POLITECNICO DI MILANO

Facoltà di Ingegneria

Scuola di Ingegneria Industriale e dell'Informazione

Dipartimento di Elettronica, Informazione e Bioingegneria

Master of Science in Electronics Engineering

Design and Implementation of a Driver for In-Wheel Brushless Motors for Unmanned Vehicles

Supervisor:

PROF. GIAMBATTISTA GRUOSSO

Co-Supervisor:

PROF. LUCA BASCETTA

Master Graduation Thesis by:

ARTURO MONTÚFAR ARREOLA
Student Id n. 840541

Academic Year 2016-2017

COLOPHON

This document was typeset using the typographical look-and-feel `classicthesis` developed by André Miede. The style was inspired by Robert Bringhurst's seminal book on typography "*The Elements of Typographic Style*". `classicthesis` is available for both LaTeX and LyX:

`http://code.google.com/p/classicthesis/`

The template has been adapted by Emanuele Mason, Andrea Cominola and Daniela Anghileri: *A template for master thesis at DEIB*, June 2015

Final Version as of December 2, 2017 (`classicthesis`).

To my family and friends:

Thanks for supporting me while I'm learning to fly

— Arturo

ACKNOWLEDGMENTS

I want to thank the following people for helping me with the development of this work:

To Professor Giambattista Gruosso, for giving me the opportunity to develop such an interesting thesis project and for his guidance through the research method for this kind of projects, which is as valuable as the knowledge I gained regarding the topic developed in this work.

To Professor Luca Bascetta, for giving me advice and material regarding the research and implementation of the motor control theory and the work already developed regarding this project.

To Professor Federico Terraneo, for his support and interest on the implementation of Miosix in this project, and to the Skyward Experimental Rocketry team, for their help with the CAN bus driver.

To Marco Bauer, for his support in the aspects related to the mechanics of the project and also for providing me with information of the work already developed.

To Professor Luigi Piegari, for his support and advice on the final tests.

To Giulio Fontana, for his support and advice regarding the manufacture of the systems developed for this work while working in the AIRLab.

To Professor Luigi Colombo, for providing me documentation regarding the thermal management proposal mentioned in this work.

To Stefano Bianchi, for helping with the fine details of the tunning of the controllers and with the development of the tests.

To Milica Jovanovic, for helping me with the documentation of the software architecture and with the flowcharts.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

LISTINGS

ACRONYMS

ADC	Analog-to-Digital Converter
AMU	Arm Management Unit
BEMF	Back Electromotive Force
BLDC	Brushless Direct Current
BMU	Battery Management Unit
CAN	Controller Area Network
DC	Direct Current
DSP	Digital Signal Processor
ECU	Electronic Control Unit
ESC	Electronic Speed Controller
FOC	Field Oriented Control
FPU	Floating Point Unit
I2C	Inter-Integrated Circuits
IC	Integrated Circuit
IGBT	Insulated-Gate Bipolar Transistor
ISR	Interrupt Service Routine
KCL	Kirchhoff's Current Law
LDO	Low Drop-Out

MOSFET	Metal–Oxide Semiconductor Field-Effect Transistor
MSB	Most Significative Bit
NTC	Negative Temperature Coefficient
OS	operating system
PCB	Printed Circuit Board
PI	Proportional-Integral
PMAC	Permanent Magnet Alternating Current
PMSM	Permanent Magnet Synchronous Motor
PP	Pole Pairs
PWM	Pulse Width Modulation
SPI	Serial Peripheral Interface
SSI	Synchronous Serial Interface
TI	Texas Instruments
UART	Universal Asynchronous Receiver-Transmitter
UI	User Interface
USB	Universal Serial Bus
VMU	Vehicle Management Unit
WMU	Wheel Management Unit

ABSTRACT

The purpose of this thesis is to study the architecture of a brushless motor driving circuit proposed by Texas Instruments and implemented as an electronic speed board in an open-source project available online, analysing the advantages and disadvantages of such design regarding the implementation of trapezoidal and sinusoidal motor driving and speed and current control techniques for an unmanned vehicle designed for robotic agriculture. After this, the implementation of such driving and control techniques was physically carried out and tested.

This thesis explains in details the most important parts regarding the physical implementation of a motor driving system in such a way that it can be fully replicated. Chapter 1 contains some initial information regarding the electric motor history and the motivation to realize this work. In Chapter 2, we explain more in detail the different reasons why this work was developed and the focus points that were stressed out. In Chapter 3, a simple but sufficient explanation about the theory behind the electric motor is given, explaining also the different existing technologies and their particular driving methods. In Chapter 4 we explain the case study of ROBI', a prototype mobile manipulator for agricultural applications, which uses the in-wheel motor for which the motor driver of this work was developed. In Chapter 5 we explain in deep detail all the work developed around the implementation of the motor driver, both in the software and the hardware fields. In Chapter 6 we explain the final results of the work, showing and comparing graphs and waveforms of the behavior of the in-wheel motor driven by the system developed. Finally, on Chapter 7, we discuss the results of the work realized and propose changes to the design of the driver, based on the implementation of the researched architecture and the problems encountered while working on the project.

ESTRATTO

Lo scopo di questa tesi è quello di studiare l'architettura di un driver per motori brushless commerciale proposto da Texas Instruments e implementato come un circuito per il controllo di velocità come un progetto open-source disponibile online, analizzando i vantaggi e gli svantaggi di tale progetto per quanto riguarda l'implementazione di eccitazione a motore trapezoidale e sinusoidale e tecniche di controllo di velocità e corrente per un veicolo senza pilota progettato per l'agricoltura robotica. Successivamente, l'implementazione di tali tecniche di eccitazione e controllo è stata effettuata e testata fisicamente.

Questa tesi spiega in dettaglio le parti più importanti riguardanti l'implementazione di un sistema di eccitazione del motore in modo tale da poter essere completamente replicato. Il capitolo 1 contiene alcune informazioni iniziali sulla storia del motore elettrico e la motivazione per realizzare questo lavoro. Nel capitolo 2, spieghiamo più in dettaglio i diversi motivi per cui questo lavoro è stato sviluppato e gli aspetti principali che sono stati evidenziati. Nel capitolo 3 viene fornita una spiegazione semplice ma sufficiente sulla teoria alla base del motore elettrico, che spiega anche le diverse tecnologie esistenti e i loro particolari metodi di eccitazione. Nel capitolo 4 spieghiamo il caso di studio di ROBI', un prototipo di manipolatore mobile per applicazioni agricole, che utilizza il motore in-wheel per il quale è stato sviluppato il driver motorio di questo lavoro. Nel capitolo 5 spieghiamo dettagliatamente tutto il lavoro sviluppato attorno all'implementazione del driver del motore, sia nel campo del software che in quello dell'hardware. Nel capitolo 6 spieghiamo i risultati finali del lavoro, mostrando e confrontando grafici e forme d'onda del comportamento del motore a ruote motrici guidato dal sistema sviluppato. Infine, nel capitolo 7, discutiamo i risultati del lavoro svolto e proponiamo modifiche al design del driver, basate sull'implementazione dell'architettura ricercata e sui problemi incontrati durante il lavoro sul progetto.

PREFACE

Knowledge is only part of understanding.
Genuine understanding comes from hands-on experience.

— Seymour Papert *Constructionism* 1991

Motor control is one of the many engineering topics that must be experienced personally to be fully understood. All the theory behind the movement of the shaft of the electric motor, which explains all the different phenomena interacting to create mechanical motion from electrical energy, should be experienced in firsthand to realize everything that is involved. This is the reason why the practical implementation of theoretical topics is always interesting. Practical implementation makes us realize that there are always challenges that might not be taken into account while they are being studied from books.

This text was written with the idea of becoming a guide on the development of a motor controller for future projects, not only for the case of study presented. This is the reason why it includes a wide explanation from, the basic physical phenomena that acts on electric motors and the important parameters to consider for the prediction of its motion, to the implementation of the hardware and software of a driving circuit and a detailed explanation on the most important factors to bring a motor controller alive and to successfully drive a brushless motor.

— Arturo Montúfar Arreola

1 INTRODUCTION

Electrical science, too, by its fascination, by its promises of immense realizations, of wonderful possibilities chiefly in humanitarian respects, has attracted the attention and enlisted the energies of the artist; for where is there a field in which his God-given powers would be of a greater benefit to his fellow-men than this unexplored, almost virgin, region, where, like in a silent forest, a thousand voices respond to every call?

— Tesla *On Electricity* 1897

The electric motors made their first appearances in the middle of the XIX century right after the invention of the battery by the Italian physicist, chemist and inventor Alessandro Volta in 1800, the discovery of the generation of a magnetic field from an electric current by the Danish physicist and chemist Hans Christian Ørsted in 1820 and the invention of the electromagnet by the English physicist William Sturgeon in 1825 (Doppelbauer, 2012). After these foundations were laid, the development of a machine that generates mechanical power from electrical power has been improving day by day, and, along with that improvement, also its utility has been increasing.

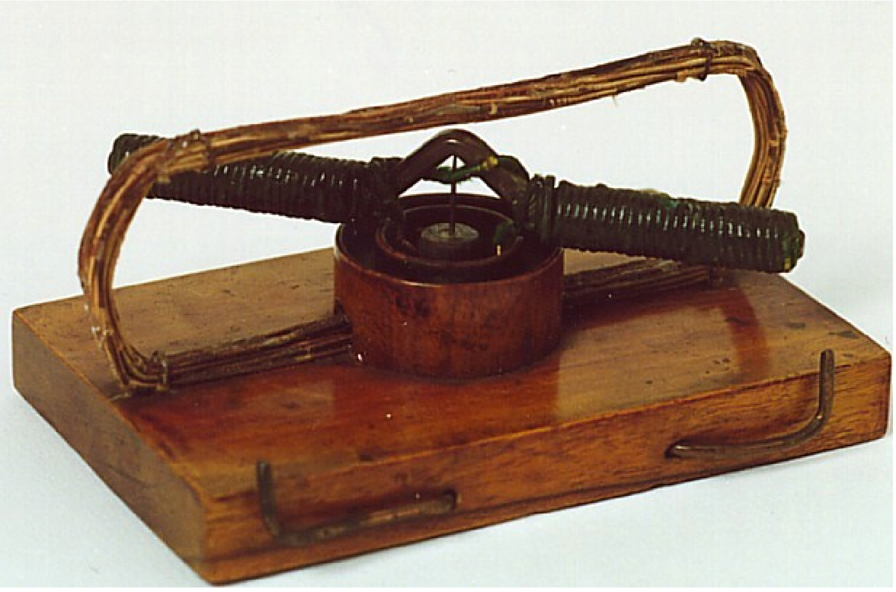


Figure 1.1: Jedlik's "Electromagnetic Self-Rotor". The historic motor created by the Hungarian physicist Ányos Jedlik still works perfectly today in the Museum of Applied Arts in Budapest.

Due to the reduction of the prices of metals and the improvement and automation of manufacture processes, electric motors became available for a large range of applications, and not only as a research topic, up to the point that nowadays we interact with them in our daily life, sometimes without even noticing it. We have electric motors in all types of devices, from small applications like home appliances and hand-held gadgets, to large applications like robotics, cars and spaceships. As the complexity of the application increases, also the need for accuracy and efficiency does, leading to the development of more advanced electric motor technologies which lead to complex motion control techniques.

One of the most complex applications for motor control is robotics. The motion in a robotic system is part of its definition of automatic movement, therefore, a robotic system needs a predictable driving system to fulfill its purpose, which implies that most of the parameters of the motors inside the robot are known and that they can be controlled in a correct and precise way.

A challenging application regarding motors in robotics is the wheel driving, since it needs to be precise and powerful at the same time to transport the robotic system around large surfaces in an unmanned way, which means that there is no person on board and controlling the robot. For example, in robotic agriculture, which is one of the main reasons why this work was developed, a robot becomes an unmanned vehicle, which must transport itself around in farms, where the road represents harsh conditions for transportation, introducing the need for a precise drive to deal with small crops, a high torque to transport the robot in uneven grounds and a good range of speeds to displace itself in large field areas in the fastest way possible.

With aims to propose a solution for the land transportation in robotic agriculture projects, different types of motor technologies were studied a priori, finding out that in-wheel permanent-magnet brushless motors are one of the most suitable and currently used solutions to develop electric vehicles, since they don't need an external system for power transmission from the motor to the wheels, taking advantage of the high stall torque of the electric motors and the reduction of the space and weight that having a motor inside a wheel represents (Bascetta, Baur, and Gruosso, 2017).

To drive permanent-magnet brushless motors it's necessary to apply a proper driving technique, which becomes a complex task when the goal is to achieve an optimal performance. Since the brushless motor technology is quite recent in comparison with other electric motors, there is still a lot of development going on with aims to reach the highest performance at the lowest cost.

The objective of this work was to develop an electronic driver to control brushless motors, taking as a case study the in-wheel motors that transport a skid-steering robot designed for robotic agriculture. We studied the available architectures to implement a motor drive, the different approaches to drive brushless motors and the peripherals required. After this, the motor driver was physically implemented and some changes in the architecture were proposed from the results and conclusions obtained from the implementation and testing of the circuit.

Figure 1.2: Rimac Automobili's electric supercar Concept S. Electric cars are becoming a trend and one of the main drivers for the development of technology related to electric motors.

This thesis explains in detail the most important information regarding the physical implementation of a low-cost motor driving system in such a way that it can be fully replicated. In Chapter 2, we explain more in detail the different reasons why this work was developed and the focus points that were stressed out. In Chapter 3, a simple but sufficient explanation about the theory behind the electric motor is given, explaining also the different existing technologies and their particular driving methods. In Chapter 4 we explain the case study of ROBI', a prototype mobile manipulator for agricultural applications, which uses the in-wheel motor for which the motor driver of this work was developed. In Chapter 5 we explain in deep detail all the work developed around the implementation of the motor driver, both in the software and the hardware fields. In Chapter 6 we explain the final results of the work, showing and comparing graphs and waveforms of the behavior of the in-wheel motor driven by the system developed. Finally, on Chapter 7, we discuss the results of the work realized and propose changes to the design of the driver, based on the implementation of the researched architecture and the problems encountered while working on the project.

2 PROBLEM STATEMENT

As motion applications complexity increases, like in the robotics field, the need for a more complex and robust driving system arises, and with it, the need to make decisions regarding the technologies to be implemented for them. For example, when a robotics project begins, with the objective to implement a solution that hasn't been implemented before, one of the main concerns is the motion system. The selection of the motor that is going to be used in a project is not an easy task, since many details must be considered, like the speed and torque required by the application, which lead to the selection of the motor technology, which might be limited by the available power source, the computing power and the economic budget of the project. These aspects have a strong dependency between each other and choosing each one of them represents a trade-off that must be studied in detail to reach an optimal result for the desired application.

2.1 TECHNOLOGICAL TRADE-OFFS

One of the main goals in the technology research and development has always been to reduce the trade-offs between the most important parameters in different engineering applications, using the most suitable technologies available, depending on the application requirements. One of the fields of engineering that has a big focus on the reduction of trade-offs is the motor drive research, since their development is crucial to drive innovation in many engineering fields.

A complicated trade-off to deal with is the cost of a motor drive, which is one of the most limiting characteristics project developments, both for research activities and for industrial and consumer applications. If a project is not expected to generate an income after its development or it's not backed up by a research institute with resources, or if the product is expected to be kept at a low price for many different reasons (like accessibility for low income communities or the possibility of large expansion), the price of a motor drive might be the weak side in a trade-off analysis and the development of the project might not be optimal, since it might require more work to reach an acceptable result.

2.2 LEARNING CURVE

The cost of a project leads to another important trade-off on engineering research and development: the learning curve. In some areas like robotics, a professional engineering project requires thousands of hours of man-work to reach a desirable result. An important amount of the time available for these projects is used in learning theory regarding the technology related to the project and how the available tools for such technology work. This time consumed in learning all the information needed regarding a project is called "learning curve". The amount of work required to reach a good result makes most of the engineering projects expensive and, in many cases, a big part of that time is consumed by the learning curve, time that could be used for developing another part of the project, increasing the speed of the development and, therefore, reducing its cost.

2.3 MOTOR DRIVER DEVELOPMENT

With the aim to solve these two important trade-offs, we decided to study the possibility to implement a low-cost motor drive system by using a cheap architecture which can be easily modified, making the development around it faster and easier for future projects.

To set up the design specifications for the motor driver to be studied, we based the requirements in the drive characteristics of a robot designed for robotic agriculture by the engineering group of the AIR-Lab of the Politecnico di Milano called ROBI' (Bascetta, Baur, and Gruosso, 2017). This robot is driven by four in-wheel brushless motors that will be described in detail in Chapter 4.

2.4 IMPLEMENTATION OF FIELD ORIENTED CONTROL

Having the objective defined, we looked for an available motor driver design with aims to test its capabilities and, based on that, propose changes on its architecture to optimally drive the in-wheel brushless motors.

The platform found is an electronic board called VESC Board. This circuit is very appealing since it has many peripherals and because its technology is easy to study, since it's based in a design proposed by Texas Instrument to drive three-phase motors.

Since this motor driver will be used in robotics applications, we had the necessity to modify the source code of the microcontroller to apply motor control algorithms that would be different to the ones available for the VESC board since the code available was designed to drive an electric skateboard. For this aim, we decided to develop our own source code, this way the implementation of the driving algorithms wouldn't require reverse engineering on the code available.

Since the motors of the wheels of the robot are Permanent Magnet Synchronous Motor (PMSM), they have a sinusoidal configuration, and therefore, a sinusoidal waveform should be applied into the motor to reach an optimal performance. This specification pointed us towards the implementation of the Field Oriented Control (FOC) driving method, which improves the torque capabilities of the motor, as it will be explained in Chapter 3.

3 MOTOR CONTROL THEORY

In this chapter, we will review the theoretical principles that concern us regarding motor control. First, we will review the physical principles that rule over the electric motors and we will explain the different motor technologies and its configurations, focusing in the Permanent Magnet Alternating Current (PMAC) motors. Later, we will review the motor drives, the configuration of a driver and the different driving techniques for the PMAC motors. Finally, we will explain the most common control methods that can be applied to them.

Most of the theory in this chapter was taken from the book *Tecnologie dei sistemi di controllo*, by Magnani, Ferretti, and Rocco, 2007 and from the compilation *Performance and Design of Permanent Magnet AC Motor Drives*, by Bose et al., 1989, as well as from the *Lecture notes in Power Electronics*, by Ghioni, 2016.

3.1 ELECTRIC MOTOR

An electric motor is an electric machine that transforms electrical power:

$$P_{electrical} = V \cdot I \tag{3.1}$$

into mechanical power:

$$P_{mechanical} = T \cdot \omega \tag{3.2}$$

by means of the electromagnetic phenomena that takes place inside the motor, which is explained by the physical principles mentioned in this chapter.

3.1.1 *Physical Principles*

The generation of torque in electric motors is based on the interaction of two magnetic fields, one generated by magnets or windings placed in the stator and the other one generated by magnets or windings placed in the rotor (Figure 3.1).

The physical laws that rule over the torque and speed generation are mainly four: The Lorentz's Law, which helps us define the torque generated by an electric charge moving inside a magnetic field; Faraday's Law of Electromagnetic Inductance and the Lenz's Law, which

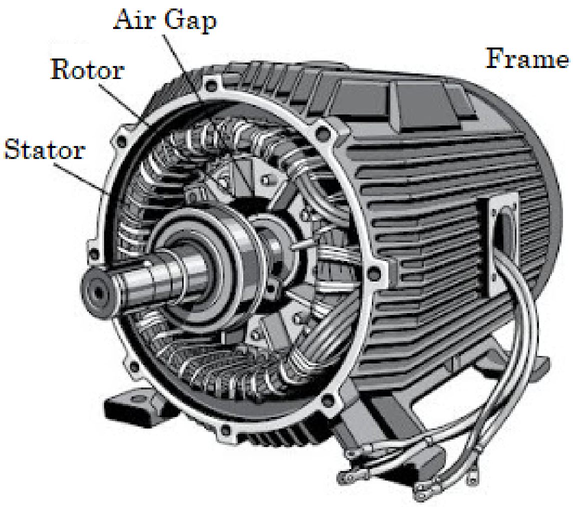


Figure 3.1: Partially Assembled Motor. An electric motor consists mainly in three parts: a rotor, which moves due to the electromagnetic interaction; a stator, the body of the motor; and a frame, which holds the rotor and the stator together. The air gap is the space between the rotor and the stator, where the electromagnetic interaction takes place.

explain the generation of the Back Electromotive Force (BEMF) in the motor coils depending on the speed of the rotor and the influence of the magnetic field generated due to this BEMF respectively; and the Ampere-Laplace Law, which allows us to calculate the magnetic field of a current loop.

LORENTZ'S LAW defines a force F, which acts over an electric charge q moving with a speed v inside a magnetic field with intensity B:

$$F = qv \times B \tag{3.3}$$

By defining a current I passing through a conductor with length l we can transform Equation 3.3 into Equation 3.4:

$$F = lI \times B \tag{3.4}$$

Considering the current I flowing through a conductive loop as the one in Figure 3.2 with side lengths l and h, we can see that there is a force F generated in the direction of the cross product of the current I and the magnetic field B. The maximum force F is generated in the sides of the loop where the direction of the current I is perpendicular to the direction of the magnetic field B (ab and cd), while on the other two sides (ad and bc) the forces generated are cancelled with each other due to the direction of the current respect to the magnetic field.

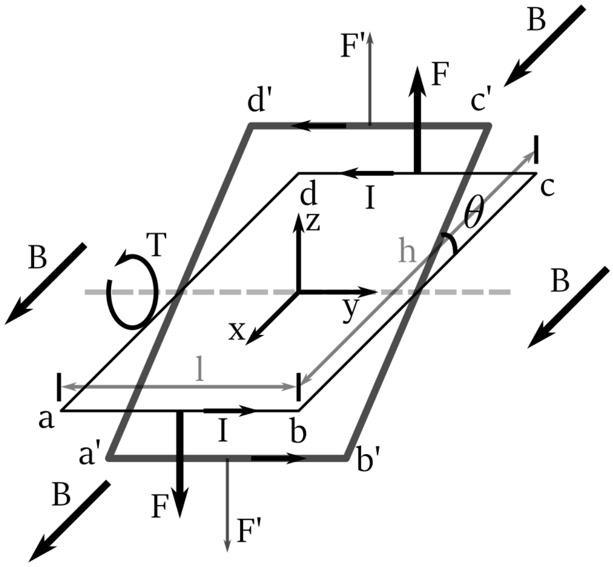


Figure 3.2: Visual explanation of the interaction of the current I and the magnetic field B generating a force F and, consecuently, a torque T around the y axis (Magnani, Ferretti, and Rocco, 2007).

Since the forces F generated on the sides ab and cd have the same magnitude but different direction, they create a torque T around the y axis:

$$T = IhlB\cos\theta \tag{3.5}$$

We can see in Figure 3.2 and in Equation 3.5 that, when the angle θ between the side ad (or bc) and the direction of the magnetic field B is $\pi/2$ radians, the torque T is zero, and when the angle θ is zero or π radians, the torque reaches its maximum possible value.

The dependency of the torque on the angle between the direction of the current and the magnetic field, introduces the need of changing the direction of the magnetic field or the direction of the current to maintain the loop spinning and a non-zero torque.

FARADAY'S LAW OF ELECTROMAGNETIC INDUCTION states that

> In every circuit under the effect of a magnetic field, an electromotive force is induced equal to the derivative respect to the time of the magnetic flux passing though the circuit, with negative sign (Jordan, 1968).

therefore, by indicating with E the electromotive force and with ϕ_m the magnetic flux, we have:

$$E = -\frac{\mathrm{d}\phi_m}{\mathrm{d}t} \tag{3.6}$$

If we consider a case like the one in Figure 3.2 we can calculate the magnetic flux passing though the loop as:

$$\phi_m = B \cdot u_N S = B \cdot u_N hl = hlB \sin\theta \tag{3.7}$$

where S is the surface of the loop and u_n is the direction normal to the plane of the loop. Therefore, we get an induced electromotive force:

$$E = -\omega hlB \cos\theta \tag{3.8}$$

where ω is the angular speed of the loop. We can see, comparing Equation 3.8 and Equation 3.5, that the induced electromotive force depends on the angular speed in the same way than the acting torque depends on the current.

LENZ'S LAW can be explained after the explanation of the induced electromotive force. It states that the induced current in a loop has the direction that creates a magnetic field that opposes the change in magnetic flux through the area enclosed by the loop, therefore, the induced current tends to keep the magnetic flux ϕ_m from changing in the circuit.

If the rotation of the loop is generated by the circulation of a current inside a magnetic field, the induced electromotive force will try to oppose to the pass of the current, that's why it's normally referred to as Back Electromotive Force (BEMF).

AMPERE-LAPLACE LAW is the last piece to understand the transformation of electrical energy into mechanical energy. It allows us to calculate the magnetic field generated by a closed loop conducting current in a point defined by a vector p as:

$$B(p) = \frac{\mu_0}{4\pi} \oint I \frac{u_t \times u_r}{r^2} dl \tag{3.9}$$

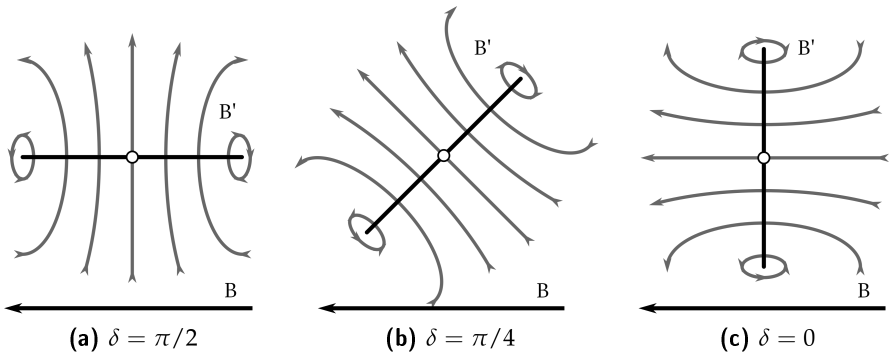
where μ_0 is the vacuum magnetic permeability constant, I is the current circulating through the loop, u_t is the versor with direction of the current in the infinitesimal element dl and u_r

and r are versor and module that define the point p respect to the infinitesimal element of the loop.

