DESIGN AND IMPLEMENTATION OF A VOLTAGE REGULATOR FOR A DRIVERLESS VEHICLE

A Degree Thesis

Submitted to the Faculty of the

Escola Tècnica d'Enginyeria de Telecomunicació de Barcelona

Universitat Politècnica de Catalunya

by

Pol Codina Vilanova

In partial fulfilment

of the requirements for the degree in

TELECOMMUNICATION TECHONOGIES AND SERVICES ENGINEERING

Advisor: Domingo Biel Solé

Barcelona, October 2020

Abstract

Formula Student is a worldwide competition where students manufacture formula race cars. Last year, the Driverless UPC team, formed by ETSEIB and ETSETB students, adapted all the electronic systems of a classic electric formula car, to transform it into an autonomous car. The new car brought in new challenges, the added computers, actuators and cooling devices increased the total current consumption from the regulated 24V supply.

The first year we solved the issue with commercial DC-DC regulators but they turned out to be expensive and did not integrate properly with the already stablished electronics package. So, for the second year we decided to develop our own converters that would suit better our needs.

This project aims to develop a proof of concept for a DC-DC converter that works with the already stablished electronics package, using the well-known STM32 microcontroller and using it as a power device for a buck-boost DC-DC converter.

Resum

Formula Student és una competició mundial on els estudiants fabriquen cotxes de carreres de tipus fórmula. L'any passat, l'equip Driverless UPC, format per estudiants de l'ETSEIB i l'ETSETB, va adaptar tots els sistemes electrònics d'un cotxe de fórmula elèctrica per transformar-lo en un cotxe autònom. El nou cotxe va comportar nous reptes, els ordinadors, actuadors i dispositius de refrigeració afegits van augmentar el consum total actual de subministrament regulat de 24 V.

El primer any vam resoldre el problema amb reguladors DC-DC comercials, però van resultar ser cars i no es van integrar correctament amb el paquet electrònic ja establert. Per tant, al segon any vam decidir desenvolupar els nostres propis convertidors que s'adaptessin millor a les nostres necessitats.

Aquest projecte té com a objectiu desenvolupar una prova de concepte per a un convertidor de DC-DC que funcioni amb el paquet electrònic ja establert, utilitzant el conegut microcontrolador STM32 i fent-lo servir com a dispositiu de control per a un convertidor de DC-DC de tipus Buck-boost.

Resumen

Formula Student es una competición mundial en la que los estudiantes fabrican autos de carreras de tipo fórmula. El año pasado, el equipo Driverless UPC, formado por alumnos de la ETSEIB y la ETSETB, adaptó todos los sistemas electrónicos de un coche de fórmula eléctrico clásico, para transformarlo en un coche autónomo. El nuevo automóvil trajo consigo nuevos desafíos, las computadoras, actuadores y dispositivos de enfriamiento agregados aumentaron el consumo total de corriente del suministro regulado de 24V.

El primer año resolvimos el problema con los reguladores DC-DC comerciales, pero resultaron ser costosos y no se integraron correctamente con el paquete electrónico ya establecido. Entonces, para el segundo año, decidimos desarrollar nuestros propios convertidores que se adaptaran mejor a nuestras necesidades.

Este proyecto tiene como objetivo desarrollar una prueba de concepto para un convertidor DC-DC que funcione con el paquete electrónico ya establecido, utilizando el conocido microcontrolador STM32 y utilizándolo como dispositivo de potencia para un convertidor DC-DC buck-boost.

Dedicated to all my family and specially to my parents and my grandparents who raised me and educated to be the man I am today. Also, to my friends from high school and college who helped me over the years.

Acknowledgements

I wish to express my sincere thanks to all the team members of the Driverless UPC project, in particular to the team leaders for the 2020 season and for the cofounders of the team, who allowed me to take part in this amazing project and have made all this possible. Also, the 2021 members of the electronics department who allowed me to keep using the workshops and will keep pushing for innovation and expand the research from this project further.

I also thank to my Project Supervisor, Domingo Biel, for guiding me through my thesis.

And finally, I place on record, my sense of gratitude to one and all who, directly or indirectly, have lent their helping hand in this venture.

Revision history and approval record

Revision	Date	Purpose
0	01/04/2020	Document creation
1	30/09/2020	Document revision

DOCUMENT DISTRIBUTION LIST

Name	
Pol Codina Vilanova	
Domingo Biel Solé	

Written by:		Reviewed and approved by:	
Date	01/04/2020	Date	30/09/2020
Name	Pol Codina	Name	Domingo Biel
Position	Project Author	Position	Project Supervisor

Table of contents

List of Figures

List of Tables:

1. Introduction

Before talking about the project itself, let's introduce what the Driverless UPC team consists on and what Formula Student is. To do that, Formula Student Germany (FSG), the best competition inside Formula Student, defines the project as:

[1]"Students build a single seat formula racecar with which they can compete against teams from all over the world. The competition is not won solely by the team with the fastest car, but rather by the team with the best overall package of construction, performance, and financial and sales planning."

Figure 1.1: Formula Student Germany 2018

"Formula Student challenges the team members to go the extra step in their education by incorporating into it intensive experience in building and manufacturing as well as considering the economic aspects of the automotive industry. Teams take on the assumption that they are a manufacturer developing a prototype to be evaluated for production."

"The challenge the teams face is to compose a complete package consisting of a well constructed racecar and a sales plan that best matches these given criteria. The decision is made by a jury of experts from the motorsport, automotive and supplier industries. The jury will judge every team's car and sales plan based on construction, cost planning and sales presentation. The rest of the judging will be done out on the track, where the students demonstrate in a number of performance tests how well their self-built racecars fare in their true environment."

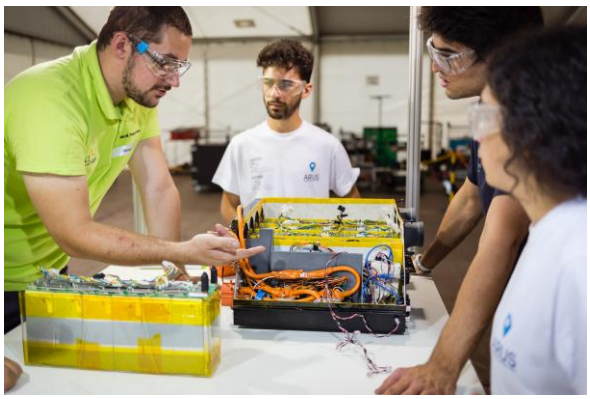


Figure 1.2: Scrutineering event at FSG

Driverless UPC was founded two years ago with a clear objective in mind: create the first Autonomous Formula Student Car in Spain. In its first year, the team did not have any kind of background, which implied a real challenge because a lot of work had to be made from scratch. However, thanks to all the team members effort the car was able to complete all dynamics event at the last competition in Spain, being one of the few cars to success in its first year. The car ran, but it did slowly (only 10 km/h) and the electrical package had some serious reliability problems which made the team lose valuable testing time.

Taking all these into account, the team members set the goal of this second year: the car should be electrically and mechanically more reliable and faster. Having a more robust electrical system would allow the team to test during longer periods of time the autonomous systems and, as a result, the team would have more time to balance and debug all the control and perception algorithms. This fact would enable us to reach higher speeds at dynamics events. Therefore, the current Hardware and Electronics department work is to gather all the designs that were developed the previous year to improve them and correct all the failures that were made due to the lack of time.

In particular this thesis describes the design and development of the proof of concept for a buck-boost DC-DC converter that aims to create a better solution for the team to other commercial alternatives.

1.1. Objectives

There are many commercially available options for buck-boost converters, but we wanted an option that could be integrated in a multipurpose PCB. Due to the limited team's budget it needed to be cost effective and cheaper than commercial alternatives.

Furthermore, since the converter was meant be used in a student developed prototype which would be constantly manipulated and pushed to the limit, it required resistance to the harsh environment besides having adequate safety measures.

Finally, we wanted to be able to monitor as many parameters of the converter as possible and transfer all these measurements into an already existing CAN network so we could troubleshoot issues with the car more easily and faster.

1.2. Requirements and specifications

1.2.1. Requirements

- Provide a stable output voltage from all the possible ranges of input voltages.
- Supply the expected current without overheating.
- Must be able to communicate with other devices through the stablished CAN bus
- Proper safety measures for overcurrent, overvoltage and undervoltage
- Fast response to current spikes

1.2.2. Specifications

- Input voltage range: 21-30V
- Output voltage: 24V
- Maximum output current: 10A
- Low power dissipation in transistors: <1W
- Low voltage ripple of the DC-DC regulators: <5%
- Low current ripple of the inductor: <30%

1.3. Work plan

The work plan is divided into two main projects:

1. Design and Simulation: A theoretical study of the power stage and the necessary control loop, accompanied by the required simulations to validate the design before implementing it.
2. Hardware validation: Design a PCB with the needed capabilities and test if the designed power stage and control behave as expected and meet the requirements.

1.3.1. Work Packages

<table>
<tr><td>Project: Power Stage design</td><td colspan="2">WP ref: 1</td></tr>
<tr><td>Major constituent: electronics design</td><td colspan="2">Sheet n of m</td></tr>
<tr><td rowspan="2">Short description: Considering the input voltage range and the desired output voltage choose a desirable configuration for the power stage</td><td colspan="2">Planned start date:01/02
Planned end date:23/02</td></tr>
<tr><td colspan="2">Start event: Set requirements
End event: Topology study</td></tr>
<tr><td>Internal task T1: Analyse commercial regulators

Internal task T2: Study advantages of different Buck-Boost configurations</td><td>Deliverables: Design of power stage</td><td>Dates:
01/02
23/02</td></tr>
</table>

Table 1.1: WP1

<table>
<tr><td>Project: Control design</td><td colspan="2">WP ref: 2</td></tr>
<tr><td>Major constituent: Simulation and analysis</td><td colspan="2">Sheet n of m</td></tr>
<tr><td rowspan="2">Short description:
Once the power stage is chosen, study its behaviour and design a suitable control</td><td colspan="2">Planned start date:24/02
Planned end date:30/04</td></tr>
<tr><td colspan="2">Start event: Analyse plant
End event: Control parameters</td></tr>
<tr><td>Internal task T1: Find the transfer functions

Internal task T2: Design the controller</td><td>Deliverables:
Design of power stage</td><td>Dates:
24/02
30/04</td></tr>
</table>

Table 1.2: WP2

<table>
<tr><td>Project: Hardware Implementation</td><td>WP ref: 3</td></tr>
<tr><td>Major constituent: Hardware design and study</td><td>Sheet n of m</td></tr>
<tr><td></td><td>Planned start date:24/05</td></tr>
</table>

Short description: Design a PCB that is capable of providing the voltage and current requirements and gives a platform for the STM32 and the CAN telemetry.	Planned end date:30/09 Start event: Design PCB End event: Validate electronics	
Internal task T1: Design a PCB with a STM32, CAN and the necessary power Stage Internal task T2: Test the hardware and upload the software and validate it.	Deliverables: optimized control	Dates: 24/05 30/09

Table 1.3: WP3

Project: Documentation	WP ref: 4	
Major constituent: Analysis of results	Sheet n of m	
Short description: Analyse the results and evaluate if the objectives are met, also document the process and study future developments	Planned start date:01/04 Planned end date:12/10 Start event: Document design End event: Study results	
Internal task T1: Analyse results Internal task T2:Study future developments	Deliverables: Documentation for future developments	Dates: 01/04 12/10

Table 1.4: WP4

1.3.2. Milestones

WP#	Task#	Short title	Milestone / deliverable	Date (week)
1	T1	Analyse commercial regulators	Commercial study	20/02
1	T2	Study Buck-Boost	Topology study	29/02
2	T1	transfer functions	Bode diagrams	08/03
2	T2	Design control	Control parameters	30/04
3	T1	Design PCB	PCB and Schematics	15/04
3	T2	Test PCB	Validation results	04/06
4	T1	Analise results	Documentation	15/06
4	T2	Future developments	Future projection	15/06

Table 1.5: Table of WPs

1.3.3. Gantt diagram

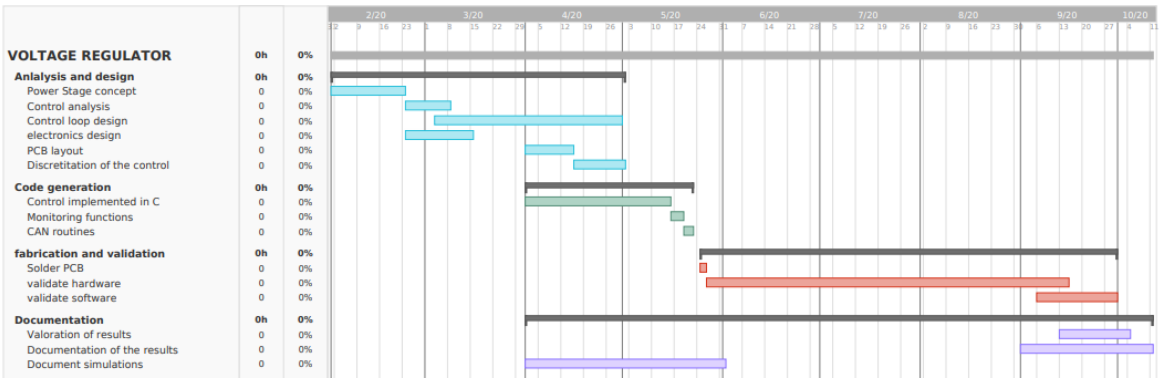


Figure 1.3: Gantt diagram

1.3.4. Meeting and comunication plan

Originally, presential weekly meetings were scheduled to evaluate the progress and set new tasks and objectives. Unfortunately due to the outbreak of COVID-19 they had to be cancelled and we adopted new means of communication such as email and occasional Google Meet videocalls.

1.3.5. Deviations from the initial plan and incidences

Hardware validation took much longer than expected, as the PCB design had some issues with the MOSFET footprint, this caused a few weeks of delay. Also, some issues caused the MOSFETs and the drivers to burn and short-circuit themselves, this caused delays in the validation of software, which was made worse by the restricted access to the workshops with hot air guns that are needed to change them.

Also due to the global pandemic of COVID-19 difficulted the hardware implementation of the design. First the manufacturing of the PCB had to be delayed until June, and from there access to the workshops was restricted and limited, during the month of August no work could be done on the campus laboratories. This delayed the presentation of the project until early October, and also limited the amount of time that could be invested in troubleshooting and debugging the hardware implementation.

The analytical side of the project, like the study of the control system and the simulations of the system could proceed as expected and followed the initial schedule.

2. State of the art of the technology used or applied in this thesis:

Throughout this chapter, the latest investigation and commercial prototypes will be described to give an overall idea of the state of the technology nowadays. Apart from that, it is going to make an emphasis in the electronics of a driverless car and the latest progresses in DC-DC regulators, as it is the main subject of this thesis.

2.1. Electronics in Autonomous Vehicles

Much of the innovation in AV systems centres on optimizing the "virtual driver," or the vehicle's brain. The virtual driver consists of machine-learning algorithms and middleware that connects to the vehicle's sensing, actuation, and communication subsystems. This technology is core to the functioning of the AV.

AVs must have a means to perceive the vehicle's surroundings to determine precise localization, accurately detect and classify fixed and moving objects, and measure the distance to those objects. Most AVs will employ a combination of sensing and perception systems, including camera-based embedded vision systems, radar, and LiDAR sensors. These sensing technologies have complementary strengths, and they can be integrated into an effective sensor suite for AVs.

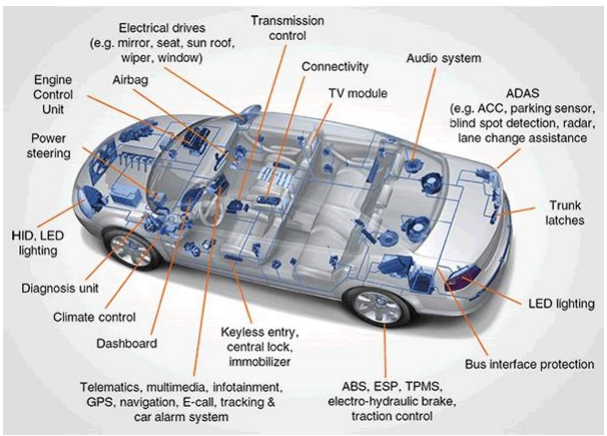


Figure 2.1: Autonomous Vehicle diagram

2.2. Electronics in a FS driverless car

The first car Driverless UPC designed took the base of the 2018 electric car and added to it the necessary hardware to make it drive autonomously. The EV car had a 600V battery pack capable of delivering 80kW, it also had a central ECU (Electronics Control Unit) designed by dSpace based on a very reliable FPGA core which communicated with the BMS (Battery management System), inverter and all the other sensors in the car. Following the trend in the automotive industry the 2018 EV car followed a distributed arquitecture for the electronics package which means that sensor data was collected by a local ECU and then send the data through the CAN bus, the idea is to reduce EMI interference in the analogue readings.

In order to make the car autonomous the software department added a central processing unit, a Nvidia development kit to run the image processing, two IMUs, a Velodyne LiDAR

and two stereo cameras. On the hardware side we added a steering actuator and two brake actuators, a pneumatic and an electric.

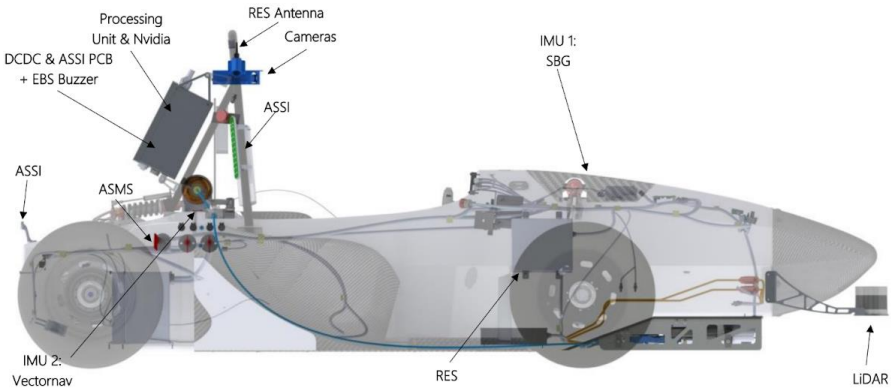


Figure 2.2: 2019 Formula Student car

The result of all the added components was a highly complex electronics package that combined a high number of analogue sensors, high speed communications and an increased power demand. In order to supply all the added components the team designed a new low voltage battery pack with a higher capacity.

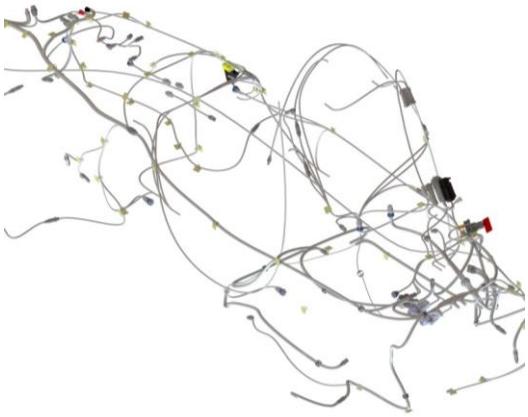


Figure 2.3: Electrical wiring distribution

The biggest power consuming devices were the actuators for the steering and braking, for the brake actuator the car had an emergency pneumatic actuator and a service brake powered by a DC brushed motor that could use up to 18A for a brief peak and 9A sustained. This actuator was powered directly from battery voltage and controlled with a PWM H-bridge driver, so it did not load the 24V supply of the car.

For the steering actuator, we decided to use a pair of industrial brushless motors, they came with their own controllers, which had to be powered from a stable 24V supply, this was not ideal as it would require another DC-DC converter that would need to supply up to 10A peak, the choice for these motors was not ideal but as the team is funded by sponsorships this where the motors we managed to acquire. For safety reasons the supply for the steering actuators would be separate from the main 24V supply for the other electronics of the car.

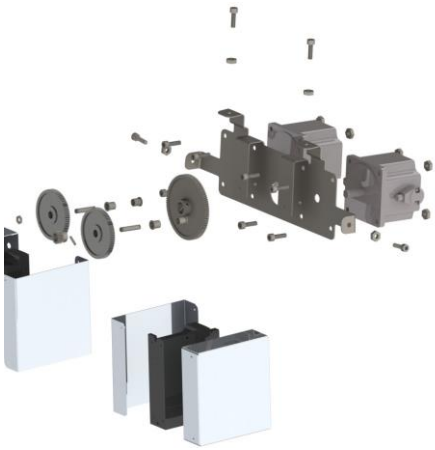


Figure 2.4: Steering actuator motor assembly

The software department decided to use two main sensors, a LiDAR and two stereo cameras, the LiDAR would be connected to a main processing unit that would run the cone detection software, in the same computer the control team also run its algorithms. The cameras would be connected to a Nvidia development kit that would run the neural network and stereo algorithms and send the data through Ethernet to the main PU. Finally, all the actuator commands were sent on the CAN bus to the main ECU, that would control the torque command to the inverter and the low level state machine for the car.

