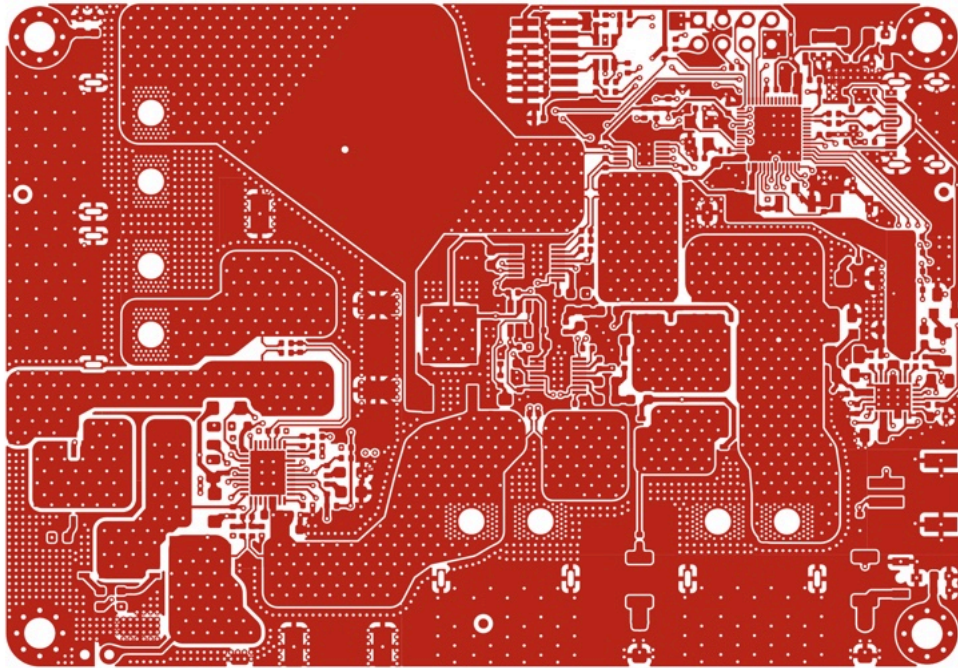
Layer	Material	Thickness (mil)	Thickness (mm)
L1	Outer Copper Weight2oz	2.76	0.0700
Prepreg	7628, RC 49%, 8.6 mil	7.99	0.2030
L2	Inner Copper Weight	1.18	0.0300
Core	1.1mm 1/1OZ with copper	40.55	1.0300
L3	Inner Copper Weight	1.18	0.0300
Prepreg	7628, RC 49%, 8.6 mil	7.99	0.2030
L4	Outer Copper Weight2oz	2.76	0.0700

Layer 1	PWR + SIGNAL
Layer 2	GND
Layer 3	GND
Layer 4	PWR + SIGNAL

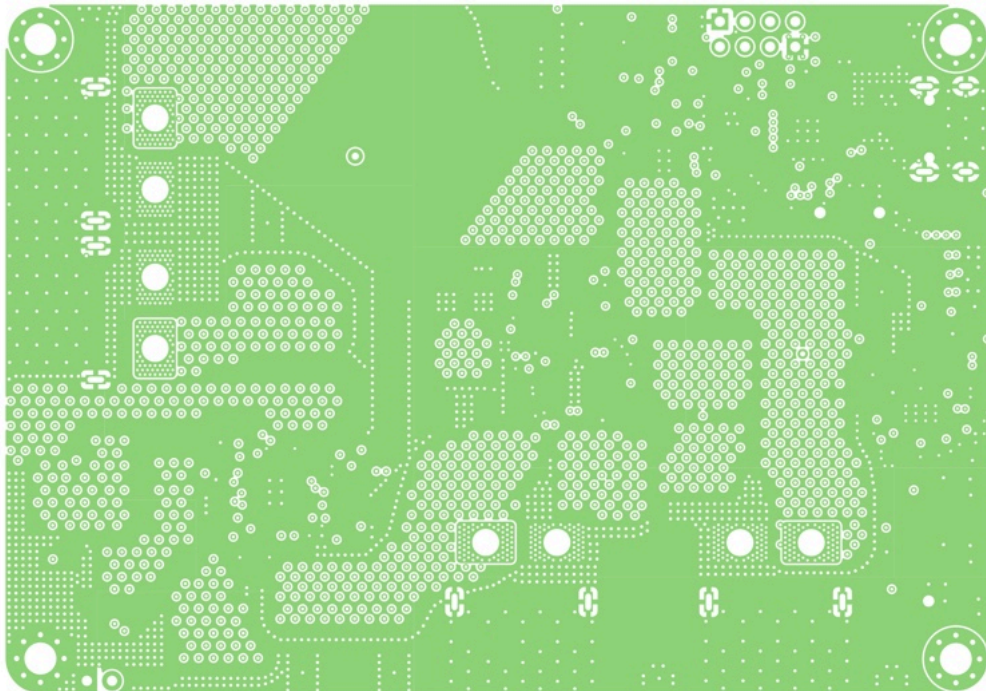
No traces nor copper pour outside of GND are routed in the 2 inner layers.