Given that the magnetic fields can be generated both by permanent magnets and by current circulation, the electromechanical conversion is obtained due to the interaction of two magnetic fields according to the alignment principle, which states that

> In a region of space which hosts two magnetic fields, there is a mechanical action that tends to align both fields. (Magnani, Ferretti, and Rocco, 2007)

If we consider the loop from Figure 3.2 and Equation 3.9, we can see that there is a magnetic field generated around the loop as seen in Figure 3.3, and due to the alignment principle, we will get the strongest coupling torque when the magnetic fields are perpendicular to each other.



(a) $\delta = \pi/2$ **(b)** $\delta = \pi/4$ **(c)** $\delta = 0$

Figure 3.3: Visual explanation of the alignment principle. In this Figure we can see the side *bc* of the loop from Figure 3.2, which is spinning around the y axis. The alignment torque between B and B' is the largest in the position 3.3a and zero in the position 3.3c.

In the case of electrical motors, there are two different magnetic fields in the airgap generated by the permanent magnets and the windings placed in the stator and the rotor, $B_r(\theta, t)$ and $B_s(\theta, t)$.

We get the electromechanical conversion from the interaction of those magnetic fields due to the torque which tends to align them:

$$T_m = kB_rB_s \sin\delta \tag{3.10}$$

where δ is the angle between the two fields. The maximum torque will be when $\delta = \pi/2$. In conclusion, by feeding the windings in the right way, we look forward to having a constant 90° angle between the two magnetic fields in aims to obtain the maximum torque generation.

In the case of the Direct Current (DC) motor, the perpendicularity between the magnetic fields is maintained by a polarity-commutating structure attached to the rotor which is connected to the windings where the current flows and generates the rotor magnetic field, which tries to the magnetic field of the permanent magnets attached to the stator. This commutating system is connected to the power supply by means of metallic brushes that energize the motor until it reaches a certain angular position, when the commutating lead changes, inverting the polarity of the windings. Therefore, the torque obtained is independent from the position of the rotor and it's proportional to the amplitude of the current.

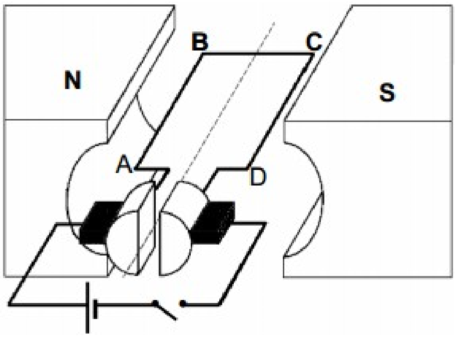


Figure 3.4: Basic schematic representation of the DC motor using the same loop represented in Figure 3.2

In brushless motors, which will be explained later in this chapter, the perpendicularity condition is maintained by feeding the windings in function of the angular position of the rotor θ.

If we clear from Equation 3.5, the torque T and the current I, grouping the rest of the variables into one constant, we can write down the following equation:

$$T = K_T I \tag{3.11}$$

where K_T is called Torque Constant and is defined by the geometry of the motor and the magnetic field in the airgap of the motor, which depend on the motor configuration and its construction technology. K_T units are Nm/A.

We can do the same with Equation 3.8, clearing the BEMF and the angular speed ω, getting the following equation:

$$BEMF = K_E \omega \tag{3.12}$$

where K_E is called Voltage Constant and its units are $V/(rad/s)$.

If we consider the loop of Figure 3.2 as an equivalent circuit like the one in Figure 3.5, where the inductance L is generated due to the windings of the coil into the rotor, and R is the windings resistance, we can write the Kirchhoff's Voltage Law on its therminals as:

$$v(t) = Ri(t) + L\frac{\mathrm{d}i(t)}{\mathrm{d}t} + bemf(t) \tag{3.13}$$

which at steady state, when the current is constant, becomes:

$$V = RI + BEMF \tag{3.14}$$

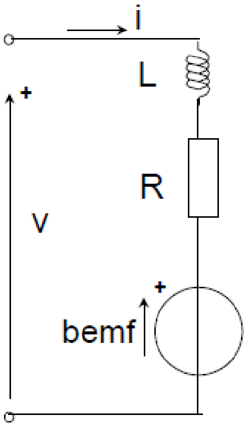


Figure 3.5: Equivalent circuit representation of a motor winding.

Based on the equivalent circuit of the windings (Figure 3.5), we can write down that the net power input into the circuit, $VI - RI^2$, is equal to the electrical power absorbed by the BEMF:

$$P_{electrical} = BEMF \cdot I = K_E \omega I \tag{3.15}$$

and we can also see that the mechanical power can be written using the new constants:

$$P_{mechanical} = T \cdot \omega = K_T I \omega \tag{3.16}$$

These two constants, K_E and K_T, synthesize the most important parameters used in the electromechanical power conversion in an electric motor. If we impose a voltage difference between the terminals of the loop of Figure 3.5, we create a current I that depends on the loop impedance. From Equation 3.11, we can see that such current will generate a proportional torque T. In the same way, from Equation 3.12, we obtain that the BEMF generated by the movement of the coil will define a proportional angular speed ω. If we rearrange the terms from Equation 3.14 using Equations 3.11 and 3.12, we can see the interrelation between the rotating speed of the motor ω and the torque T being generated:

$$\omega = \frac{V}{K_E} - \frac{R}{K_T K_E} T \tag{3.17}$$

which defines, by imposing a voltage difference between the terminals of the loop, the torque T vs angular speed ω curve that characterizes the mechanical behavior of an electrical motor.

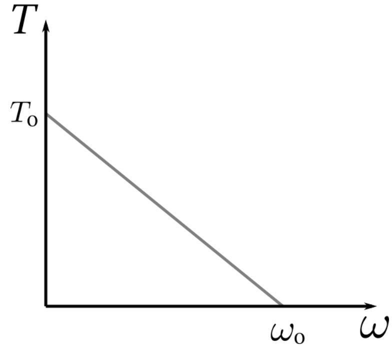


Figure 3.6: Basic representation of the Torque vs Angular Speed characteristic curve of an electric motor

If the speed of the motor ω is equal to zero, then we can rearrange the terms and see that the torque T is defined by $K_T V/R$, which is the maximum torque that the electric motor can provide by a given voltage, also named stall torque, T_0. We can also define a no-load speed ω_0, for the case when there is no load influencing over the movement of the rotor, as V/K_E.

The relationship between torque and angular speed stablished in Equation 3.17 will help us select the voltage that we need to applpy into the motor terminals in order to control the dynamics of the motor.

It is important to mention that these equations also work if there is an imposed angular speed ω. If we impose an angular speed to the rotor, the motor works as a voltage generator:

$$V = K_E \omega + R \frac{T}{K_T} \tag{3.18}$$

therefore, if we connect the motor to a load, it generates a current proportional to the speed and to the torque driving the rotor. This behavior is used in the so-called regenerative braking, since we can obtain energy when we want to stop a motor which is moving due to an inertial torque.

3.2 PMAC MOTORS

The Permanent Magnet Alternating Current (PMAC) motor is a type of electrical motor that doesn't need the mechanical commutators mentioned in 3.1.1 to be driven as in the case of the DC motor, but its windings need to be energized in a specific sequence to commutate and function correctly by means of an inverter. Since it doesn't need the mechanical commutators, it also doesn't need the brushes that energize the windings, so it can be said that it's a brushless motor. As seen in Figure 3.7, there are permanent magnets attached to its rotor. The stator consists of stacked steel laminations on which the coils are winded in a three-phase configuration.

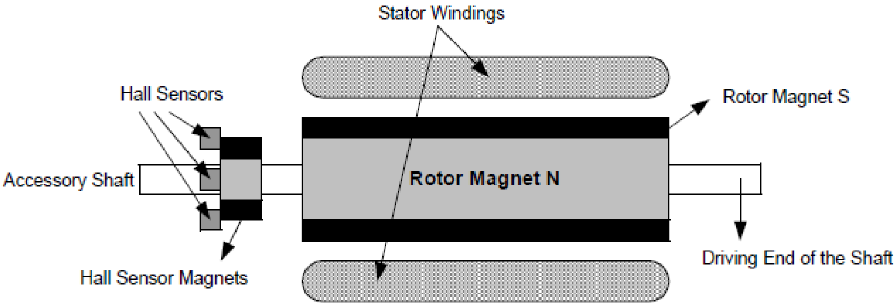


Figure 3.7: Brushless motor transverse section (Yedamale, 2003).

These three phases are fed alternatively in such a way that the magnetic field, generated by the relative currents passing through the coils, should always be perpendicular to the magnetic field generated by the rotor's permanent magnets, which is why the motor is defined as synchronous.

To maintain the synchronization, it's necessary to commute, by means of an inverter, the currents in the windings of the stator, taking as a reference the angular position of the rotor, which must be obtained by a sensor.

The number of commutations needed to generate one revolution of the rotor is determined by the number of magnetic Pole Pairs (PP) of the permanent magnets attached to the rotor. One PP is defined by 2 magnets attached to the rotor, one with the north pole facing out in one side of the rotor and one with the south pole facing out in the opposite side of the rotor. Normally, the PMAC motors have many PP (four or more) in order to have a lower torque ripple since the alignment of the magnetic fields would be every $360/(\text{PP} \times \Phi_N)$ degrees of the rotor, where Φ_N is the number of phases in the motors (three in this case).

For the electromechanical conversion, the angular position of the rotor is substituted by an electrical angular position, which is controlled by the commutator and is defined by:

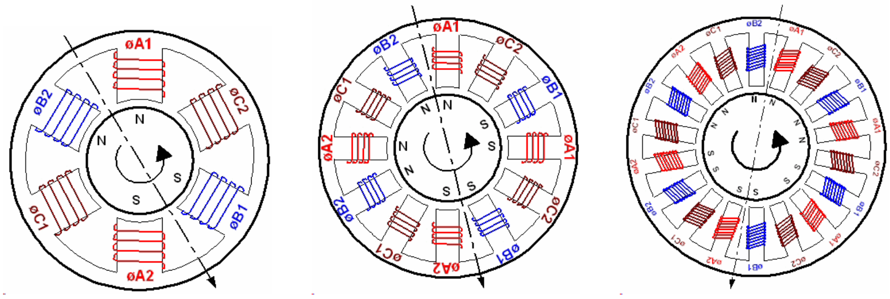
$$\theta_{electrical} = \mathrm{PP} \cdot \theta_{mechanical} \tag{3.19}$$

which also represents a relationship between the mechanical speed of the rotor and the electrical speed of commutation:

$$\omega_{electrical} = \mathrm{PP} \cdot \omega_{mechanical} \tag{3.20}$$

so, for example, if we have a motor with 6 PP and we want to drive it at $1kHz$, we must commutate the polarity of the inverter at

$$f_{commutation} = \mathrm{PP} \cdot f_{mechanical} = (6)(1kHz) = 6kHz \tag{3.21}$$



(a) 2 Pole Pairs **(b)** 4 Pole Pairs **(c)** 6 Pole Pairs

Figure 3.8: Different pole pair configurations (BASIL Networks, 2003)

The main characteristics that make the brushless motor a better option for some applications than the DC motor are the following:

- better weight-to-power ratio,
- a more linear acceleration,
- a low inertia,
- a higher reliability,
- smaller dimension,
- reduced need for maintenance,
- high rotation speed,
- ideal for working in hostile environments

but there are two disadvantages to this motor technologies:

- need of a rotation sensor,
- need of a complex logic to commutate the currents flowing through the coils

Both of these disadvantages are mainly reflected in a higher price respect to the DC motor.

It is possible to identify mainly two types of brushless motors: Brushless Direct Current (BLDC) and Permanent Magnet Synchronous Motor (PMSM).

3.2.1 *BLDC Motors*

The stator winding for each one of the three phases of the BLDC motor consists of a uniform distribution of turns over $N = \text{PP}$ sectors of a width equal to 60°. The magnets attached to the stator cover an arc of 180 electrical degrees and, at any instant, each magnet interacts for 120 electrical degrees with a winding carrying current.

Due to this "discrete" interaction every 120°, the three-phase commutation between the currents of the stator should happen when the edge of the magnet attached to the rotor reaches the boundary between windings every 60°, therefore, a trapezoidal BEMF is generated in its coils.

The boundary of the rotor magnets with respect to the windings position is detected by three sensors, one every 120°, which send a signal to the driving circuit in order to change the polarity of the coils depending on such signal value and on the desired direction. The sensors to detect the angular position will be explained in 3.3 and the driving sequences will be explained in 3.4.

3.2.2 *PMSM Motors*

The stator of the PMSM motor is fitted with three-phase windings with $N = \text{PP}$ turns of each phase distributed sinusoidally around the periphery. If the stator windings are feed by sinusoidal currents, there is a linear current density around the stator periphery and a sinusoidal BEMF is generated in its coils.

Since the sinusoidal feeding of the current into the coils depends continuously on the magnetic field of the rotor permanent magnets, it is necessary to use an angular position sensor attached to the rotor. The sensors to detect the angular position will be explained in 3.3 and the driving algorithm will be explained in 3.4.

3.3 ELECTRIC DRIVES

> An electric drive can be defined as an electromechanical device that converts electrical energy into mechanical energy to impart motion to different machines and mechanisms for various kinds of process control (Ghioni, 2016).

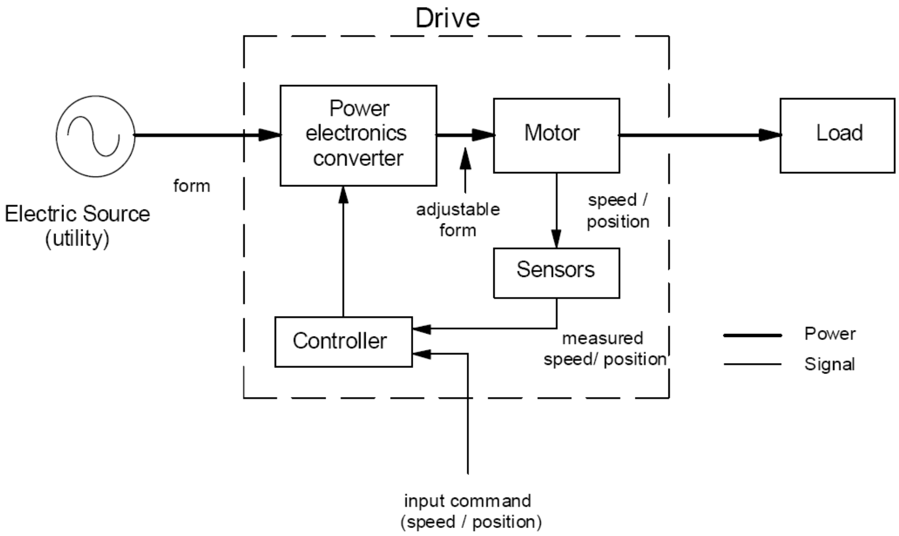


Figure 3.9: Electric Drive Scheme

The brushless motor drive, therefore, consists not only on the electric motor and the inverter, but it also includes the position, speed and current feedback systems, these being external sensors, embedded circuits or just algorithms, the vector, current, speed and position controllers and the DC power supply which can be a battery for mobile applications or a rectified power source for industrial applications.

3.3.1 *Inverter*

The commutation of the polarities in the windings of the PMAC motors is done by means of a three-phase inverter. The inverter avoids the need of the mechanical commutation used in the DC motors, which creates sparks between the commutator and the brushes due to the discharge of the electromagnetic energy stored in the windings of the rotor.

This three-phase inverter configuration consists on 3 branches with 2 switches each one. The switch connected to V_{DC} is called high-side switch, while the switch connected to $0V$ is called low-side switch. The middle point between the two switches in each branch is connected to each one of the phases of the motor.

To feed the coils of the motor, we need to activate the switches in such a way that the current comes into one of the phases an comes out through another phase. For example, if we want to feed a current

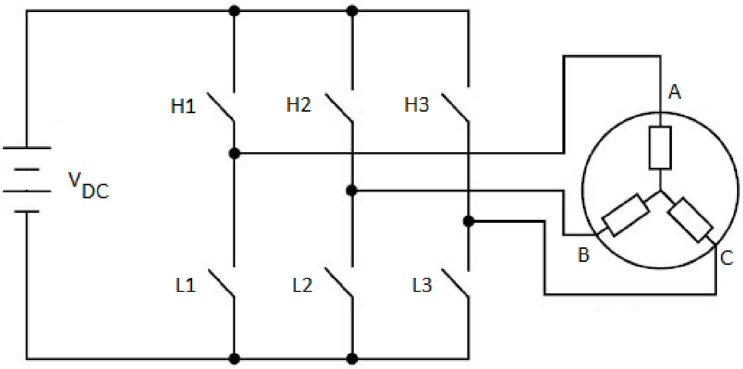


Figure 3.10: Schematic representation of a three-phase inverter

through phase A and we need it to come out though phase B to generate a magnetic field in a certain angle, we would enable switches $H1$ and $L2$, having a phase-to-phase voltage in the A and B coils equal to V_{DC}. The correct feeding sequence should be synchronous with the rotor position to generate an angular rotation of the rotor with the lowest possible torque ripple.

The switches in the three-phase inverter are implemented by transistors as seen in Figure 3.11. The transistors are chosen accordingly to the power requirements of the application on which the inverter is being used. If the DC voltage is lower than $1000V$ and the current is lower than $100A$, it is recommended to use the Power Metal–Oxide Semiconductor Field-Effect Transistor (MOSFET) technology. If the power consumption of the motor is larger, different technologies should be considered, like the Insulated-Gate Bipolar Transistor (IGBT) (Ghioni, 2016).

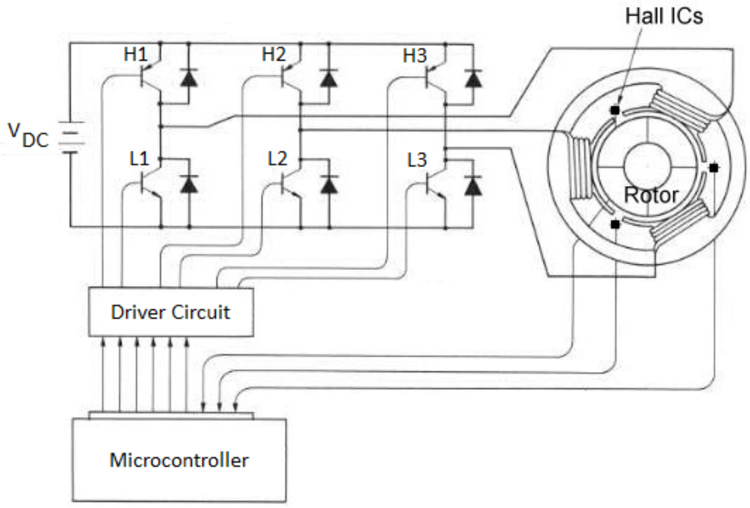


Figure 3.11: Schematic representation of a three-phase inverter built with transistors

The use of recirculation diodes is necessary to avoid damage in the transistors due to the overvoltage generated by the current transients in the windings LdI/dt that takes place between the switches of a same branch of the inverter while switching from one state to another. For example, when a transistor is suddenly turned off, the current flowing through the coils doesn't instantly disappear, but instead it recirculates through the diodes until it vanishes.

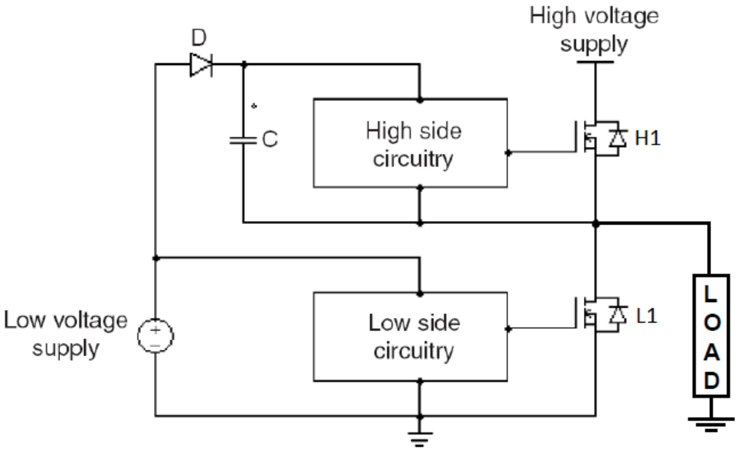


Figure 3.12: Power MOSFET devices require a dedicated circuit to provide enough voltage and current to drive their gates

Since Power MOSFETs have large parasitic capacitances, they can't be driven by typical CMOS (0 to $3V$) or TTL (0 to $5V$) logic signals, which typically have a low current driving capability. Instead, to drive a power MOSFET is necessary to use a more complex driving circuit to rapidly charge the capacitance of the gate and reach a value of V_{GS} large enough to switch the device. Some integrated circuits provide solutions to solve this problem.

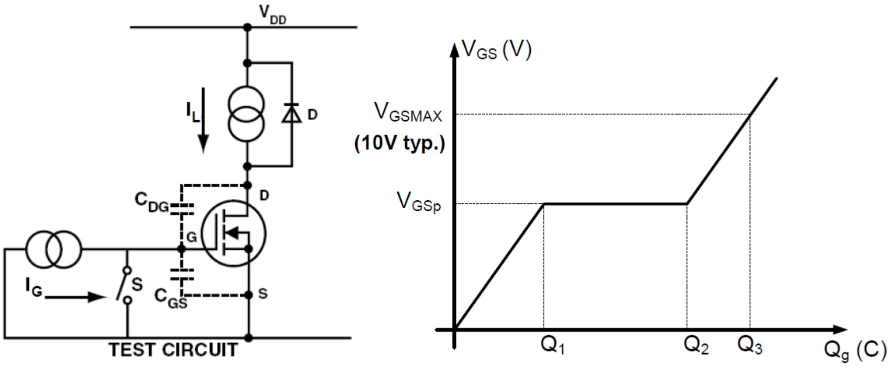


Figure 3.13: Schematic representation of the components influencing on the driving of a Power MOSFET

3.3.2 *Angular Position Sensors*

It is strictly necessary to obtain the angular position of the rotor in order to energize the coils of a PMAC motor in a synchronous way. The angular position detection can be made in different ways, but the solutions are mainly separated into two groups: sensored and sensorless.

The sensored angular detection is made by an external device, which creates a signal that allows us to know the angular position of the rotor. Typically two different approaches are used for the sensored angular detection. The first approach, and the simplest one, is the use of Hall effect sensors.

Hall effect sensors are transducers that variate an output voltage signal in response to a magnetic field acting over them, and depending on their configuration the output can be digital (high or low signals) or analog (the output voltage variates proportionally to the detected magnetic field strength). In the case of the brushless motors, since the rotor has permanent magnets attached to it, three Hall effect sensors are mounted inside the motor frame in order to read the magnetic field of those permanent magnets and create a digital output that helps us determine the angular position of the rotor.

The second approach to the sensored angular detection is the use of absolute rotary encoders. These encoders are attached to the shaft of the motor and, therefore, provide the angle of the rotor continuously. There are different technologies used for this approach. The simplest one is the resistive encoder, which is practically a potentiometer attached to the rotor's shaft, so it is prone to mechanical disturbances and noise, but is very cheap. The next approach is the optical encoder, which identifies the absolute angular position by means of attaching a disk with holes or with a pattern and optical sensors, like infrared LEDs and infrared detectors. The optical encoder is a very precise approach, but is very prone to mechanical disturbances, so it is mostly used in applications where the conditions to the system don't represent a problem to the encoder. The approach that is being used in this project to detect the absolute angular position is the magnetic encoder, which also uses Hall effect sensors for the absolute angular position determination. The difference between these two approaches is that, in this case, the Hall effect sensors of this encoder provide an analog signal proportional to the angular position of the magnetic field of an external permanent magnet, which is attached to the shaft. The absolute angular position detection is used mainly in the PMSM technology, since it is necessary to know the angular position of the rotor continuously to create a sinusoidal signal, to drive the sinusoidally distributed coil windings, synchronously to the magnetic field of the permanent magnets attached to the rotor.

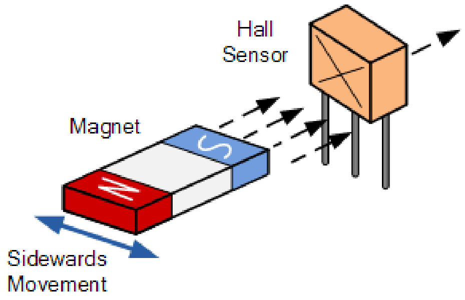


Figure 3.14: Hall Effect Sensor

One important thing that has to be considered when using absolute rotary encoders is that, even if these encoders provide the absolute angle of the rotor, they don't necessarily provide the angular position of the magnetic field of the rotor, introducing an angular slip $\Delta\theta$ which is equal to the difference between the value of the absolute rotary encoder angle and the angular position of the magnetic field.

The sensorless approach to obtain the angular position of the rotor consists in reading the BEMF on an undriven motor terminal during one of the drive phases. This is done by means of connecting a voltage divider to the middle point of each inverter branch, in order to read the analog voltage signal proportional to the BEMF and calculate the angular position of the rotor. The sensorless approach reduces the size and the cost of the motor drive, and it's also useful in applications where the rotor runs immersed in fluid. Even if this method is more complex than just reading the angular position, the main disadvantage is that the motor must be rotating at a minimum rate to generate a BEMF that can be detected by the controller, therefore, this approach is not convenient in applications where a motor must run at low speeds.

3.3.3 *Current Sensors*

Knowing the amount of current that is circulating through the motor windings can help us apply different control strategies to control the torque as defined in Equation 3.11. Also, reading the current can help us detect overcurrent conditions.