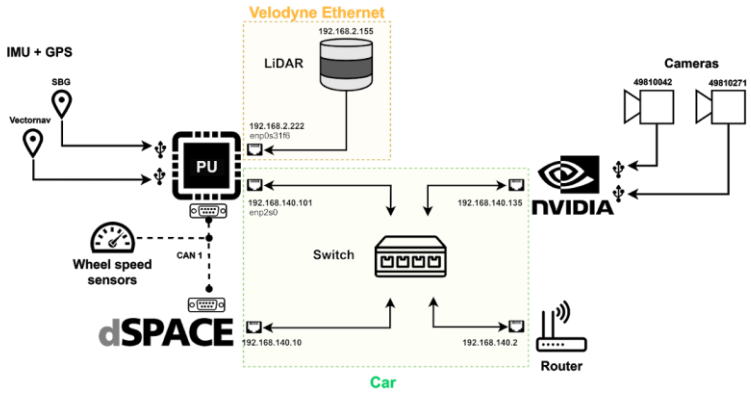


Figure 2.5: Hardware for the autonomous system

The PU (Processing Unit) was an industrial grade computer, with passive cooling and vibration proof, it had a 7th generation intel I7 processor and it could consume up to 5A but the highest current consumption measured was at 2A.

The Nvidia Jetson Xavier Developer Kit has a 512-core Volta GPU with Tensor Cores, it is powered from 19V and it could consume up to 3A continuous. The major challenge with this computer was the peak current demand. If the DC-DC converter that powers it could not deliver the current peak while maintaining a stable enough voltage, the computer shut down. Some of the DC-DC converters initially used did not have a fast-enough response time and made the computer shut down unexpectedly.

Figure 2.6: Nvidia Jetson Xavier for perception algorithms

2.3. Previously used DC-DC converters

Practical electronic converters use switching techniques. Switched-mode DC-DC converters convert one DC voltage level to another, which may be higher or lower, by storing the input energy temporarily and then releasing that energy to the output at a different voltage. The storage may be in either magnetic field storage components (inductors, transformers) or electric field storage components (capacitors). This conversion method can increase or decrease voltage. Switching conversion is often more power-efficient (typical efficiency is 75% to 98%) than linear voltage regulation, which dissipates unwanted power as heat. Fast semiconductor device rise and fall times are required for efficiency; however, these fast transitions combine with layout parasitic effects to make circuit design challenging. The higher efficiency of a switched-mode converter reduces the heatsinking needed, and increases battery endurance of portable equipment. Efficiency has improved since the late 1980s due to the use of power FETs, which are able to switch more efficiently with lower switching losses at higher frequencies than power bipolar transistors, and use less complex drive circuitry.

2.3.1. Complete hybrid circuit module

The first idea was to buy an already made DC-DC converter, that would save us time in a first year where a lot of new systems had to be designed. After some research the candidates for integrated converters turned out to be really expensive. As expected, the converter worked out of the box but had horrible efficiency under low loads (up to 11W of power consumption with no load) [2], and it would result in overheating without a cooling fan.

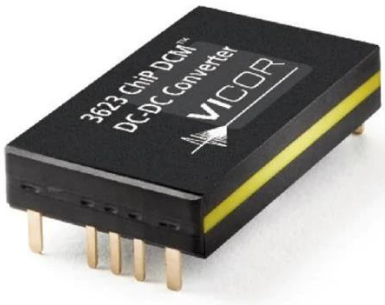


Figure 2.7: Previously used commercial DCDC converter

Also, as they are integrated packages it is impossible to repair them in case of failure, which happened often as some of them experienced short circuits and other improper uses, as it is expected from a prototype car. Also, in case we wanted to provide some telemetry like current consumption we needed to place an external current sensor on the PCB, increasing cost and complexity.

2.3.2. DC-DC controller

A cheaper alternative that was proposed is to purchase an integrated IC that controls the converter and we add the external components and design the power stage to suit our needs. This has the benefit that we can choose the newest and most efficient MOSFETs to have the best power efficiency, also all the components integrate into our PCBs acquiring the shape that best fits it.

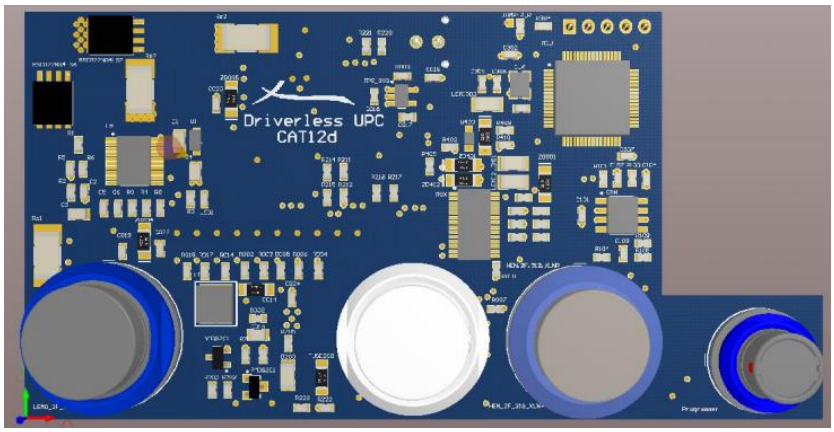


Figure 2.8: Rear ECU with DCDC converter

The results worked mostly as expected, we could fit a DC-DC converter of the same power capabilities in a similar area, the efficiency increased specially in the low current range, also the cost was greatly reduced. The only issues we had where during the initial troubleshooting we struggled to find the reasons why the IC would not work as expected, as the control IC was all integrated it was hard to find issues. Also, it did not give us the data communication we wanted, so the microcontroller had to be included either way.

3. Methodology / project development:

A key parameter that defines this design is the range of input voltages, as they come from a 7s3p lithium battery pack, the input voltage ranges from 21V to 30V, this means that a buck-boost topology was needed to obtain the 24V output desired.

After some consideration and evaluating all the commercial alternatives, we decided to design our own control stage. As all the ECUs in the car already have an STM32 microcontroller with powerful timers and ADCs, it could also run the control loop and the reading of all the necessary sensors, also the STM32 has a CAN bus module that would allow it to communicate with the rest of the car.

3.1. Control design:

First, we need to choose a topology and study its behaviour, this way we can design the digital control for our microcontroller. In order to simplify the research, we decided to start from a commercial DC-DC controller and its proposed topology, the LT8390 from Linear Technologies [3].

3.1.1. Power stage analysis:

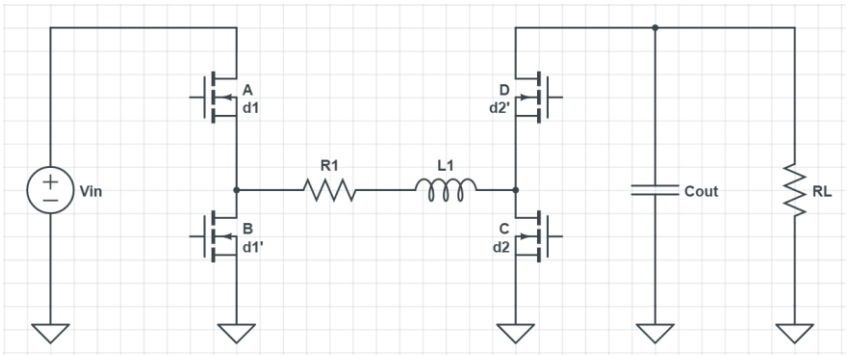


Figure 3.1: Basic topology of converter

The chosen topology for our DC/DC converter is a synchronous 4-switch buck-boost, it can regulate the output voltage above and below the input voltage, it is built around a single inductor and 4 N-channel MOSFET switches, with a current sense resistor to sample inductor current.

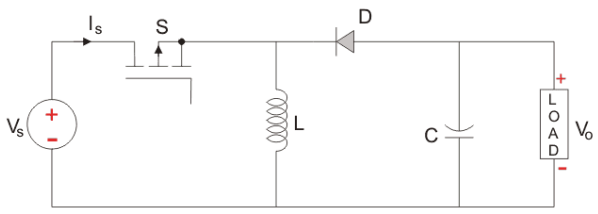


Figure 3.2: Traditional Buck-Boost topology

The 4 switch topologies present and advantage in respect to 2 switch buck-boost converters in the reduction of inductor ripple current and inductor average current, the

equations for the 2 switch controller are well known, and in a later stage we will calculate the ones for the 4 switch, here is the comparisons.

Worst case analysis	2 switch	4 switch
$\hat{i}_L$ [A]	6	3.5
$<i_L>$ [A]	17.9	13.7

Table 3.1: Buck-Boost comparison

After performing a worst case scenario study (21V Vin and 10A output current), you can see the 2-switch topology has a much higher mean current, this would require a much larger inductor, also higher resistive losses in it, the higher ripple current also is an issue in order to achieve the objectives. So, in conclusion the simplicity of the 2-switch is not worth the losses in performance for us, also as this is meant to be a proof of concept we choose the more complex option to prove the capabilities of the STM32 microcontroller.

3.1.2. Mathematical model:

In order to properly design the control loop we need to understand the behaviour of our plant, the 4 switch regulator has three modes depending which switches has closed, this pattern will repeat itself in one cycle:

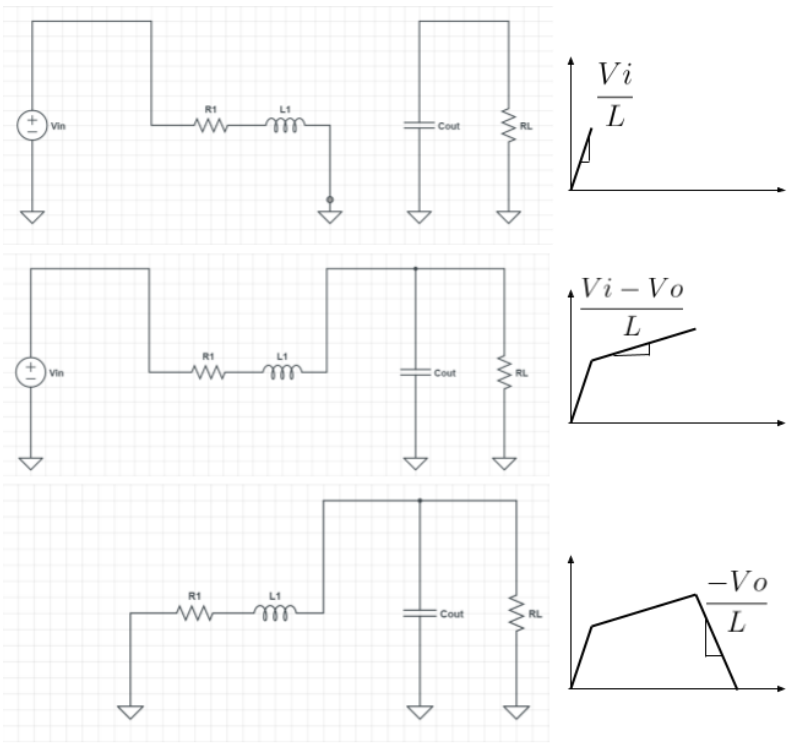


Figure 3.3: Current evolution over a period

As seen in Figure 3.3 the cycle begins with MOSFETs A and C closed, charging the inductor at a constant voltage V_{in}, this will last d2*Ts (d2 is the duty cycle of MOSFET C, Ts is the period of the cycle).

The second stage is when C opens and D closes, A is still closed, so the inductor charges to the constant voltage V_{in}-V_{out}, this will last a time of (d1-d2)*Ts, the slope will be positive when in buck region or negative in boost (d1 is the duty cycle of MOSFET A) .

Finally A opens and B closes, discharging the inductor at the constant voltage -V_{out}, this will last (d2'-d1)*Ts (d2' is 1-d2 or the duty cycle of MOSFET D).

In order to understand the behaviour of the regulator we develop the controlled supply model.

	A	B	C	D	V_{OA}	i_{OA}	V_{OB}	I_{OB}	V_{PC}	i_{PC}	V_{PD}	i_{PD}
d1=1	ON	OFF	X	X	0	is_{1N}	Vs_{1N}	0	X	X	X	X
d1=0	OFF	ON	X	X	Vs_{1F}	0	0	is_{1F}	X	X	X	X
d2=1	X	X	ON	OFF	X	X	X	X	0	is_{2N}	Vs_{2N}	0
d2=0	X	X	OFF	ON	X	X	X	X	Vs_{2F}	0	0	is_{2F}

Table 3.2: Controlled supply

Following this table we can create a controlled supply model for the converter, as we can choose from two possible models we select one and represent it to analyse it further:

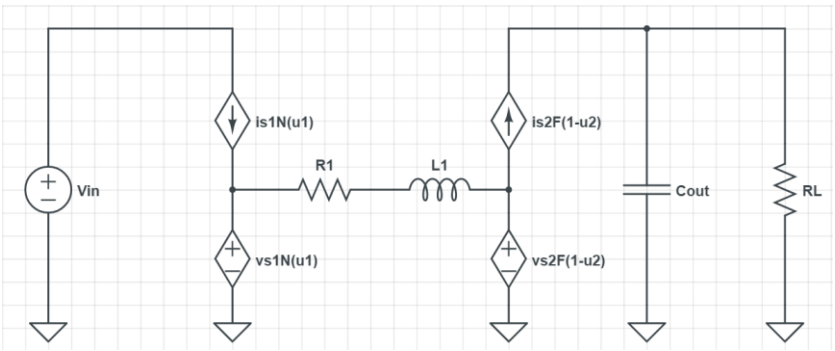


Figure 3.4: Controlled supply model

This is the complete circuit but we can represent it in a more simplified form that will allow us to perform an electrical analysis.

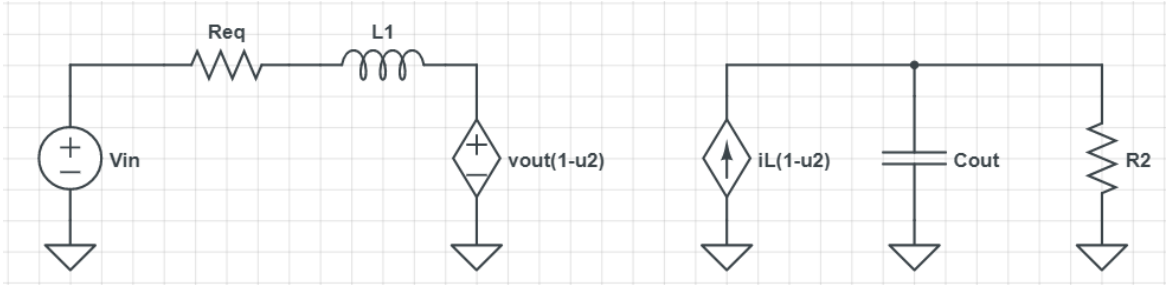


Figure 3.5: Simplified model

With this model we can extract the system of equations using basic circuit analysis techniques:

$$\begin{cases} V_{IN}u_1 = R_{eq}i_L + L\frac{\mathrm{d}i_L}{\mathrm{d}t} + V_o(1-u_2) \\ i_L(1-u_2) = i_C + \frac{V_o}{R_L} \end{cases}$$

Once we have the simplified equation we will expand it to the big and small signal model

$$\begin{cases} (V_{IN}+\hat{v}_{IN})(D_1+\hat{d}_1) = R_{eq}(I_L+\hat{i}_L) + L\frac{\mathrm{d}(I_L+\hat{i}_L)}{\mathrm{d}t} + (V_o+\hat{v}_o)(1-(D_2+\hat{d}_2)) \\ (I_L+\hat{i}_L)(1-(D_2+\hat{d}_2)) = C\frac{\mathrm{d}(V_o+\hat{v}_o)}{\mathrm{d}t} + \frac{(V_o+\hat{v}_o)}{R_L} \end{cases}$$

Now if we want to study the steady state model we can use superposition to simplify the equation:

$$\begin{cases} V_{IN}D_1 = R_{eq}I_L + V_oD_2' \\ V_o = D_2'I_LR_L \end{cases}$$

But because the voltage drop at R_{eq} is very small we can neglect it, so get the input output relation:

$$\frac{V_o}{V_i} = \frac{d_1}{1-d_2}$$

This equation will be very useful as it gives us the relation between the input/output voltages and the duty cycles. Both duty cycles affect the relationship, and we see how by playing with them the converter will behave as a buck or a boost converter.

3.1.3. Relation of d1 and d2

Playing with the relation d1 to d2' we transition from a boost behaviour to buck. If d1=1 then it behaves as a classic 2 switch boost converter, and if d2=0 then it operates as a regular buck converter. These two modes could be useful as their efficiency is higher because we only commutate 2 MOSFETs, but because our input voltage is always close to our output voltage we choose to operate in buck-boost mode.

When operating in buck-boost what defines the conversion factor is the relation between d1 and d2', so technically we could operate in the range of duty cycles we wanted. One option would be to control simultaneously d1 and d2, for example d1=d2. Another option is to set one fixed and change the other.

In order to decide the most efficient strategy we analyse the effect of the duty cycle relation with the current ripple and average current. The current ripple can be measured at the last stage of four cycle for buck mode and the first stage for boost mode as seen in Figure 3.3, we know the slope and also the time duration, it gives us the following formulas

Buck: $\hat{i}_L = \frac{V_o}{L}\frac{1-d_1}{f_s}$

Boost: $\hat{i}_L = \frac{V_i}{L}\frac{d_2}{f_s}$

Another important parameter is the inductor current in relation to the output current, as we know in buck-boost converters the inductor current is not the same as the output current

$$< i_L >= \frac{i_o}{d_2'}$$

If we choose the d1=d2 control method we quickly realise that at Vi=Vo the average current at the inductor is double the output current, this is an issue as our output current could arrive at 10A this would require a very large inductor with a saturation current of more than 20A, also the resistive losses on the inductor would increase considerably.

If on the other hand we choose to fix one of the duty cycles and control the other we can set d2=0.15 in buck mode, so our inductor current is similar to the output current and in buck d1 will be close to 1, reducing our inductor current ripple. For Boost mode we fix d1=0.85 and control d2 this will change our inductor current but should still be close to the output current. Here we see the ranges of duty cycles:

	Boost	Buck
d1	0.85	0.68-0.85
d2	0.15-0.29	0.15

Table 3.3: Duty cycle ranges

We use this duty cycles and the equations stated above to perform the comparison of worst case scenario shown previously.

3.1.4. Transfer functions

In order to design a suitable control we need to understand the behaviour of the plant, in buck and in boost mode. First we analyse the output voltage in relation to our control signal, the duty cycle and because in buck and in boost we have different control signals we search for both functions:

To find the transfer function for buck mode voltage control we eliminate the input variance and the d2 variance, leaving us with the equation system:

$$\begin{cases} Ls\hat{i}_L(s) = -\hat{v}_o(s)D_2' + V_i\hat{d}_1(s) \\ Cs\hat{v}_o(s) = \hat{i}_L(s)D_2' - \frac{\hat{v}_o(s)}{R} \end{cases}$$

From this system of equations we can obtain the following transfer function:

$$G_{c1}(s) = \frac{\hat{v}_o(s)}{\hat{d}_1(s)} = \frac{V_o {D_2'}^2}{D_1} \frac{1}{LCs^2 + \frac{Ls}{R} + {D_2'}^2}$$

To find the transfer function for boost mode voltage control we eliminate the input variance and the d1 variance, leaving us with the equation system:

$$\begin{cases} Ls\hat{i}_L(s) = -\hat{v}_o(s)D_2' + V_o\hat{d}_2(s) \\ Cs\hat{v}_o(s) = \hat{i}_L(s)D_2' - I_L\hat{d}_2(s) - \frac{\hat{v}_o(s)}{R} \end{cases}$$

If we operate this system we arrive at the final transfer function:

$$G_{c2}(s) = \frac{\hat{v}_o(s)}{\hat{d}_2(s)} = \frac{V_o}{RD_2'} \frac{-Ls + RD_2'^2}{LCs^2 + \frac{Ls}{R} + D_2'^2}$$

Now with the help of MatLab we plot this transfer functions, because we know that at Vi=Vo the transfer function changes from Gc1 to Gc2 we can plot both of them at the same time.

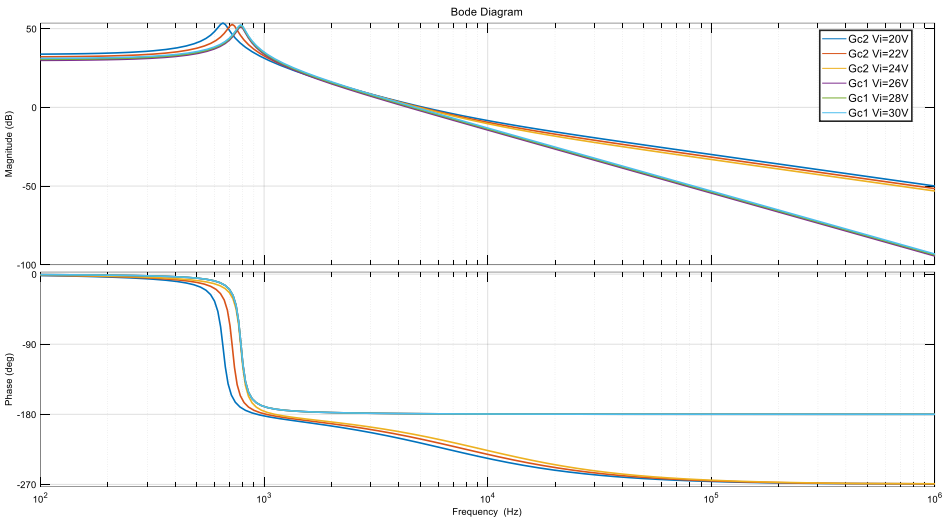


Figure 3.6: Transfer functions for different input voltages

As you can see when the plant is working on buck mode it could achieve stability with a simple PI control, as the phase gets stable at 180°, but in boost mode a 90° shift would not be enough to achieve the desired phase margin, so we need to study a more complex control solution.