4.2 Power Routing

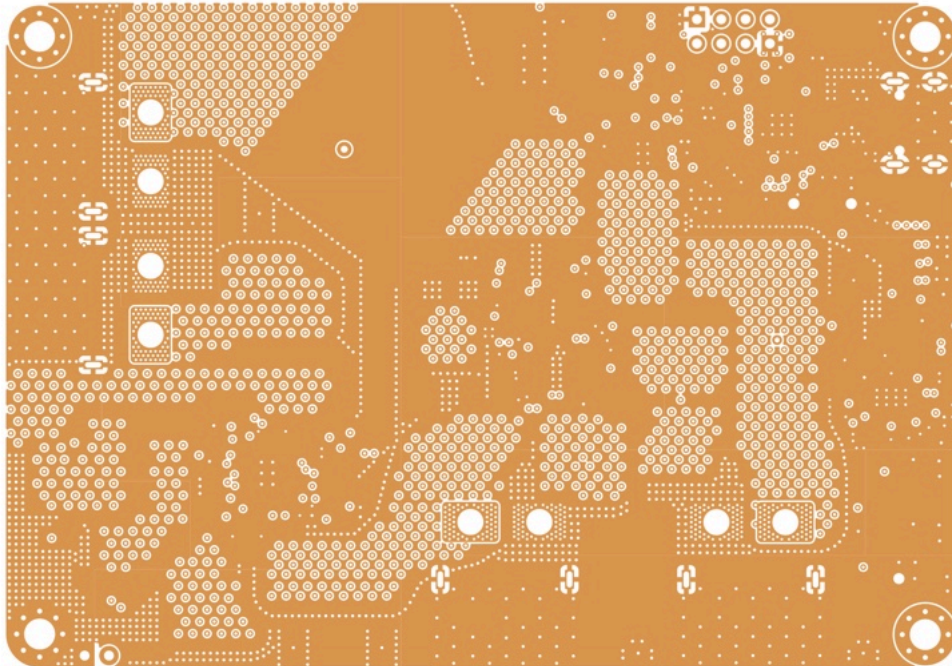
L1 (Sig, PWR)(Scale 1:1)