To read the current we must install an external sensor. The simplest approach to measure current, which is also the approach used in this project, is the use of a shunt resistor. The shunt resistor is an electrical resistor with a low (but well controlled) resistance value. This shunt resistor is placed in the path of the current flow of the motor's coils, in such a way that, when the current flows through the resistor, we have a proportional voltage drop across it:

$$I_{phase-to-phase} = \frac{V_{shunt}}{R_{shunt}} \tag{3.22}$$

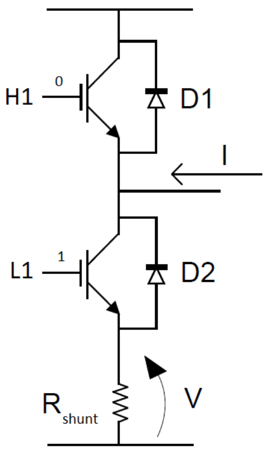


Figure 3.15: Measurement of the current flowing through a load using a shunt resistance

Since we need to take care of the power dissipated in the shunt resistor $P_{shunt} = I^2R$, small resistance values R with large power dissipation capability are selected, leading to the problem that this generates a very small voltage drop and that is prone to noise disturbances. For example, if we have a current of $100mA$ flowing through a shunt resistance of 0.001Ω, we would get a voltage drop across the shunt resistance equal to $0.1mV$, which is a very small value to be read by a typical Analog-to-Digital Converter (ADC). To solve this problem, we need to include an amplifier that allows us to read such a small voltage with a normal ADC, therefore, we can calculate the equivalent current as:

$$I_{phase-to-phase} = \frac{V_{amplifier}}{R_{shunt}G} \tag{3.23}$$

where G is the gain of the voltage amplifier.

3.3.4 *Controller*

The last piece on the motor drive is the controller, which executes all the logic behind the commutation of the inverter. Previously, motor controllers were made by analog electronic circuits. Nowadays, these controllers are implemented using digital circuits that can be programmed, like microcontrollers.

The microcontroller has the task to run embedded software that contains the logic that generates a driving signal for the inverter which will depend in the desired dynamic conditions and the actual conditions of the motor, which are sensed and stored by the microcontroller.

The microcontroller handles also the interface between other devices, allowing us to modify control parameters, like the desired speed or torque, and to read different physical variables from the motor drive that are detected by the system.

3.4 PMAC MOTORS DRIVING METHODS

Due to the differences between BLDC and PMSM motors, there are two methods to apply a voltage into the coils, depending on the generated BEMF. To obtain the best torque using a simple driving method in BLDC motors, we can apply the Trapezoidal Driving method, which is the driving method for which BLDC motors are designed. This method can also be used in the PMSM motors, but it won't allow us to obtain the best torque available for this motor technology. To obtain a better torque in both motors, we can apply the Sinusoidal Driving method, which allows to have a better current density, mainly in the PMSM motor, but it also implies a more complex and expensive hardware architecture and the implementation of a more complex algorithm.

3.4.1 *Trapezoidal Drive*

The trapezoidal drive is intended to create a trapezoidal BEMF by energizing positively one of the windings, connecting it to V_{DC} (current coming in), and negatively a second winding, connecting it to $0V$ (current going out), while the third winding remains floating (no current flowing through it), using the three-phase inverter explained in 3.3.1. This energizing will generate a magnetic field in a certain angle that will interact with the magnetic field of the rotor, producing a torque between the rotor and the stator and, therefore, a rotation of the rotor if the load is smaller than the torque generated. The sequence of commutation that is followed to create the trapezoidal BEMF is called "six-step" commutation.

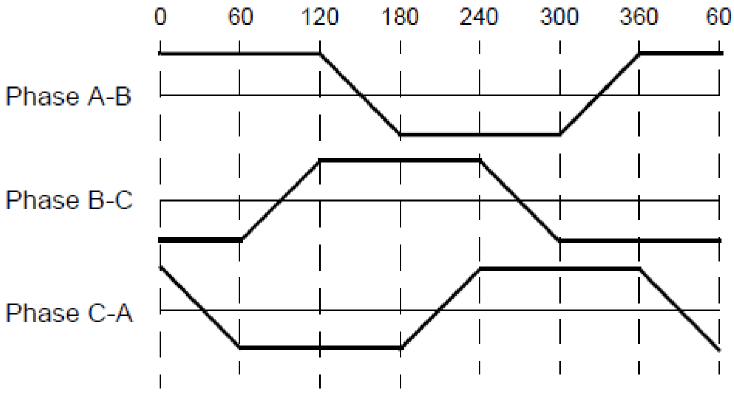


Figure 3.16: Trapezoidal BEMF

The six-step commutation is produced by following the sequence described in Table 3.1, which depends on the rotor position, obtained

Table 3.1: Six-Step Commutating Sequence

Clockwise Direction								
Step #	Hall Sensor Input			Active Gates		Phase Voltage		
	A	B	C	High Side	Low Side	A	B	C
1	1	0	0	$H2$	$L1$	$0V$	V_{DC}	-
2	1	0	1	$H3$	$L1$	$0V$	-	V_{DC}
3	0	0	1	$H3$	$L2$	-	$0V$	V_{DC}
4	0	1	1	$H1$	$L2$	V_{DC}	$0V$	-
5	0	1	0	$H1$	$L3$	V_{DC}	-	$0V$
6	1	1	0	$H2$	$L3$	-	V_{DC}	$0V$

Counter-Clockwise Direction								
Sequence #	Hall Sensor Input			Active Gates		Phase Voltage		
	A	B	C	High Side	Low Side	A	B	C
1	1	0	0	$H1$	$L2$	V_{DC}	$0V$	-
2	1	0	1	$H1$	$L3$	V_{DC}	-	$0V$
3	0	0	1	$H2$	$L3$	-	V_{DC}	$0V$
4	0	1	1	$H2$	$L1$	$0V$	V_{DC}	-
5	0	1	0	$H3$	$L1$	$0V$	-	V_{DC}
6	1	1	0	$H3$	$L2$	-	$0V$	V_{DC}

by the Hall effect sensors, and on the expected direction of rotation. This sequence generates a current circulation like the one explained in Figure 3.17, which produces a BEMF and a torque like the one shown in Figure 3.18. We can see that the torque in Figure 3.18 presents a ripple due to the non-continuous shape of the current in the coils, generated by the rectangular commutation of the six-step sequence. This current shape is defined by the electrical model of the coils of an electric motor, which follows a model similar to the one described in Figure 3.5 and Equation 3.13.

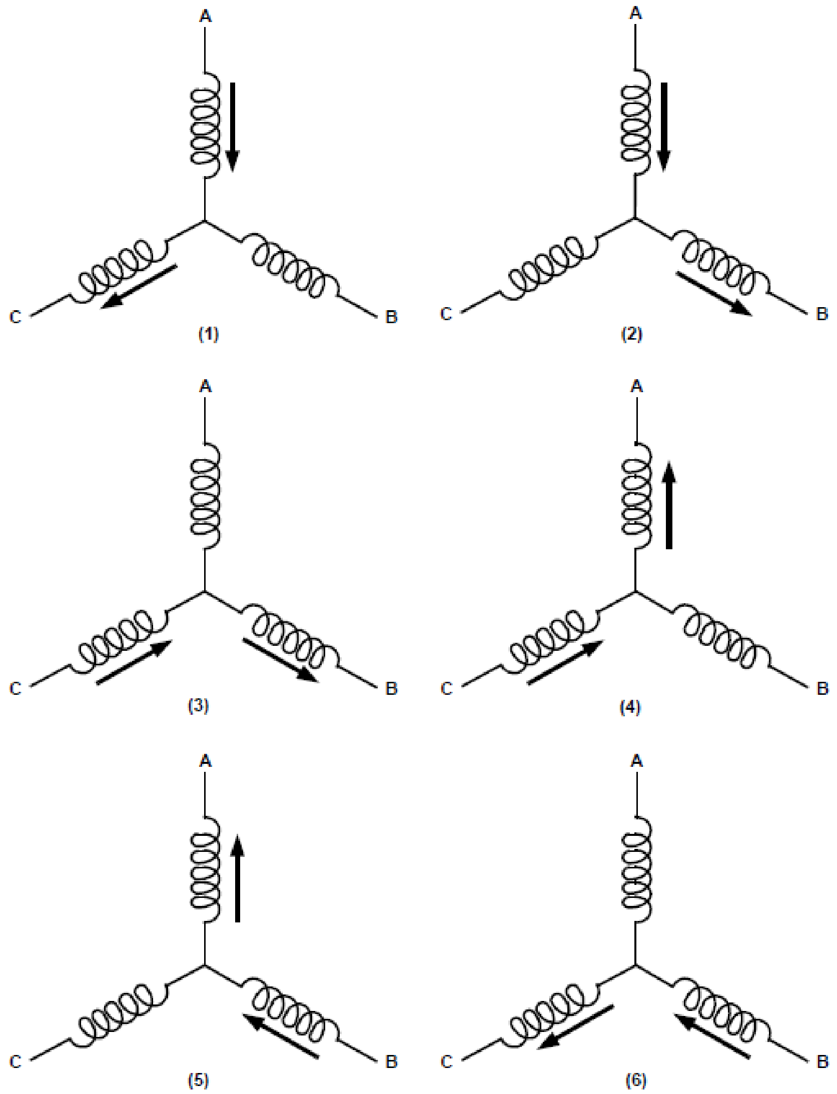


Figure 3.17: Current flowing through the coils of the PMAC motor in the 6-step commutation sequence (Yedamale, 2003)

Due to the synchronous configuration of the PMAC motors, the six-step commutation frequency is defined by the rotation speed and not vice versa, unlike in induction motors. Therefore, to reduce the rotation speed, we need to reduce the current flowing through the wind-

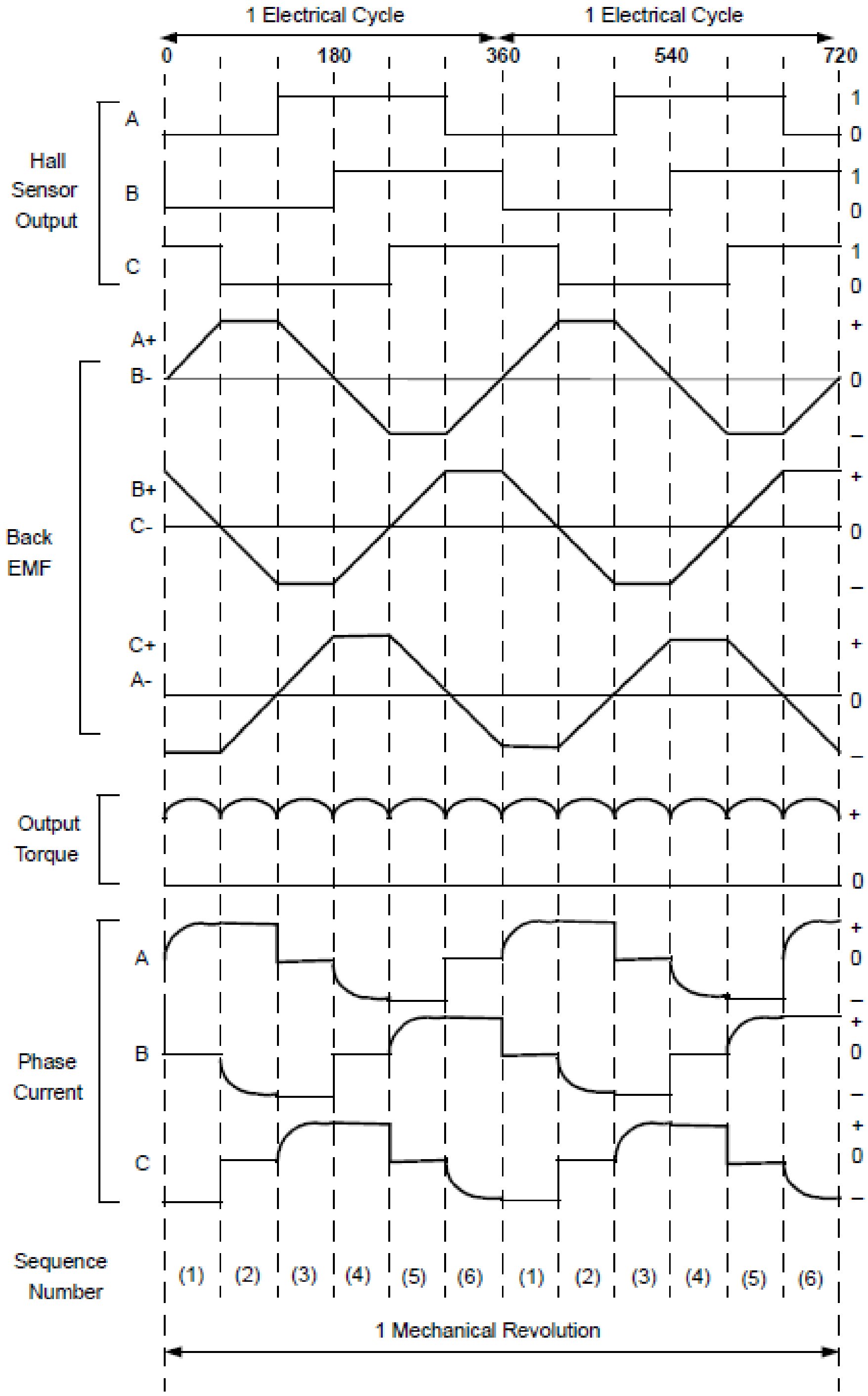


Figure 3.18: Six-Step commutation elements: Hall effect sensor signal, BEMF, current and output torque (Yedamale, 2003)

ings by reducing the voltage applied into the windings. To achieve this, we must follow the six-step sequence applying a Pulse Width Modulation (PWM) signal into the gates of the transistors with a frequency higher than the inverse of the time constant $\tau = L/R$ of the motor windings, which is also much higher than the six-step commutating frequency. By doing this we apply an average voltage $D \cdot V_{DC}$ into the windings, where D is the duty cycle of the PWM signal.

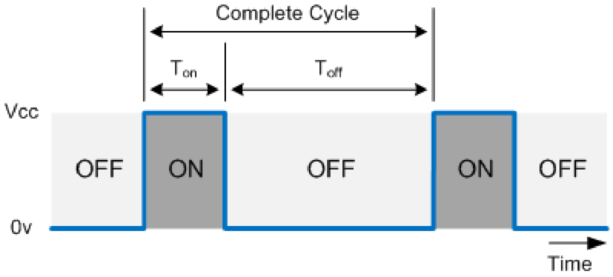


Figure 3.19: Pulse width modulation signal components

The simplest way to do this is by applying the PWM signal to either the high-side transistor or to the low-side transistor. Due to the recirculation diodes, the voltage in the windings energized would commutate from V_{DC} to $0V$, as in a step-down voltage converter. A more complete approach to reduce the speed would be to apply a complementary PWM signal to the low-side transistor on the same branch of the high-side transistor indicated on the sequence, while the low-side transistor indicated on the sequence stays enabled. This approach would allow us to brake the motor, enabling also the regenerative braking, since, by applying a complementary signal to the high-side transistor in the low-side transistor, we are allowing the flow of current not only from the power supply to the motor, but also from the motor to the power supply.

3.4.2 *Sinusoidal Drive*

The sinusoidal feeding for the PMAC motors is similar to feeding a three-phase induction motor, to which we need to provide a three-phase voltage in order to energize the windings and create a three-phase current. The difference in the power supply with the induction motor would be that the PMAC motor three-phase power supply needs to be synchronous with the rotor position. Therefore, after determining the absolute angular position of the rotor with any of the methods described in 3.3.2, we generate independent voltages for each phase using the three-phase inverter.

To generate these three independent voltage signals using the inverter, we can consider each one of the branches of the inverter as a voltage generator for each one of the phases. We can modulate a sinusoidal signal by using a PWM signal with a duty cycle D proportional

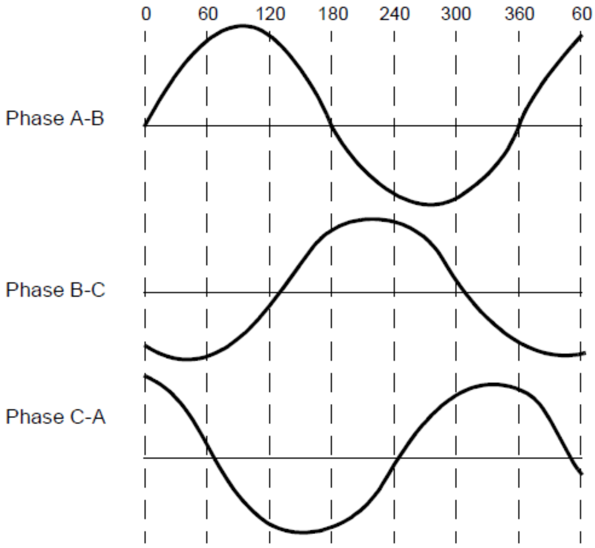


Figure 3.20: Sinusoidal BEMF

to the amplitude of the sinusoidal signal, which can go from $0V$ up to V_{DC}:

$$D_{PWM} = \frac{v_a(\theta)}{V_{DC}} + 0.5 \tag{3.24}$$

By commutating the high-side transistor of the bridge with the PWM signal, and the low-side transistor with the complementary signal, in the three branches of the inverter, we generate a three-phase signal as the one seen in Figure 3.20.

It is important to recognize that if the voltage to be applied is larger than $V_{DC}/2$, the duty cycle D becomes larger than 1. Since a duty cycle larger than 1 doesn't make sense, the duty cycle will remain as 1 until the voltage applied becomes lower than $V_{DC}/2$, generating a trapezoidal waveform, which would have the same effect over the motor of the trapezoidal drive, losing the advantages of the sinusoidal drive.

This driving method is normally not used by itself, but it is applied in the Field Oriented Control (FOC) method, which will be explained at the end of this chapter.

3.5 electric motor control methods

As we have seen previously, there is a strong dependence between the rotating speed of the electric motor and the torque generated as a function of the voltage applied to its phases. Due to this, and to the fact that there are many other external factors that might affect our system parameters, in order to obtain a desired speed or a desired torque, we can't just apply a previously calculated voltage that would give us those speed or torque values, instead, we need to apply a control loop.

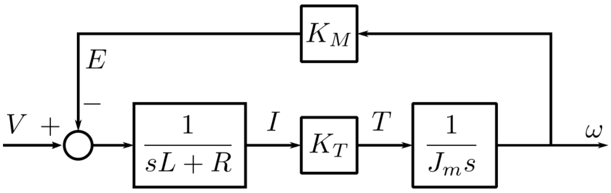


Figure 3.21: Coupling between mechanical and electrical dynamics on an electric motor

3.5.1 *Speed Control*

The speed control loop of an electric motor is achieved by means of applying a driving voltage into the motor coils, proportional to the error signal between a reference speed and the measured speed. This driving voltage is obtained by means of a Proportional-Integral (PI) controller. The PI controller creates a driving signal that is calculated by adding two signals. The first signal is the proportional error, which is equal to the error ε multiplied by a proportionality constant K_p. The second signal is the integral error, which is equal to the error ε multiplied by an integral constant K_i integrated over the time that the algorithm has been running (Figure 3.22).

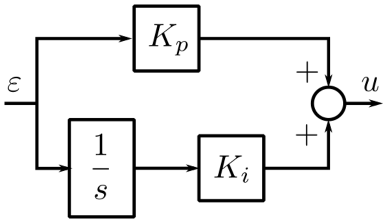


Figure 3.22: Blocks Diagram of a Proportional-Integral Controller

To apply the speed loop into the PMAC motors, we need to obtain the speed of rotation of the rotor. This can be easily calculated by calculating the rate of change of the angular position of the rotor. After this, we calculate the error signal ε, pass it through the PI controller to generate a reference value that is applied to the inverter, which feeds the motor with the resulting voltage. As an example, we can see the speed control applied using the trapezoidal drive in Figure 3.23.

3.5.2 *Torque Control*

The torque control is based in Equation 3.11:

$$T = K_T I \tag{3.11}$$

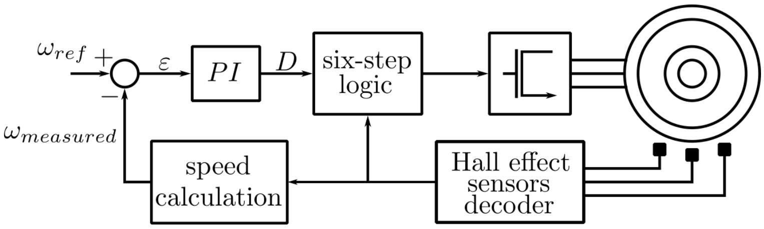


Figure 3.23: Speed Controller with Trapezoidal Drive

where we can see that the torque applied by the motor is equal to the current flowing through its windings multiplied by the torque constant K_T. Therefore, we need to read the current flowing through the motor windings to calculate the torque that the motor is applying, by multiplying the current $I_{measured}$ it by the torque constant K_T, and comparing it with the torque T_{ref} that we want the motor to apply. As we can see in Figure 3.24, the rest of the loop is similar to the speed loop.

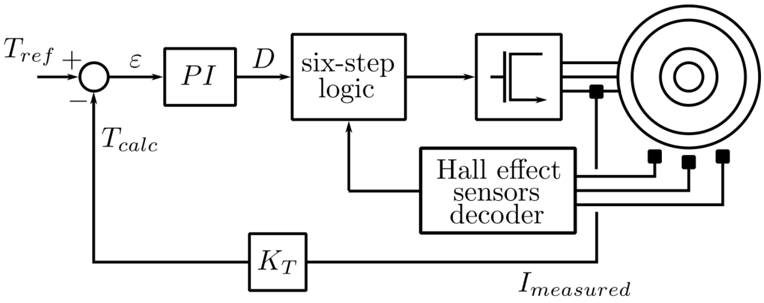


Figure 3.24: Torque Controller with Trapezoidal Drive

3.5.3 *Field Oriented Control*

The Field Oriented Control (FOC) or Vector Control, is technique that is applied with aims of controlling the current (and therefore the torque) of the PMAC motor in the same way that the current of a DC motor is controlled. This is done because the DC motor has the advantage that, in any case, the magnetic field of the rotor is always synchronized with the magnetic field of the permanent magnets attached to its stator due to the mechanical commutators, which makes it very easy to control. Instead, the PMAC motors must be synchronized by means of applying different driving algorithms depending on the rotor angular position, as mentioned before in this chapter.

To achieve a current control on the PMAC motors that is similar to the one on DC motors, we need to make mathematical transformations to the voltage and current values and controller variables of the system in order to control the motor using only two variables: a direct current I_d and a quadrature current I_q.

The direct current I_d is a vector that is facing always in the same direction of the north-pole of the magnetic field of the rotor, while the quadrature current vector I_q is displaced from the direct current vector by 90°. Due to this angular displacement, we can control the current and the direction of rotation by simply changing I_q, as in a DC motor we would only need to reduce the armature current in order to reduce the torque, which is the scope of the FOC, since the maximum torque generation of any electric motor is when the magnetic fields of the rotor and the stator are displaced between each other by 90°.

The control of the direct current vector I_d allows the implementation of "flux weakening", which is a technique that aims to push the angular speed of the electric motor beyond its rated speed by reducing the strength of the magnetic field B, in order to reduce the generated BEMF, which limits the current flow in the windings. If a larger current circulates through the windings, a larger torque is developed, reaching a higher speed until the BEMF increases, preventing the current to keep increasing. By this point the electric motor is running at a higher speed but at a lower torque.

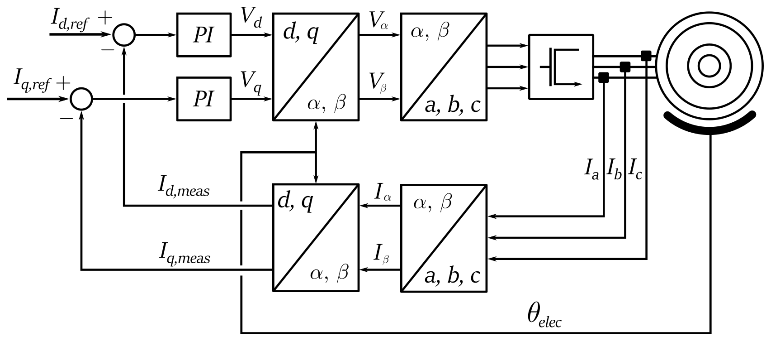


Figure 3.25: Field Oriented Control Blocks Diagram

To achieve the FOC we start by transforming the current signals that we acquire from the current sensors using the Clarke Transformation, which converts the three current phasors of the motor into two equivalent currents located on the α and β axis:

$$\begin{bmatrix} i_\alpha \\ i_\beta \end{bmatrix} = \frac{2}{3} \begin{bmatrix} 1 & -\frac{1}{2} & -\frac{1}{2} \\ 0 & \frac{\sqrt{3}}{2} & -\frac{\sqrt{3}}{2} \end{bmatrix} \begin{bmatrix} i_a \\ i_b \\ i_c \end{bmatrix}$$

After this, in order to obtain I_d and I_q, which don't change in function of the angular position of the rotor θ, we apply the Park Transformation, which converts the currents i_α and i_β into two currents that are mounted into an axis that rotates with the same angle as the rotor:

$$\begin{bmatrix} I_d \\ I_q \end{bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta \\ -sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} i_\alpha \\ i_\beta \end{bmatrix}$$

We can compare these two currents, I_q and I_d, with the reference currents $I_{q,ref}$ and $I_{d,ref}$, generating error signals ε_q and ε_d that pass through PI controllers in order to generate two voltage values V_q and V_d, which can be transformed into sinusoidal voltages v_a, v_b and v_c.

To obtain these voltage values, we apply the Inverse Park transform to the quadrature and direct voltages V_q and V_d:

$$\begin{bmatrix} v_\alpha \\ v_\beta \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} V_d \\ V_q \end{bmatrix}$$

and then, the Inverse Clarke Transform:

$$\begin{bmatrix} v_a \\ v_b \\ v_c \end{bmatrix} = \frac{3}{2} \begin{bmatrix} \frac{2}{3} & 0 \\ -\frac{1}{3} & \frac{\sqrt{3}}{3} \\ -\frac{1}{3} & -\frac{\sqrt{3}}{3} \end{bmatrix} \begin{bmatrix} v_\alpha \\ v_\beta \end{bmatrix}$$

Finally, we apply these v_a, v_b and v_c values to the inverter applying the sinusoidal drive method explained in 3.4.2.