3.1.5. Control design

In power electronics it is typical to apply a double control loop, it works with an internal current control loop and an external output voltage control loop. The internal current loop is much faster so it adds stability to a system that would not be stable with a simple output voltage control loop.

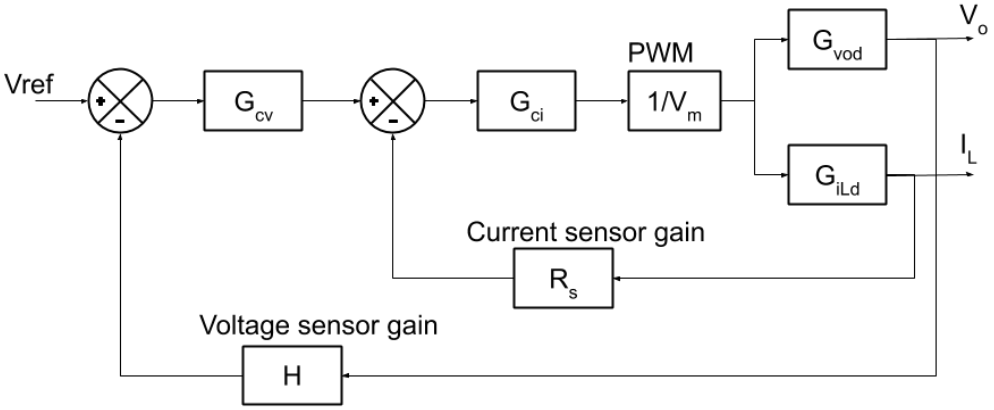


Figure 3.7: Proposed control model

We have a double control loop with an inner loop with the inductor current that will provide the stability and an outer loop with the output voltage. In order to work this new system, first we find the transfer functions for the inductor current in respect to the control signal, we analyse the system in buck and boost mode.

First, we find the transfer function for d1 control signal

$$\begin{cases} Ls\hat{i}_L(s) = -\hat{v}_o(s)D_2' + V_i\hat{d}_1(s) \\ Cs\hat{v}_o(s) = \hat{i}_L(s)D_2' - I_L\hat{d}_2(s) - \frac{\hat{v}_o(s)}{R} \end{cases}$$

If we operate the system we get the following transfer function

$$G_{I1} = \frac{\hat{i}_L(s)}{\hat{d}_1(s)} = \frac{V_i(Cs + \frac{1}{R})}{s^2LC + s\frac{L}{R} + {D_2'}^2}$$

Now we find the transfer function for the d2 control signal

$$\begin{cases} Ls\hat{i}_L(s) = -\hat{v}_o(s)D_2' + V_o\hat{d}_2(s) \\ Cs\hat{v}_o(s) = \hat{i}_L(s)D_2' - I_L\hat{d}_2(s) - \frac{\hat{v}_o(s)}{R} \end{cases}$$

If we operate the system we get the following transfer function

$$G_{I2} = \frac{\hat{i}_L(s)}{\hat{d}_2(s)} = \frac{V_0Cs}{s^2LC + s\frac{L}{R} + {D_2'}^2}$$

Now using MatLab we represent a sweep of input voltages, in order to do it we need to stablish some parameters, L=29.5µH, C=1mF and R=2.4Ω, to show that with current control even in boost mode a simple PI can achieve stability.

Inductance: Chosen from a limited set of commercial inductors, needed to have a high enough saturation current and highest possible inductance.

Output resistor: Necessary resistance to have an output current of 10A at 24V

Capacitance: Chosen from available capacitors. High capacitance to reduce output voltage ripple, this converter could be used to power computers that need a stable voltage. Also, the hardware implementation will use a combination of electrolytic and ceramic capacitors to reduce ESR.

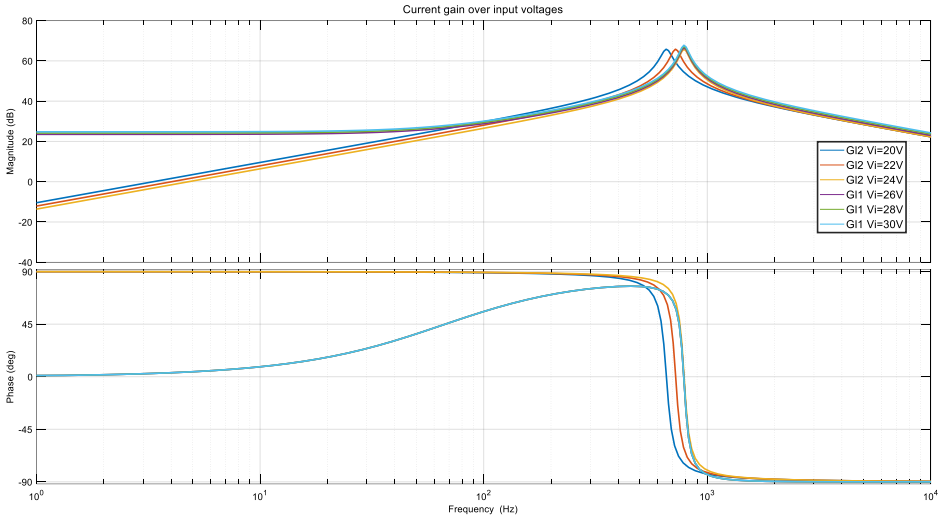


Figure 3.8: Current gain over input voltages

So first we design this current mode control, buck and boost mode have a similar response, we take the worst-case scenario: Vin=20V (Boost mode) and Iout=10A.

The current control loop has an open loop gain of:

$$T_c = \frac{R_S}{Vm} \cdot G_{ci}(s) G_{li}(s)$$

First, we plot the T_c without any control compensation so $G_{ci}(s)=1$.

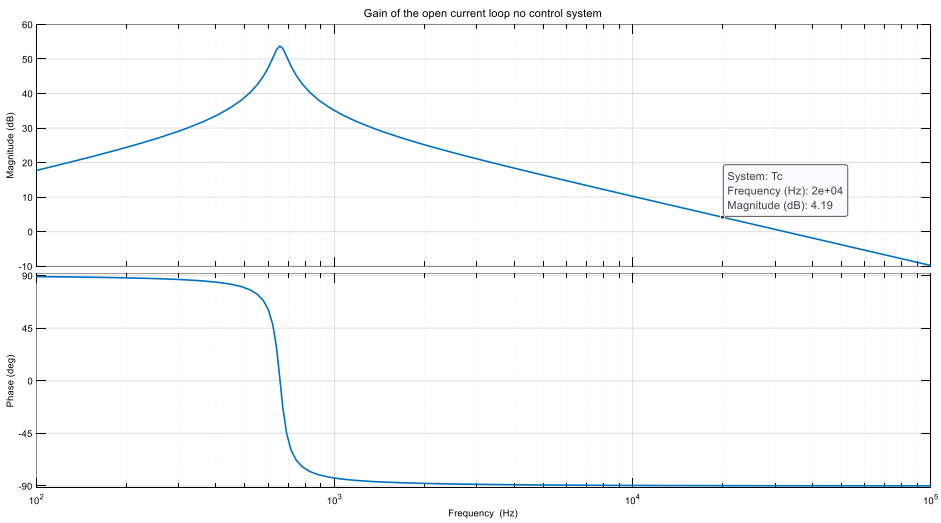


Figure 3.9: Gain of the open loop no control system

One design criteria dictates for the new crossover frequency to be five times lower than the commutation frequency. Then as shown in Figure 3.9 the gain at 20kHz is 4.21dB, so our proportional gain has to be:

$$[20log(k_p) = -4.21dB \Rightarrow k_p = 0.61]$$

We choose to locate the zero at a frequency 10 times lower than our cut-off frequency:

$$w_z = \frac{w_c}{10} = 2kHz = 12.56 \cdot 10^3 rad/s$$
$$\frac{k_i}{k_p} = 12.56 \cdot 10^3 rad/s \Rightarrow k_i = 7739$$

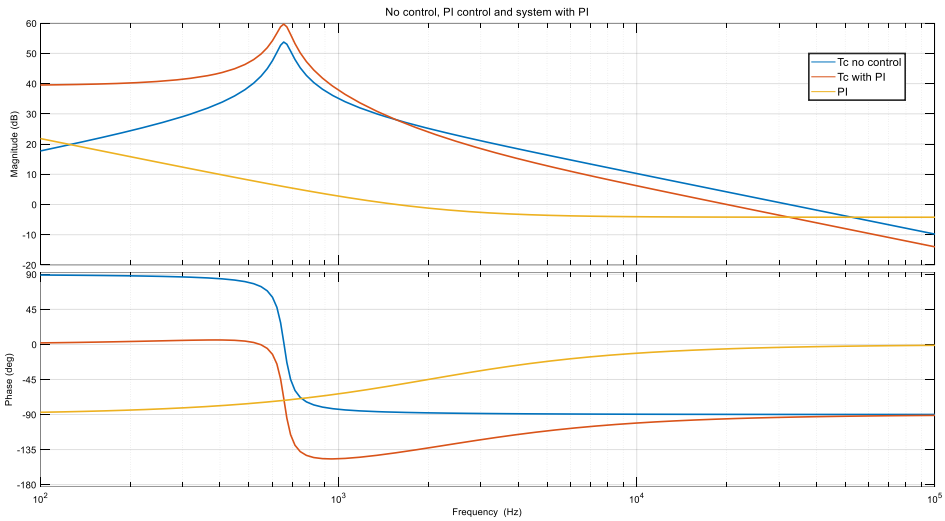


Figure 3.10: No control, PI control and system with PI

Finally, in Figure 3.10 we see how the PI control changes the system to the targeted parameters. At this point the inner current loop is completed and we can target the outer voltage control loop.

This double control loop assumes the outer loop is much slower than the inner one, so as we propose the outer loop gain we decide to ignore the inner loop gain, as it does not affect, so we set the inner loop gain to one, giving us the transfer function for the outer loop:

$$T_v(s) = G_{cv}(s)\frac{H}{R_s}\frac{G_{v_o d}}{G_{i_L d}}$$

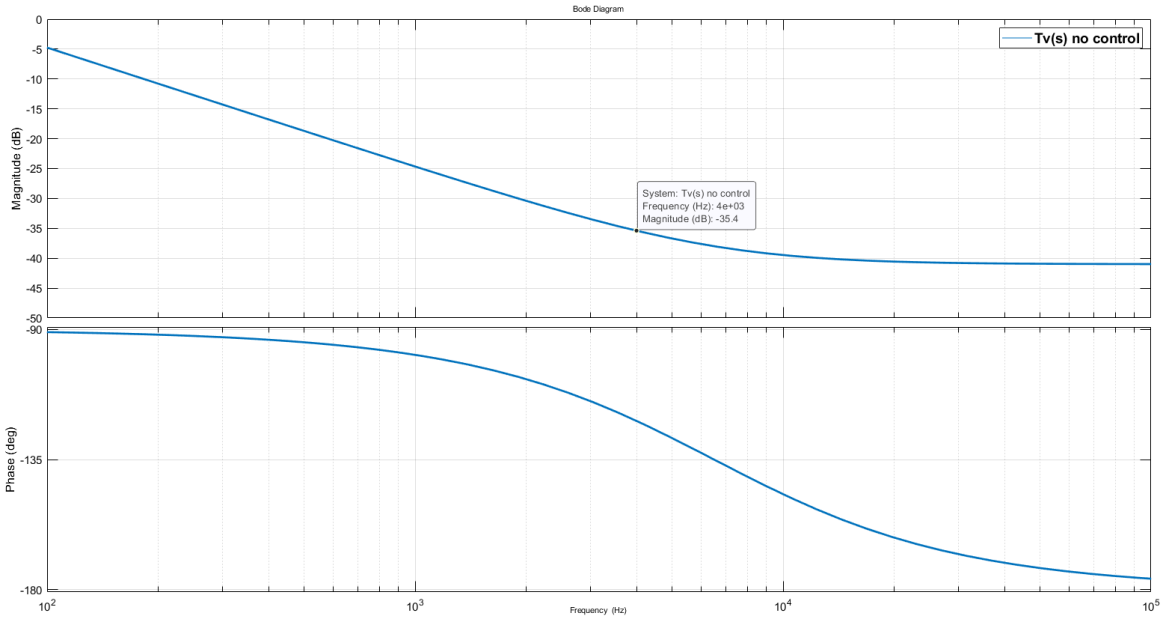


Figure 3.11: Outer loop without control

In order to have the outer loop slower than the inner we choose a frequency 5 times lower than the outer, so 4kHz where we have a gain of –35.4dB as seen in Figure 3.11

$$20log(k_p) = 35.4dB \Rightarrow k_p = 5.87$$

We choose to locate the zero at a frequency 10 times lower than our cut-off frequency, and we apply the same process as before to calculate the integral component:

$$k_i = \frac{4000}{10} 2\pi \cdot 5.87 = 14752$$

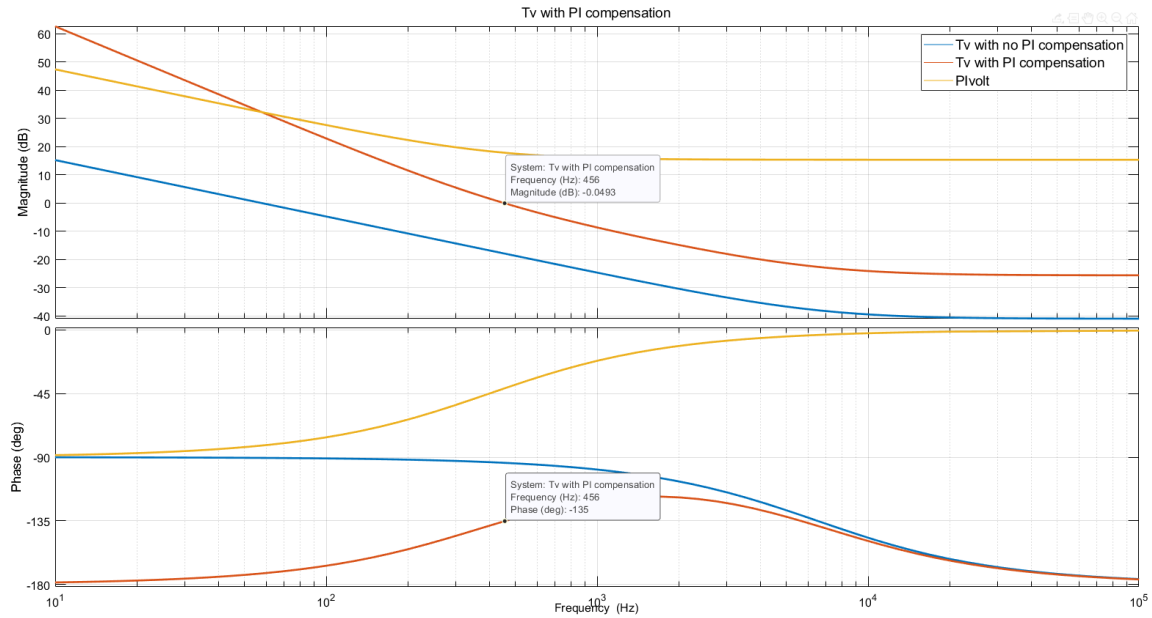


Figure 3.12: Outer loop with control, PI control, original system

As seen in Figure 3.12 the resulting phase margin will be 45°. Knowing we will have a stable response we can plot the final closed loop step response to have an idea of the theoretical performance, as ESR or resistive loses in the inductor are not considered.

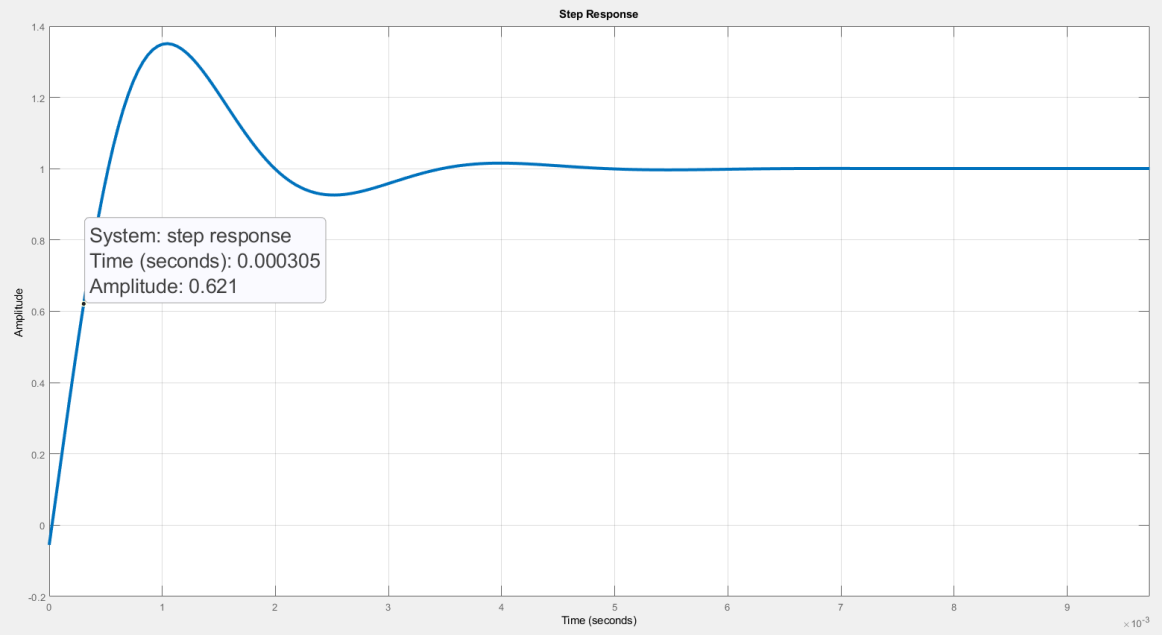


Figure 3.13: Step response of closed loop

With the final response of the system in a continuous time domain, we analyse if it can be transferred to a discrete system in order to run on the microcontroller. Following the discretisation criteria, the system needs to sample 20 times for the time constant of the system.

In the final step response the time it takes to achieve 63% of the final value is 300us as seen in Figure 3.13, and as the control loop in the microcontroller will run at 100kHz (same as the PWM frequency) the system samples 30 times in the duration of the time constant of the system. So, we can expect to implement the PI loops without modifying the constants. [4]

Also, we have to validate the system works in Buck mode, we follow the same procedure as before but using the buck equations and parameters

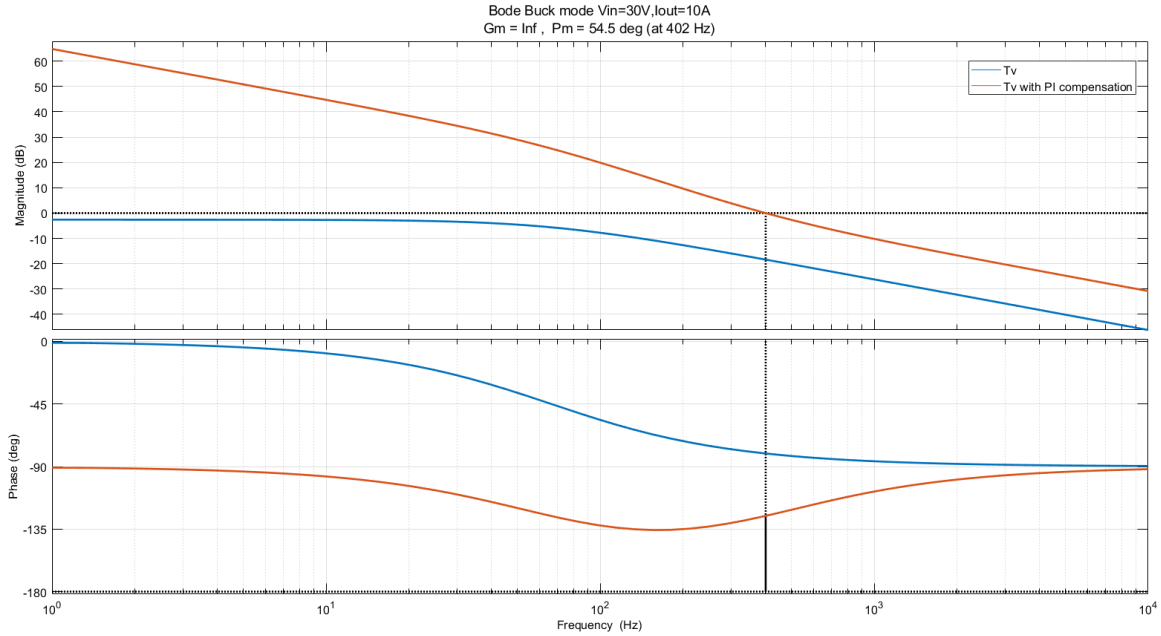


Figure 3.14: System in buck mode with and without control

As expected the system is more stable than in boost mode with the same control loop, now the phase margin is 54°.