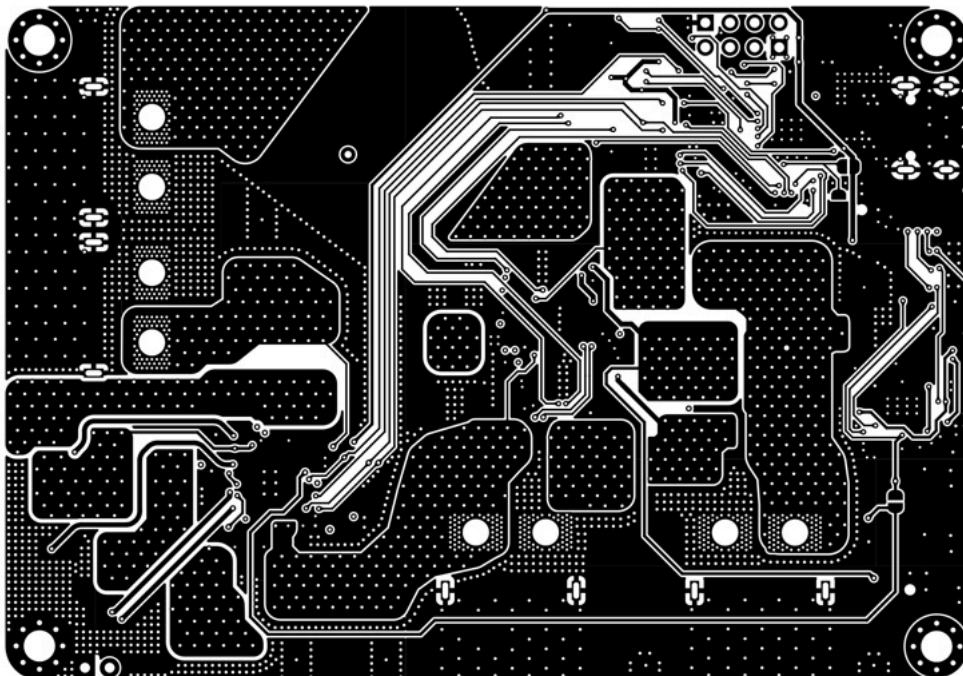
L2 (GND)(Scale 1:1)



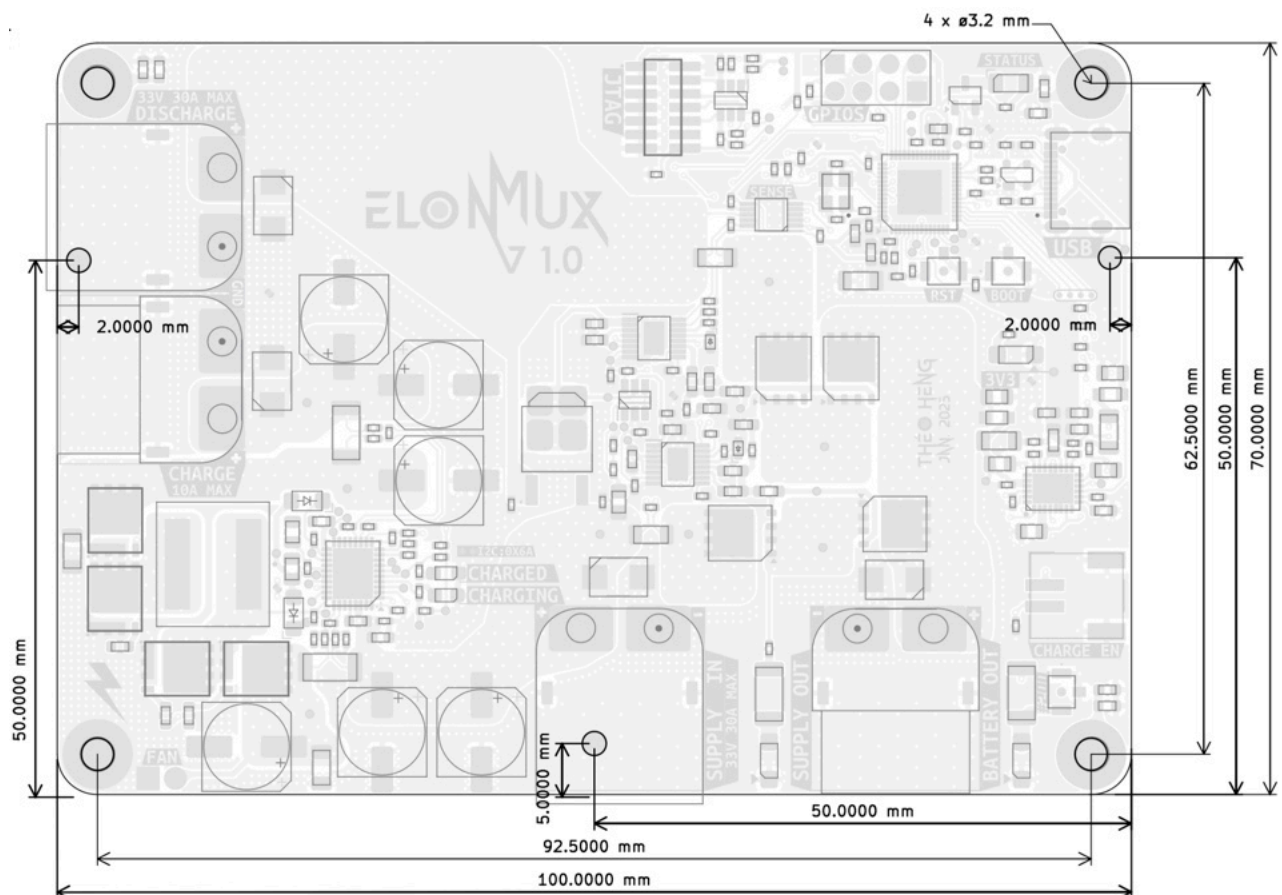
L3 (GND)(Scale 1:1)