3.5.4 *Speed Control with Field Oriented Control*

Since by applying the FOC method to PMAC motors we can achieve a good current control, we can therefore control the torque produced by the motor with the same performance. Due to this, the best option to control the speed of a PMSM motor is to close a speed loop around the FOC loop as the one shown in Figure 3.26.

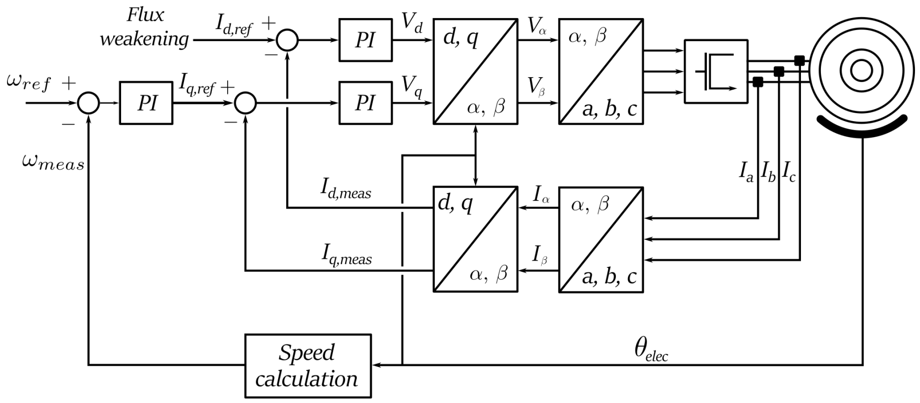


Figure 3.26: Field Oriented Control Applied in Speed Control

4 CASE OF STUDY

This thesis was proposed as part of a robotic agriculture project which consists in the development of a mobile manipulator for agricultural applications called ROBI'. In this chapter we will explain the main characteristics of the project, focusing in the electronic architecture and in the systems that should be controlled by the motor driver developed in this thesis. The information for this chapter was obtained from Bascetta, Baur, and Gruosso "ROBI': A Prototype Mobile Manipulator for Agricultural Applications" 2017 and from Bascetta, Baur, and Gruosso "Electrical Unmanned Vehicle Architecture for precision farming applications" 2017.

4.1 ROBI'

ROBI' is a mobile manipulator for agricultural applications. The development of this project has the objective of becoming a platform for research, development and testing of different perception and control algorithms that could help in agriculture and in similar applications. The robot was designed following low-cost, low-weight, simplicity, flexibility and modularity requirements, in order to allow an easy implementation of the work developed as part of this project.

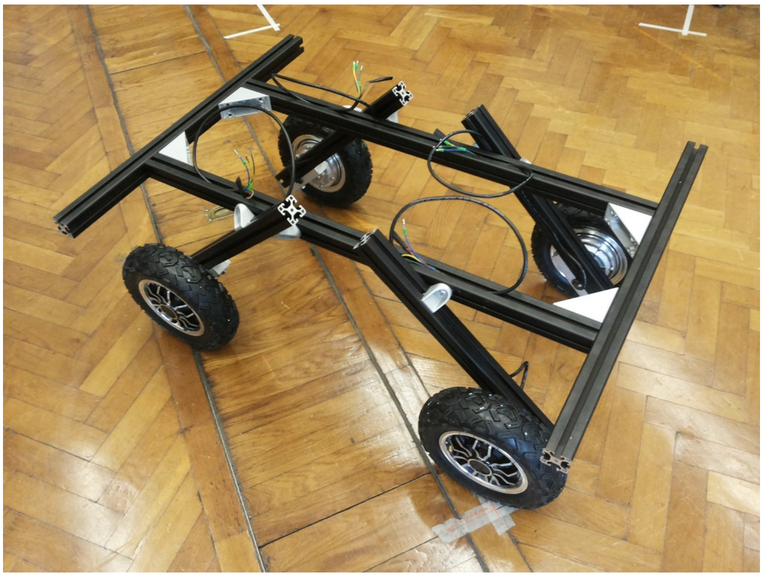


Figure 4.1: ROBI' Basic Configuration

Table 4.1: HUB10GL In-Wheel Motor Main Characteristics

Motor mass	3.5 kg
Motor and tire mass	5.7 kg
Tire diameter	254 mm
Power	500 W
Voltage	36 V
Maximum speed	1350 RPM
Phase-to-phase resistance	160 mΩ
Phase-to-phase inductance	0.76 mH
Pole Pairs	10
K_E	0.36 V/(rad/s)
K_T	0.4506 Nm/A

4.2 IN-WHEEL MOTORS

One of the most important components of ROBI' for this thesis is the in-wheel motor system used as motor drive. As mentioned in the introduction, in-wheel motors have the advantage that there is no need for an external power transmission mechanism, due to the fact that the motor is embedded into the wheel. This also reduces the space needed for the motor drive system, as well as the weight of the robot, increasing the payload capacity of the electric motor.

It is also interesting to mention that this configuration of the drive system allows the control of each one of the wheels of the robot independently, which allows the implementation of different control methods for this robot, like skid steering, which relies on the slip effect of the wheels in the terrain and requires a good control of the torque and the speed of each one of the motors.

The in-wheel motor installed in the robot is a PMSM motor and its main characteristics are mentioned in Table 4.1.

In order to take advantage of the sinusoidal distribution of the windings of this motor, an absolute angular position sensor was installed in the drive system in order to be able to apply a FOC loop. The sensor was attached to the wheel by means of a toothed pulley system in order to keep it away from the ground, as seen in Figure 4.2.

The angular position sensor is an Orbis magnetic encoder, manufactured by RLS. This is a magnetic encoder which works by reading the orientation of the magnetic field of a permanent magnet attached to a shaft, as explained in 3.3.2. The encoder has a resolution of 14 bits with a sensitivity of $\pm 0.3°$and transmits the angular position information through a modified Serial Peripheral Interface (SPI) called

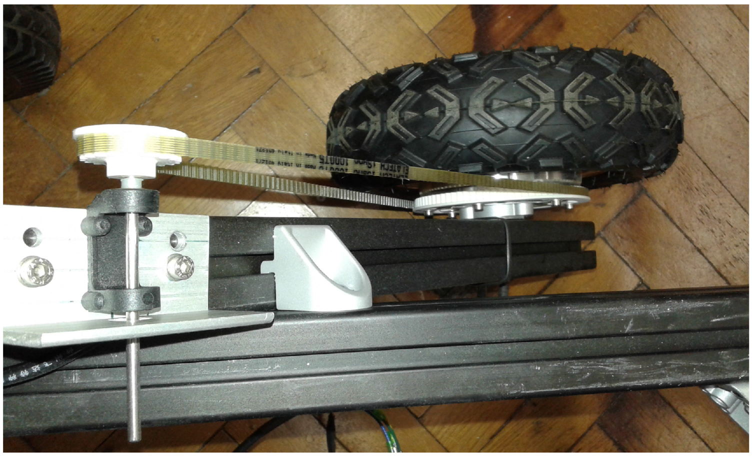


Figure 4.2: Encoder transmission configuration

Table 4.2: Battery Main Characteristics

Manufacturer	FIAM
Model	FGC22703
Type	Sealed lead acid
Voltage	12 V
Capacity	27 Ah

Synchronous Serial Interface (SSI), which sends the 24 bits in slave write-only mode through an $RS422$ transceiver.

In this case, the shaft of the sensor is not the shaft of the rotor, but it's the shaft of the pulley system. This transmission system has a 4.6 size ratio due to the space constraints of the system. This motor includes three Hall effect sensors, even if it's a PMSM motor, as mentioned in 3.3.2.

4.3 POWER ARCHITECTURE

The power supply consists on three $12V$ lead acid deep-cycle batteries connected in series to provide $36V$ to the different components of the robot.

4.4 ELECTRONIC ARCHITECTURE

It is important to understand the expected electronic architecture in order to consider the peripherals that the microcontroller has to include to communicate with the rest of the systems inside the robot.

The electronic architecture of ROBI' was designed to be similar to an automotive Electronic Control Unit (ECU) based system. In this architectures, all the ECUs interact with each other through a communication bus called Controller Area Network (CAN).

As it can be seen in Figure 4.3, the ECUs network consists on four units: Wheel Management Unit (WMU), Battery Management Unit (BMU), Vehicle Management Unit (VMU) and Arm Management Unit (AMU). The one that is being developed for this thesis is the WMU.

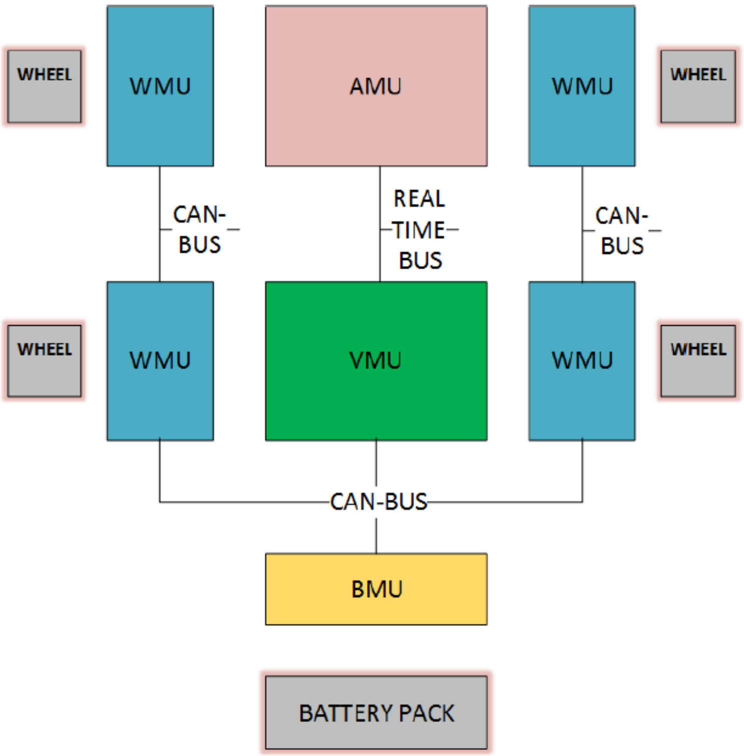


Figure 4.3: ECUs architecture: functional block diagram

5 PROJECT DEVELOPMENT

All the theory explained in Chapter 3 and the details of the case study presented in Chapter 4 lead us to the definition of the requirements for the implementation of the motor driver needed by the ROBI' project. This driver, defined as the WMU, should be able to apply both driving techniques, trapezoidal and sinusoidal, and to execute different motor control methods for the PMSM in-wheel motors. The driving and control methods should be selected by future developers, depending on the application and on the algorithm needed for the control of the overall displacement of ROBI'.

This chapter is divided into three sections: the first section contains a description of the hardware implementation of the driver; the second section explains the implementation of the driving techniques and control methods through embedded software; the last section describes the setup mounted to test the driving capabilities of the WMU developed.

5.1 HARDWARE

The WMU's electronic circuit is based on an open architecture proposed by Texas Instruments (TI) to be implemented together with the use of their Integrated Circuit (IC) MOSFET inverter driver DRV8302. This architecture was implemented in a project developed by the Swedish engineer Benjamin Vedder called VESC, which consists in the design of an Electronic Speed Controller (ESC), its Printed Circuit Board (PCB) layout design and the implementation of the source code used to drive a BLDC motor (Vedder, 2015). The hardware design of the VESC board was not modified for this thesis in order to identify, after the implementation and the testing of the motor control algorithms, what could be changed to obtain a better performance from the in-wheel motors mounted on ROBI'.

5.1.1 *VESC Board*

The VESC board was conceived and designed to be used as an ESC in electric skateboards. Due to this, it has a small form-factor and it can be used for different applications with similar power demand. The hardware can also be modified by changing specific components to drive motors with higher power demand or by modifying the PCB layout design following the schematic design. The original schematic files can be found attached in Appendix A for reference.

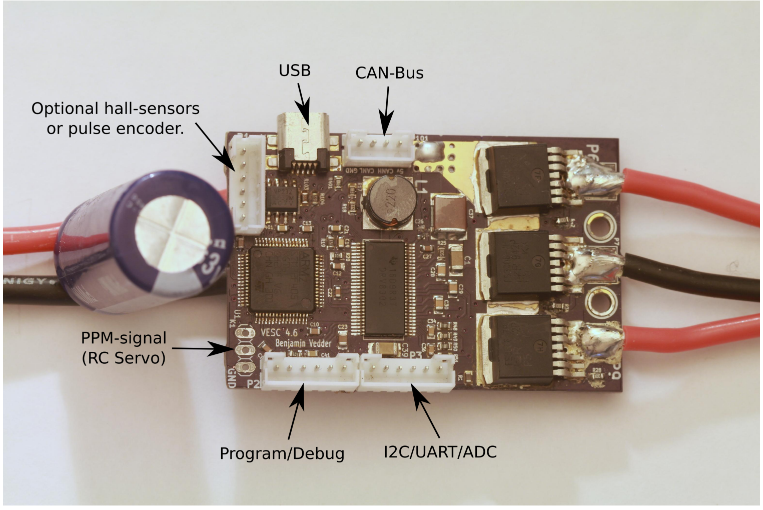


Figure 5.1: Front Side of the VESC Board. The three wires at soldered at the right edge of the board are connected to the three phases of the brushless motor (Vedder, 2015).

The PCB layout provided as part of the VESC project is a four-layer, $40mm \times 60mm$, FR-4 board, with pads for components on the both external layers. The use of both external layers to place components and the four layers to interconnect the circuits allows the board to keep a small form factor.

The different functional components of the board and their considerations regarding the implementation of the WMU will be explained in this section.

5.1.2 *Power Supply Input*

The power is supplied into the board by means of 2 wires soldered into large pads placed close to the three-phase inverter. V_SUPPLY and GND are the positive and negative nodes of the power supply source respectively. It is specified in the schematics that there should be a bulk capacitor connected in parallel to the power supply port.

5.1.2.1 *Power Supply Range*

The voltage range of the VESC board is defined by the MOSFET driver IC $DRV8302$, which specifies an operating supply voltage range from $8V$ to $60V$, but allows a maximum voltage supply up to $65V$ **??**. All the other components of the board, mainly capacitors and power MOSFETs, must be selected according to this voltage range.

The current range of the board is limited by the MOSFETs current limit and by the width of the traces in the PCB.

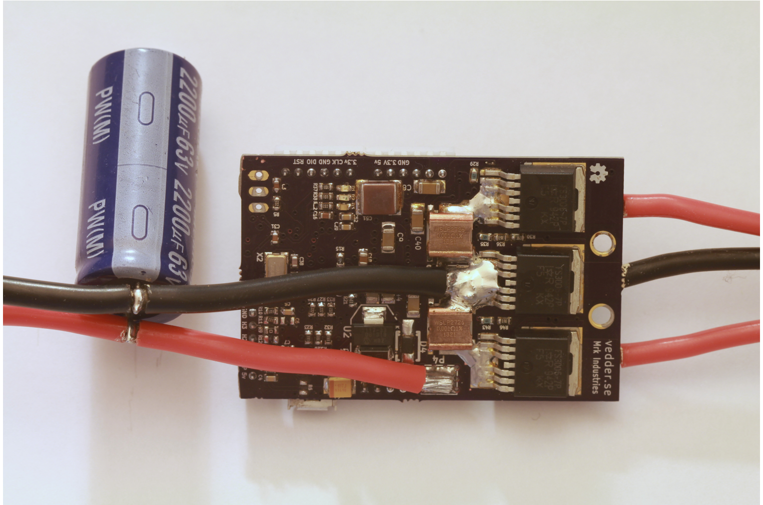


Figure 5.2: Back Side of the VESC board. The red and black wires are the power supply input V_SUPPLY and GND respectively.

Table 5.1: VESC Board Operating Ranges

Input voltage	8 to $60V$
Maximum output current	$50A$
Peak output current	$240A$

5.1.2.2 *Bulk Electrolytic Capacitor*

It is important to place a bulk capacitance with an appropriate capacitance value and voltage range in parallel to the input of the power supply of a motor drive system. The main objective of a bulk capacitance is to control the voltage deviation at the input of a system when the converter is responding to an output load transient, meaning that if there is a load increase and the power supply is not able to provide the current instantaneously for the motor, the current must be provided by the bulk capacitor. The higher the capacitance at the input of the system, the lower the deviation at the load.

The datasheet of a motor driver should normally provide a recommended value for the bulk capacitor, but it is necessary to calculate and test the value to determine if it's appropriate. The voltage rating of a bulk capacitor must be higher than the value of the power supply to provide a margin for cases in which the motor transfers energy to the supply, which would increase its voltage and would damage a capacitor with a smaller value than the resulting voltage. We can calculate the capacitance value by using the following formula (Arrigo, 2006):

$$C = \frac{1.21 \cdot I_{tr}^2 \cdot L}{\Delta V^2} \tag{5.1}$$

Since there is no filtering inductance in the system, we can consider a stray inductance of $50nH$ in the calculation. If we consider a transient current I_{tr} equal to the rated current of the in-wheel motor, $14A$ approximately, and a voltage ripple of $100mV$, we obtain a capacitance value of $1185.8\mu F$. Looking for available capacitor values, we increased the value of the capacitor to $2200\mu F$ and, considering the maximum input voltage value of the motors, the common practice is to double that value, so we defined a voltage range of $75V$, which is a typical value for a commercially available capacitor.

It is important to note that since the value of this capacitance is large, also its size will be large, therefore it's important to consider its dimensions to setup a mechanically stable position for such capacitance with aims of avoiding hardware malfunction.

5.1.2.3 *Power Supply Wiring*

The strategy of soldering the wires directly to the pads in the PCB helps reduce the size and cost of the board since it avoids the use of large power traces in that would be needed if a regular receptacle connector was to be used.

Anyway, the practice of soldering a wire directly to a PCB is discouraged mainly in circuits that will be mounted in a system designed to be in movement (i.e. automotive or aerospace applications) because the wire used in these applications is made of delicate copper strands which can break easily with movement of the board or of the wires and cause a loose contact or a short circuit, both of which might lead to catastrophic failures. Therefore, there is a latent risk in this board since it will be mounted in a system that will be in constant movement (RS Components Ltd., 2012). Since this board was designed this way to reduce space consumption, the trade-off between safety and size was balanced into the size constraints, therefore some cautions must be taken to avoid problems, mainly because these wires carry a considerable amount of current and they pass over many components of the board as seen in Figure 5.2.

Since the heat exchange calculation of the wire due to the voltage drop across it would require many assumptions, it's a normal practice to select the type of wire using tables that consider the temperature rise according to different currents and cores. In the case of the PMSM motor used in ROBI', the maximum rated current would be approximately $14A$, which would lead us to select a wire with a diameter between $2.1mm$ and $2.6mm$, based on Table 5.2. This selection would depend on the number of cores from which the wire is constructed: the bigger the amount of cores, the more flexible the wire is, but also the diameter becomes larger.

Table 5.2: American Wire Gauge current load ratings in function of the number of cores of the wire (The Engineering ToolBox, 2015)

AWG	D [mm]	Maximum Current Load Ratings [A]					
		1	2 - 3	4 - 6	7 - 24	25 - 42	> 43
16	1.3	15	10	8	7	6	5
14	1.6	24	15	12	10	9	7.5
12	2.1	34	20	16	14	12	10
10	2.6	52	30	24	21	18	15

Table 5.3: IRFS7530-7PPbF Absolute Maximum Rating

Continuous drain current	$240A$
Maximum power dissipation (25°)	$375W$
Linear derating factor	$2.5W/°C$
Drain-to-Source breakdown voltage	$60V$
Drain-to-Source on-resistance	$1.5m\Omega$

5.1.3 *Power MOSFET Three-Phase Inverter*

The three-phase inverter used in the VESC board is based on the power MOSFET IRFS7530-7PPbF, manufactured by the semiconductor company International Rectifier. The schematic design of the inverter can be seen in Figure A.2 in Appendix A. The PMAC motor is connected to the board by soldering its phase terminals directly to the node pads in the middle of each branch of the inverter, which is located at the bottom edge of the board, as seen in Figure 5.1.

This MOSFET is based on the HEXFET Power MOSFET technology, which is a Vertical Double-diffused MOS transistor (VDMOS) with hexagonal elementary cells in parallel which maximizes the W/L ratio of the transistor, providing a low on-resistance value $R_{DS(ON)}$ (International Rectifier, 2015).

It is important to note that the maximum power dissipation implies a junction temperature of $25°C$, which means that the heat of the MOSFET package should be dissipated by a heat sink, which is not implemented nor considered in the VESC board. If the heat is not dissipated from the transistors, the power dissipation capacity of the transistor decreases following the linear derating factor.

5.1.3.1 *Switching Times*

As mentioned in 3.3.1, power MOSFETs have large parasitic capacitances that need to be charged in order to allow the activation of the

Table 5.4: IRFS7530-7PPbF Dynamic Electrical Characteristics

Total gate charge	236 nC
Gate-to-Source charge	62 nC
Gate-to-Drain charge	73 nC
Turn-On delay time	24 ns
Rise time	102 ns
Turn-Off delay time	168 ns
Fall time	79 ns
Input capacitance	12.9 nF

transistor. This behavior also affects the switching time, which is the time that a transistor takes to turn on and turn off completely.

The switching time of a transistor is therefore defined by the capability of the driving circuit to provide current to charge the parasitic capacitances in the gate and to draw current from them.

To determine the switching time of a MOSFET is necessary to analyse the driving circuit and the parasitic capacitances of the transistor. The information regarding the switching time of the transistor is normally available in the device datasheet.

It is important to notice also the conditions on which the electrical characteristics of the device where obtained. In this case, it is stated in the datasheet that the transistor had a gate resistance R_G of 2.7Ω, a gate-source voltage V_{GS} of $10V$, a V_{DC} equal to $30V$ and the current flowing through the device was $100A$.

In the case of the $IRFS7530 - 7PPbF$, we can calculate that the minimum period of time to turn on and turn off the transistor is $373ns$, which is equal to the sum of the turn-on delay time, rise time, turn-off delay time and fall time (Table 5.4). The inverse of the minimum switching period defines the maximum switching frequency of the transistor $f_{switch,max}$ which is equal to $2.68MHz$. Since the time constant $\tau = L/R$ of the motor is equal to $4.75ms$, which is much larger than $373ns$, we can see that the switching frequency of the IRFS7530-7PPbF doesn't represent a limitation for implementing the driving methods explained previously. This switching frequency limitation should be considered instead at the moment of the implementation of the PWM signal generation.

If we look at the MOSFETs on the inverter in the VESC board, we can see that there is a 4.7Ω resistance connected between the driver and the gate, which might change the driving current of the device. We must also keep an eye on the V_{GS} value applied by the MOSFET driver not to be lower than $10V$.

5.1.4 *MOSFET Driver*

The central component in the VESC board, around which the rest of the circuit was designed, is the IC *DRV*8302, a brushless motor inverter driver manufactured by TI. The complete functional block diagram can be found in Appendix B.

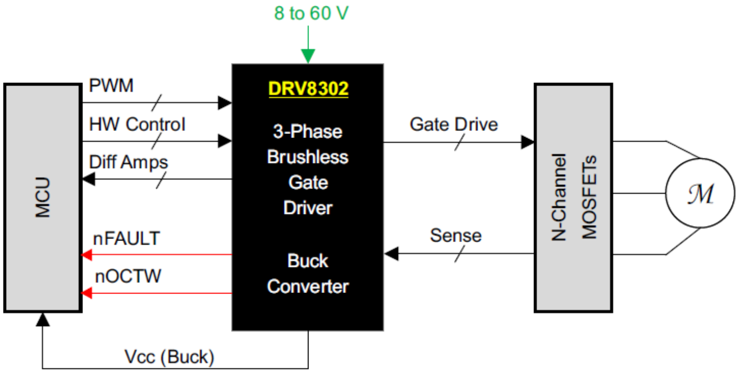


Figure 5.3: DRV8302 Simplified Schematic

The *DRV*8302 provides three half bridge drivers capable of driving two N-channel MOSFETs each, which solves the need to implement a high-side and a low-side circuitry to turn on each one of the transistors on the inverter, as mentioned in 3.3.1. For this mean, the device provides up to $30mA$ at an output voltage V_{GS} between $9.5V$ and $11.5V$ to drive the transistors, using a bootstrap gate driver architecture with trickle charge circuitry to support a duty cycle of 100%.

5.1.4.1 *Operating Range*

The *DRV*8302 can operate from a single power supply with a wide range from $8V$ to $60V$. This voltage range limits both the minimum operating voltage and the maximum operating voltage of the VESC board. The minimum operating voltage shouldn't be a problem in applications like ROBI', where the power is supplied by battery packs with a voltage larger than 12V, but if V_{DC} is smaller than $8V$, the driver won't operate correctly.

It is important to point out that the *DRV*8302 also includes an integrated switching mode buck converter with adjustable output and switching frequency, which can provide up to $1.5A$. In the case of the VESC board, the logic level voltage V_{DD} is $3.3V$, which is obtained by regulating $5V$ with a Low Drop-Out (LDO) regulator. The $5V$ are obtained from the integrated buck converter of the *DRV*8302. Due to this, having a large voltage drop on the power supply, which can be caused by a big current transient, might lead to a malfunction of the motor drive system since the power supply of the microcontroller could be interrupted. This is one of the reasons why the bulk capacitance at the input of the power supply is important.

The other problem that might arise is on the other side of the voltage supply range of the device, which is $60V$. As mentioned at the end of 3.1.1, it is possible that the motor behaves as a power supply if there is a speed imposed on its rotor. Also, we explained in 3.4.1 that is possible to brake the motor if we direct the current into the power supply. These behaviors lead to the posibility of a voltage increase in the motor terminals equal to the double of the voltage value of the power supply, due to the energy stored in the windings when the motor starts to brake. In the case of ROBI', this voltage might reach up to $72V$, which can lead to failure of the $DRV8302$, so we must be careful of this when braking or we must install an electronic circuit which protects the device.