3.2. Electronics design

In order to validate our design we design a PCB with all the necessary components to test the power electronics and the capabilities of the microcontroller. The especifications for the PCB are:

- Implementation of the DCDC converter capable of supporting the 10A 30V
- MOSFET drivers
- The STM32 with the required support components (clock, decoupling capacitors)
- CAN filter and transceiver IC.
- 3V3 and 12V DCDC converters to power the uC and MOSFET drivers

3.2.1. DCDC converter and implementation

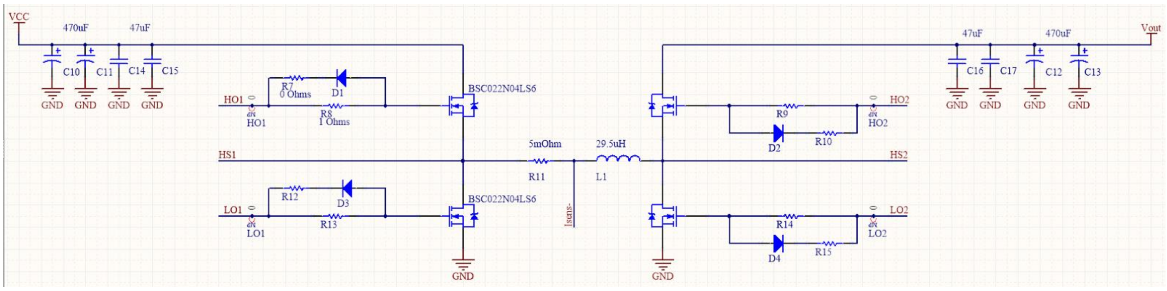


Figure 3.15: Power stage design

The implementation of the theoretical design is straightforward, the capacitor of 1000uF used in the theoretical design is split between two electrolytic capacitors of 470uF and two ceramic of 47uF in order to reduce to combined ESR, both of them rated for 35V, even the ones at the output in order to reduce the different number of components.

The MOSFETs were selected from the distributor of choice Mouser, at the time of selection they were the newest models that met the criteria and had the best specifications for power losses. The gates of these MOSFETs are connected to the drivers with a protection circuit for reverse currents, this is provided by the driver manufacturer in their datasheet.

In order to choose the best suited MOSFET we had to balance the losses by conduction and the switching losses [5], here is a table with a comparison of some popular MOSFETs from a distributor. From this table we selected the BSC022N04LS6

MOS	Rds(4.5V) [mOhm]	Qgd [nC]	Pcond	Psw	Pd
CSD18540Q5B	2,6	6,7	0,2106	0,2191	0,4297
TPN2R304PL	2,8	6,2	0,2268	0,2027	0,4295
CSD17522Q5A	10	1,1	0,81	0,0360	0,8460
NVTFS5C454NL	6,9	2,1	0,5589	0,0687	0,6276
RJK0305DPB-02	10	1,5	0,81	0,0490	0,8590
CSD18503Q5A	4,7	4,3	0,3807	0,1406	0,5213
BSC022N04LS6	2,4	3,6	0,1944	0,1177	0,3121

Table 3.4: MOSFET comparison

The inductor was chosen because it was the highest inductance for the package that could handle the current. The inductor is also shielded which protects the rest of the circuit from EMI.

3.2.2. MOSFET drivers

The requirements for the MOSFET driver was for it to be able to control two N-Channel MOSFETs with the highest efficiency possible.

The schematics for this project was designed following the application manual provided in the datasheet by the manufacturer [6]. The only addition was the option to select a pull-up or pull-down resistor for the enable pin which is controlled from the microcontroller.

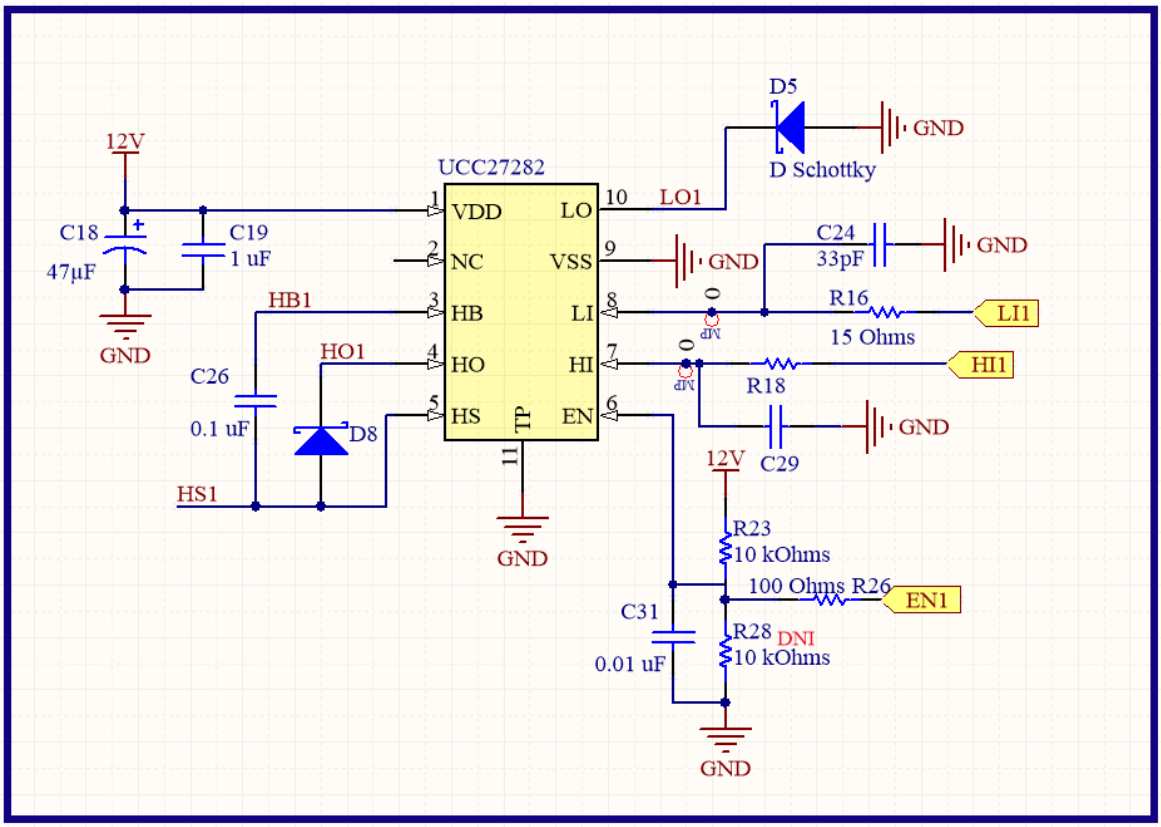


Figure 3.16: MOSFET driver

3.2.3. Current and voltage sensors

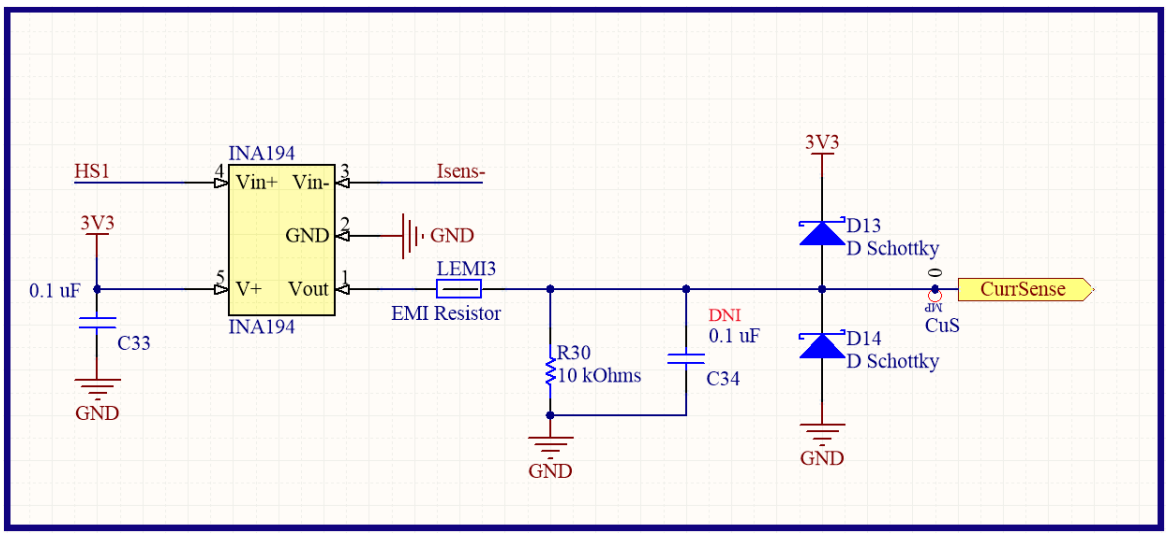


Figure 3.17: Current amplifier

The current sensor is a very simple integrated package with an opamp with a very high CMRR with a gain of 50. There is also some filtering with a ferrite bead and a possible RC if needed, two Schottky diodes designed to protect the microcontroller in case the sensor failed and delivered more than 3V3 to the ADC of the microcontroller.

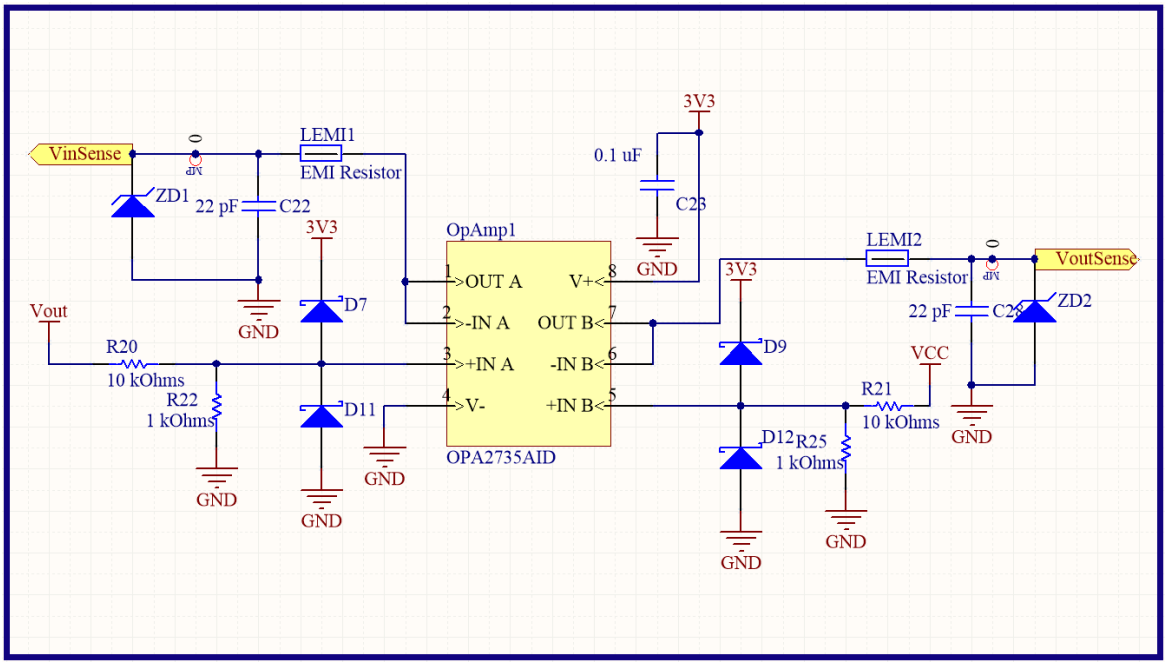


Figure 3.18: Voltage amplifier

For the voltage sensing, we first install a 1/11 voltage divider with the protection diodes and use an opamp in voltage following mode in order to provide a stable voltage for the ADC, when the ADC samples a small amount of current flows and it would disrupt the simple voltage divider and give a false reading. Also before we connect it to the microcontroller, we add a ferrite bead and a protection diode, a 3V3 Zenner diode.

3.2.4. Microcontroller

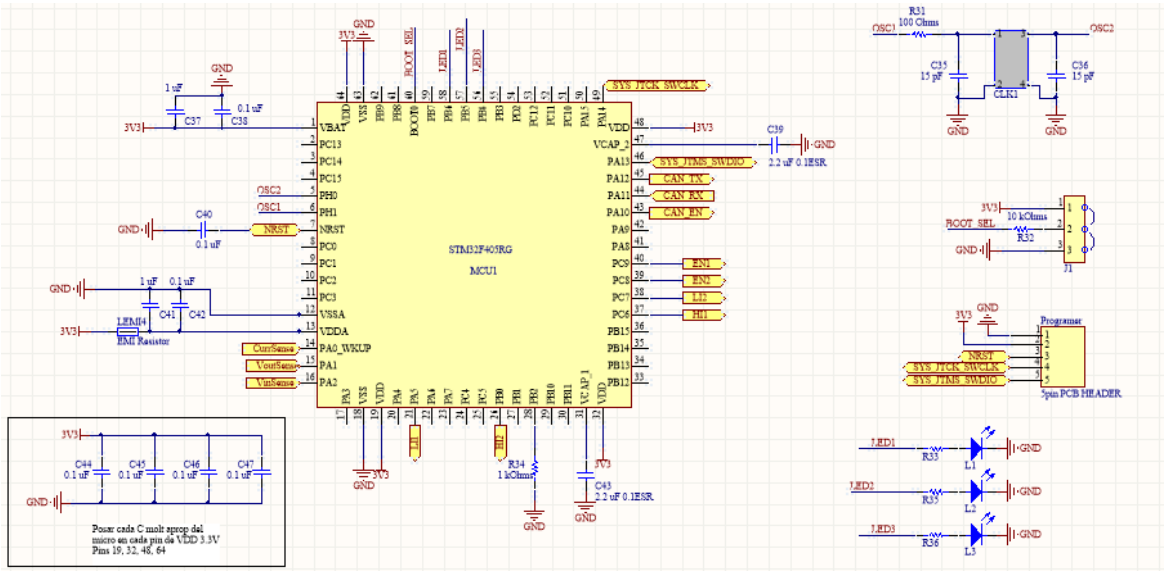


Figure 3.19: Microcontroller implementation

The microcontroller is the STM32f405RG. It has 3 ADC, a CAN module 8 timers and it can run up to 168MHz. This specific module was chosen for its availability and experience from previous projects in the Formula Student environment, also the support components like

the decoupling capacitors, crystal ceramic clock and debugging LEDs are an established design used previously in the team.

3.2.5. CAN filter and transceiver

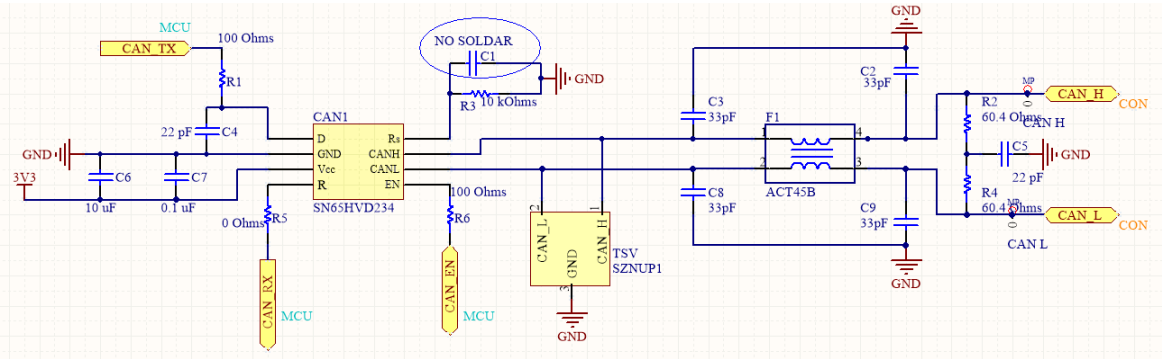


Figure 3.20: CAN module and filter

This schematic is also taken from the Formula Student team and has been validated on many designs. First, we have the terminating resistors 120Ohms in total, then a common mode filter and a dual TVS diode, and finally a CAN transceiver.

3.2.6. 3V3 and 5V regulators

In order to power the microcontroller, CAN transceiver and the OpAmps we have a small LDO. It's not very efficient but it provides low power.

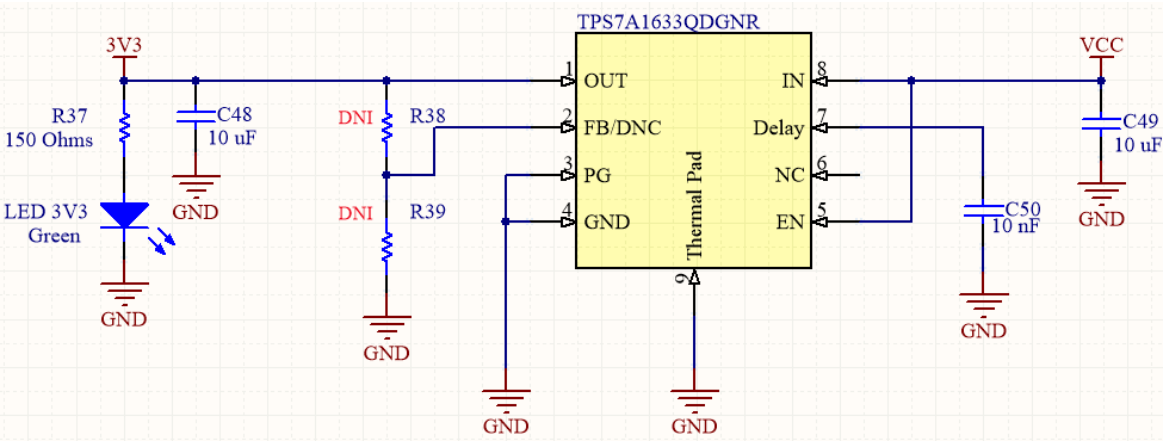


Figure 3.21: 3V3 LDO

The 12V regulator is capable of sourcing 1A continuous and it has been tested previously.

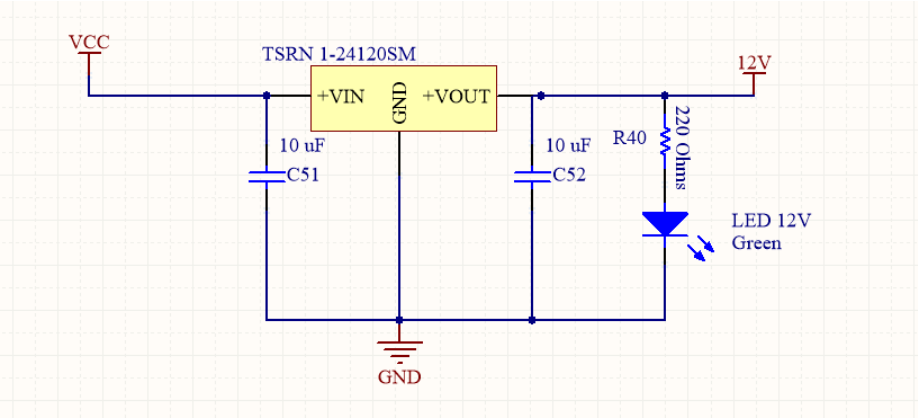
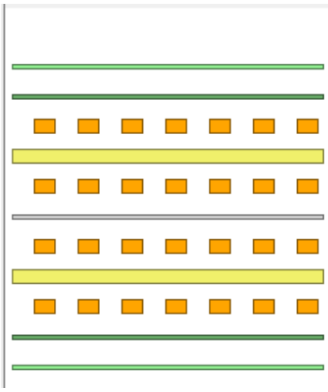


Figure 3.22: 12V regulator

3.3. PCB layout

Before we begin with the PCB layout, we have to stablish the design rules, these are given by the manufacturer's fabrication capabilities, they define trace width, trace spacing, via hole diameter, copper thickness and other parameters that need to be stablish before we start routing.

One important decision was the number of layers required, more layers allows us to run power planes and route signals more directly, but it increases cost. For this project we selected 4 layers as it's a nice balance between cost and function.



Layer Name	Type	Material	Thickness (mm)	Dielectric Material	Dielectric Constant
Top Overlay	Overlay				
Top Solder	Solder Mask/Co...	Surface Material	0.01016	Solder Resist	3.5
Top Layer	Signal	Copper	0.03556		
Dielectric 1	Dielectric	Core	0.254	FR-4	4.2
Signal Layer 1	Signal	Copper	0.036		
Dielectric 3	Dielectric	Prepreg	0.92856		4.2
Signal Layer 2	Signal	Copper	0.036		
Dielectric 2	Dielectric	Core	0.254		4.2
Bottom Layer	Signal	Copper	0.03556		
Bottom Solder	Solder Mask/Co...	Surface Material	0.01016	Solder Resist	3.5
Bottom Overlay	Overlay				

Figure 3.23: PCB layers

For the two internal layers one was used for GND and the other was split, half of it for 12V for the MOSFET drivers and the other half for 3V3 for the microcontroller. In order to start placing components into the board, we have to follow a criteria and personal guidelines. For this specific project, there was four main design specifications:

First the components for the power converter and the high current path had to be separated from the more sensitive analog amplifiers and the microcontroller.