L4 (Sig, PWR)(Scale 1:1)



5 Mechanical considerations and dimensions



Dimensions: 100x70x14mm

Weight: TBD

4 mounting holes in a 92.5x62.5mm square.

Step file of the PCB can be found in the STEP folder.

Step file of the casing can be found in the STEP folder.

6 Orderables Specifications and Price History

6.1 PCB options to select for order

Here is a list of the most important options to select on JLCPCB to ensure proper PCB function:

- 4 layers
- Outer Copper weight: 2 oz (necessary to have enough current capabilities)
- Inner copper weight: 1 oz (necessary to have enough current capabilities)
- Via covering: Epoxy Filled & Capped (necessary because of via-on-pads)
- Minimal via hole size/diameter: 0.25mm/(0.35/0.4mm) (necessary to route dense areas)
- Impedance control: Yes -> no requirement default stackup JLC041621-7628 (necessary for USB)
- Stencil: custom size + electropolishing (necessary to reduce costs of shipping and ensure additional precision for thin-pitched ICs)

6.2 Price History

V1.0

5 PCB on JLC PCB	114.57\$ (shipping and taxes excluded)
Components for 2 PCBs on Mouser	194,85 CHF (shipping and taxes included)
Total for order:	≈ 300 CHF
Total for 1 assembled PCB:	≈ 150 CHF

7 Testing and Results

7.1 Assembly

The assembly process went perfectly fine, and no problems whatsoever were observed.

However the stencil ordered with this pcb is a bit small and the solder paste was a bit hard to lay out perfectly but it's doable. **For V1.1, order a bigger stencil.**

7.2 Charger

Charger works great.

7.3 Power Path

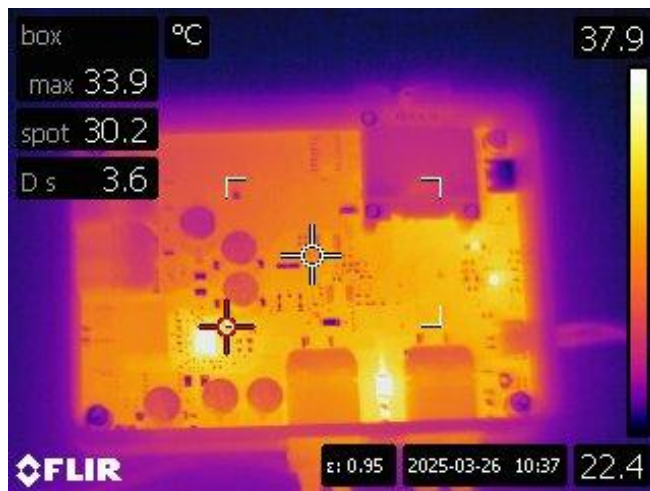
The power path works perfectly, however the delay between the switch is a tiny bit too long and it reboots the ESP32 MCU when switching from one source to the other. However this is fine when a large capacitive load is connected like Pollux III.