5.1.4.2 *Switching Frequency*

It is stated in the datasheet of the $DRV8302$ that the maximum switching frequency that can be achieved on the MOSFET inverter by using this driver is $200kHz$, which depends on a gate charge equal to $25nC$. Since the gate charge of the MOSFETs used in the inverter is much larger, we need to calculate the minimum switching period in order to be sure that this value is the same for our case. The minimum switching period is obtained from the following Equations, using the data from the dynamic electrical characteristics of the transistor (Table 5.4) and the gate charge curve of the transistor (Figure5.4) (Ghioni, 2016):

$$t_{d(ON)} = \frac{Q_{GS}}{I_G} = 2.07\mu s \tag{5.2}$$

$$t_{rise} = \frac{Q_p}{I_G} = 2.433\mu s \tag{5.3}$$

$$t_{fall} = \frac{Q_p}{I_G} = \frac{Q_p \cdot R_G}{V_{GS,p}} = 97ns \tag{5.4}$$

$$t_{d(OFF)} = R_G \cdot C_{iss(MAX)} \ln(\frac{V_{GS,MAX}}{V_{GS,p}}) = 144.3ns \tag{5.5}$$

where Q_p is the plateau charge in Figure 5.4, equal to the Gate-to-Drain charge charge, specified in Table 5.4; I_G is the current capacity of the driver, equal to $30mA$; R_G is the gate resistance of the circuit, defined by the gate resistance of the MOSFET, 2.2Ω, in series to the resistance between the driver and the gate, 4.7Ω, obtaining a total R_G equal to 6.9Ω; V_{driver} is the driver voltage, considered as $10V$; $V_{GS,MAX}$ is equal to the maximum Gate-to-Source voltage in the gate charge curve; $V_{GS,p}$ is the gate-to-source voltage in the plateau of the gate charge curve; and the maximum gate capacitance $C_{iss(MAX)}$ can be calculated as:

$$C_{iss(MAX)} = \left.\frac{\mathrm{d}Q}{\mathrm{d}V}\right|_{ohmic} = 25nF \tag{5.6}$$

Since the charge mechanism is defined by the current that can be provided by the driver, the calculation of the turn on and rise time are done considering this current as the limiting parameter. For the turn off and discharge time, the current discharge depends on the voltage on the gate and on the gate resistance.

We can see that the values obtained are much larger than the ones mentioned by the datasheet under the test conditions, therefore, we calculate a new minimum switching period equal to $4.74\mu s$, which represents a maximum switching frequency of $211kHz$, therefore, we should considering the maximum switching frequency as the one stated by the datasheet of the driver. In any case, this doesn't represent a constraint in the driving of the in-wheel motor, but it should be taken into account while implementing the PWM signal generation, as mentioned in 5.1.3.1.

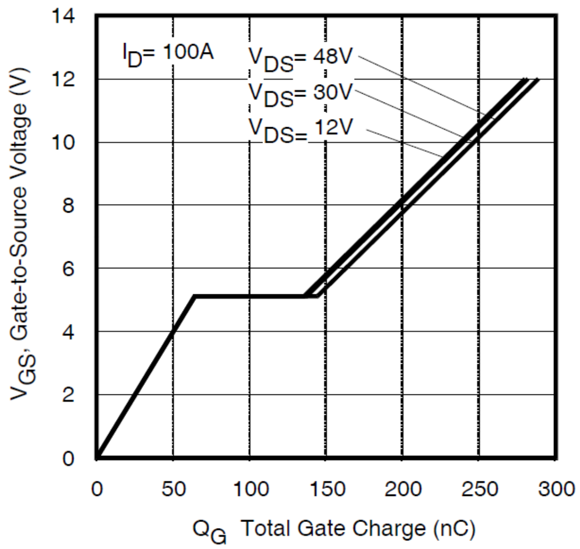


Figure 5.4: Gate Charge vs. Gate-to-Source Voltage

5.1.4.3 *Driver Protection*

The configuration from which the three-phase inverter is implemented has a dangerous failure condition called "shoot-through". This is a characteristic of any circuit based on a "half-bridge" configuration and it can occur when the two transistors in one branch of the inverter are enabled, which would create a low resistance path ($2R_{DS(ON)}$) for the current from V_{DC} to $0V$, which would practically be a short circuit that can be catastrophic. In order to prevent this failure condition, the $DRV8302$ uses an automatic hand shaking when the high side or low side MOSFET is switching, meaning that, if one of the transistors of the bridge is enabled, the other transistor can't be enabled.

The $DRV8302$ provides overcurrent, overtemperature, and undervoltage protection. The overcurrent protection is done by sensing the drain-to-source voltage of the transistor V_{DS} while the transistor is enabled, which is equal to the product of the current flowing through the MOSFET I_D, and the on-resistance $R_{DS(ON)}$ of the same. This voltage is compared against a trip point, which determines when there is an overcurrent. This trip point is defined by applying a voltage to the OC_ADJ pin in the IC using a resistive voltage divider, enabled by the pin $DVDD$. In the case of the VESC, the pin OC_ADJ is connected directly to the pin $DVDD$, which disables the overcurrent protection feature of the device. Also due to this, the pin $nOCTW$ which informs about an overcurrent fault is not connected to the microcontroller.

The undervoltage protection of the $DRV8302$ consists on driving low the gate drive outputs of the inverter whenever the voltage of the pins $PVDD$ and $GVDD$ are below their undervoltage thresholds ($6V$ and $8V$ respectively). $GVDD$ is the voltage of the gate driver regulator and is usually $11V$. The device has two $PVDD$ inputs: $PVDD1$ and $PVDD2$. $PVDD1$ is the power supply of the gate driving circuit, while $PVDD2$ is the power supply of the integrated buck converter. In the case of the VESC board, both power supplies are connected to the same node V_SUPPLY. An undervoltage brownout of $13\mu s$ in $PVDD1$ can cause the device to become unresponsive to external inputs until the device is reseted. This behavior was present while doing the tests of the board when the motor drive demanded a current larger than the limit of the power supply, which caused a voltage drop and the device became unresponsive, even if the buck converter was still working. An undervoltage error flag is provided by the device through the pin $nFault$.

The datasheet states that there is also an overvoltage protection, which turns off the date driver and the charge pump in case than the voltage $GVDD$ reaches $16V$. This voltage is generated from an internal regulator, which delivers the voltage to drive the gates of the MOSFETs of the inverter. Therefore, an overvoltage on this pin means that there is a hardware problem and the board must be inspected.

The overtemperature protection of the device works in two levels. The first level delivers an overtemperature warning through the pin $nOCTW$ (which is disconnected in the VESC board). The second level of the protection shuts down the gate driver and the charge pump until the temperature on the driver decreases and the EN_GATE pin is reset.

The EN_GATE pin enables the operation of the inverter gate driver. If the EN_GATE pin is low, the driver won't respond to the driving signals applied on the switching signal input pins INH_X and INL_X.

One of the main features of this MOSFET inverter driver is that there is a built-in buck converter. This allows the $DRV8302$ to provide a power supply for the microcontroller and other lower voltage components. The buck converter is based on the TI's $TPS54160$ step-down converter, which can provide up to $1.5A$. The output voltage of this converter is defined by the external components connected to the pins dedicated to this, which close the feedback loop. In the case of the VESC board, the converter is configured to provide $5V$, which can be used to supply power to different low voltage devices. From this $5V$ power supply, a $3.3V$ LDO voltage regulator is connected to supply power for the microcontroller, as mentioned before.

In order to sense the current flowing through the MOSFETs of the inverter, and therefore, in the windings of the motor, the $DRV8302$ includes two current shunt amplifiers. The working principle of the shunt resistance voltage drop amplifier is explained in 3.3.3. In the case of the $DRV8302$, the voltage drop is measured in two differential amplifiers from which we can select the gain. The approach of using only two shunt resistances, instead of three, in order to read the current flowing through all the phases of the motor, is based on the fact that the currents on the motor behave as in a nodal analysis, so if we apply the Kirchhoff's Current Law (KCL) in the center of the motor, we obtain that:

$$I_{phase1} + I_{phase2} + I_{phase3} = 0 \tag{5.7}$$

therefore, by measuring the current in, for example, phase 1 and phase 3, like in the case of the VESC board, we can calculate the current flowing through phase 2 as:

$$I_{phase2} = -(I_{phase1} + I_{phase3}) \tag{5.8}$$

The voltage obtained as the result of the conversion (Equation 5.9) presents an off-set in order to be able to read currents bidirectionally. This offset is set by applying the reference voltage V_{REF} in the pin REF. In the case of the VESC board, this pin is directly connected to the output of the LDO regulator ($3.3V$).

$$V_O = \frac{V_{REF}}{2} - G \cdot (SN_X - SP_X) \tag{5.9}$$

From 5.9 we can therefore calculate the phase currents I_{phase1} and I_{phase3} as:

$$I_{phase} = \frac{SN_X - SP_X}{R_{shunt}} = \frac{V_O - \frac{V_{REF}}{2}}{R_{shunt}G} \tag{5.10}$$

where $SN_X - SP_X$ is the voltage drop across the shunt resistance. In the case of the VESC board, the gain is hardwired to 10 and the value of the shunt resistance R_{shunt} is 0.001Ω.

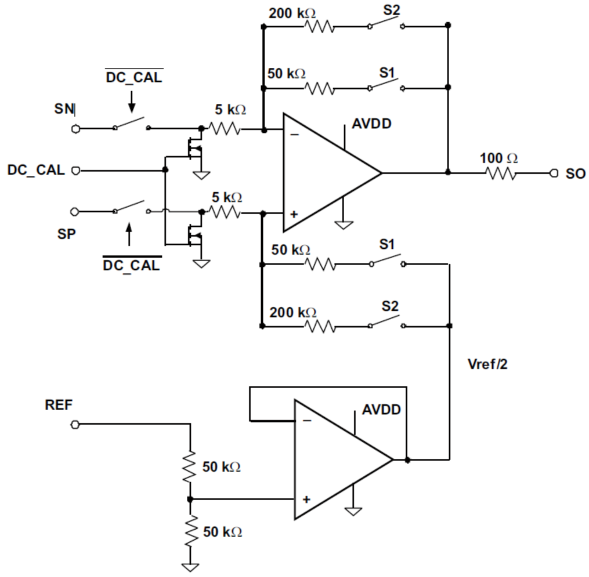


Figure 5.5: Current Shunt Amplifier Simplified Block Diagram

5.1.5 *Microcontroller Unit*

The processor for which this board was designed is the STM32F405rg: a 64-pin IC microcontroller of the STMicroelectronics family of 32-bit devices with an ARM Cortex-M4 CPU and a core operating frequency up to 168 MHz. The Cortex-M4 core features a Floating Point Unit (FPU) single precision which supports all ARM single-precision data-processing instructions and data types. It also implements a full set of Digital Signal Processor (DSP) instructions, which makes it ideal for the signal processing required to drive PMAC motors. This microcontroller also includes three 12-bit ADCs and supports USART, I2C, SPI, USB and CAN communication interfaces, improving the versatility of the WMU. The memory of the microcontroller consists in 1024 kB of Flash memory and 192 kB of SRAM memory.

The microcontroller is programmed by uploading a *.hex* file into its memory by using the programming tool $ST - LinkV2$.

5.1.6 *Peripherals*

The VESC board has many peripheral circuits in order to improve its versatility. In the case of ROBI', these peripherals are indispensable for the implementation of the WMU.

5.1.6.1 *USB*

The board has a micro-Universal Serial Bus (USB) port, which is connected to the corresponding pins D+ and D- of the microcontroller. This Universal Serial Bus (USB) port can be used as a normal USB device, which requires the implementation of the USB drivers and of a compatible program developed to communicate through the port. It can also be used as a USB-Serial device, which creates a virtual serial COM port in the computer to which it is connected in order to communicate to it through a serial therminal, like PuTTY. STMicroelectronics provides drivers to implement both uses of the USB port.

The USB port was not used in this project since the implementation of a compatible USB software would require much time. Also, the USB-Serial library of STMicroelectronics was not used since it presented many problems in the development stage.

5.1.6.2 *CAN*

The Controller Area Network (CAN) is a communication protocol developed by the German company Robert Bosch GmbH, based on a bus topology for the transmission of messages in a distributed environment, considering a solution to the communication management between many ECUs (Lawrenz and Obermöller, 2011). This communication protocol is the most used one in automotive applications and it also can be found in many other applications with a multi-drop bus architecture. In the case of ROBI', the idea is to implement the communication between its ECUs using CAN.

The microcontroller includes hardware support for CAN communications. The physical layer is controlled by the IC CAN transceiver SN65HVD232, by acTI. The implemented circuit can be found in Figure A.4 in Appendix A.

The CAN bus drivers for this project were provided by the Skyward Experimental Rocketry project team, and its implementation is yet to be tested.

5.1.6.3 *Hall Enconder*

The signal of the Hall effect sensors embedded in the PMAC motor connected to the board is conditioned by the circuit that can be found in Figure A.5 in Appendix A. This circuits connects the output of the Hall effect sensors to a pull-up resistor since Hall effect sensors might have an open-collector configuration.

5.1.6.4 *Communication Interfaces*

Other types of communication protocols can be implemented into the board, like Universal Asynchronous Receiver-Transmitter (UART), Serial Peripheral Interface (SPI) and Inter-Integrated Circuits (I2C), by means of connecting the respective nodes of the board through the pins available in the connector $P3$, which can be located in the schematic overview (Figure A.1, Appendix A).

5.1.6.5 *Temperature Sensors*

The board considers the use of two temperature sensors, one connected externally and one embedded on the same board.

The external temperature sensor can be selected in any technology as long as the output is inside the ranges of the ADC of the microcontroller, which will be explained later.

The internal temperature sensor is a Negative Temperature Coefficient (NTC) thermistor. A thermistor is a type of resistor whose resistance value changes with a larger dependence o temperature than normal resistors (Micro-Chip Technologies, 2010). The thermistor is used as part of a voltage divider, with the output connected to an ADC port of the microcontroller (Figure A.3, Appendix A).

The temperature of the board can be calculated based on the voltage drop on the thermistor due the change of its resistance in function of its calculated dependence on temperature:

$$R = R_0 \exp(B - (\frac{1}{T_0} - \frac{1}{T})) \tag{5.11}$$

which can be solved for the temperature as:

$$T = \frac{B}{\ln(R/r_\infty)} \tag{5.12}$$

and since the value of the other resistance in the voltage divider is equal to the rated resistance R_0 of the thermistor, we can calculate the temperature of the thermistor by measuring the output voltage of the voltage divider as:

$$T = \frac{B}{T_0} - \frac{B}{\ln(V_{temp} - 1)} \tag{5.13}$$

5.1.6.6 *Indicator LEDs*

There are three LED indicators that can be used to visually determine the status of the processes running in the board. Two of these LEDs can be controlled by the microcontroller. The other LED is connected directly to the $3.3V$ voltage supply in order to see if the power supply is connected into the board and the microcontroller is energized.

5.1.6.7 *RC Servomotor Port*

There is a port provided to connect an RC servomotor to the board, which is connected to a free PWM port on the microcontroller. This port can also be used for other purposes. For example, in this project, the port was used in order to visualize signals with the aid of an oscilloscope.

5.1.7 *Angular Position Sensor*

As mentioned in Chapter 4, the angular position sensor of the rotor is determined by the Orbis absolute rotation magnetic encoder.

Since the encoder uses the SSI interface to communicate with other decives, a signal conditioning circuit was needed because the VESC board doesn't support the physical layer of this communication protocol. In order to solve this problem, an $RS422$ transceiver circuit (Figure 5.6) was connected between the Orbis sensor and the board. The transceiver purpose is to manage the physical layer interconnection between the SSI interface of the Orbis encoder and the SPI interface available in the board. The clock signal, which defines the transmission frequency, was connected from the port SCK of the board to the D input of the transceiver, which converts it into a differential clock signal with output in the ports Y and Z, which is the type of signal that is read by the SSI protocol. The data input to the board was through the port $MISO$, connected to the output R of the transceiver, which converts the differential output signal of the encoder A and B into a CMOS logic-levels signal.

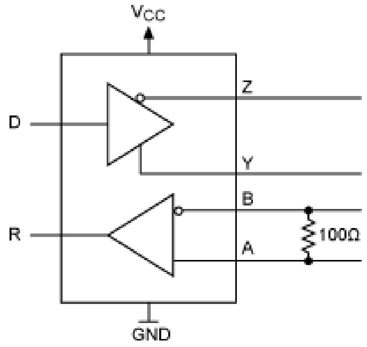


Figure 5.6: RS422 Transceiver Diagram

Another problem encountered with the encoder setup is that the pulley system previously implemented to transfer the angular position information from the in-wheel motor to the Orbis sensor (Figure 4.2) converted the absolute encoder into an incremental encoder due to the 4.6 size ratio between the gears attached to them. This aspect ratio increases the resolution of the angular position obtained but it doesn't provide an absolute value of the actual angular position of the rotor. A solution for this could be to use the Hall effect sensors embedded in the motor to define the actual angle range (6 steps of 60°) and add the difference between two angles provided by the Orbis encoder as an increment to the angle defined by the Hall effect sensors. The problem with the implementation of this solution is that it would be necessary to have the motor already rotating at a known speed to calculate the angular difference to be added to the value defined by the Hall encoder.

The solution to this problem was to change the transmission size ratio from 4.6 to 10. This would provide not the absolute position of the rotor, but it would provide the absolute position of the electrical position of the rotor, which is explained in Chapter 3:

$$\theta_{electrical} = \mathrm{PP} \cdot \theta_{mechanical} \tag{3.19}$$

The system was implemented as a gear transmission instead of a pulley, since the small gear size didn't allow a propper setup using the pulley. This brings the sensor closer to the ground, which was one of the things that was avoided by the pulley system, so further development on this topic should be done.

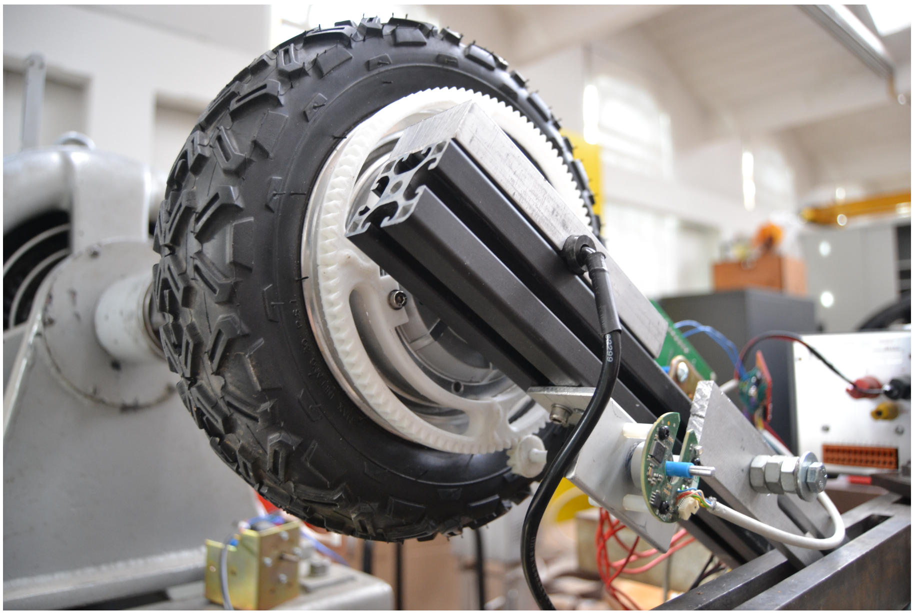


Figure 5.7: New angular position transmission system

5.2 EMBEDDED SOFTWARE

The term "embedded software" is applied to the computer software programmed into the embedded systems, which usually have computing power and memory constrains, like microcontrollers, in order to maintain a lower price, lower power consumption and a higher robustness than a typical computer processor. This software is designed to be programmed only once into the microcontroller, which will execute a sequence of algorithms during its lifetime, unless there is an update needed.

Embedded software implementation deals with a topic that is not very usual in higher programming levels: the use of peripheral registers. These registers allow us to use and communicate with the different peripheral components in a microcontroller that interact with physical variables though its pins. For example, to apply a high logic signal into one of the pins of a microcontroller, we must generate a code that will access the register that controls the physical state of that pin and change its value, which would toggle the voltage of such pin from GND to VDD.

All the algorithms explained in 3.4 are implemented into the motor driver circuit by writing them in embedded code. The inputs to the algorithm are physical variables obtained by means of interpretation of digital signals, analog signals converted into digital signals by means of an Analog-to-Digital Converter (ADC), and communication interfaces, like CAN, SPI or UART. The outputs from the algorithm are mainly analog signals, which are implemented into the motor driver as a PWM signal, which, by applying it to the three-phase inverter which commutates the V_{DC} voltage in the windings of the motor, generates an analog signal.

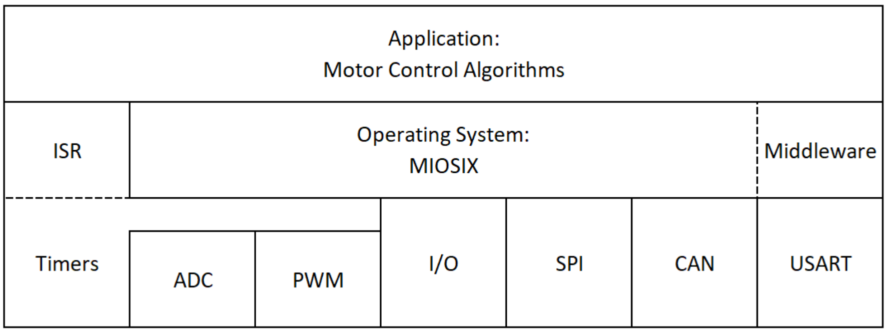


Figure 5.8: Embedded software architecture diagram implemented for the WMU

For the WMU development, we decided not to keep working with the source code developed for the VESC board, since there was no development regarding FOC, and because the implementation of the driving algorithms in ROBI' might have different considerations regarding the algorithms developed for an electric skateboard.

5.2.1 *Operating System*

An embedded system can run without an operating system (OS) if the tasks that are going to be executed on it are simple. For this project we decided to use an existent embedded OS with aims to help us with the implementation of many functions that might need to run in parallel.

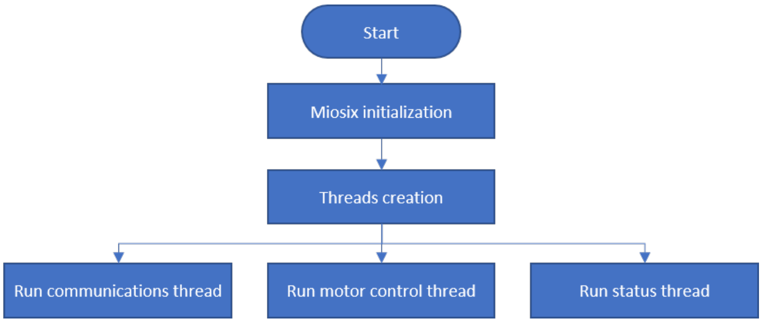


Figure 5.9: Threads Execution Flowchart

The OS chosen to be implemented in this project was Miosix. Miosix is an OS kernel designed to run on 32*bit* microcontrollers, in active development since 2008. Miosix supports C++, which allows the implementation of embedded software with an objects-oriented architecture. Due to the use of C++, it also supports both a single process and multiple threads application models, which simplifies the implementation of threads for this project (Terraneo, 2014). Miosix was chosen due to its support for the STM32 microcontrollers.

Three threads were implemented, using Miosix multi-thread support, with aims to simplify the implementation of the motor control algorithms, running along with the communication protocol between the WMU and a User Interface (UI). Later on, the implementation of new threads can be done to control CAN communications or other new functionalities and algorithms. The first thread implemented in this project controls the communication though an Universal Asynchronous Receiver-Transmitter (UART) port, which can be connected to a computer with the implemented UI or to another board with aims to control the reference values of the motor control algorithms or to monitor the different variables of the system. The UART port was configured in the Miosix environment to be used as in a computer based C++ environment, by the use of the `iostream.h` library commands, `cin` and `cout`, in such way that we don't have to worry about configuring the UART related registers. Also, while the system is expecting an input value after using the `cin` command, the thread blocks itself until it receives the expected input. The second thread runs the speed control algorithm. A third thread was implemented only to toggle a LED every second in order to detect if the system is correctly running or if there is any error with the board.

5.2.2 *Pulse Width Modulated Signal Generation*

The microcontroller family *STM32F4* has two timers that are designed to be able to control three-phase motor inverters. For this, the microntroller is able to generate two complementary PWM signals in three different channels for each timer, intended to command the high-side and the low-side signals of a power inverter driver as the *DRV8302*. These timers are called "advanced timers", and are identified as *TIM1* and *TIM8* (STMicroelectronics NV, 2016). The board connects the PWM outputs from timer *TIM1* to the driver.

A PWM signal is defined by a frequency and a duty cycle, as seen in Chapter 3. The actual implementation of the PWM signal depends on five variables: a counting frequency, a counter, a compare value, a top value and an output signal. The counter normally starts from zero and the compare value should be smaller than the top value, both larger than zero. The output signal is a digital reference signal, which starts as set (high) when the PWM signal is enabled. This output signal is usually the PWM signal generated. When the PWM counter is enabled, the output signal is set and the counter starts increasing its value at the counting frequency until it reaches the compare value, which resets the output signal (low). After this, the counter keeps increasing until it reaches the top value. After the counter reaches the top value, it counts down or starts from zero, depending on the PWM mode.

The PWM mode is defined by the motor drive algorithm. In the case of the trapezoidal drive, since the PWM signal is only applied to one branch at a time, the counter restarts from zero after the top value is reached, switching the output signal immediately after this. This mode is called Edge-Aligned PWM (Figure 5.10).