Second, the layout of the MOSFET drivers had to be as compact as possible, in order to avoid parasitic inductances, also the bootstrap capacitors and diodes needed a clean path as they are recharged with a high current peak. The manufacturer also provides more layout recommendations.

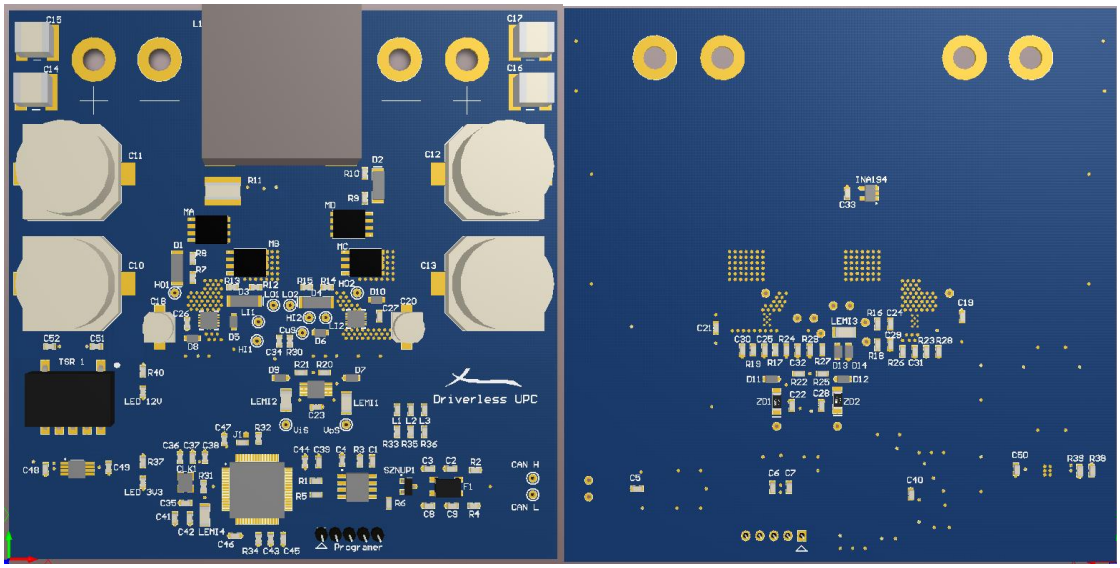


Figure 3.24: PCB 3D view

Third large copper planes and thermal vias were used to dissipate heat from the MOSFETs and the drivers.

Finally for the microcontroller components, we placed them as close as possible, with direct paths to 3V3 and GND.

3.4. Microcontroller configuration and code

A microcontroller is a small computer; it contains a CPU along with memory and input/output peripherals. These devices have the advantages that are relatively cheap, small and easy to program and that's why they are used all kind of embedded applications.

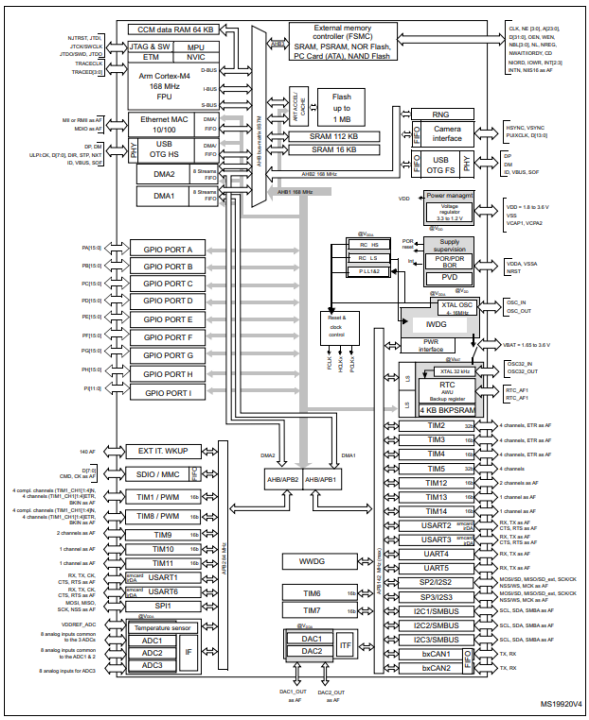


Figure 3.25: STM32F4 internal features

As seen in Figure 3.25 the STM32F405RG contains 3 ADCs, 2DAC, 14 timers, UART, SPI, I2C, CAN and multiple GPIO ports, it is also equipped with an efficient but powerful ARM-cortex M4 CPU capable of running up to 168MHz. Also, it is equipped with an internal watchdog capable of monitoring if the code has crashed and reset the microcontroller. And all the peripherals are equipped with powerful features that will facilitate the creation of tasks that could be complex in software but are easily performed with dedicated hardware, like PWM generation. Also, it has 2 DMA modules capable of transferring data from the ADCs to memory. [7]

Now that we have an idea of what peripherals we can use, we need to understand how to connect them with each other and how they interact with the CPU, that can only run a single task at a time and is not capable of running tasks with the precise timing we need by itself.

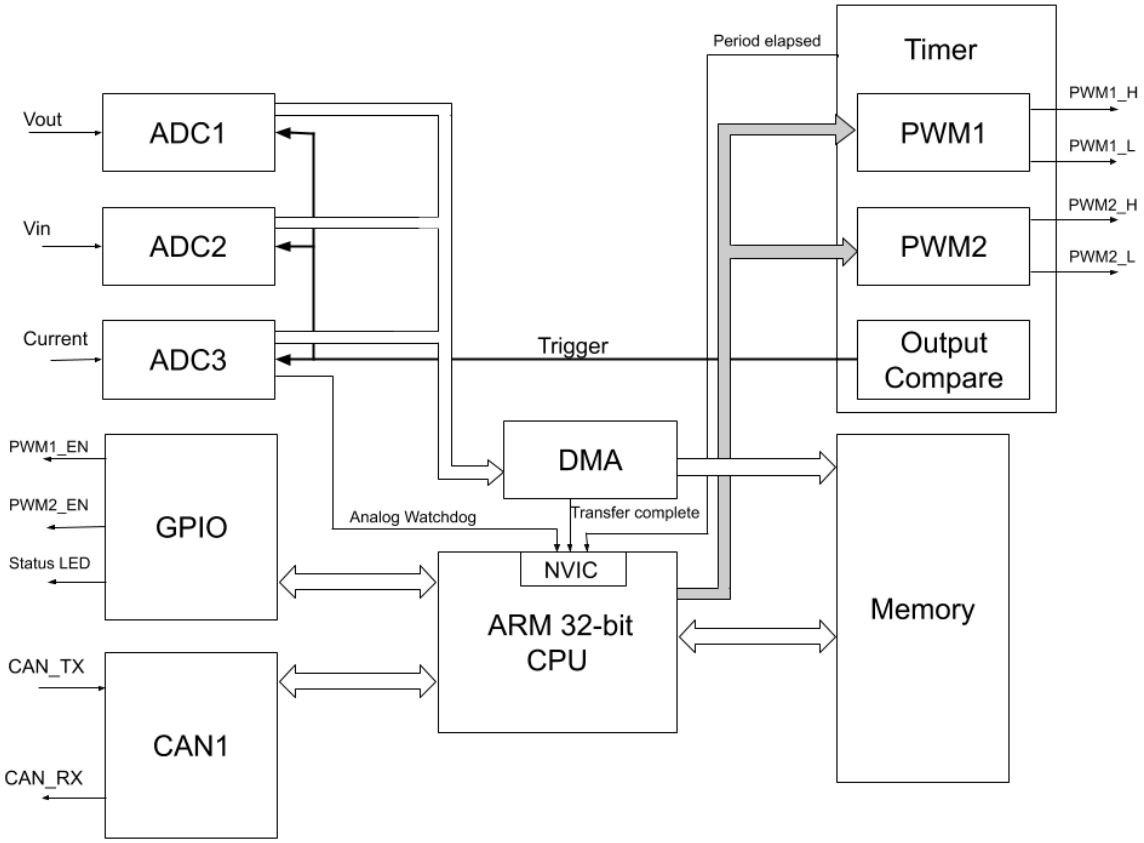


Figure 3.26: Proposed microcontroller utilization

First, we need to use one of the timers at our disposal, only two of them have the PWM generation modules that we need, also this timer will use the output compare function to trigger the ADCs to start a conversion.

The ADCs will perform a conversion and then in order to not use CPU time they will automatically use the DMA to transfer the data to memory. Also, the ADCs have their own integrated analog watchdog that will trigger an interrupt in the CPU if a value goes out of bounds.

Finally, we have the modules for the CAN transmission and the GPIOs used to control the enable signals and the status LEDs.

The microcontroller programming was carried out using CubeMX (Microcontroller configuration) and Keil software (Program code). In this section, the program is described and explained.

3.4.1. Clock configuration

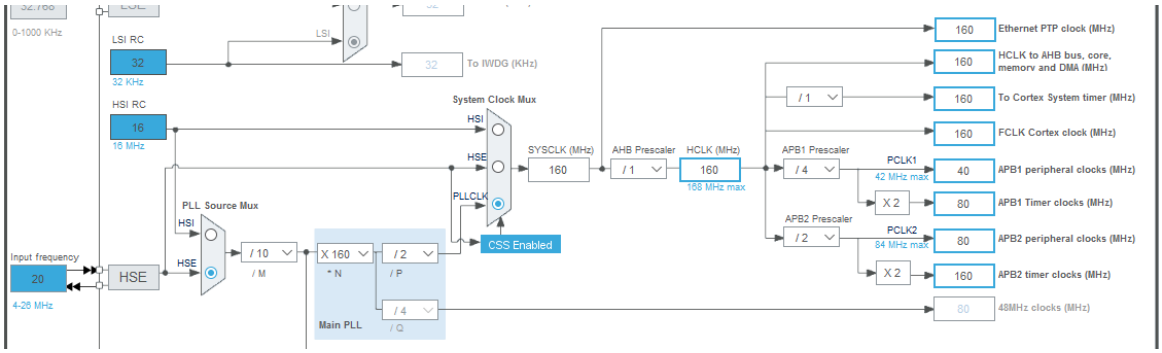


Figure 3.27: CubeMX clock configuration

First, the clock configured: it was added an external Crystal/Ceramic Resonator with a 20MHz frequency and it was set a 160MHz internal clock. With that, the software automatically set the internal PLLs (Phase-Locked Loop) to achieve the desired frequency and it allows to observe the frequency used in each system clock (APB1 peripheral, APB1 timer clocks, APB2 peripheral, APB2 timer clocks).

3.4.2. Timer configuration

STM32s are equipped with powerful and precise timers, we will use one of these to generate our PWM for both drivers (high and low), set the trigger for the ADCs and control the timing for the CAN messages and other operations.

First we configure our timer, we will use Cube MX as it provides an easy interface to do so.

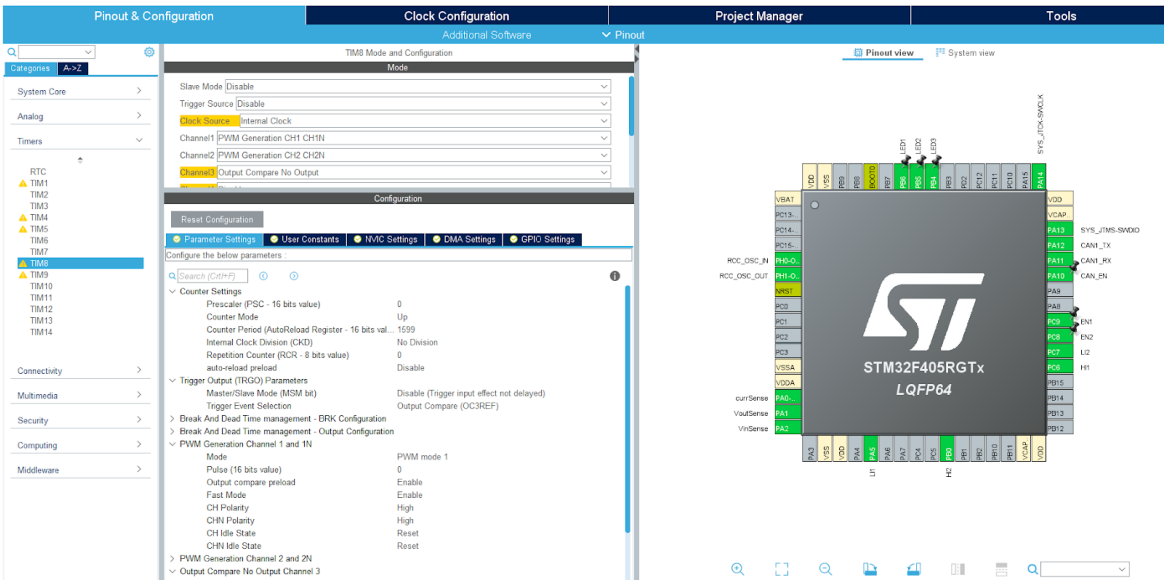


Figure 3.28: CubeMX Timer configuration

First we need to set the period of our timer, the clock frequency is 160MHz, so if we want a frequency of 100kHz (the proposed control frequency) we need to divide the clock by 1600, STM32s timers allow for prescalers, but in this case we do not need it as our counter frequency is high enough and we want as much precision as we can, so we set the counter to 1599.

The PWM is enabled and we can select the setting where it generates the signal and its negated state, we need to habilitate two channels, for the input and output drivers.

Also we activate the output compare for the channel 3, this feature will create an internal trigger which we will use to start an ADC conversion that will be synchronized with the PWM. This conversion will happen at the half point of the cycle.

3.4.3. ADC configuration

The ADCs for the STM32 have multiple channels that feed into the a single buffer and ADC, this gives us the potential to sample multiple signals without the need to add many ADC peripherals that would increase the cost for the microcontroller, but using a single ADC and multiple channels implies that only one conversion can be performed at the same time, for our application we need to read the input voltage, output voltage and inductor current at a very precise moment of the cycle and we need the values to be sampled at the same time, so for this application we will use the 3 ADC that our chosen microcontroller has available, as it is a higher end model.

In order to properly set up the 3 ADCs we will use Cube MX again.

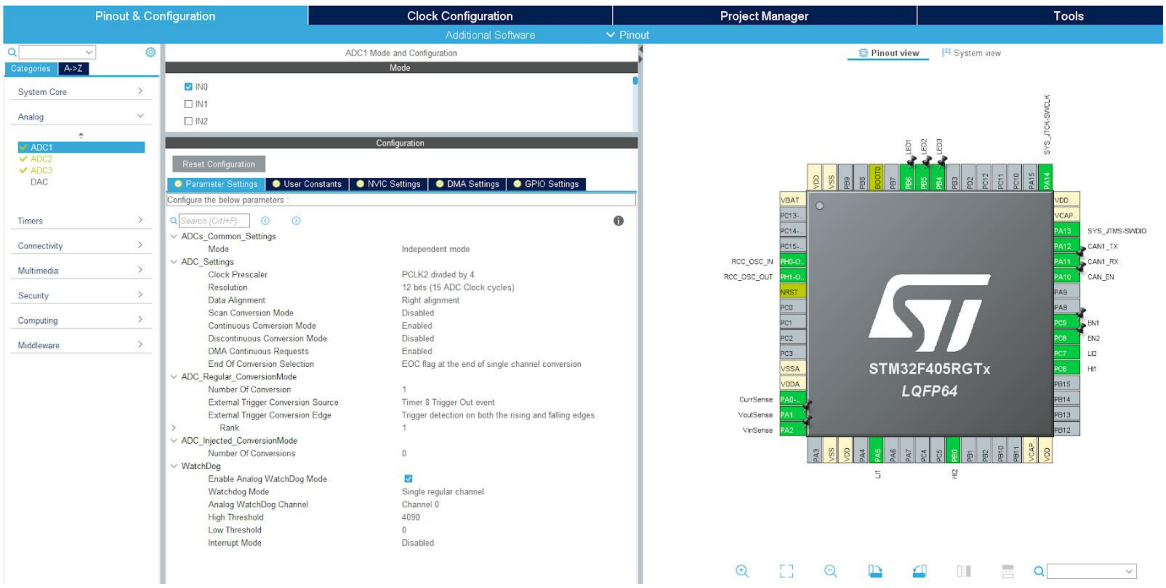


Figure 3.29: CubeMX ADC configuration

First we select which channel we activate from all 3 ADCs, then we select the prescaler for the internal ADC clock, we will select the fastest possible clock frequency for our ADCs, as the selected clock frequency for the APB2 bus where the ADCs peripherals are connected is 80MHz we need a prescaler of 4 to reach the maximum established frequency of 20MHz.

We also can choose the resolution of the ADC, a higher resolution will need a higher conversion time, for 12 bits its 15 clock cycles, t_C=750ns.

The sampling time can be configured to last a certain amount of clock cycles, less sampling time requires a lower input resistance in order to achieve the desired accuracy, the manufacturer provides a formula to calculate the maximum input resistance for an acceptable error, but in our case because we feed all 3 ADCs with voltage following opamps we are able to select the fastest option of 3 clock cycles ($f_{CLK\ ADC}$=20MHz) t_S=150ns.

Because our microcontroller has 3 ADC modules we can perform the current and the voltage reading at the same time, giving us an approximate time for the ADC of 0.9us. The manufacturer specifies some extra time for trigger latencies and other factors, but they are much lower than the two previously mentioned factors.

Another important parameter is the trigger for the ADCs, all 3 are connected to the previously mentioned output compare signal from the timer 8, that will trigger at the halfway point of the cycle.

Also this ADCs have the option to enable an analog watchdog which is integrated in the ADC peripheral, the watchdog is constantly and independently comparing the sampled values of the ADC to two defined values for maximum and minimum values, if we enable this watchdog it will trigger an event that will stop the code in which we can actuate as necessary to protect the circuit.

As an example we setup the watchdog for the current sampling, we want to establish a maximum current of 14A, so we need to convert that to the equivalent sampled value:

$$14 \cdot 0.005 \cdot 50 \cdot \frac{2^{12}}{3.3} = 4344$$

(14 Amps, Rsense resistor of 5mOhm, 50 gain of current sensor, 12 bits, 3.3V of ADC voltage).

Also we use Cube MX to set up the DMA (Direct Memory Access) which is a system that after the conversion of each ADC it will take the sampled value and move it to memory where it can be accessed by the CPU, after completing the move it will interrupt the CPU so we can start with the computation of the control for the new values.

3.4.4. C code

The entire configuration done in the CubeMX software is initialised in the code. GPIOs, clock, CAN, ADCs, timers and the middleware implemented are included in the code automatically with the setup configured.

Also in order to help coding the microcontroller, we use HAL (Hardware Abstraction Layer), the HAL drivers are designed to offer a rich set of APIs and to interact easily with the application upper layers. [8]

3.4.4.1. Initialization

The code starts to run at the main() function, here Cube MX already initializes some peripherals and clock for us with the proper configuration, but we have to start them.

First we start the timer, we need to start the two PWM counters and the high and low outputs, also we have to start the output compare trigger for the ADCs.

The ADCs need to be started in a DMA mode where we give it the memory location to store the sampled values, the DMA requests a buffer to store the values, and later we could take

this buffer and do a mean calculation, but in our case as we will read the value after each conversion we will just have a buffer of 1 slot.

Also we will activate the enable signals for the MOSFET drivers and the CAN transceiver.

```
//Start the PWM channels of TIM8, needed to start channel high and low
  HAL_TIM_PWM_Start(&htim8,TIM_CHANNEL_2);
HAL_TIMEx_PWMN_Start(&htim8,TIM_CHANNEL_2);
HAL_TIM_PWM_Start(&htim8,TIM_CHANNEL_1);
HAL_TIMEx_PWMN_Start(&htim8,TIM_CHANNEL_1);
HAL_TIM_Base_Start_IT(&htim8);

//Start the output compare channel to generate the trigger for the ADCs
HAL_TIM_OC_Start(&htim8,TIM_CHANNEL_3);

//Start the ADCs as DMA
HAL_ADC_Start_DMA(&hadc1, (uint32_t*)current_buf, ADC_BUF_LEN);
HAL_ADC_Start_DMA(&hadc2, (uint32_t*)vin_buf, ADC_BUF_LEN);
HAL_ADC_Start_DMA(&hadc3, (uint32_t*)vout_buf, ADC_BUF_LEN);

HAL_GPIO_WritePin(EN1_GPIO_Port,EN1_Pin,GPIO_PIN_RESET);  //enable PWM drivers
HAL_GPIO_WritePin(EN2_GPIO_Port,EN2_Pin,GPIO_PIN_SET);
HAL_GPIO_WritePin(CAN_EN_GPIO_Port,CAN_EN_Pin,GPIO_PIN_SET);   //enable CAN module
```

As all the code is performed with interrupts and triggers from the timers and ADCs there is no code running in the while(1) loop.