7.4 Additional features

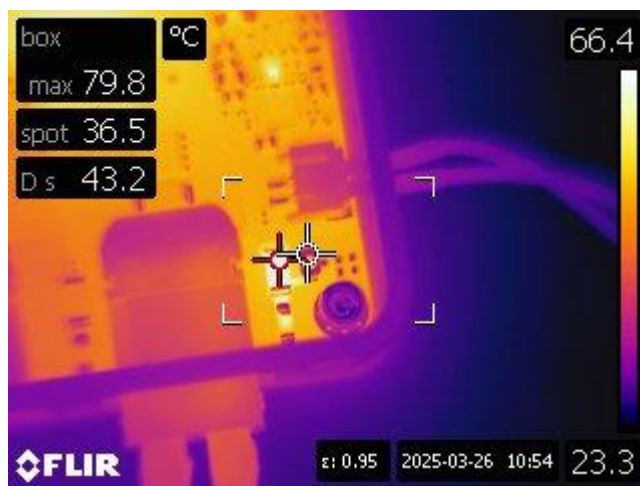
- The I2C screen works perfectly and can be used to display various information.
- The 3v3 switching regulator works perfectly and it is possible to get from it the current power source for the MCU.
- Fan can be controlled by a switch on the left side of the casing
- CC (fast charge) Current can be controlled via the 2 small buttons on the top of the casing

7.5 Thermal images

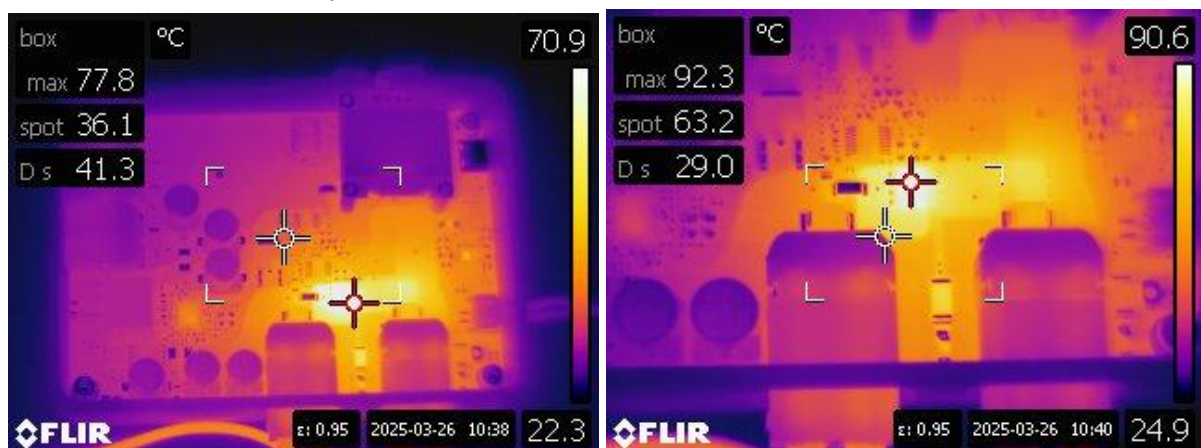
Regular operation powered by 24V supply. Heat is concentrated in the current limiting resistor of the LEDs, and the ICs.



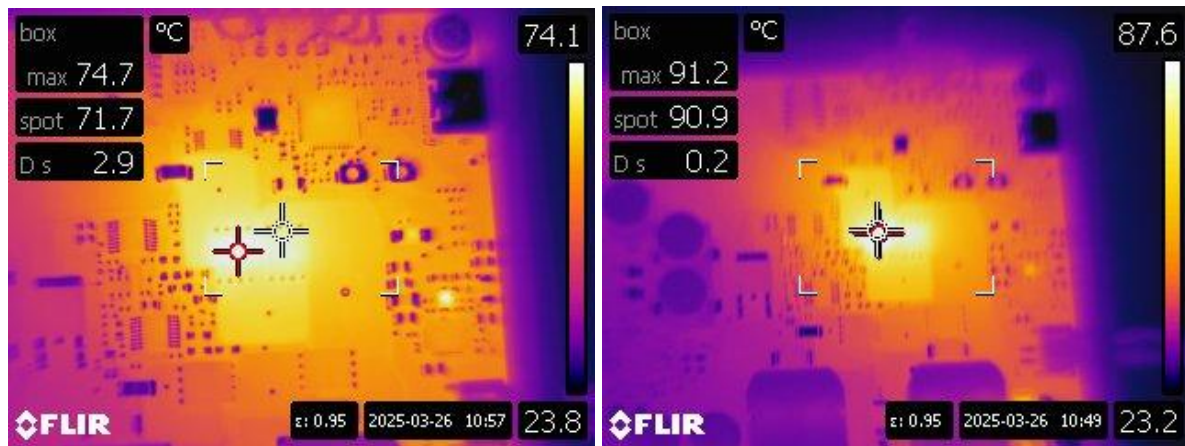
Output resistors of the output LEDs are always around 50 degrees due to the large forward drop across them. They will heat up to around 80 degrees under heavy discharge current due to the overall increase in temperature of the PCB. Active cooling solves this.