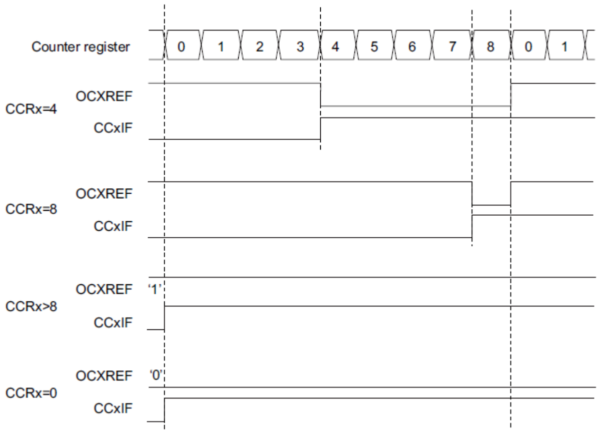


Figure 5.10: Edge-aligned PWM waveforms (ARR=8)

In the case of the sinusoidal drive, since the three branches have PWM signals driving the transistors all the time, in order to reduce the harmonic signals generation due to the commutation between

$0V$ and V_{DC}, a normal solution is to center the PWM signals by, after counting up, counting down from the top value back to zero, resetting the output signal when the counter reaches the compare value. This way, we don't have three branches commutating from $0V$ to V_{DC} at the same time (Figure 6.16).

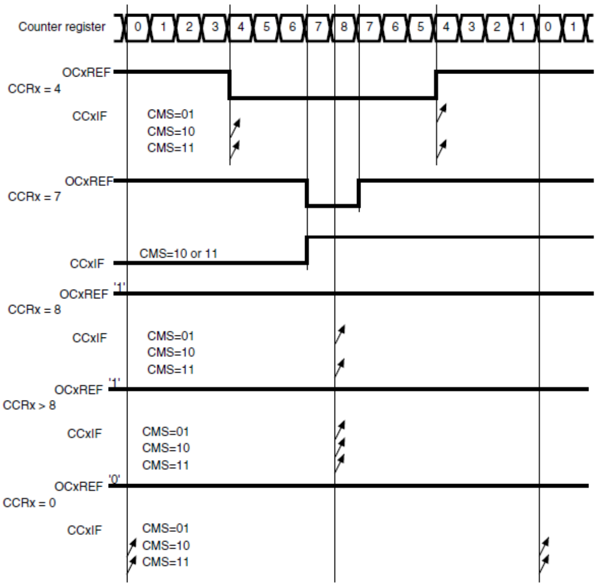


Figure 5.11: Center-aligned PWM waveforms (ARR=8)

The implementation of both motor driving techniques implied also the generation of a signal which is complementary to the reference PWM signal. This complementary signal is obtained by simply inverting to the output PWM signal by modifying one register in the microcontroller. One important aspect to consider regarding this complementary signal, is that it is possible to generate a shoot-through effect in the inverter, even if the driver shouldn't allow it. This shoot-through is generated due to the non-instantaneous activation of the MOSFET gates in the inverter. For example, if the driver disables the gate on the high-side, and then enables the gate on the low side, the gate on the high-side might still be enabled since the capacitance of the gate might still be charged, which creates a shoot-through in the inverter with the duration of the difference between the charge and discharge time of the gates (Maxim Integrated, 2016).

Due to this shoot-through effect due to the gate capacitances, we must generate a dead-time between the main PWM signal and its complementary. This delay is defined as the difference between the activation and deactivation time of the MOSFET equal to $4.26\mu s$. The activation time is the sum of the on-time plus the rise time and the deactivation time is the sum of the fall time plus the off-time, obtained in 5.1.4.2. The dead-time is also configured by modifying the registers of the advanced timer $TIM1$.

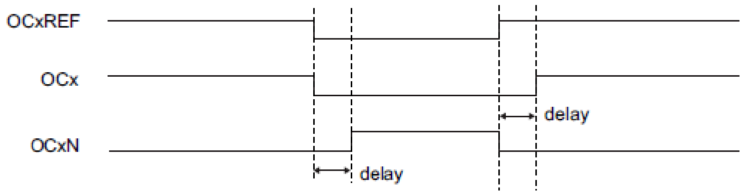


Figure 5.12: Complementary output with dead-time insertion

5.2.3 *Analog-to-Digital Converter*

STMicroelectronics' microcontrollers are famous for having one of the most complete ADC systems in the market. For this project, this is a plus, since the input of analog values is necessary to apply some of the motor drive algorithms. The ADC on the $STM32F4$ microcontrollers family is $12bit$ successive approximation analog-to-digital converter.

The most important implementation of the ADC in this project is used for the current measurement to be applied in the current (and torque) control methods. The ADC was configured in this project to read the two available current signals and a signal proportional to V_{DC} at the same time.

In a continuous model, the current measured from the system would be continuously updated to be compared with the reference values and to be introduced into the PI controller to generate an output voltage signal, which would be applied into the inverter. Since we are dealing with digital models, the implementation differs, but we must approximate it as much as possible to the continuous model to obtain a good performance of our system.

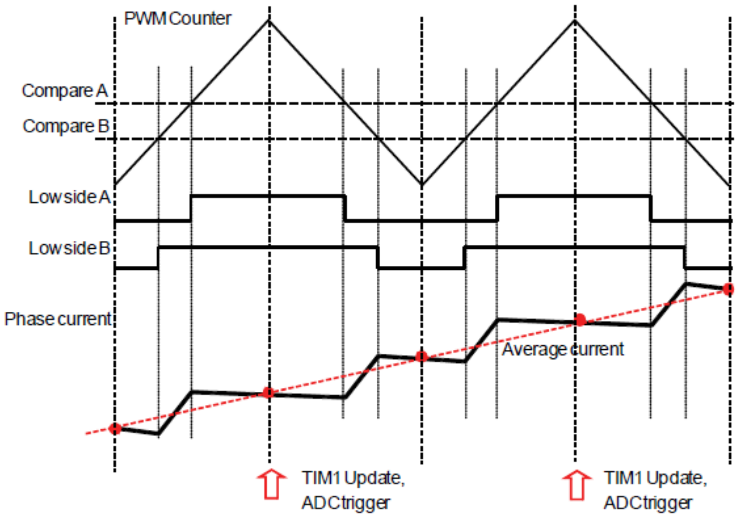


Figure 5.13: Current reading in shunt resistor (STMicroelectronics NV, 2016)

The implementation of the current measurement is explained in Figure 5.13. Based on the windings charge and discharge, which depends on the voltage applied to them, we can obtain the average value of the current flowing through the coil by measuring the current at the middle point of the PWM signal generated to drive the inverter. In order to do this, the ADC was synchronized with the fourth channel of the advanced timer $TIM1$, which doesn't have a physical output, but it generates a reference signal as well as the other three channels that drive the inverter (Figure 5.14).

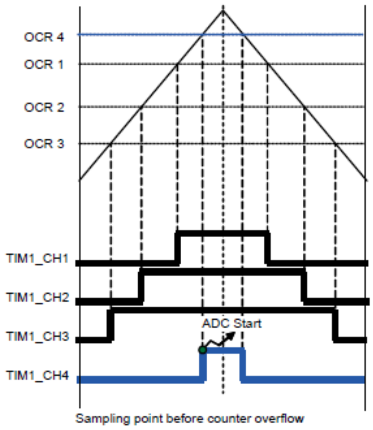


Figure 5.14: ADC and PWM Synchronization

After obtaining with the $12bit$ ADC the value of the voltage drop in the shunt resistor, which was amplified by the driver $DRV8302$, we can calculate the current flowing in the branches 1 and 3 by translating the voltage value obtained into an equivalent current:

$$I_{shunt} = (\frac{ADC_{Vdrop} \cdot 3.3V}{2^{12}} - \frac{3.3V}{2}) \cdot (\frac{1}{G \cdot R_{shunt}}) \qquad (5.14)$$

Another important thing to consider is the possibility of an offset at the output of the $DRV8302$ amplifier. The datasheet of the driver specifies that there can be an offset of $\pm 4mV$, which would translate into a phase current of $400mA$, which is not negligible. In the case of the FOC, this current reading offset could create a DC offset in the generated voltage driving signals and this could significantly reduce the performance of the motor drive. Due to this, we need to implement a calibration algorithm in order to suppress this offset. This calibration is implemented to be done every time that the board is initialized, but it might be necessary to be done every time that the board stops driving the motors.

To do the offset calibration of the current measurement, the $DRV8302$ provides a pin that disconnect the input of the amplifiers in order to obtain the DC offset value. The algorithm implemented creates an average of ten thousand samples of the offset voltage and then defines a current offset value I_{offset} that should be subtracted every time that we calculate a current value.

$$I_{shunt} = (\frac{ADC_{Vdrop} \cdot 3.3V}{2^{12}} - \frac{3.3V}{2}) \cdot (\frac{1}{G \cdot R_{shunt}}) - I_{offset} \quad (5.15)$$

5.2.4 *Algorithms Implementation*

The driving algorithms implemented in this thesis for the WMU are two. The first algorithm is the trapezoidal drive, which can be used in an open loop and in a speed control loop. The second one is the sinusoidal drive, which can be used in an open loop, in FOC only controlling the current and in FOC with a speed control loop. These algorithms depend on other algorithms, like the speed calculation and the PI controller.

The trapezoidal and FOC driving algorithms are executed by a call of an Interrupt Service Routine (ISR) of one of the timers at a specific frequency that will be explained later.

The speed loop controllers are PI controllers executed in a thread of the OS every millisecond.

5.2.4.1 *Speed Calculation*

The speed calculation was made by means of reading the rate of change of the Hall effect sensors value. One clockwise electrical revolution is obtained every time that the Hall effect sensor goes from position 6 to position 1, while a counter-clockwise electrical revolution is obtained when the Hall effect sensor goes from 1 to 6. Following Equation 3.19 we can calculate the mechanical speed of the in-wheel motor, and with this we can estimate the dispalcement speed of the whole robot. The speed calculation function is called every time that a driving algorithm is called in order to update the driving speed and the Hall effect sensor values.

5.2.4.2 *Continuous Angular Position Definition*

As mentioned before, the Orbis angular position encoder is connected to the microcontroller through an $RS422$ transceiver that manages the interaction between their different physical layers. The SSI interface is practically an SPI interface, with the difference that the SSI implemented in the Orbis sensor uses $24bit$ messages, while the SPI protocol uses $8bit$ or $16bit$ messages.

The message of the encoder is structured as follows: $bit23$ to $bit10$ contain the encoder position data; $bit9$ and $bit8$ provide the general status of the encoder (data valid or invalid); $bit7$ to $bit0$ provide a detailed status about the previous two bits.

Listing 5.1: Code snippet of the basic PI controller implemented

```
error = reference_value - measured_value;
output = (error * kp) + integral_error;
  if (output > MAX_VALUE)
    output = MAX_VALUE;
  else if (u < MIN_VALUE)
    output = MIN_VALUE;
  else
    integral_error = integral_error + ((ki * error) * dt);
```

The SPI is configured as receive-only master mode, which sends the clock signal to the encoder, expecting a response signal at the same transmission rate. Since the largest message length that the SPI protocol of the microcontroller supports is $16bit$, we send only a 16 pulses clock signal, obtaining only the 16 Most Significative Bit (MSB) of the message from the encoder.

To obtain the angular position, we do a bit-wise operation in order to read only the 14 MSB of the message, which contain the angular position. This binary value is translated into radians, to which an angular slip is added (explained in 3.3.2) to define the absolute electrical angular position of the encoder due to the rotor angular position transmission adopted (5.1.7).

It is stated in the datasheet of the encoder that the maximum clock frequency is $500kHz$, unless a "Delay First Clock" function is implemented, which would increase the frequency to $2.5MHz$, but such function is not explained.

If we consider a clock frequency equal to $500kHz$ for a $24bit$ message, the message has a duration of $48\mu s$. After every message we must wait a transfer timeout t_M equal to $16\mu s$ and a pause time t_p equal to $20\mu s$ in order to get a new message from the encoder. These delays add up to the message duration to give a total transmission duration of $84\mu s$, which defines the maximum transmission frequency of the encoder, which is equal to $11.9kHz$.

5.2.4.3 *Proportional-Integral Controller*

The PI controller algorithm was based on a pseudo-code algorithm proposed by Magnani, Ferretti, and Rocco, 2007. The idea was to prove the ease of implementation of a simple code with a real functionality in a system that can look complex due to all the peripheral configuration and management.

The PI controller implemented compares a measured value with an input value (which can be input through the UI or by another control loop) to generate an error signal and then follows the loop explained in 3.5.1. The difference between this controller and a theoretical one is that, in real cases, the integral part might become very large, and due to this, try to apply a very large control signal, which can damage the system.

For example, in the FOC loop, the PI controller outputs a driving signal depending on a current calculation and its directly applied to the windings. If, by any reason, the in-wheel system gets stuck, the integral part of the system, which increments its value proportionally to the time that has passed and to the error signal, would command the driver to implement an every-time larger driving signal to the coils, which might damage the drive system. To avoid this, the PI algorithm implemented has a saturation condition. If the output of the controller is larger than a `MAX_VALUE`, then the output becomes `MAX_VALUE` and the same happens in with the opposite case using `MIN_VALUE`.

In the speed loop implementation for the trapezoidal drive, the saturation values are 0 and 1, since the duty cycle must be always a number between 0 and 1. In the FOC implementation, the saturation value is a percentage of the bus voltage V_{DC} in order not to generate a large output current. In the speed loop which uses FOC, since the output of the controller is a current which becomes an input to the FOC loop, the saturation value is a maximum current, which applies as a maximum value and as a minimum value by changing its sign. These maximum driving voltage and current in the FOC related algorithms can be implemented as a safety measure in order to avoid drawing too much current from the power source, which might lead to malfunction of the system.

5.2.4.4 *Trapezoidal Drive*

The trapezoidal drive was implemented by following the six-step sequence with a PWM signal in the high-side transistor, a complementary PWM signal on the low-side transistor of the same branch (positive branch), and enabling a low-side transistor in another branch (negative branch), as explained in 3.4.1.

The algorithm was implemented in a function that is called by the **ISC!** (ISC!) at a frequency that can be specified by the user . Also the PWM signal is driven at a frequency that can be modified depending on the motor characteristic time τ.

One of the first steps of the algorithm is to compare the current status of the Hall effect sensors with the previous one, in order to avoid modifying registers if it is not needed and to reduce the usage of the microcontroller resources. This condition applies only when the speed is larger than $5rad/s$, because, if the motor hasn't started moving, the hall-effect sensors won't change and the algorithm wouldn't be executed.

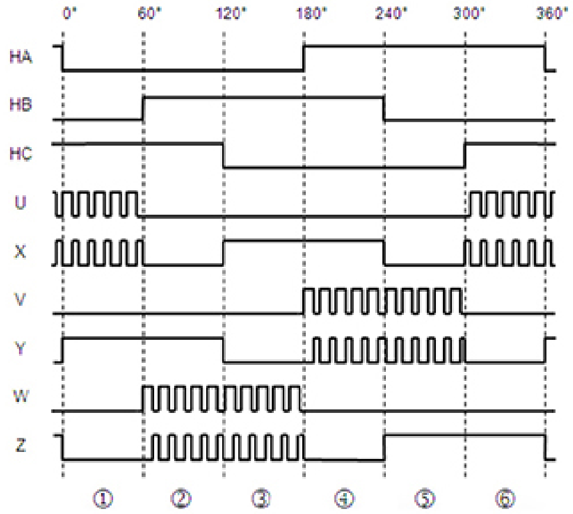


Figure 5.15: Trapezoidal drive signals with complementary PWM in leading branch

5.2.4.5 *Field Oriented Control Implementation*

The FOC algorithm was implemented following the explanation from 3.5.3, which is a straight sequence. The maximum frequency at which the FOC loop can be called is determined by the maximum frequency of the Orbis encoder, since the angular position of the rotor must be detected every time that the loop is executed. Due to this, the ISR calls the FOC loop every $100\mu s$.

One important thing to be considered in the case of ROBI' for the implementation of the FOC, is that the system

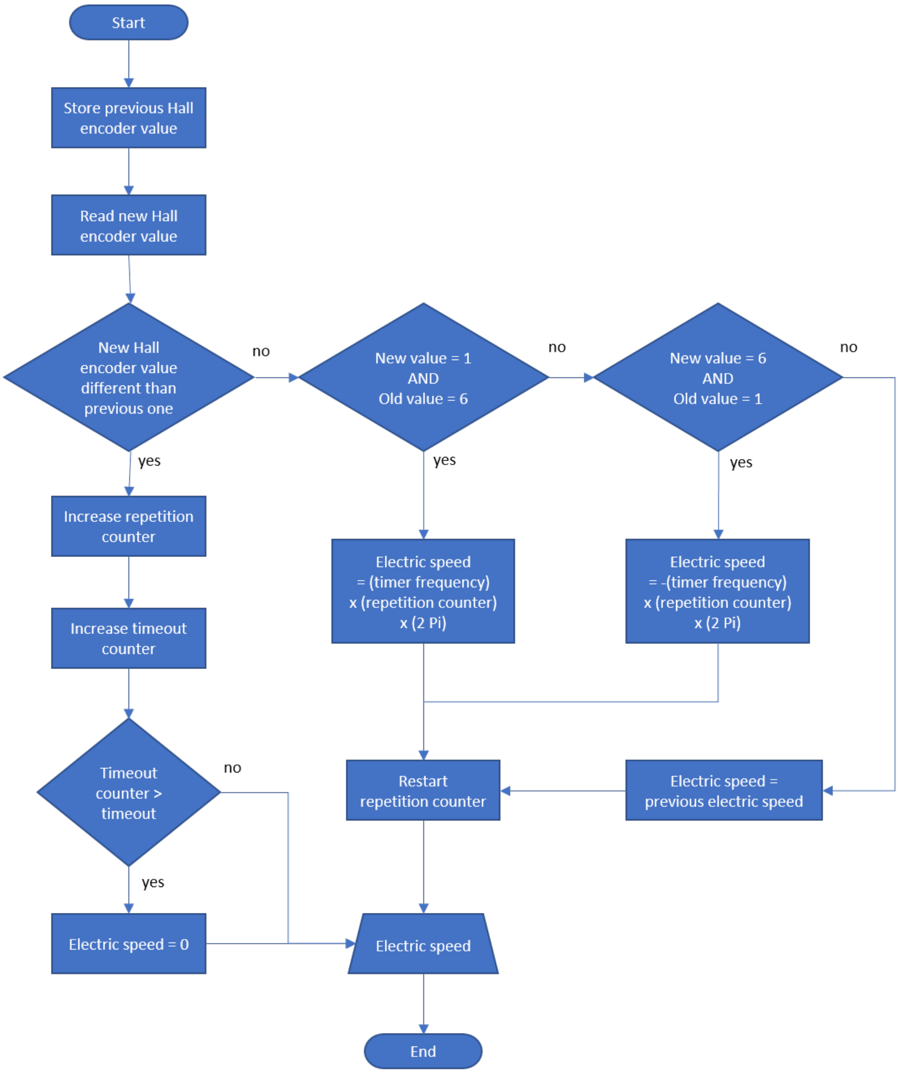

Start
Store previous Hall encoder value
Read new Hall encoder value
New Hall encoder value different than previous one
no
yes
New value = 1 AND Old value = 6
no
yes
New value = 6 AND Old value = 1
no
yes
Increase repetition counter
Increase timeout counter
Electric speed = (timer frequency) x (repetition counter) x (2 Pi)
Electric speed = -(timer frequency) x (repetition counter) x (2 Pi)
Timeout counter > timeout
no
yes
Restart repetition counter
Electric speed = previous electric speed
Electric speed = 0
Electric speed
End

Figure 5.16: Speed Calculation Flowchart

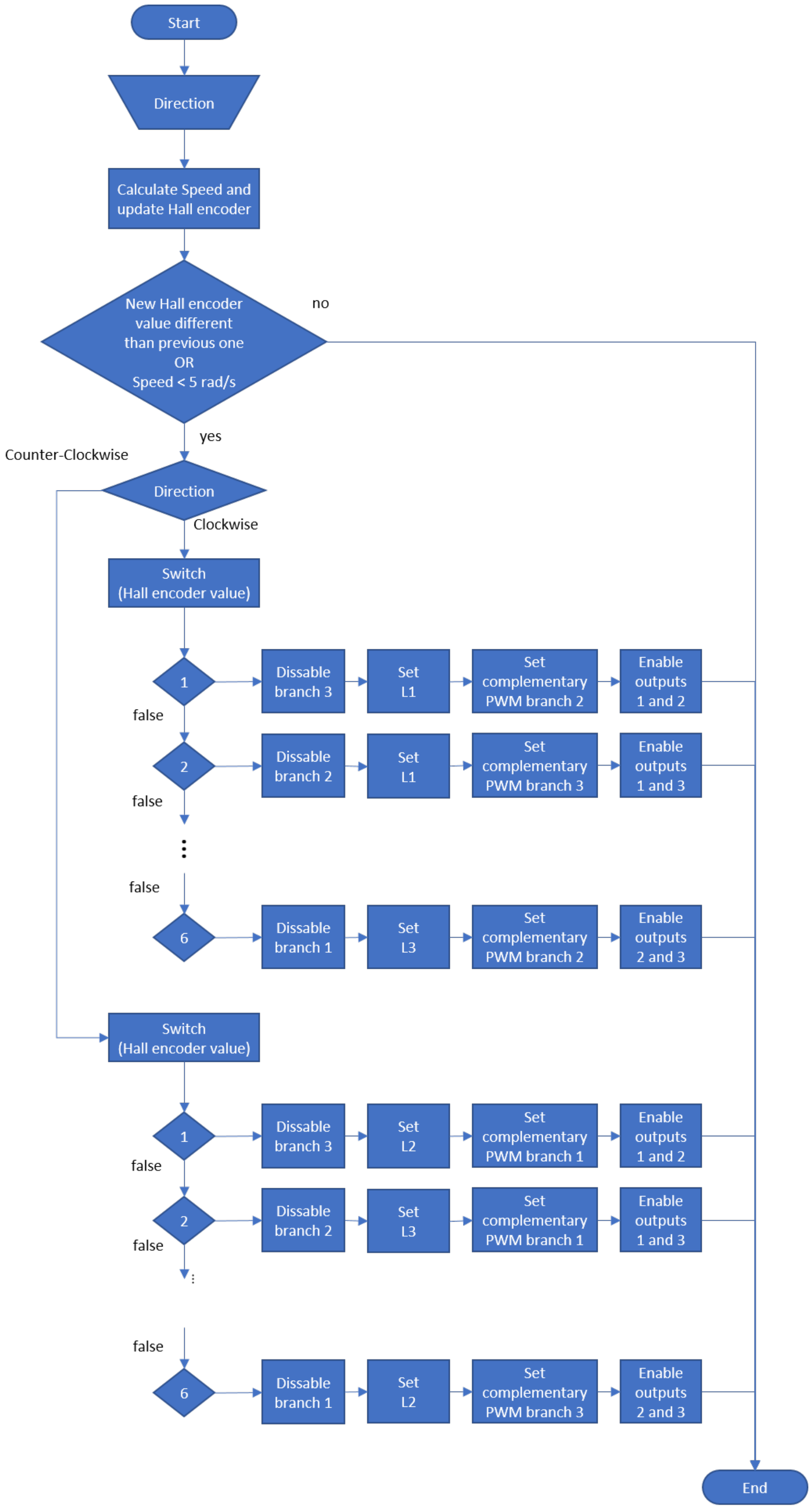


Figure 5.17: Trapezoidal Drive Flowchart

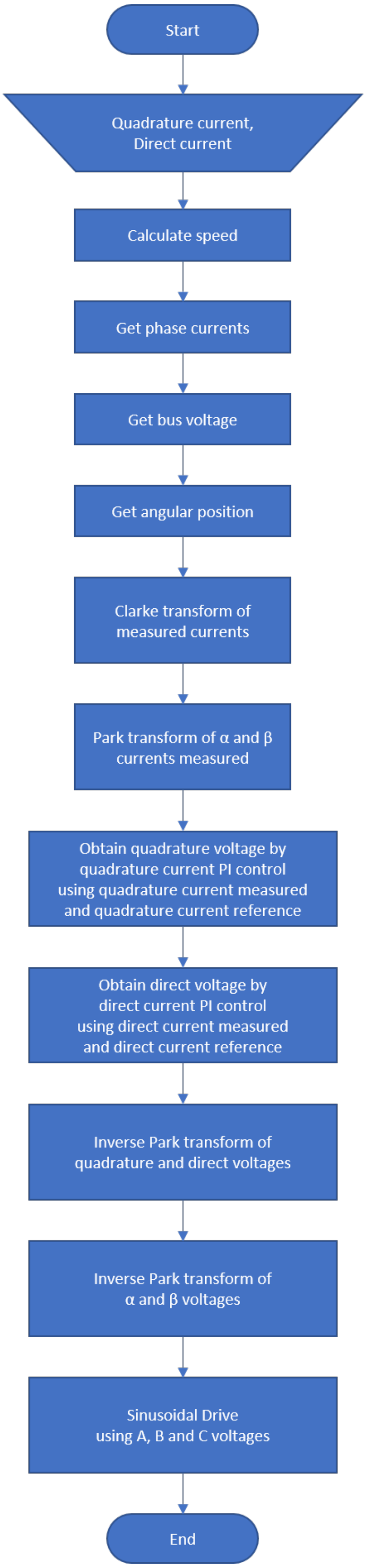

Start
Quadrature current,
Direct current
Calculate speed
Get phase currents
Get bus voltage
Get angular position
Clarke transform of
measured currents
Park transform of α and β
currents measured
Obtain quadrature voltage by
quadrature current PI control
using quadrature current measured
and quadrature current reference
Obtain direct voltage by
direct current PI control
using direct current measured
and direct current reference
Inverse Park transform of
quadrature and direct voltages
Inverse Park transform of
α and β voltages
Sinusoidal Drive
using A, B and C voltages
End

Figure 5.18: Field Oriented Control Flowchart

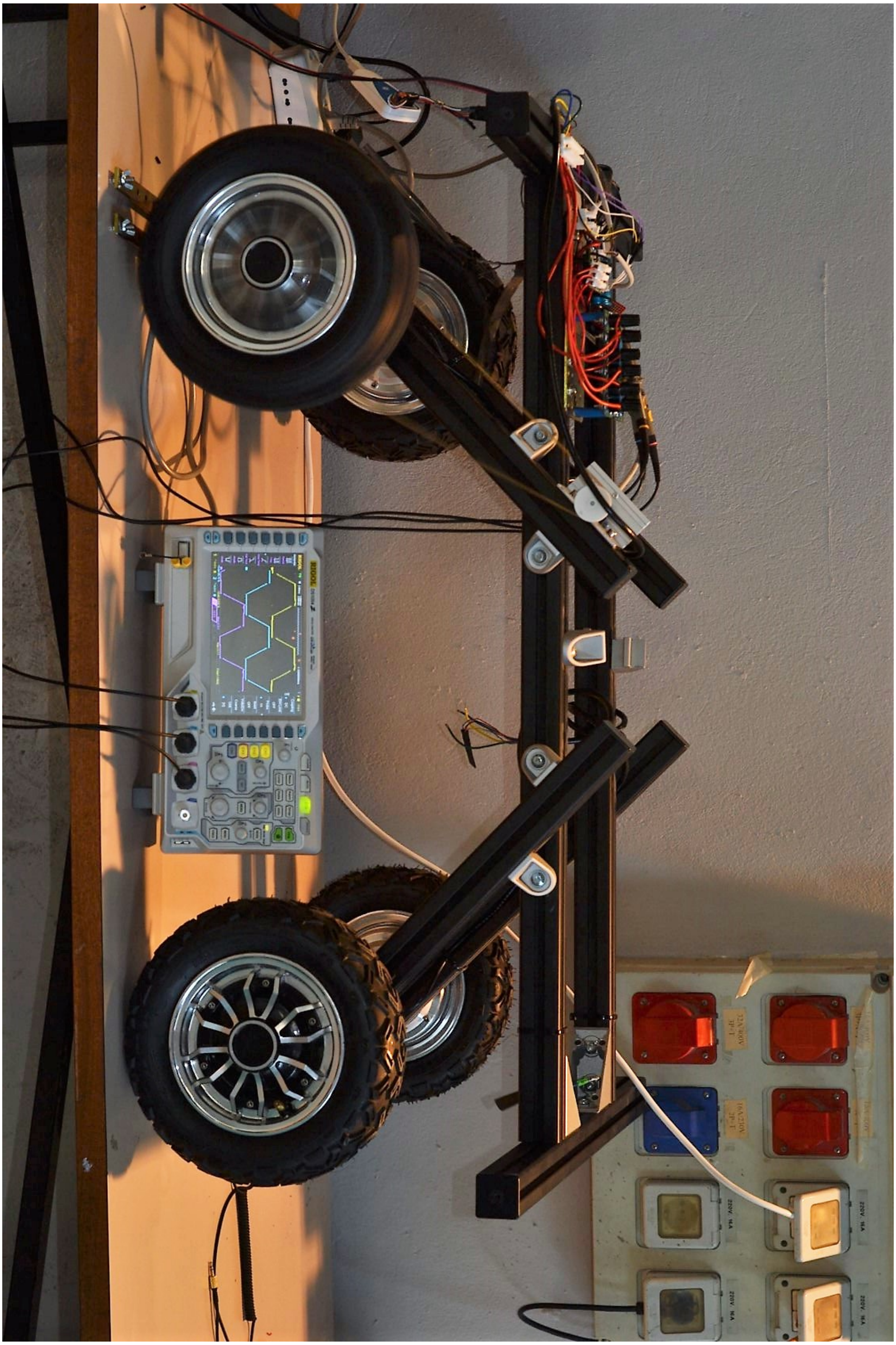


Figure 5.19: Trapezoidal drive implemented on in-wheel motor mounted on ROBI'

5.3 TEST SETUP

This section explains the implementation of the test bench developed to measure the driving capabilities of the WMU. It is also a preamble to the next chapter, which shows the results of the implementation of the different driving methods under different test conditions.