3.4.4.2. End of period callback

The timer that is running the PWM and ADC trigger can activate an interrupt once it reaches the end of the period, we will use this interrupt to write on the timer registers for the PWM duty cycles the calculated values for the next cycle, also we want to use this point to send the CAN telemetry, but as this portion of the code is accessed every 10us and we want to send the CAN messages every reasonable amount of time we will write a counter that will divide the frequency so we only send the CAN message every 10ms.

In order to send a CAN message we need to fill the 8 available bytes with our data, we will send the values of current, voltage and the output value of our current control loop also the two duty cycles. Once we have filled the bytes we use HAL to put the message into the buffer for the CAN peripheral that will send it whenever it is available to do so without taking time from the CPU.

```
//Timmer finished one cycle
void HAL_TIM_PeriodElapsedCallback(TIM_HandleTypeDef *htim)
{       HAL_GPIO_WritePin(LED1_GPIO_Port, LED1_Pin,GPIO_PIN_RESET);
  //Change the PWM registers at the begining of the cycle.
  htim8.Instance->CCR1=duty1;
  htim8.Instance->CCR2=duty2;

  if(countCAN ==CAN_PERIOD_COUNT){
    HAL_GPIO_TogglePin(LED2_GPIO_Port,LED2_Pin);
    countCAN =0;

    data2send[0] = (uint8_t) vin_buf[0];
    data2send[1] = (uint8_t) vout_buf[0];
    data2send[2] = (uint8_t) current_buf[0];
    data2send[3] = (uint8_t) currentCommand;
    data2send[4] = (uint8_t) duty1;
    data2send[5] = (uint8_t) duty2;
    data2send[6] = 0;
    data2send[7] = 0;

    HAL_CAN_AddTxMessage(&hcan1,&txHeader,data2send,&ptxMailbox);
  }

  countCAN++;

  UNUSED(htim);
}
```

3.4.4.3. ADC conversion complete callback

As we have 3 ADCs performing a conversion at the same time we need to check individually if they have finished, it should take them the same amount of time but there could be a small variance. Each ADC will interrupt the CPU after it has finished converting, so in this interrupt function we check which one it is and mark a completion flag, in this case we will only check for the current and output voltage as they are the ones we need to calculate the control. Once we check they have both finished we execute the control function.

```
// function triggers when DMA transfer is completed, need to check which ADC finished
// once the current and Vout ADCs completed the conversion to start the control computation
void HAL_ADC_ConvCpltCallback(ADC_HandleTypeDef* hadc) {
    HAL_GPIO_WritePin(LED1_GPIO_Port, LED1_Pin,GPIO_PIN_SET);
  if(hadc->Instance == ADC1){
        HAL_GPIO_WritePin(LED1_GPIO_Port, LED1_Pin,GPIO_PIN_SET);
    currentSampleCompleted=1;}

  else if(hadc->Instance == ADC2){
    HAL_GPIO_WritePin(LED2_GPIO_Port, LED2_Pin,GPIO_PIN_SET);
    voutSampleCompleted=1;  }

  if(currentSampleCompleted && voutSampleCompleted){control();}
}
```

3.4.4.4. Control function

The control function first checks if the converter is in the buck or boost region comparing the input voltage to the 24V (converted to 12bit digital is 2481 taking into account the voltage divider). After we set one duty cycle to 0.15 and for the other we calculate the PI control. Also we have to remember to accumulate the error for the integral calculation and take into account the integration time.

```
//control calculation function executes after the DMA finishes the transfer
void control(){

  ERRORvoltage=VOUT_TARGET-vout_buf[0];
  IntegralVoltage= IntegralVoltage+ERRORvoltage*ITERATION_TIME;
  currentCommand=KPvoltage*ERRORvoltage+KIvoltage*IntegralVoltage;

  ERRORcurrent=currentCommand-current_buf[0];
  IntegralCurrent=IntegralCurrent+ERRORcurrent*ITERATION_TIME;

  if(vin_buf[0]<2481){
    duty1=239;  //set PWM1 duty cycle to 15%
    duty2=KPcurrent*ERRORcurrent+KIcurrent*IntegralCurrent;
  }
  else{
    duty2=200;  //set PWM2 duty cycle to 15%
    duty1=KPcurrent*ERRORcurrent+KIcurrent*IntegralCurrent;
  }

  currentSampleCompleted=0;voutSampleCompleted=0;
}
```

3.4.4.5. ADC watchdog callback

When the ADC watchdog triggers an interrupt we will disable the driver enable pins and set the duty cycle to 0%, also we trigger a status LED on the board. Also, because we are able to know which ADC triggered the interrupt we know which error is, so 3 error flags can be activated depending on the ADC, later this is useful when debugging so it's easier to know where the error comes from.

```
//Analog watchdog triggers when current sense is too high.
void HAL_ADC_LevelOutOfWindowCallback(ADC_HandleTypeDef* hadc)
{
  /* Prevent unused argument(s) compilation warning */
  UNUSED(hadc);
  HAL_GPIO_WritePin(EN1_GPIO_Port,EN1_Pin,GPIO_PIN_RESET);  //disable PWM drivers
  HAL_GPIO_WritePin(EN2_GPIO_Port,EN2_Pin,GPIO_PIN_RESET);
  htim8.Instance->CCR1=0;                  //Set PWM to 0
  htim8.Instance->CCR2=0;
  HAL_GPIO_TogglePin(LED3_GPIO_Port,LED3_Pin);  //Toggle Status LED
  overERROR=1;
  if(hadc==&hadc1){CE=1;}
  if(hadc==&hadc2){OE=1;}
  if(hadc==&hadc3){IE=1;}
}
```

4. Results

4.1. Simulation results

First, in order to validate our design we use Simulink, this allows us to further tune our design and more easily make changes before implementing it.

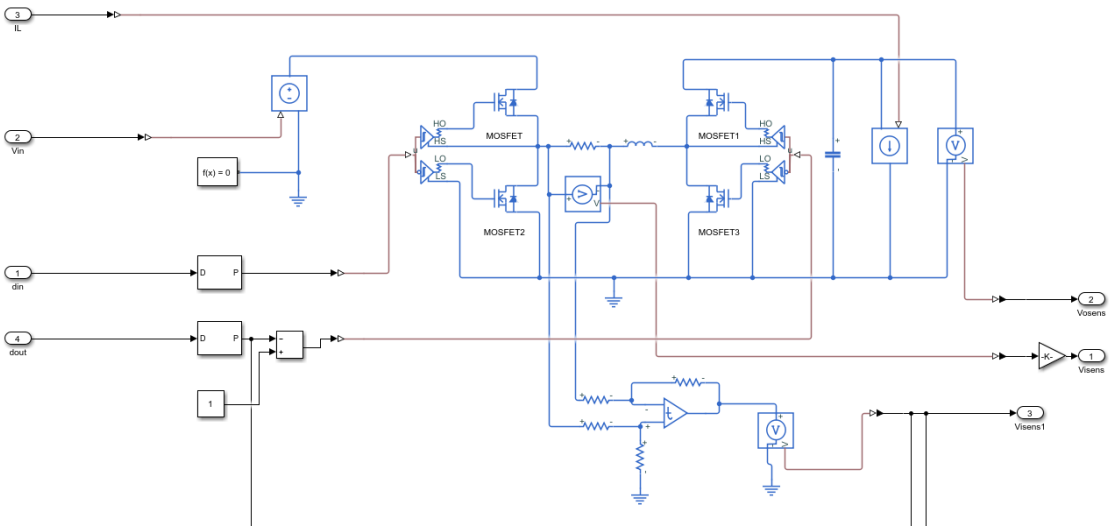


Figure 4.1: Simulink implementation

Simulink has a power systems module [9] that allows us to integrate electrical simulation with the usual Simulink tools for control systems. The first step is to model the converter, not only the power stage but also the sensing with the proper amplification and the generation of PWM

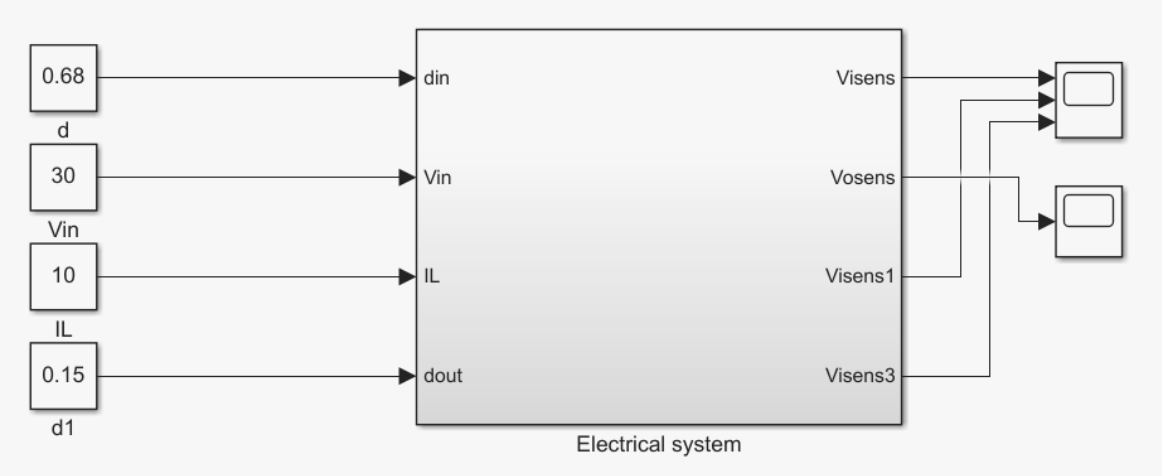


Figure 4.2: Simulation setup

After encapsulating the system we can provide some simulation parameters, in this case we aim to validate the ripple and mean current through the inductor for the given voltages and duty cycles

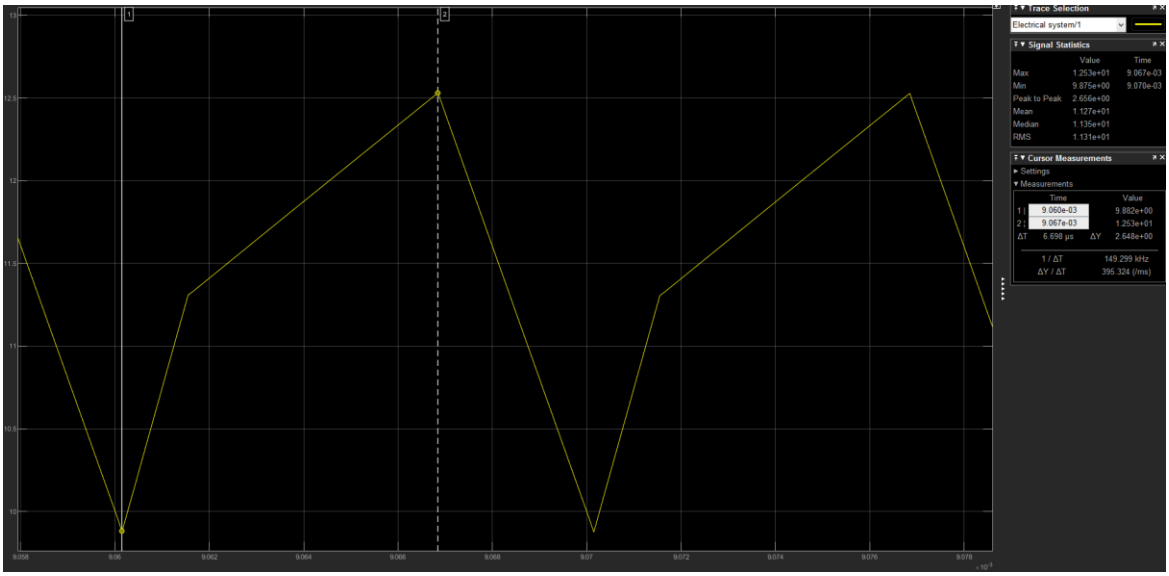


Figure 4.3: Inductor current simulation

$\hat{i}_L$ [A]	2.64A
$<i_L>$ [A]	11A

Table 4.1: Current measurements

Now that we have validate the power stage we implement our control system, placing the two PI controllers with the calculated values and all the feedback loops associated. Also we can input different functions for the voltage and current, in this case we will use a triangular function for the voltage, that will go from 20V to 30V and the current will step from 1A to 10A, so we can see how the control deals with a sudden change in load.

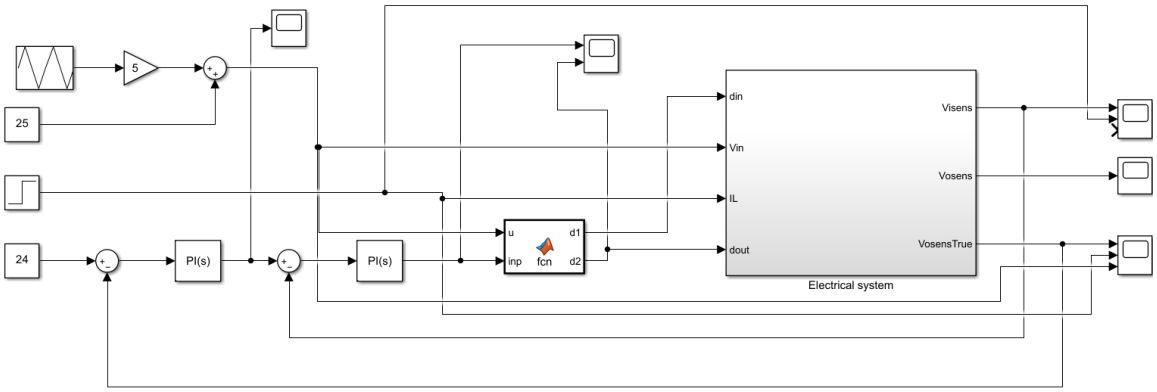


Figure 4.4: Simulink with control

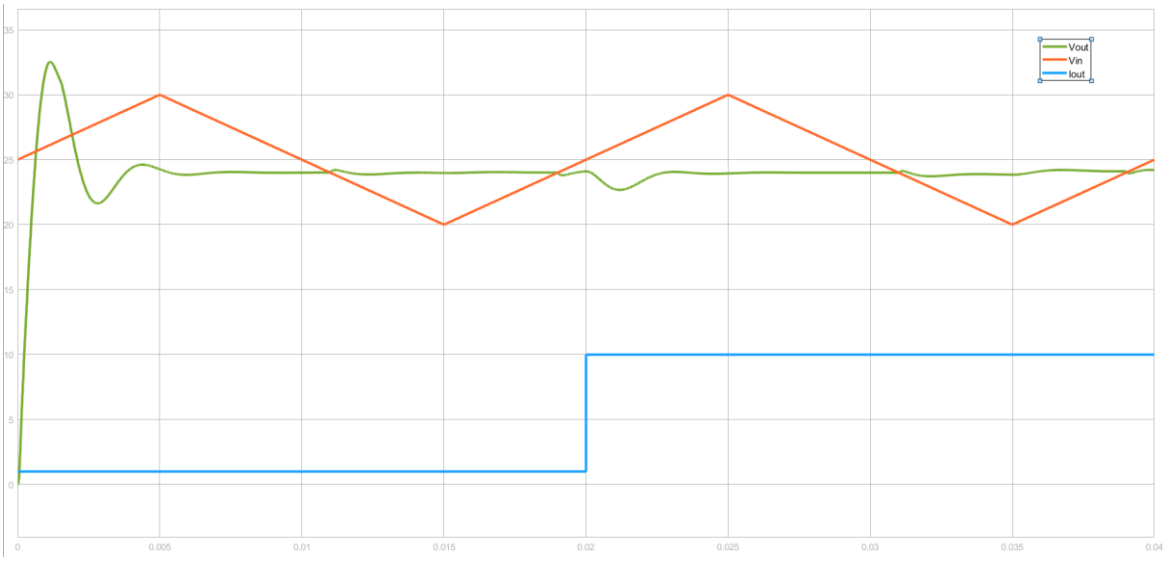


Figure 4.5: Step response simulation

As predicted in the control design phase there is a small overshoot in voltage when we power up the converter, but it quickly settles to the 24V, generally it deals without issues with the change in voltage except in the transition from buck to boost where it has a small variation in voltage of 200mV.

The 9A step in current also causes a small variation in voltage where it falls to 22.6V, so a variation in voltage of 5.8%, even though the drop is significant is a worst case scenario where it is to be expected a small variation.

Because we want to implement this control into a discrete system we have to validate if it is still valid, in order to prove it we first use Simulink, it provides the Zero-Order Hold blocks that will simulate the behaviour of an ADC. Also the PI blocks can easily be setup into a discrete operation.

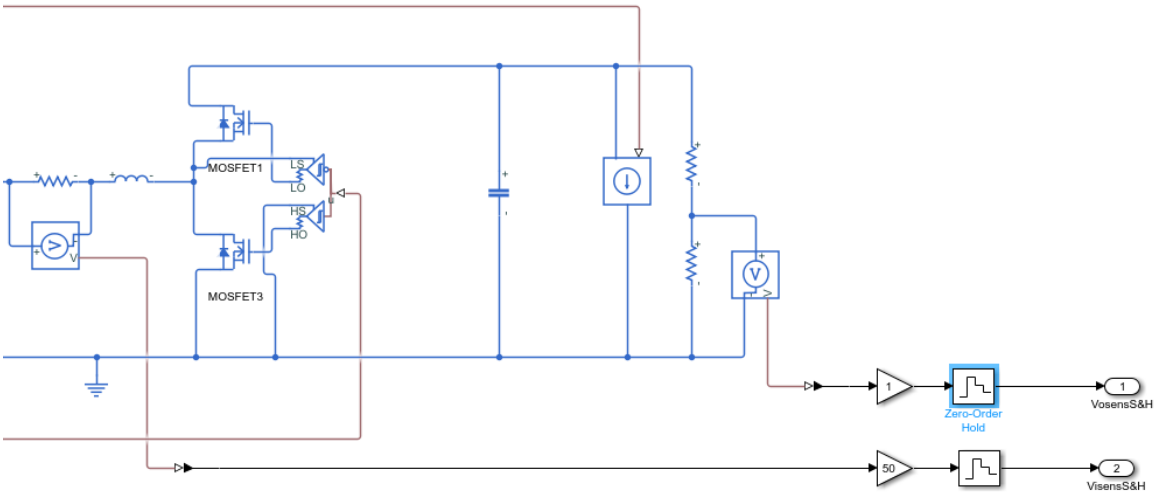


Figure 4.6: Discretization of the simulation

If we plot the output voltage after the sampling which is the same values that are used for the control we see the discretization, and how it does not affect at the stability of the system.

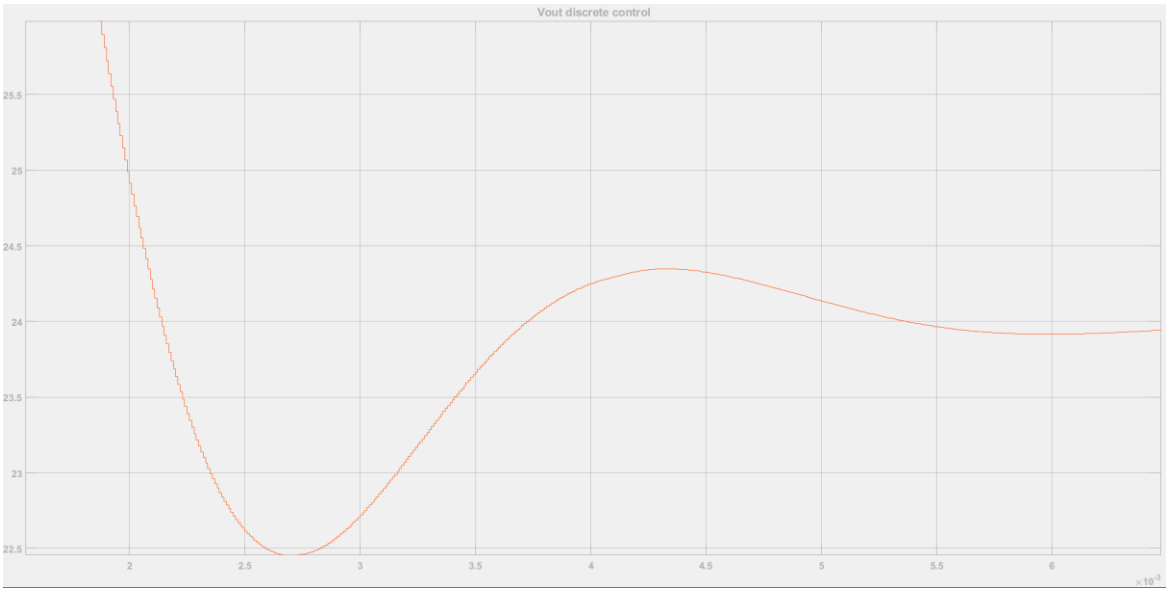


Figure 4.7: Output with discrete control

4.2. Laboratory results

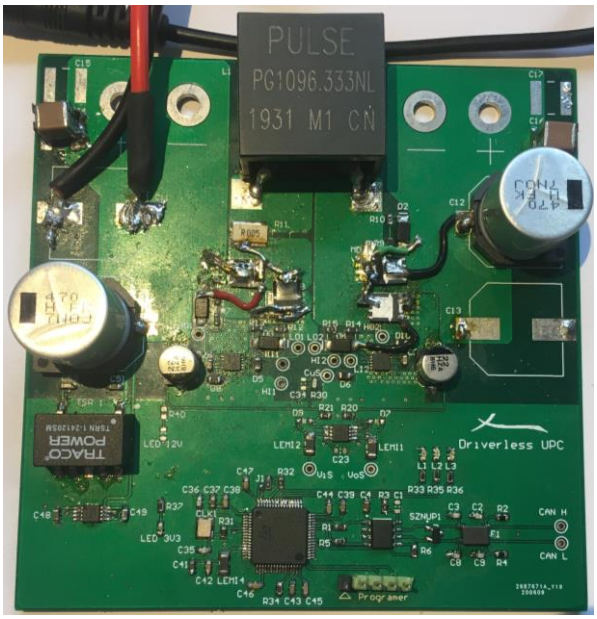


Table 4.2: PCB assembled

The PCB was easily soldered, with the help of a heat gun, and a microscope to see the finer pitch elements, but as soon as it was plugged into the power supply it went to constant current mode, the issue turned out to be the MOSFETs, there was a confusion with the footprint, and the source and the drain was reversed. In order to fix it, the MOSFETs were soldered upside down, with wires and solder bridges connecting them to the PCB, this would be enough to test the converter and the control, but it makes it impossible to test the converter at the maximum current capabilities as the MOSFETs would overheat as they do not have a copper plane to dissipate the heat.