Temperature of the supply path MOSFETs after 1-2min at 24V-30A continuous.

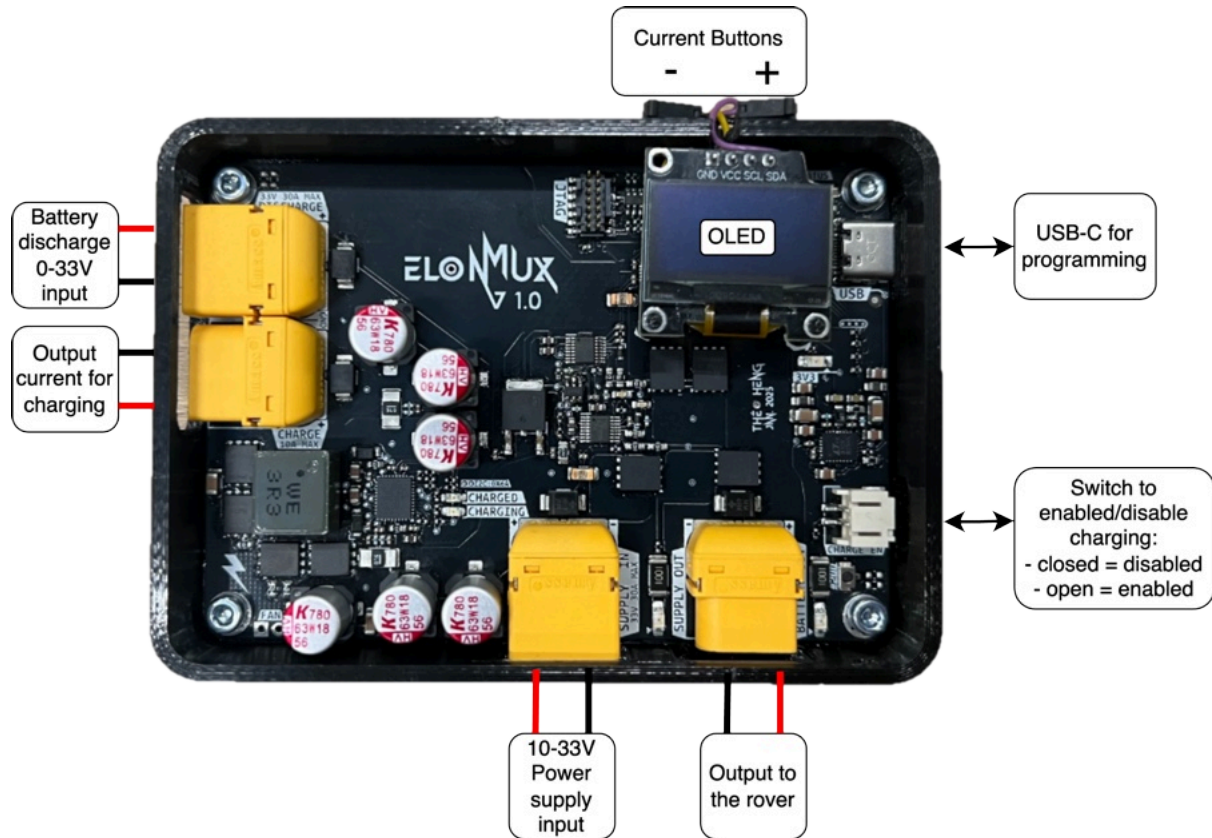


Temperature of the battery discharge path MOSFETs after 1-2min at 28V-30A continuous.