5.3.1 *Measurement Board*

In order to be able to obtain the voltage and current waveforms of the motor, it was necessary to condition those signals to be measured by an oscilloscope or by any other device. With this aim, a measurement board was developed. The schematic files of this measurement board can be found in Appendix C.

The board was designed to acquire data by means of connecting an oscilloscope to the circuit through test points or by setting up an embedded system to acquire the information, transform it and transfer it to a UI by USB or any other digital communication protocol available. The embedded system can be setup by soldering an Atmel microcontroller directly to the board or by connecting an Arduino-like device. Since the Arduino doesn't have a sampling frequency fast enough to read the analog signals of this project, we used the approach of connecting an oscilloscope directly to the board through the test points.

For future reference, it is possible to use other more powerful devices instead of the Arduino to acquire the signals, like a Nucleo board, by STMicroelectronics, which has the same pinout of an Arduino board.

The operating voltage levels of the board can be selected depending on the available power supply and the operating logic levels of the embedded system.

In order to obtain the information of the motor, its terminals must connected to the board by means of three large "banana" connectors. The current of the motor flows through wide copper pads, which are connected to voltage test points, passing through the current transducers and connected to other three banana terminals, which are connected to the WMU.

5.3.1.1 *Voltage Measurement*

The voltage of the motor windings was measured from the motor terminals with reference to ground. This voltage can be acquired by connecting an oscilloscope to the test points available for this, or it can be measured by an embedded system.

In order to condition the signal to be read by the embedded system, it was necessary to implement a circuit with operational amplifiers to reduce the measured voltage levels of the motor to logic levels. The gain of the amplifiers can be set by modifying the dip-switches in the board.

5.3.1.2 *Current Measurement*

The current of the motor was measured between the driver and the motor terminals. It was obtained by means of the LEM current transducer HXS 20-NP, which is a Hall effect sensor that can be configured to measure up to 60A, changing the sensitivity of the transducer by modifying its pinout interconnection, which changes the current distribution inside the sensor, increasing or reducing its effect over the Hall effect sensor.

As in the case of the voltage measurements, the signal can be also obtained by the embedded system and its conditioned by operational amplifier circuits with an adjustable gain.

5.3.2 *User Interface*

The board was connected to a UI by means of a USB-Serial transceiver, which was connected to the available UART port.

The UI was developed in LabView because of the ease of implementation of automatic serial communication algorithms which allow us to control different variables of the WMU, like the reference input values ($I_{q,ref}$, $I_{d,ref}$, ω_{ref}) and the parameters of the controllers implemented (K_P and K_I). It is also possible to obtain and plot different values measured and calculated by the microcontroller, like the angular position and speed of the rotor and the values of the measured currents and voltages.

This UI allowed us to run the tests of the capabilities of the WMU.

5.3.3 *ROBI′ Setup*

The WMU and the measurement board were mounted into ROBI′ with the objective of having a mechanically stable arrangement of the components.

In order to test the torque development of the in-wheel motors, ROBI′ was mounted in a torque test bench, which consists on a DC current generator with a resistance connected to its output. The DC current generator was manufactured by Ercole Marelli & CSA – Milano.

One of the in-wheel motors mounted on ROBI′ was attached to the shaft of the generator by means of a custom-designed coupler, which was 3D printed in PLA plastic. This coupler was attached to the rim of the in-wheel motor by replacing three of the $M3$ screws that attach the rim to the wheel for larger ones in order to be able to attach it.

Table 5.5: Marelli DC Current Generator Characteristics in Function on its Rotation Speed

Power	$10/40kW$
Voltage	$110/440V$
Current	$91A$
Speed	$750/3000RPM$
Excitation voltage	$220V$
Excitation current	$1.5A$

The generator is mounted into a large base, holding it by its rotor (with the use of bearings) and not by its stator. This configuration allows the measurement of the torque generated by means of a load cell, which measures a compression or a tension force applied to it, proportional to the torque applied to the generator, which depends on the power that is being generated to energise the load connected to its terminals. The power generation depends on the field excitement of the generator, therefore, we were able to define a constant torque at a given speed by changing its excitation voltage.

The load cell information was obtained by means of an instrumentation amplifier, model HX711, which has a digital interface. The amplifier output was connected to an Arduino, which translated the information of the sensor into an applied torque, equal to the force measured multiplied by the distance from the center of the generator to the load cell, whose direction is practically tangential to the generator circumference. Before each test was realized, the torque measurement was calibrated, following a procedure like the one applied for the current offset calibration (5.2.3).

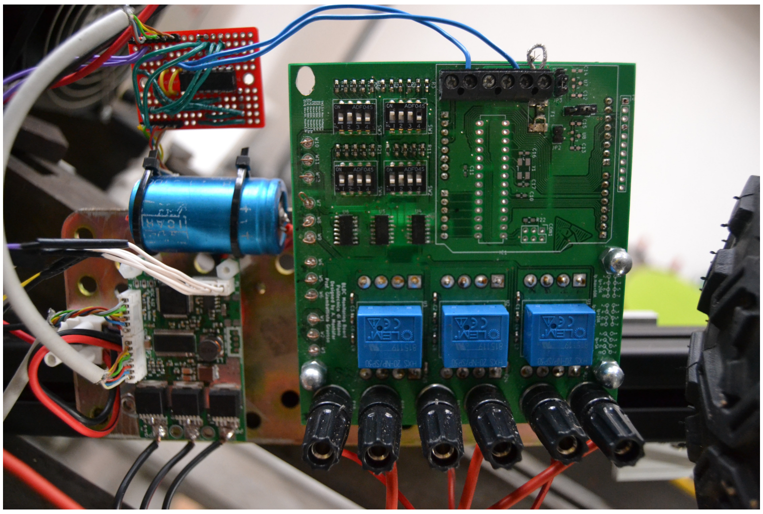


Figure 5.20: WMU connected to measurement board, mounted in ROBI's structure

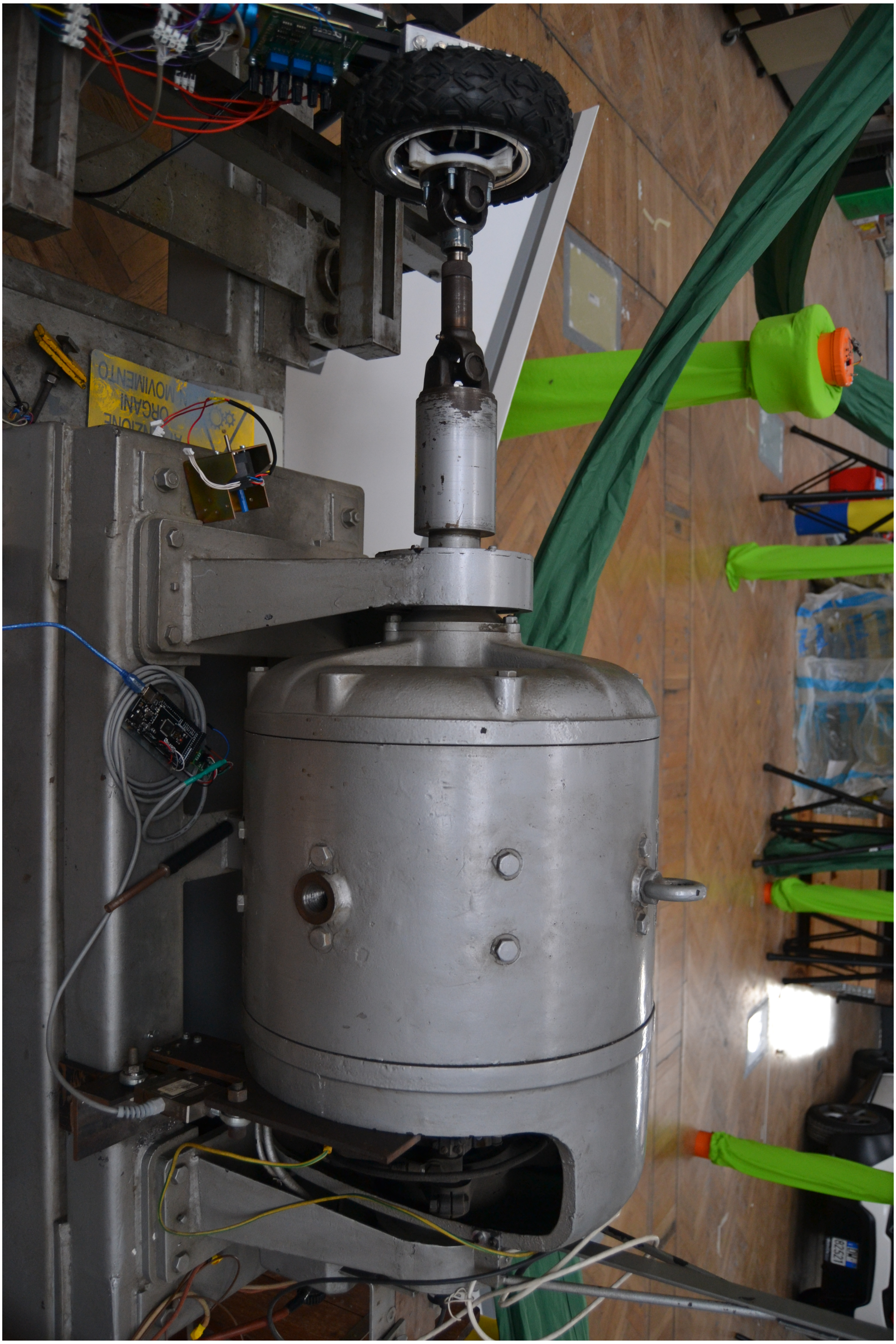


Figure 5.21: In-wheel motor attached to Marelli DC generator

6 RESULTS

This chapter contains waveforms and plots referent to the tests realized to determine the driving ability of the WMU and their explanations. The tests were realized on the in-wheel motor by applying two types of driving methods: trapezoidal and sinusoidal, both at open loop and closed loop. These driving methods were tested under two conditions: loaded and unloaded.

The power was supplied by an Elind KL Series power supply with a current limit of 15 A. The analog signals were obtained by a Rigol oscilloscope, model $DS1054$. The speed readings were obtained through the implemented LabView UI. All the plots and waveforms were processed in MATLAB for their analysis.

6.1 TRAPEZOIDAL DRIVE – OPEN LOOP

The trapezoidal open loop driving was executed by setting a duty cycle directly into the six-step sequence PWM. The algorithm was executed at $50kHz$ and the PWM signal has a frequency of $15kHz$.

Figure 6.1 shows the different speeds achievable by applying different duty cycles with different torque loads. The curves on the plot start on different duty cycles since the torque load couldn't reach a steady value until a certain speed was established. For example, for the measurement with a load equal to $3Nm$, the curve starts at a duty cycle equal to 30%, since at a duty cycle lower than this or a speed lower than $10rad/s$, the load can't be generated by the test bench, which would apply around $2Nm$, therefore those measurements were not considered for the plot. On the other hand, the curves that don't show large duty cycle values (5 and $6Nm$), is because the power supply couldn't feed enough current to the driver to reach such torque.

Figures 6.2, 6.3, 6.4 and 6.5 show the voltage waveforms measured on the terminals of the motor. The trapezoidal BEMF generated on its coils due to the six-step sequence algorithm can be identified as the ramp-up and ramp-down signals between rectangular drives, mainly when the duty cycle is 100%. From these figures we can also identify that when the load increases, the speed of the motor decreases since there is no speed control loop.

Figures 6.6, 6.7, 6.8 and 6.9 show the current waveforms. We can see from its shape, mainly when the duty cycle is 100%, that the BEMF of the motor is sinusoidal, even if the trapezoidal drive is applied. As the load increases, along with the reduction of the speed, the current consumption increases. It can also be appreciated that the current waveforms present a larger ripple as the duty cycle and the load increase, which represents a torque ripple that reduces the performance of the motor.

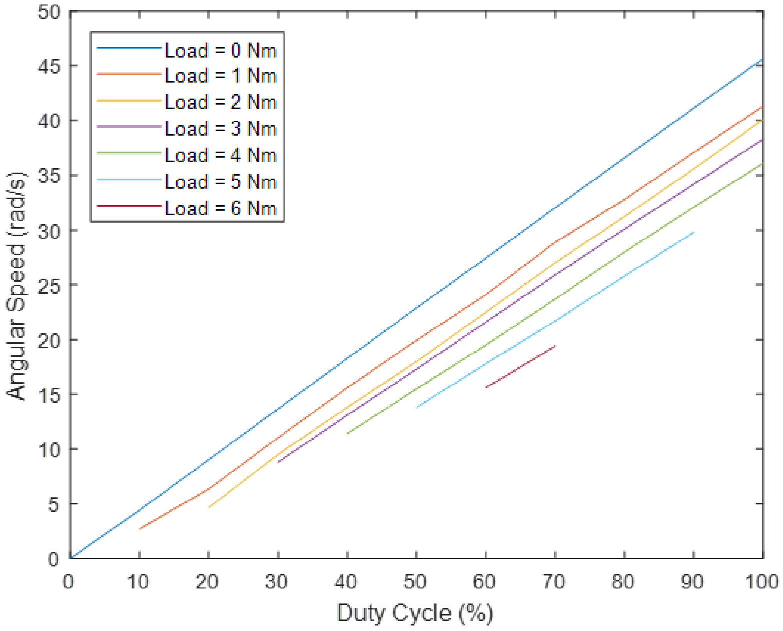
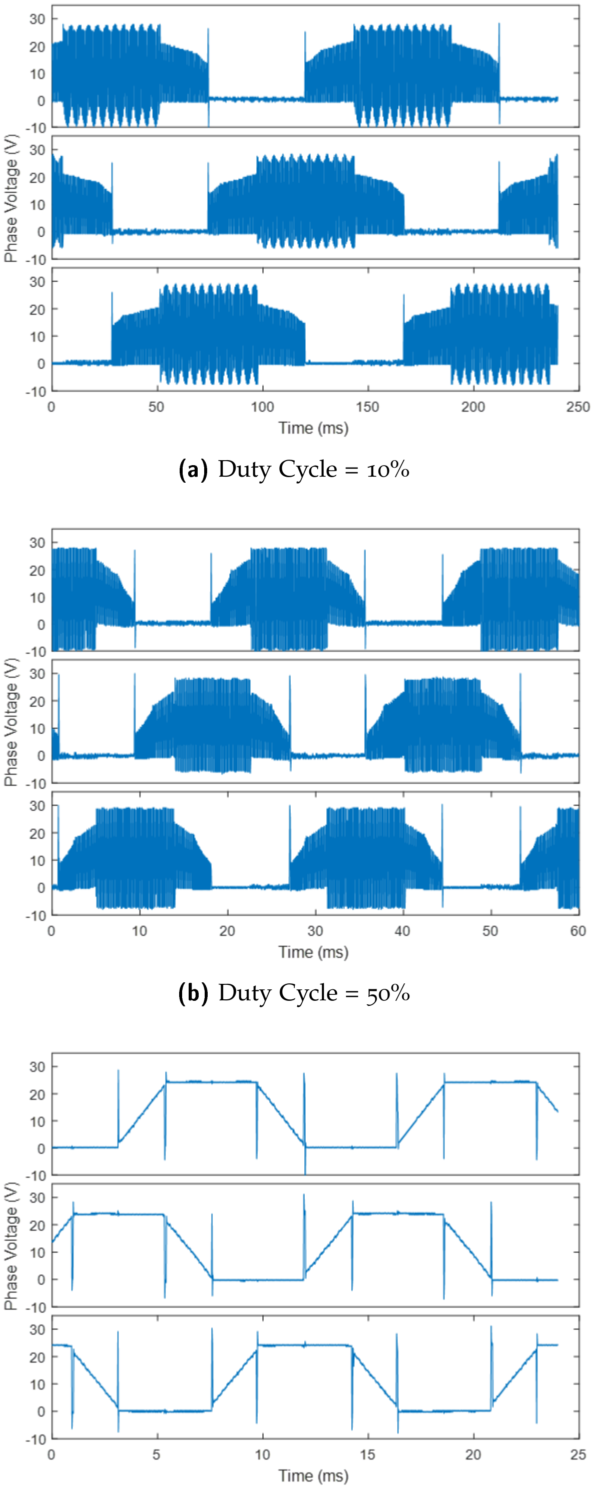


Figure 6.1: Plot of the different speeds achieved applying a known duty cycle using an open loop trapezoidal drive with different loads

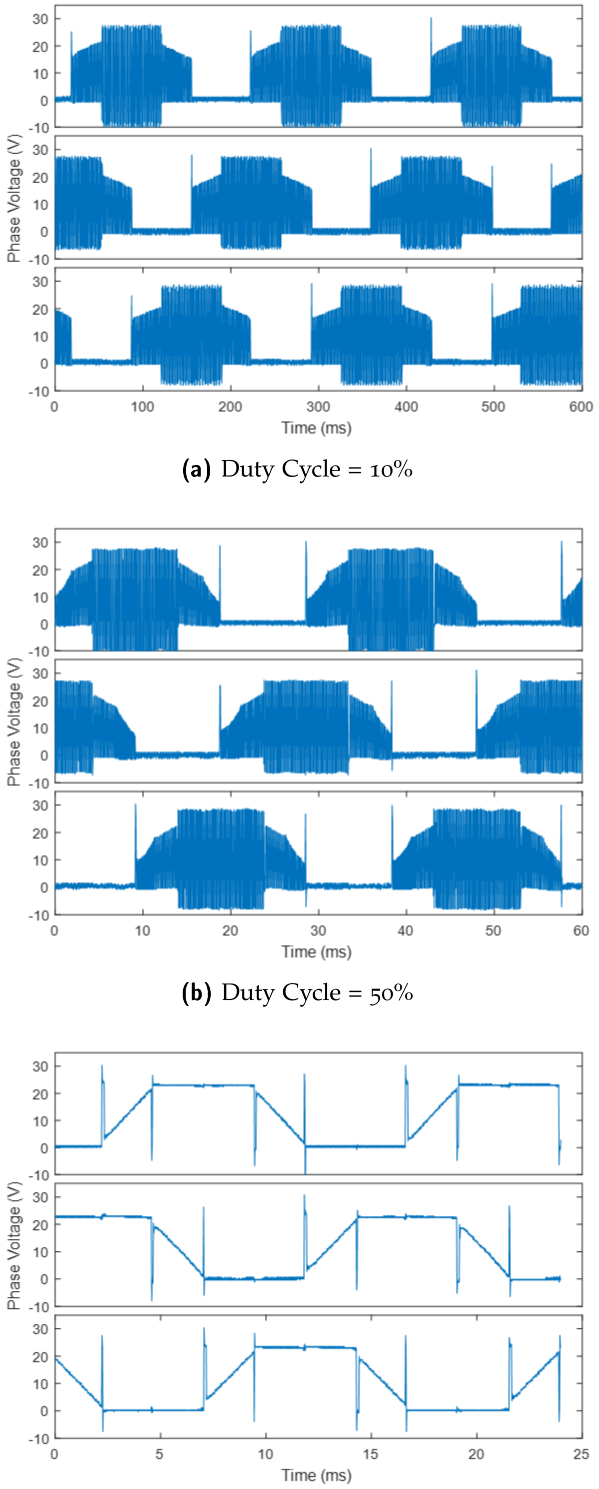


(a) Duty Cycle = 10%

(b) Duty Cycle = 50%

(c) Duty Cycle = 100%

Figure 6.2: Trapezoidal drive voltage waveforms without load applied

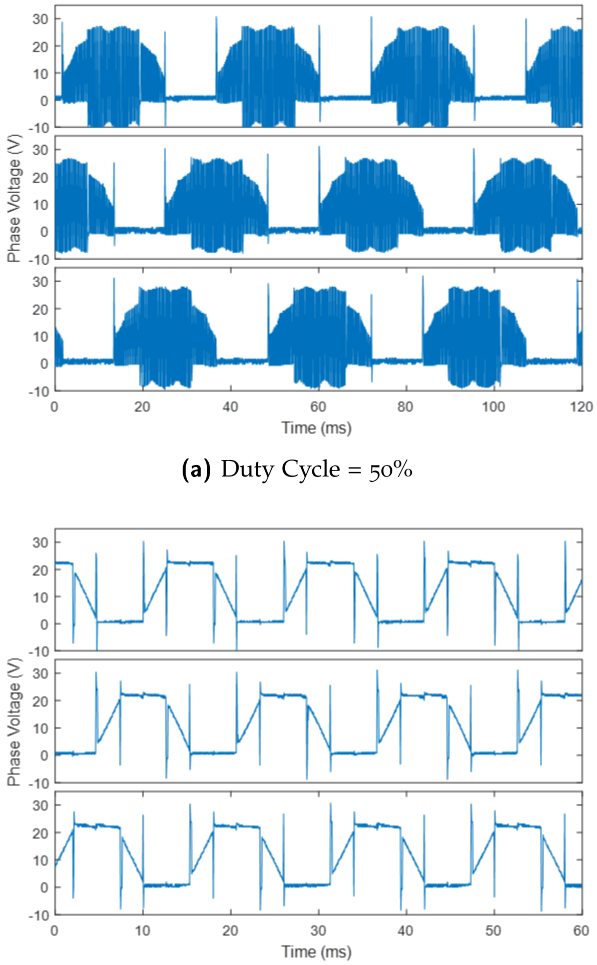


(a) Duty Cycle = 10%

(b) Duty Cycle = 50%

(c) Duty Cycle = 100%

Figure 6.3: Trapezoidal drive voltage waveforms with load = 1 Nm



(a) Duty Cycle = 50%

(b) Duty Cycle = 100%

Figure 6.4: Trapezoidal drive voltage waveforms with load = 3 Nm

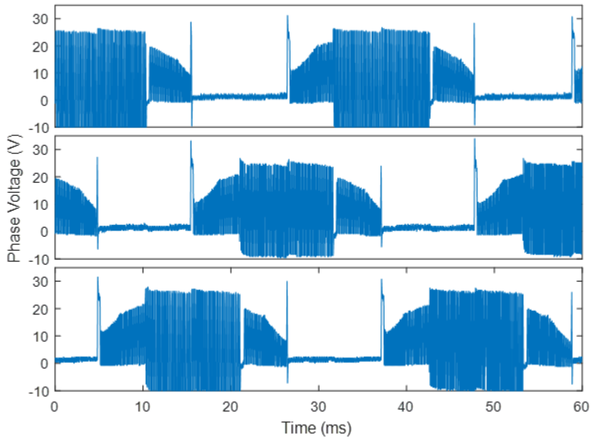
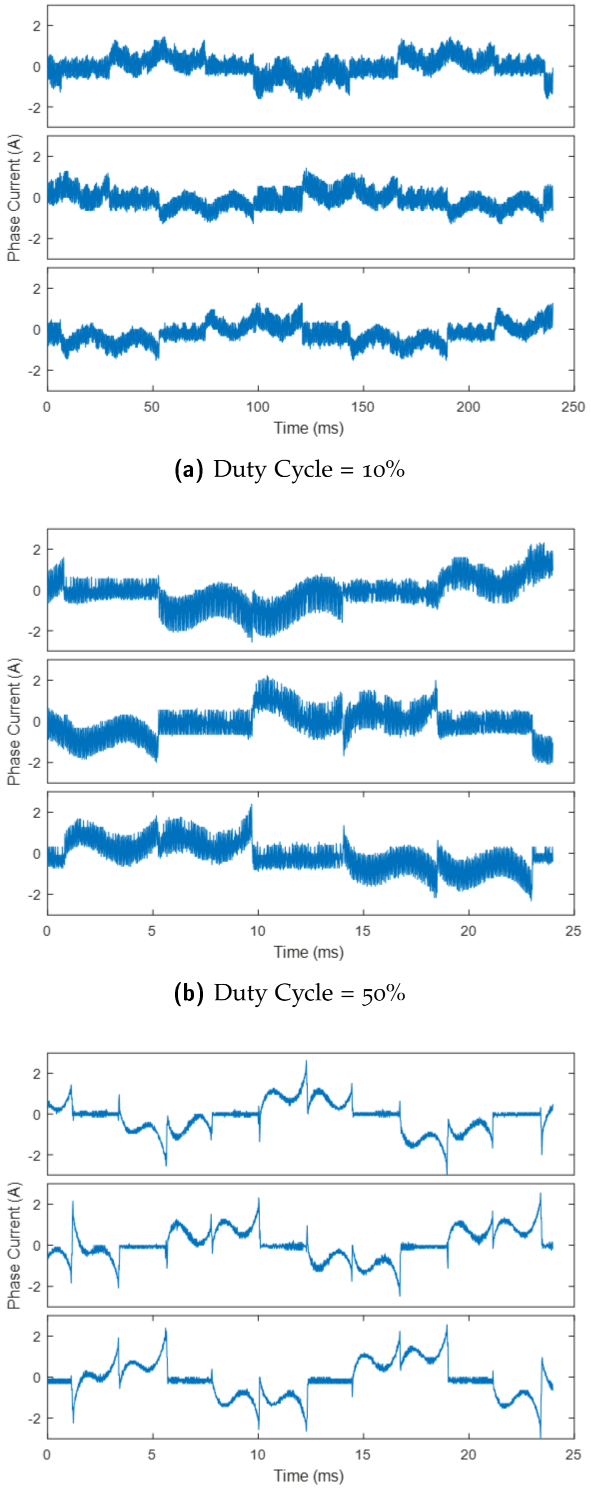