Once the PCB is assembled first we check that all the systems work as expected before we can validate the control that we have designed. First we want to validate that the microcontroller is operating the internal timer by creating the PWM expected.

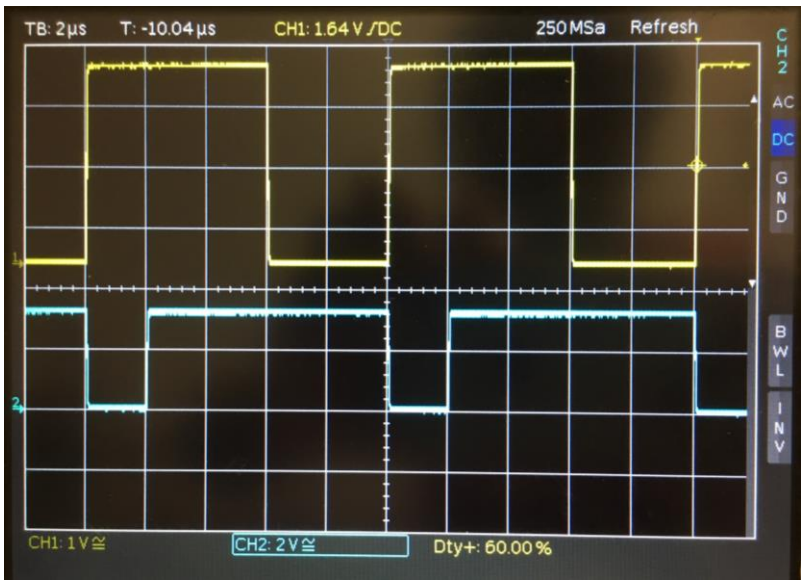


Figure 4.8: PWM generation

In the Figure 4.8 we see the two PWM sequences for d1 high and d2 high, with duty cycles of 0.6 and 0.8, and the frequency of 100kHz. Now that we know the PWM is as expected we analyse the MOSFET drivers.

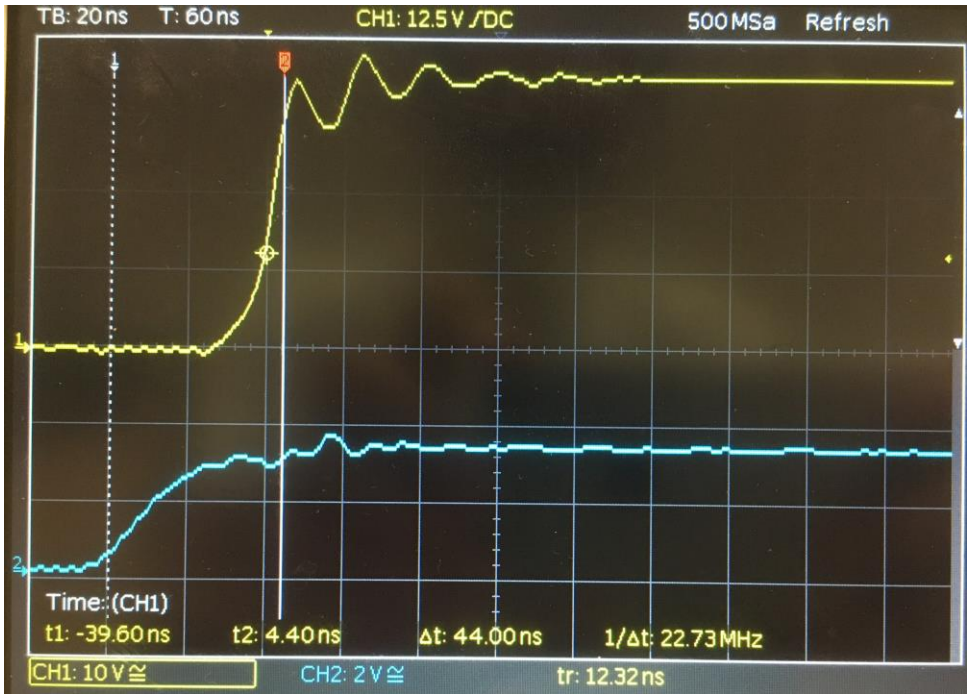


Figure 4.9: Half-bridge step response

In Figure 4.9 with blue we see the output from the microcontroller, and in yellow the output from the MOSFET, we can analyse the rise time at 12.32ns which is as expected, as the manufacturer specifies a rise time of 14ns in the datasheet [6, 6]. Also, we can study the total delay of 44ns. Overall, the response is as expected and accordingly to the manufacturer.

Now that we have validated the MOSFETs and the microcontroller we can start to validate the power stage and analyse if it works as designed. First, we will attempt to boost an input voltage of 21V to the expected 24V, for this initial test the input bridge is always open so the converter will operate as a traditional boost, at this stage we only want to validate the ADCs and the behaviours of the components.

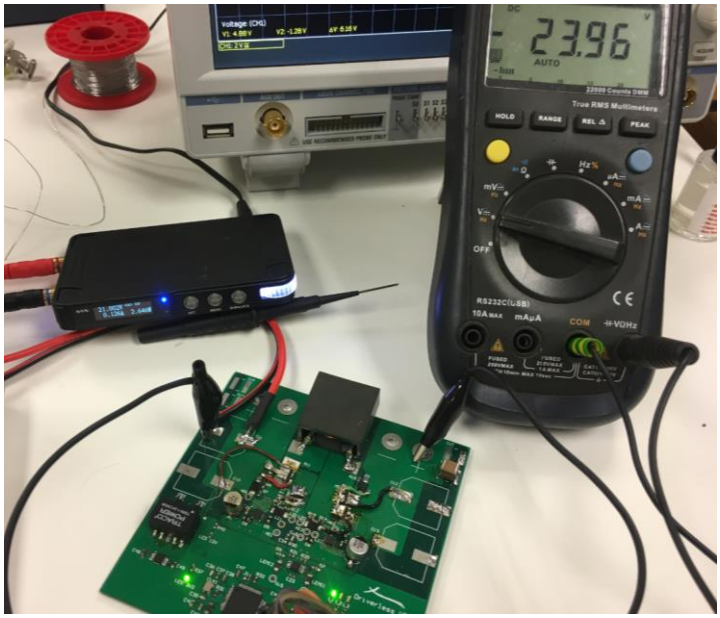


Figure 4.10: Converter working in boost mode

With an external multimeter we verify that the converter is working as expected for a traditional boost, and the MOSFET drivers, PWM generator and ADCs are also working as expected. The next step is to validate all the safety measures, overvoltage and overcurrent safety limits before we start to implement our control, as in case it does not work as expected we do not damage any components, as some of them are limited to 35V like the output capacitors, or the MOSFETs that are only rated for 40V.

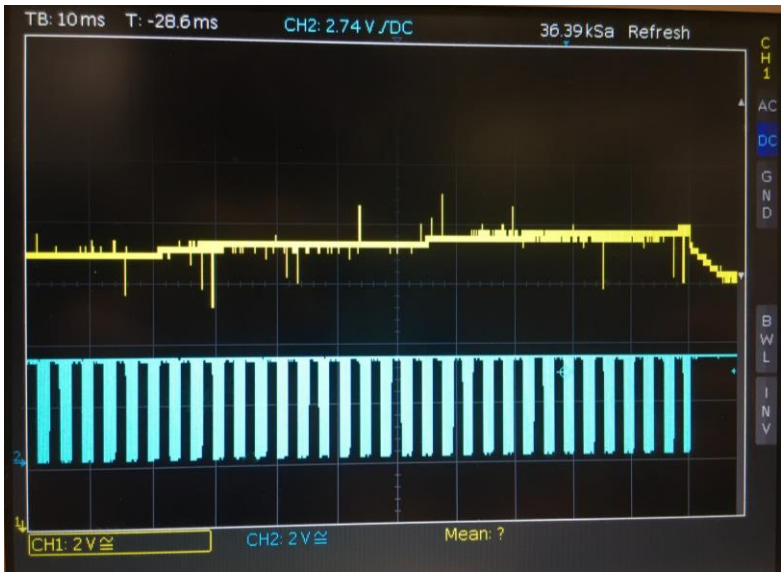


Figure 4.11: Overvoltage protection

As seen in the Figure 4.11 once the output voltage goes over a certain threshold, the PWM for the MOSFETs stops, this happens within once cycle of the control loop. This gives us the confidence to test the full capabilities of the converter as we are protected for overvoltage and overcurrent.

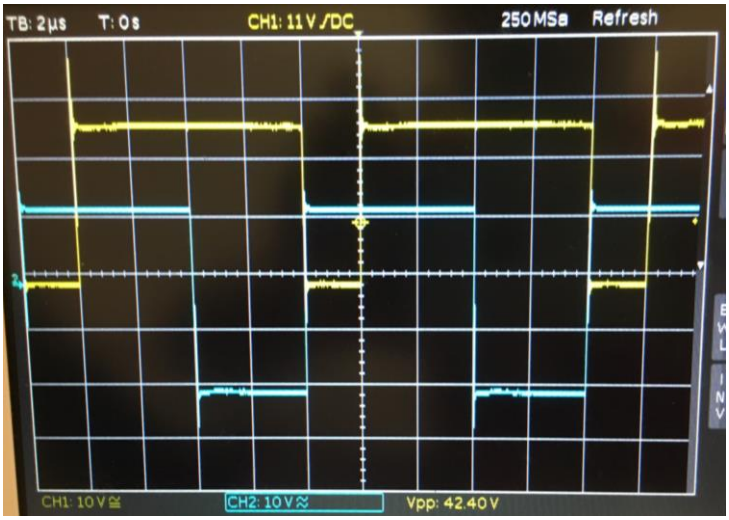


Figure 4.12: Gate voltages for the high MOSFETS at the input and output

As seen in Figure 4.12 now we activate both bridges so we can validate the full model of the converter operating in the 4-switch buck-boost mode. Channel one (yellow) represents the high MOSFET of the input bridge, the gate voltage has a peak value of 42V, but this is not the actual V_{gs} of the MOSFET, as the ground of the oscilloscope is at the 0V but the source of the MOSFET is at the input voltage, in this case it was 15V as we wanted to test the Boost capabilities, so because the MOSFET driver is powered with 12V, Vg will be 12 on top of the 15V, so 27V, which is the stable value we see in the oscilloscope, the 42V measured at the peak is the initial oscillation of the MOSFET driver when it switches, this is within the margins expected so it will not damage the MOSFET. In conclusion we see both bridges are commutating as expected and with the proper timing, so the PWM generation from the microcontroller and the MOSFET drivers are working as expected.

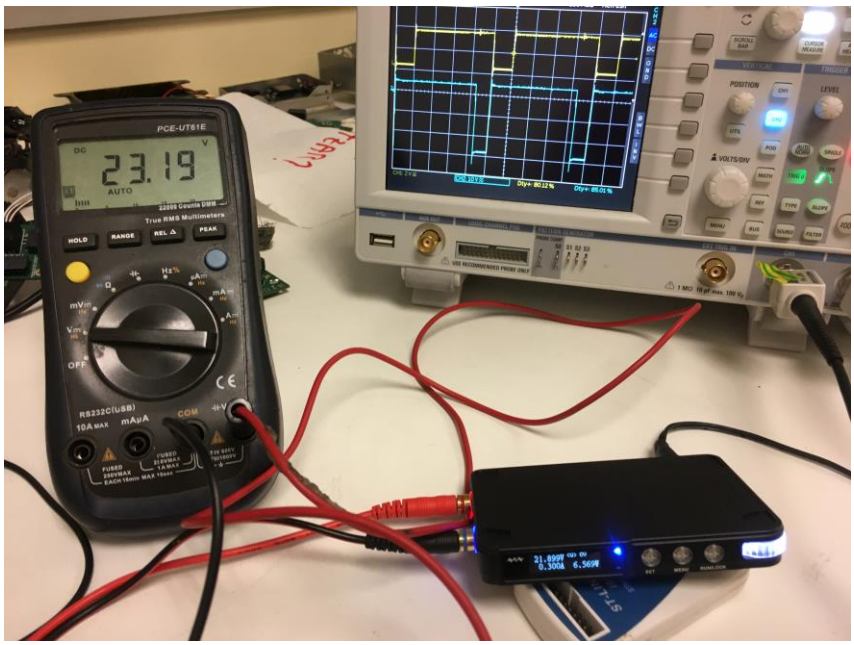


Figure 4.13: Boosting 21V to 24V and oscilloscope with MOSFET gates

Now that the 4-switch operation is validated, we will sample the output of the current sensor, to see that it is operating as expected and giving reasonable values.

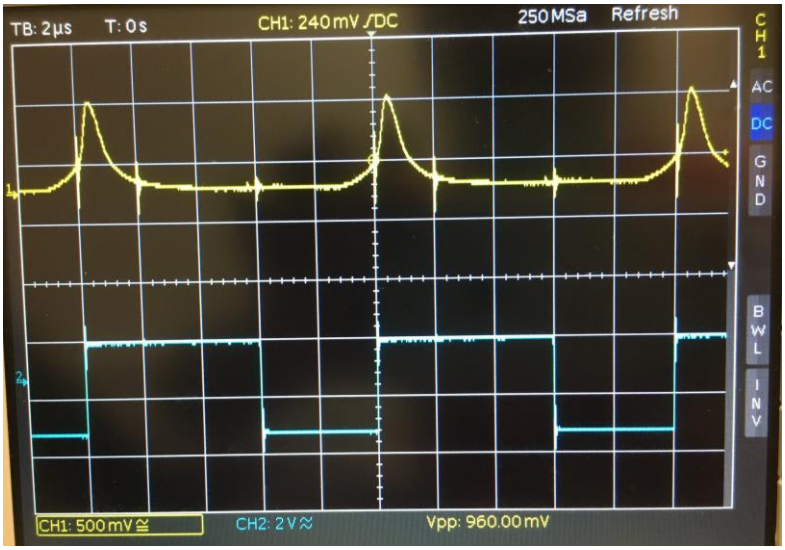


Figure 4.14: Output of current sensor

As you can see in Figure 4.14 the oscilloscope is connected to the output of the current sensor (yellow) and the PWM of the output bridge (blue), on a first view the shape of the inductor current does not look like what we would expect, but this is the voltage at the output of the amplifier, which is powered with 3V3, so it cannot output negative voltages, this means that for negative inductor currents the output will just saturate to 0V.

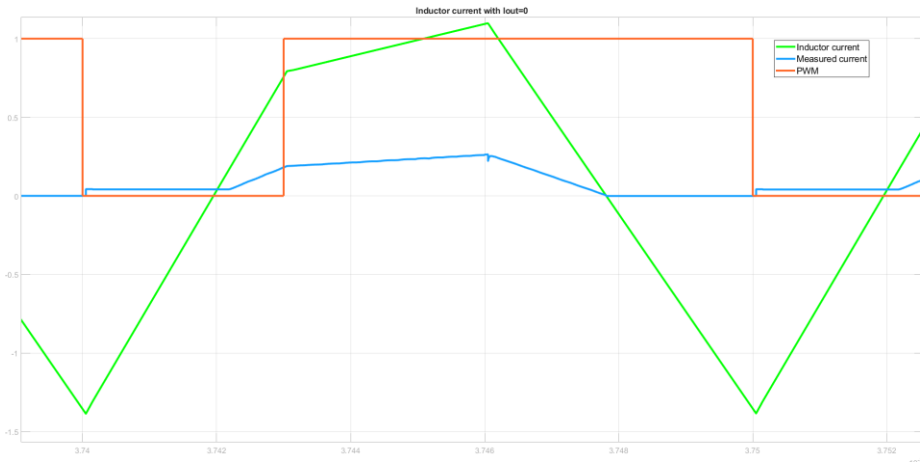


Figure 4.15: Simulation for Iout=0A

In order to have a clear view of what is happening we use Simulink again, in Figure 4.15 we see the inductor current [A](green), and the simulated output of the sensor [V](blue), as you can see only when the current is positive the sensor outputs a voltage, this only happens when the inductor current is close to 1A. With the simulation result we can validate that the output of the real sensor is coherent and it gives a reasonable value.

In order to conclude the validation of the power stage and the model we decide to perform a sequence of voltage changes, for a fixed Vin=21V, we will step through multiple duty cycles that transition from buck to boost. In order to do this first we need to calculate the necessary duty cycles using the mathematical models.

21	Vin		
Vout	d1	d2	mode
16	0.65	0.15	buck
18	0.73	0.15	
20	0.81	0.15	
22	0.85	0.19	boost
24	0.85	0.26	
26	0.85	0.31	

Table 4.3: duty cycles to validate converter

Then we substitute the control function for a simple counter that every second will change the duty cycle. As the control function executes every 10us, if we count 200.000 times the duty cycles will change every 2 second.

```
void control(){
  if(ChangeVout<=200000)                                {duty1=1039;duty2=239;ChangeVout++;}
  else if(ChangeVout<=400000 && ChangeVout>200000){duty1=1167;duty2=239;ChangeVout++;}
  else if(ChangeVout<=600000 && ChangeVout>400000){duty1=1295;duty2=239;ChangeVout++;}
  else if(ChangeVout<=800000 && ChangeVout>600000){duty1=1359;duty2=303;ChangeVout++;}
  else if(ChangeVout<=1000000 && ChangeVout>800000){duty1=1359;duty2=415;ChangeVout++;}
  else if(ChangeVout<=1200000 && ChangeVout>1000000){duty1=1359;duty2=495;ChangeVout++;}
  else{ChangeVout=0;}
}
```

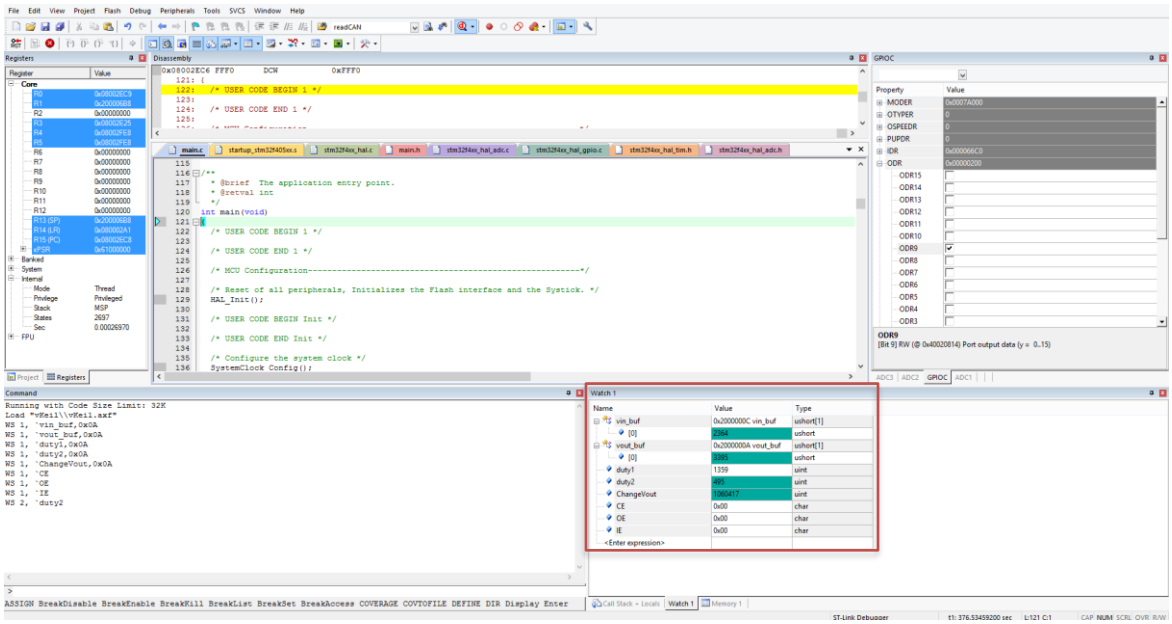


Figure 4.16: Keil debugger

In order to validate the new control, we use the Keil integrated debugger, it allows us to monitor variables live, and even change them, it also allows to access peripheral registers. Figure 4.14Figure 4.16 shows the debugging mode, in it the watch window shows some useful variables, like the error flagsfor input and output voltages and the two duty cycles. In this case, the counter is at 1060417 which is at the range for d1=1359 (1359/1599=0.85) and d2=495 (495/1599=0.31), if we look at the output voltage value it is Vout=3395 which converts to the expected output voltage.

$$\frac{3395}{2^{12}} \cdot 3.3 \cdot 11 = 30V$$

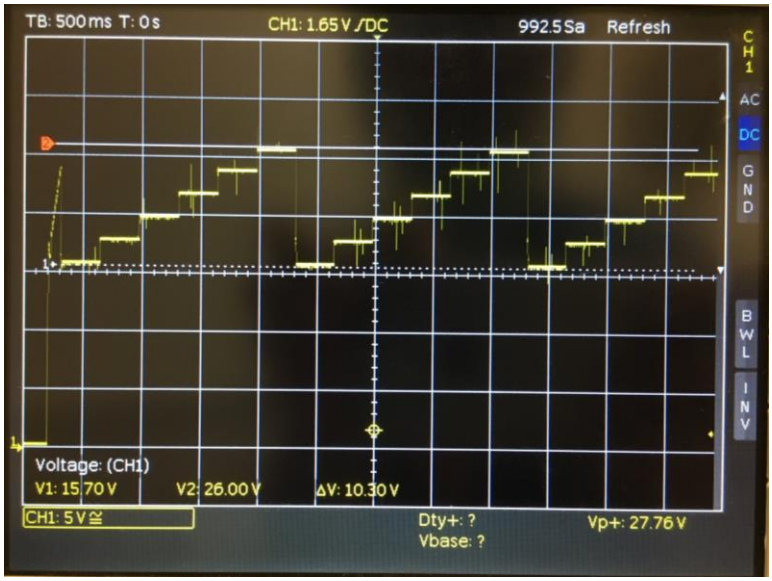


Figure 4.17: Voltage steps oscilloscope capture

As seen in Figure 4.17 the voltage steps from 16V to 26V as expected, transitioning from buck to boost without issues, so with this final experiment we can assure the converter model is working as calculated in the design phase. Now we can move with confidence into validating the control design.