8 How-to-Use

8.1 Wiring diagram



8.2 Charging procedure

To start the charge simply flip the charge switch. The current will be automatically adjusted depending on the battery voltage, but can be adjusted at any moment during the CC phase via the 2 current control buttons.

The default current values are:

- $0V < V_{BAT} < 20.48V \rightarrow PRECHARGE = 0.5A$
- $20.48V < V_{BAT} < 23V \rightarrow CC = 1A$
- $23V < V_{BAT} < 24V \rightarrow CC = 1.5A$
- $24V < V_{BAT} < 28.3V \rightarrow CC = 3A$

9 Discussion

For a V1.0 Elon works great, but a few things could be improved in an eventual V1.1

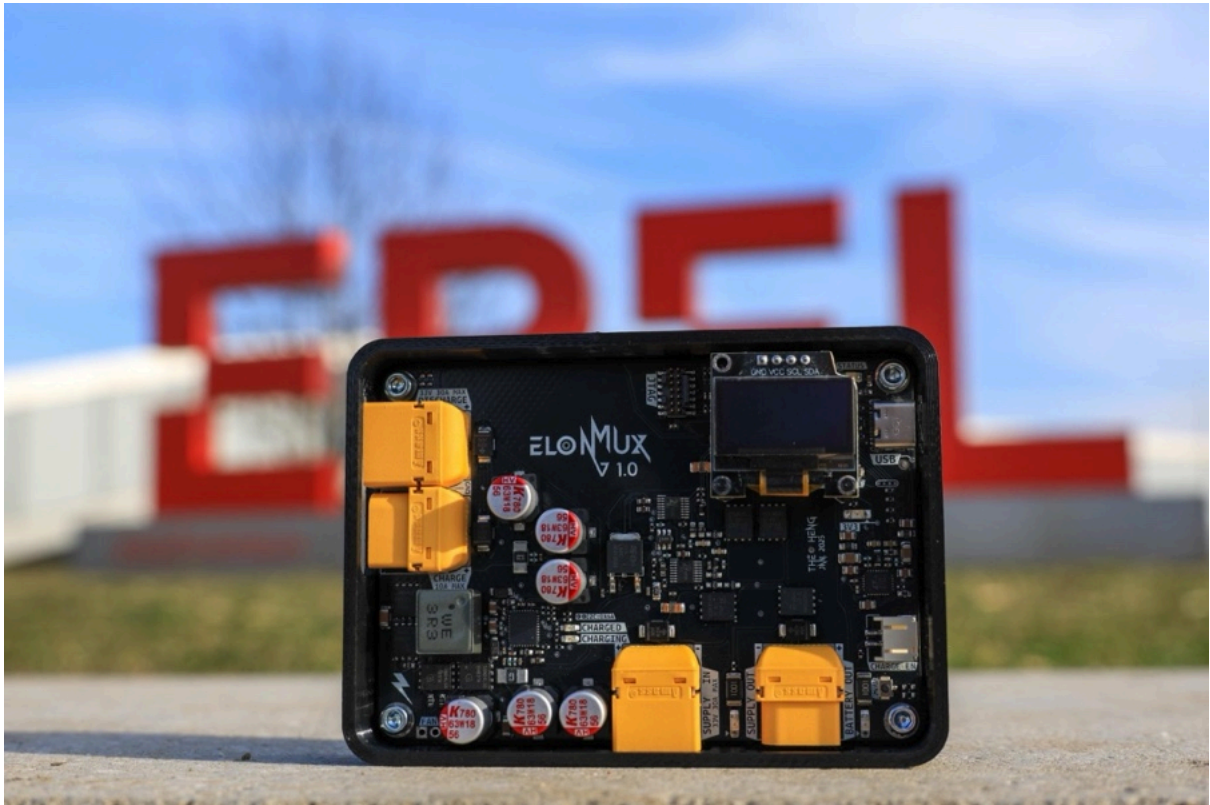
9.1 Suggestions for V1.1

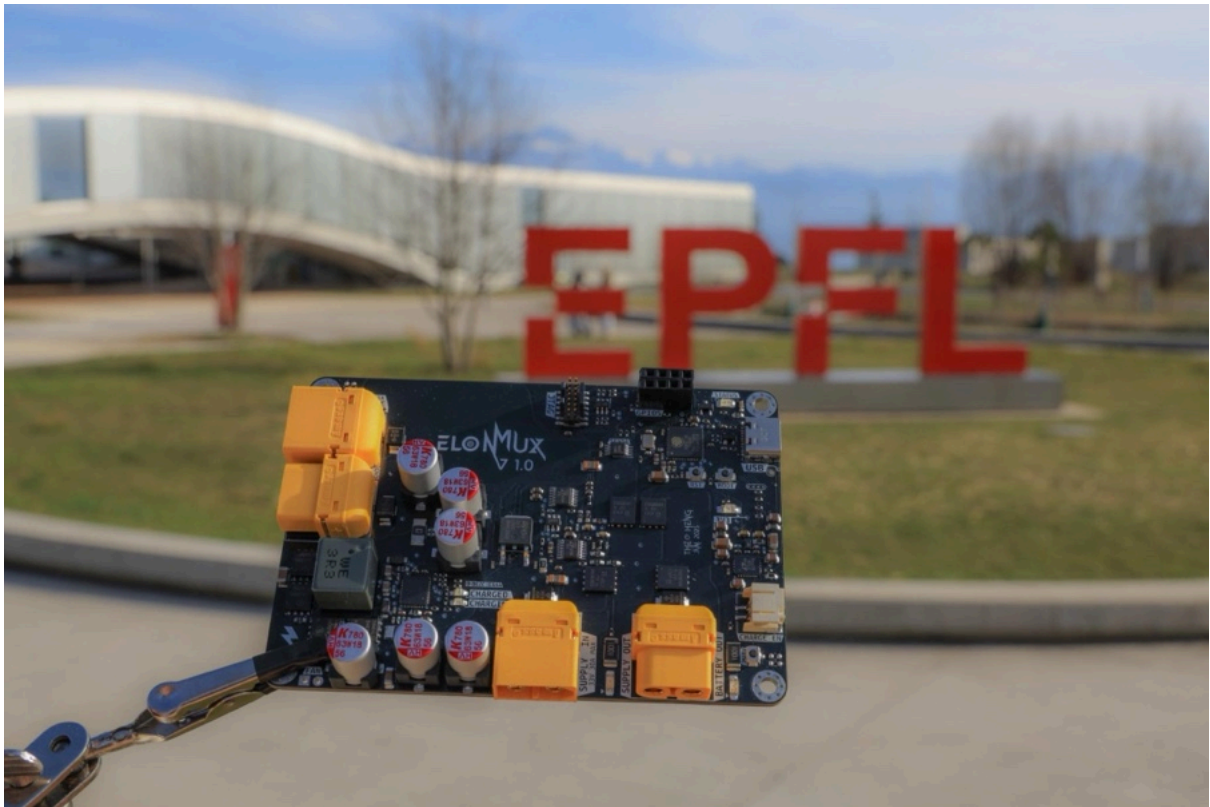
- Implement reverse polarity protection on all XT-60 connectors. Even though the connectors are placed in a logical and standardized way, a wrongly solder wire connected to ElonMux could break it.
- Go with 6 layers to increase current capabilities, and save PCB space to optimize routing further, or to go with 0.5oz internal layers if it saves cost.
- Fixe charging LEDs
- Add control for the fan
- Order bigger stencil
- change the power path mosfets for ones with lower R_{DSOn} and larger current rating
- remove the INA chip (useless) to save money

10 References

Vincent Nguyen Kicad template: https://github.com/nguyen-v/KiCAD_Templates

11 Photos





Thanks for reading !



BTW, did you spot the small hardware easter egg ?