Before testing the complete control system designed previously, we start with a simple proportional control in buck mode, with an input voltage of 28V

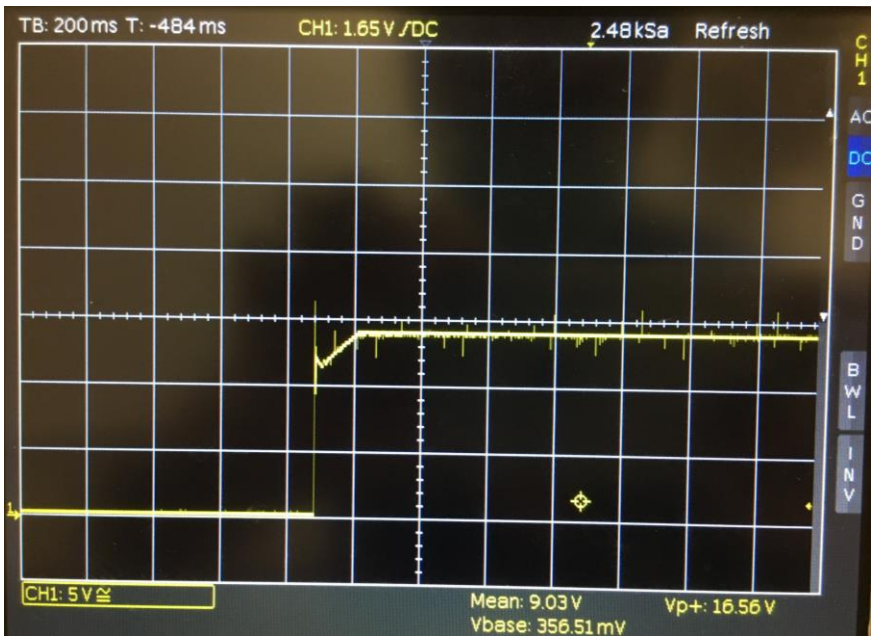


Figure 4.18: System response with a P control

With this attempt we do not expect to get a perfect response, but to test that the voltage conversions and the control loop is working as expected. In Figure 4.18 we see the response of the system when it's activated, first it has a very steep climb, this is because the input side MOSFET is 100% open, so the voltage quickly rises, after that the control does a more progressive rise, but it settles to an output voltage of 14V, which is not the target, this is because the system does not have an integral component.

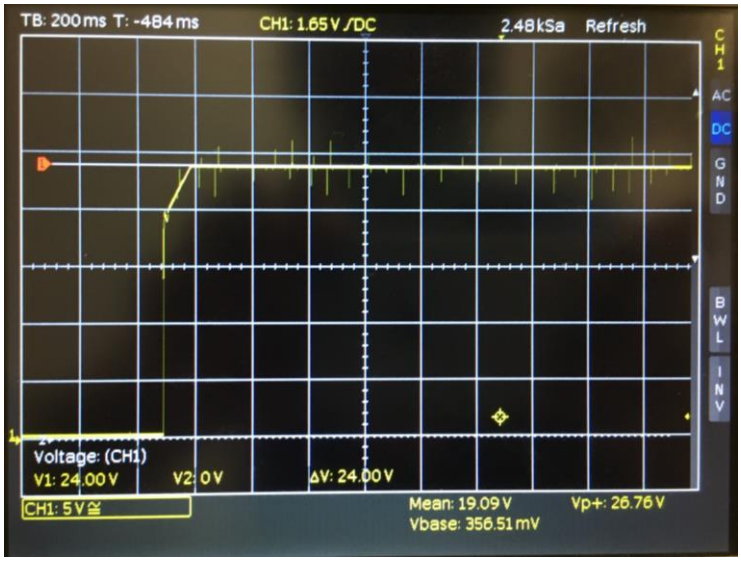


Figure 4.19:Vout response with PI control

Once we include an integral component to the control loop, the output voltage quickly stabilizes into the desired output voltage of 24V, without any disturbances or overshoots. This PI control loop would be enough for a buck converter, but as we desired a buck-boost we would need to test the designed double loop PI control.

At the point in time this thesis was submitted, The double loop PI control designed previously did not perform as expected, after some tuning we could not manage to implement it as it had stability issues, this could be due to discretization issues not taken into account.

Figure 4.20: CAN sequence being transmitted

As seen in Figure 4.20 the CAN module in the microcontroller is transmitting a CAN message, and the timing is precise, without any disturbance from the other code running. In the oscilloscope we see CAN high, low and the subtraction of the two, so we can appreciate how the CAN protocol is resilient to EMI, as the disturbances in each of the CAN lines gets cancelled.

5. Budget

This project budget could be divided in the designing and simulation phase, the PCB and the components used to manufacture the DC-DC voltage regulator cost.

5.1. Design and simulation

The design phase involved the software used to develop the schematics and the PCB layout (Altium), the microcontroller programming and simulation (CubeMX and Keil), and the simulation of the DC-DC regulators (MatLab); and the hours implied in the designing phase. Most of this software is acquired by free licenses or by free keys handled by the company thanks to some sponsorship.

Resource	Provider	Price	Observations
Design	Pol Codina	2240€	expected working time 224h
Altium license	Altium	0€	Free acquired(sponsored)
MatLab license	MatLab	0€	Student license
Cube MX license	ST electronics	0€	Free software
Keil license	arm Keil	0€	Free software

Table 5.1: Design and simulation costs

5.2. PCB costs

The PCB was manufactured by the Chinese company JLCPCB, the minimum order is 5 PCBs, but for this cost exercise we only account for 1, also the shipping costs that can go up to 18€ are not accounted as the order was shared. The assembly time and testing is estimated for 76h but at 0€/h as this is a voluntary project and not part of a company.

Resource	Provider	Price	Observations
Assembly	Pol Codina	0€	expected working time 76h
4 layer PCB	JLC PCB	5.92€	shipping costs saved by joint order and cost of single PCB
Tin	Driverless UPC	0€	Free acquired
Flux	Driverless UPC	0€	Free acquired
Soldering station	Driverless UPC	0€	Free acquired
Lab equipment	Driverless UPC	0€	Free acquired

Table 5.2: PCB costs

The most expensive side of the project was the cumulative cost of all the electronic components, here is a cost breakdown and total.

Comment	Designator	mouser	Quantity	Price	Total
Capacitor	C2, C3, C8, C9, C24, C25, C29, C30	81-GCM1885C1H330J16D	8	0,14€	1,14€
Capacitor	C4, C5, C22, C28	80-C0603C220J5G3190	4	0,22€	0,86€
Capacitor	C6, C48, C49, C51, C52	81-GRM188R6YA106MA3J	5	0,44€	2,21€
Cap Pol2	C18, C20	667-EEE-1CA470WR	2	0,30€	0,59€
Capacitor	C19, C21, C37, C41	581-0603DD105KAT2A	4	0,60€	2,41€
Capacitor	C23, C26, C27, C33, C34, C38, C40, C42, C44, C45, C46, C47	810-CGA3E2X7R1H104K	12	0,07€	0,85€
Capacitor	C31, C32, C50	581-06035F103MAT2A	3	0,22€	0,65€
Capacitor	C35, C36	77-VJ0603D150FXPAJ	2	0,69€	1,39€
Capacitor	C39, C43	81-GRM188R6YA225KA2D	2	0,21€	0,41€
SN65HVD234	CAN1	595-SN65HVD234DR	1	2,48€	2,48€
Clock2	CLK1	815-M8AIG-20-12-1ZT	1	0,66€	0,66€
Diode	D1, D2, D3, D4	755-RFN1LAM6STFTR	4	0,40€	1,58€
D Schottky	D5, D6, D7, D8, D9, D10, D11, D12, D13, D14	581-SD0805S040S0R5	10	0,29€	2,90€
ACT45B	F1	810-ACT45B5102PTL003	1	1,95€	1,95€
INA194	INA194	595-INA194AIDBVR	1	2,26€	2,26€
29.5uH	L1	673-PG1096.333NLT	1	5,31€	5,31€
EMI Resistor	LEMI1, LEMI2, LEMI3, LEMI4	710-742792116	4	0,23€	0,92€
NMOS DC BUCK destined	MA, MB, MC, MD	726-BSC022N04LS6ATMA	4	1,41€	5,64€
STM32F405RG	MCU1	511-STM32F405RGT6	1	9,83€	9,83€
OPA2735AID	OpAmp1	595-OPA2735AID	1	1,52€	1,52€
Resistor	R8, R9, R13, R14	667-ERJ-PA3J1R0V	4	0,10€	0,40€
Resistor	R11	588-FCSL64R005DER	1	1,05€	1,05€
Resistor	R16, R17, R18, R19	667-ERJ-3RED15R0V	4	0,14€	0,58€
TSV	SZNUP1	863-SZNUP2105LT3G	1	0,34€	0,34€
TPS7A1633QDGNR	TPS7A1633QDGNR	595-TPS7A1633QDGNRQ1	1	2,60€	2,60€
12VDC Switching regulator	TSR 1	495-TSRN1-24120SM	1	10,53€	10,53€
Mosfet Driver	UCC1, UCC2	595-UCC27282DRCT	2	2,41€	4,82€
Zener Diode	ZD1, ZD2	771-BZT52H-C3V3-T/R	2	0,15€	0,30€
TOTAL					66,18€

Table 5.3: Electronic components cost breakdown

Even if the cost of PCB + components is still lower than some equivalent DC-DC alternatives, the cost represented here also includes the microcontroller and CAN filtering which would be included either way in the actual implementation of the converter in the team, so the actual savings are even higher.

6. Conclusions and future development:

As mentioned before this project is built around the Formula Student team Driverless UPC, it aims to provide a proof of concept for a self-developed DC-DC converter that integrates better with the already existing electronics package of the team.

The project was successful at proving that a microcontroller is powerful and safe enough to provide the control loop for a 4-switch buck-boost topology and still be able to perform other tasks like communications. However, some mistakes were made in the implementation of the design and prevented it from performing at its maximum potential, the erroneous footprint of the MOSFETs made it impossible to load the system with the proposed 10A of maximum current, so high-power delivery could not be tested.

Moreover, it has proven to be challenging to control the 4 switches and to design a stable control, so for future cars the ideal solution would be to design a new battery configuration with a higher voltage range (27V-38V, a 9s-2p) that way the regulator only needs to work in buck mode, so a simple 2 switch topology and a voltage controlled PI would be enough.

The power stage analysis and control design turned out to be accurate and could be validated with Simulink and the hardware implementation. Also, the proposed control algorithm was able to run on the microcontroller with the expected timing and without any issues while performing other tasks like communicating over CAN, and the safety measures were tested and were as effective as expected.

In conclusion, some of the techniques and studies performed in this project will result useful for the future development of controllers in the team, even if they operate with different topologies or different control methods, this project opens the door to new DC-DC converters that are better suited to the team electronics package and are more affordable and easier to repair.

7. Bibliography

[1] "Formula Student Germany," 2020. [Online]. Available: https://www.formulastudent.de/about/concept/.

[2] Vicor. [Online]. Available: https://www.mouser.es/datasheet/2/685/DCM3623x50M26C2y7z_ds-1144754.pdf.

[3] Linear Technologies, "Analog Devices," [Online]. Available: https://www.analog.com/media/en/technical-documentation/data-sheets/8390fa.pdf.

[4] K. Ogata, "Discrete-Time Control Systems," Prentice Hall, p. sec. 4.5.

[5] Texas instruments, [Online]. Available: https://www.ti.com/lit/an/slyt664/slyt664.pdf?ts=1601748882612&ref_url=https%253A%252F%252Fwww.google.com%252F.

[6] Texas Instruments, [Online]. Available: https://www.ti.com/lit/ds/symlink/ucc27282.pdf?ts=1601463462353&ref_url=https%253A%252F%252Fwww.ti.com%252Fstore%252Fti%252Fen%252Fp%252Fproduct%252F%253Fp%253DUCC27282DRCT.

[7] STMicroelectronics. [Online]. Available: https://www.st.com/resource/en/datasheet/stm32f405rg.pdf.

[8] STMicroelectronics, [Online]. Available: https://www.st.com/resource/en/user_manual/dm00105879-description-of-stm32f4-hal-and-ll-drivers-stmicroelectronics.pdf.

[9] Matlab, Mathworks, [Online]. Available: https://www.mathworks.com/products/simscape-electrical.html.

Appendices:

7.1. MatLab code

```
clear all
close all
C=1000e-6;
L=29.5e-6;
R=24/10;
Vin=30;
Vout=24;
D1=0.85;
D2p=0.85;
d1=0.68;
d2=0.29;

P=bodeoptions;
P.FreqUnits='Hz';
P.Grid='on';
P.PhaseWrapping='off';
P.PhaseMatching='on';

s=tf('s');

Gc1=(Vout*D2p^2/d1)/(L*C*s^2+L*s/R+D2p^2)
subplot(2,2,1)
margin(Gc1)
%title 'Vout/d1'

Gl1=Vin*(C*s+1/R)/(L*C*s^2+s*L/R+D2p^2)
subplot(2,2,2)
margin(Gl1)
%title 'IL/d1'

Vin=21;
Vout=24;
D1=0.85;
D2p=0.71;
d1=0.85;
d2=0.29;

Gc2=(Vout/(D2p^2*R))*(-L*s+R*D2p^2)/(L*C*s^2+L*s/R+D2p^2)
subplot(2,2,3)
margin(Gc2)
%title 'Vout/d2'

Gl2=Vout*(C*s)/(L*C*s^2+s*L/R+D2p^2)
subplot(2,2,4)
margin(Gl2)
%title 'IL/d2'



figure()
hold on;
for Vin=20:2:30
    if(Vin>Vout)
        d1=0.85*Vout/Vin
```

```
        D2p2=0.85;
        Gc1=(Vout*D2p^2/d1)/(L*C*s^2+L*s/R+D2p^2);
        bode(Gc1,P)

set(findall(gcf,'type','line'),'linewidth',2);
    else
        D2p=0.85*Vin/Vout
        D1=0.85;
        d2=1-D2p;
        Gc2=(Vout/(D2p^2*R))*(-L*s+R*D2p^2)/(L*C*s^2+L*s/R+D2p^2);
        bode(Gc2,P)

set(findall(gcf,'type','line'),'linewidth',2);
    end

end


figure()
hold on;
for Vin=20:2:30
    if(Vin>Vout)
        d1=Vout/Vin*0.85
        D2p2=0.85;
        Gl1=Vin*(C*s+1/R)/(L*C*s^2+s*L/R+D2p^2)
        bode(Gl1,P)
        set(findall(gcf,'type','line'),'linewidth',2);
    else
        D2p=0.85*Vin/Vout
        D1=0.85;
        d2=1-D2p;
        Gl2=Vout*(C*s)/(L*C*s^2+s*L/R+D2p^2)
        bode(Gl2,P)
        set(findall(gcf,'type','line'),'linewidth',2);
    end

end

%%
%Control Design
clear all;
close all;
C=1000e-6;
L=29.5e-6;
R=24/10;
Vin=20; %worst case
Vout=24;
D1=0.85;
D2p=0.708;
d2=1-D2p;
Rsense=5e-3;
currGain=50;
Rs=Rsense*currGain; %current sensor gain
H=1/11;  %voltage sensor gain
P=bodeoptions;
P.FreqUnits='Hz';
P.Grid='on';
P.PhaseWrapping='off';
P.PhaseMatching='on';
```

```
s=tf('s')
%Transfer function of Vo/d
figure()
Gc2=(Vout/(D2p^2*R))*(-L*s+R*D2p^2)/(L*C*s^2+L*s/R+D2p^2)
bode(Gc2,P)
set(findall(gcf,'type','line'),'linewidth',2);

%Transfer function of Il/d
figure()
Gl2=Vout*(C*s)/(L*C*s^2+s*L/R+D2p^2)
bode(Gl2,P)
set(findall(gcf,'type','line'),'linewidth',2);

%inner loop no control
Vm=1;
PIcurr=1;
Tc=PIcurr*Rs*Gl2/Vm;
figure();
bode(Tc,P)

%PI compensation inner loop(current)
Vm=1;
PIcurr=0.6158+7735/s
Tc=PIcurr*Rs*Gl2/Vm;
hold on;
bode(Tc,P)
bode(PIcurr)
hold off;

%outer loop no control
Gvi2=Gc2/Gl2
PIvolt=1;
Tv=Gvi2*H*PIvolt/Rs
figure()
pzplot(Tv)
figure()
bode(Tv,P)
set(findall(gcf,'type','line'),'linewidth',2);
hold on;
%compensation outer Loop
PIvolt=5.87+14752/s;
Tv=Gvi2*H*PIvolt/Rs;
bode(Tv)
bode(PIvolt);
hold off;
set(findall(gcf,'type','line'),'linewidth',2);

%%

%In buck mode
close all
Vin=30; %worst case
Vout=24;
D1=0.68;
D2p=0.85;
d2=1-D2p;
Gc2=(Vout*D2p^2/D1)*(1)/(L*C*s^2+L*s/R+D2p^2)
Gl2=Vin*(C*s+1/R)/(L*C*s^2+s*L/R+D2p^2)
```

```
%PI compensation inner loop(current)
Vm=1;
PIcurr=0.6158+7735/s
Tc=PIcurr*Rs*Gl2/Vm;
figure();
hold on;
bode(Gl2,P)
margin(Tc)
hold off;
%compensation outer Loop

Gvi2=Gc2/Gl2
PIvolt=1;
Tv=Gvi2*H*PIvolt/Rs;
figure()
bode(Tv,P)

PIvolt=5.87+14752/s;
Tv=Gvi2*H*PIvolt/Rs;
hold on;
margin(Tv)
set(findall(gcf,'type','line'),'linewidth',2);
```

Glossary

A list of all acronyms and the meaning they stand for.

ECU Electronic Control Unit

DC Direct Current

PCB Printed Circuit Board

HV High Voltage

LV Low Voltage

CAN Controller Area Network (Communication protocol used in automotive)

WP Work Packages

FSG Formula Student Germany

AV Autonomous Vehicles

PI Proportional Integral

PWM Pulse Width Modulation

MCU Microcontroller Unit

STM32 STMicroelectronics 32-bit integrated Microcontroller

ADC Analogic-to-Digital Converters

DMA Direct Memory Access

APB Advanced Microcontroller Bus

EMI Electromagnetic Interference

SOIC Small outline integrated circuit