Figure 6.5: Trapezoidal voltage waveforms drive Duty Cycle = 70% and load = 6 Nm

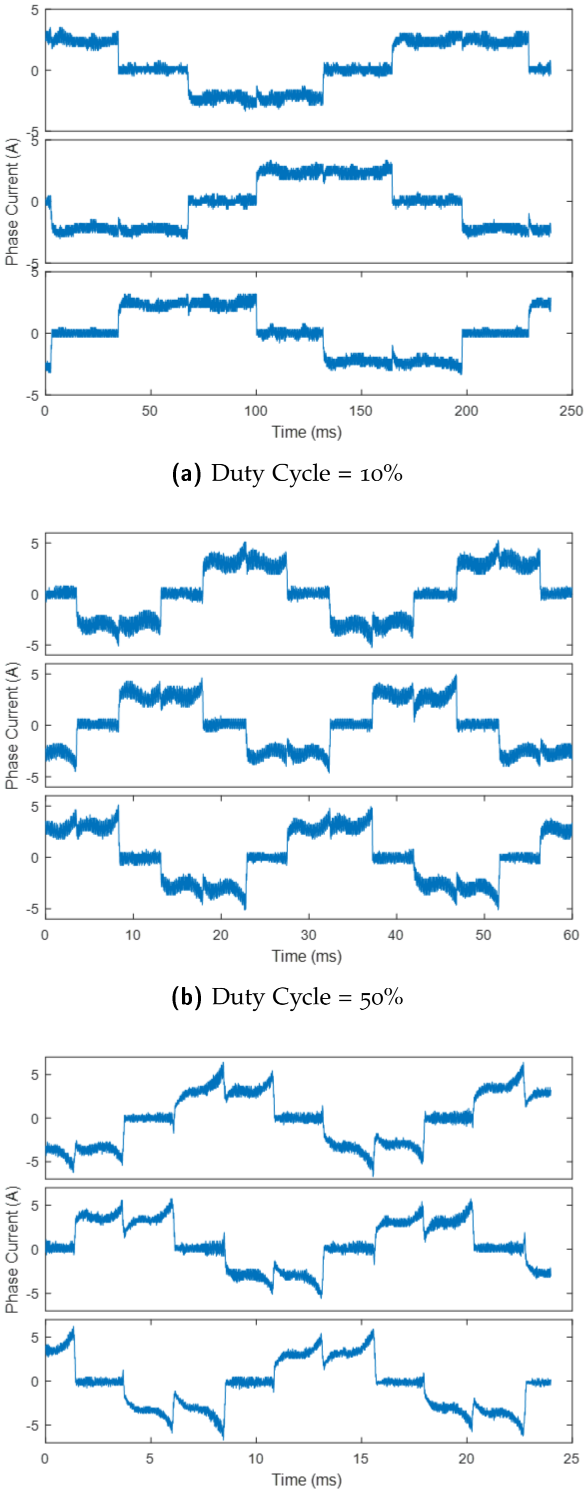


(a) Duty Cycle = 10%

(b) Duty Cycle = 50%

(c) Duty Cycle = 100%

Figure 6.6: Trapezoidal drive current waveforms without load applied

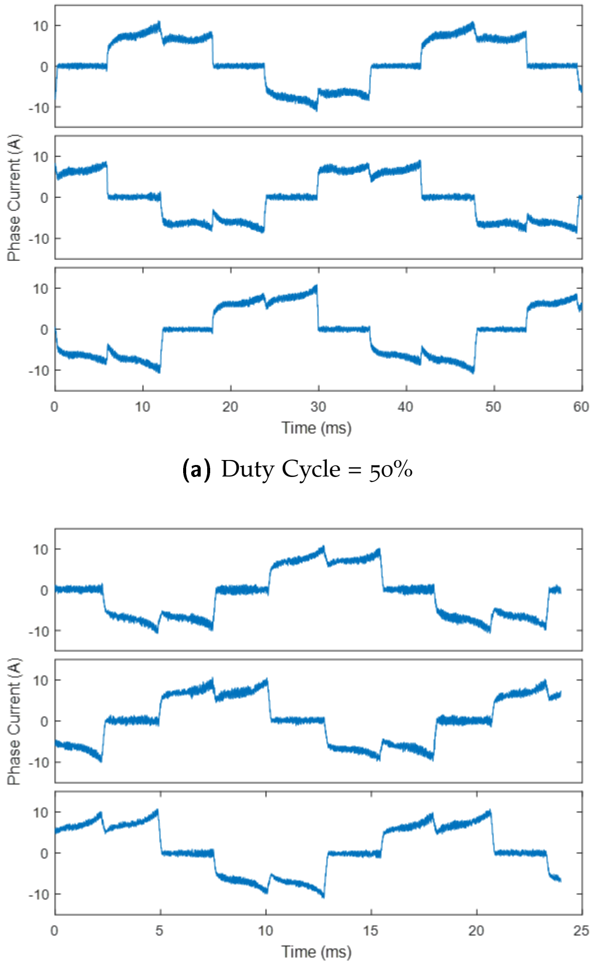


(a) Duty Cycle = 10%

(b) Duty Cycle = 50%

(c) Duty Cycle = 100%

Figure 6.7: Trapezoidal drive current waveforms with load = 1 Nm



(a) Duty Cycle = 50%

(b) Duty Cycle = 100%

Figure 6.8: Trapezoidal drive current waveforms with load = 3 Nm

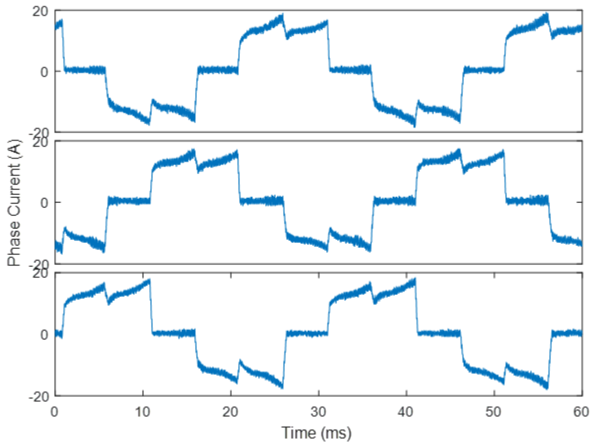


Figure 6.9: Trapezoidal drive current waveform with duty cycle = 70% and load = 6 Nm

6.2 speed control with trapezoidal drive

To see the behavior of the speed control, different step signals were applied with different loads and different controller parameters.

The torque loads were setup first at a steady speed to know the correct excitation voltage of the field, then the motor was stopped and step signal was applied.

Figures 6.10, 6.11 and 6.12 show the different step responses between the same step signal applied and the same load but with different control parameters K_p and K_i. It can be identified that, by changing the value of K_i we obtain different damping behaviors on the motor.

In Figure 6.13 we compare two signals with different K_i values, but in this case, in the test to obtain the signal with $K_i = 0.03$ the current was limited by the power supply. This shows that, even if K_i is larger, the slew rate of the speed depends on the current limit of the power supply.

It is possible to see also the speed behavior of the system compared with the duty cycle applied by the control algorithm on Figures 6.14 and 6.15.

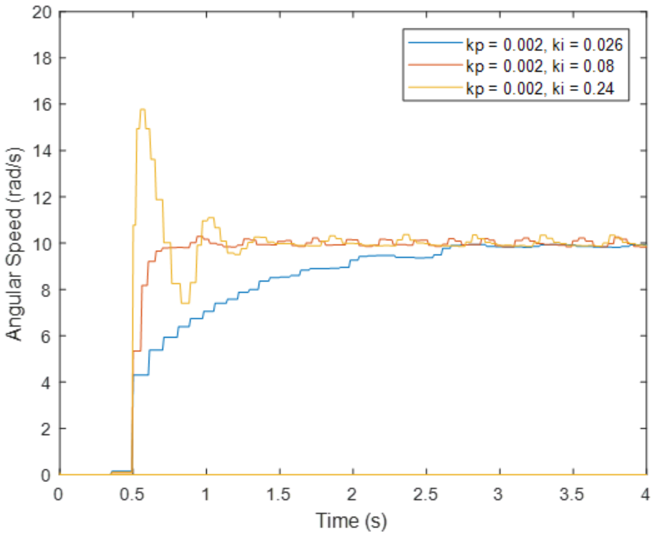


Figure 6.10: Step response to a 10 rad/s signal without load applied

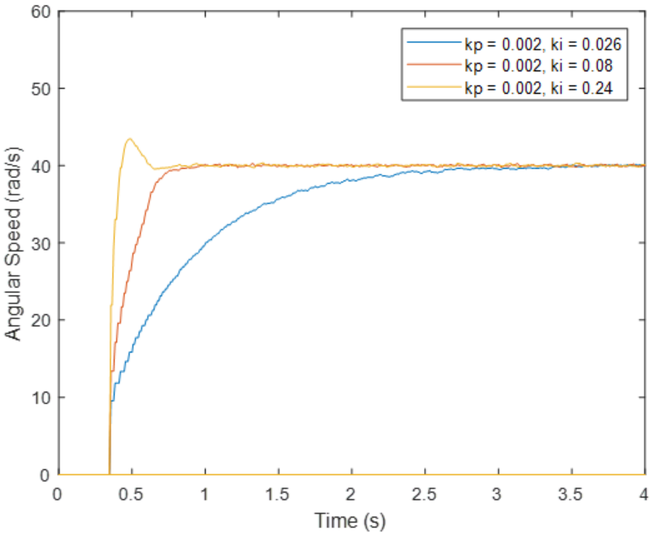


Figure 6.11: Step response to a 40 rad/s signal without load applied

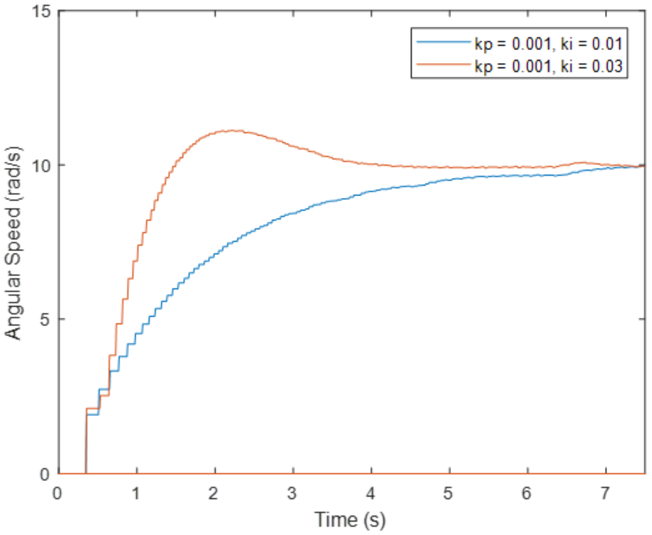


Figure 6.12: Step response to a 10 rad/s signal with a 3 Nm load

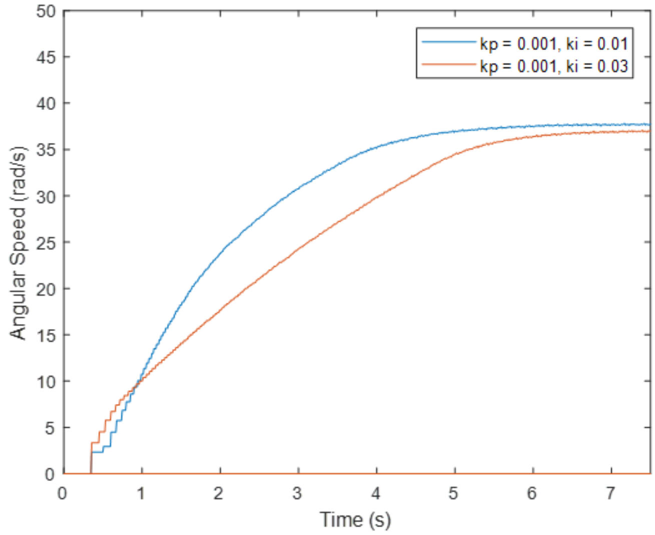


Figure 6.13: Step response to a 40 rad/s signal with a 3 Nm load

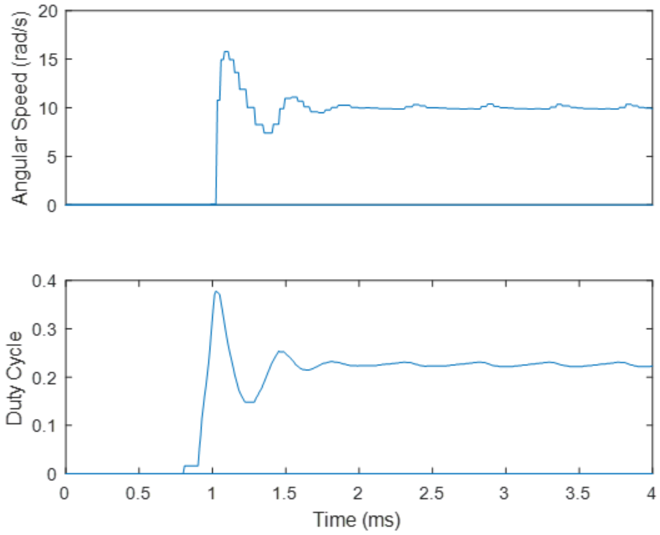


Figure 6.14: Step response to a 10 rad/s signal without load, compared to its driving duty cycle signal

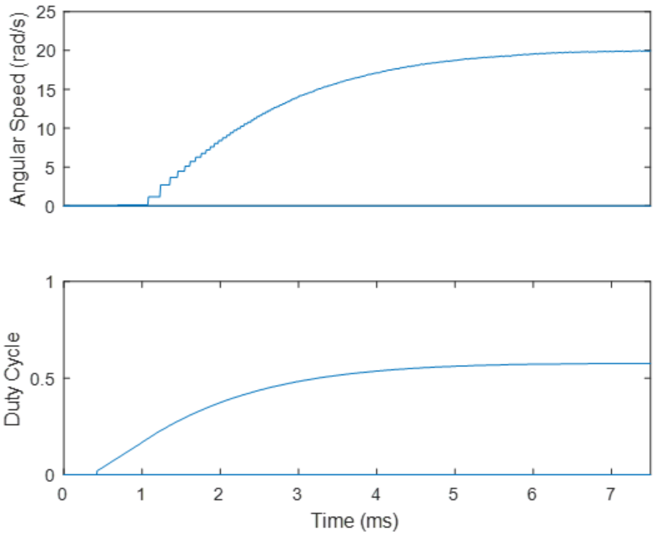


Figure 6.15: Step response to a 20 rad/s signal with a 3 Nm load, compared to its driving duty cycle signal

6.3 sinusoidal drive – open loop

This test was applied at the beginning of the development of the Field Oriented Control to test the generation of a sinusoidal waveform depending on the angular position of the rotor. The signals were generated by applying a constant amplitude value as a quadrature voltage without any load attached to the motor.

Figure 6.16 shows the center-aligned PWM signal generated by the inverter in the motor terminals. We can identify the ringing effect on the rising and falling edges but it has a small duration, therefore it can be neglected.

It can be appreciated in Figure 6.17c that the current in the highest and lowest side of the signal are limited by the LEM current sensor of the measurement board at $20A$. While doing these tests, one wire of the measurement board started melting since it didn't have the required diameter for this amount of current. Also the LEM transducers became hot, since they were configured to measure signals of $\pm 5A$ to provide a better resolution.

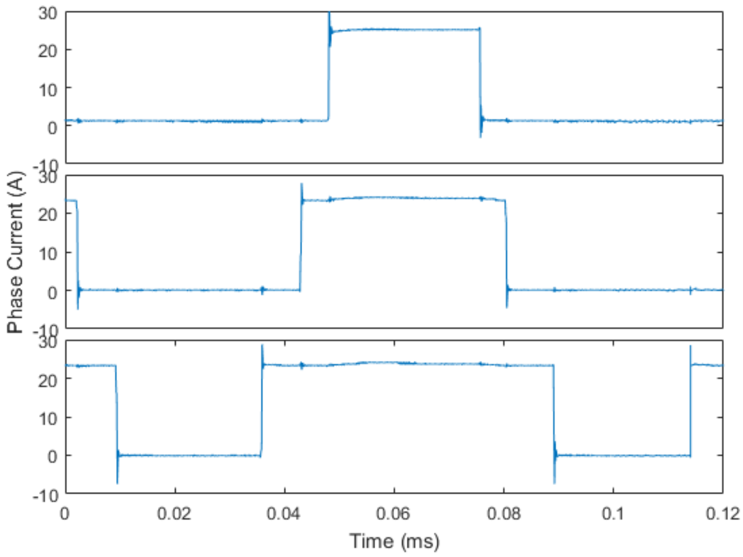
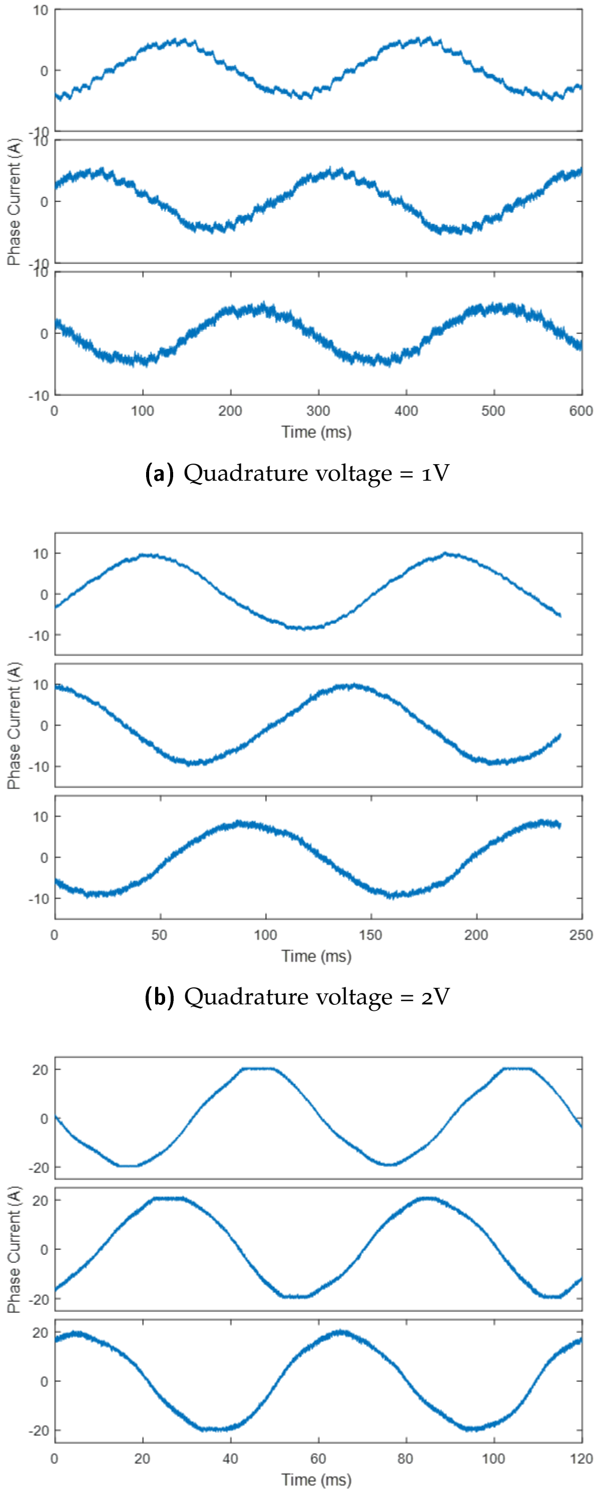


Figure 6.16: Center-aligned PWM voltage waveform



(a) Quadrature voltage = 1V

(b) Quadrature voltage = 2V

(c) Quadrature voltage = 5V

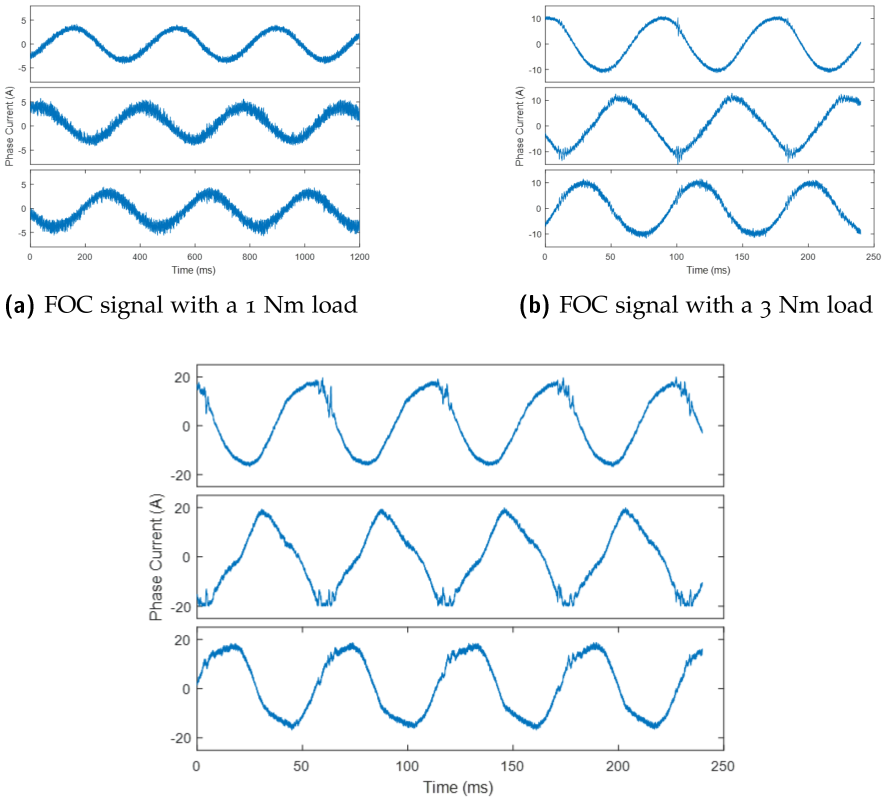
Figure 6.17: Sinusoidal drive current waveforms without load applied

6.4 field oriented control

These tests were applied by exciting the field of the DC generator to the maximum available ($30V$), so by setting up a reference quadrature current on the controller, the motor would reach a steady state speed when the torque applied by the load is equal to the torque applied by the motor.

The FOC loop is executed at $10kHz$ due to the limitation imposed by the Orbis encoder.

From these tests we can compare the torque measured with the torque applied by multiplying the amplitude of the current waveform by the torque constant of the motor K_T equal to $0.4506Nm/A$. By doing this, we see that there is a loss of about 30% of the torque on the tests with 1 and $3Nm$ loads, which can be due to mechanical losses and to measurement errors on the load torque. In the case when the load is $5Nm$ the torque loss is around 50%. We can identify that this current presents a distortion on its sinusoidal waveform and an offset of almost $1A$, which can be the reason of such power loss. This offset can be generated by the offset error in the current measurement of the $DRV8302$, which generates a wrong calculated current in the branch without shunt resistor, increasing the offset of the generated signal.



(a) FOC signal with a 1 Nm load

(b) FOC signal with a 3 Nm load

(c) FOC signal with a 5 Nm load

Figure 6.18: FOC with different loads

6.5 SPEED CONTROL WITH FIELD ORIENTED CONTROL

The behavior of the system with different reference speeds and load values can be appreciated from the remaining figures of this chapter.

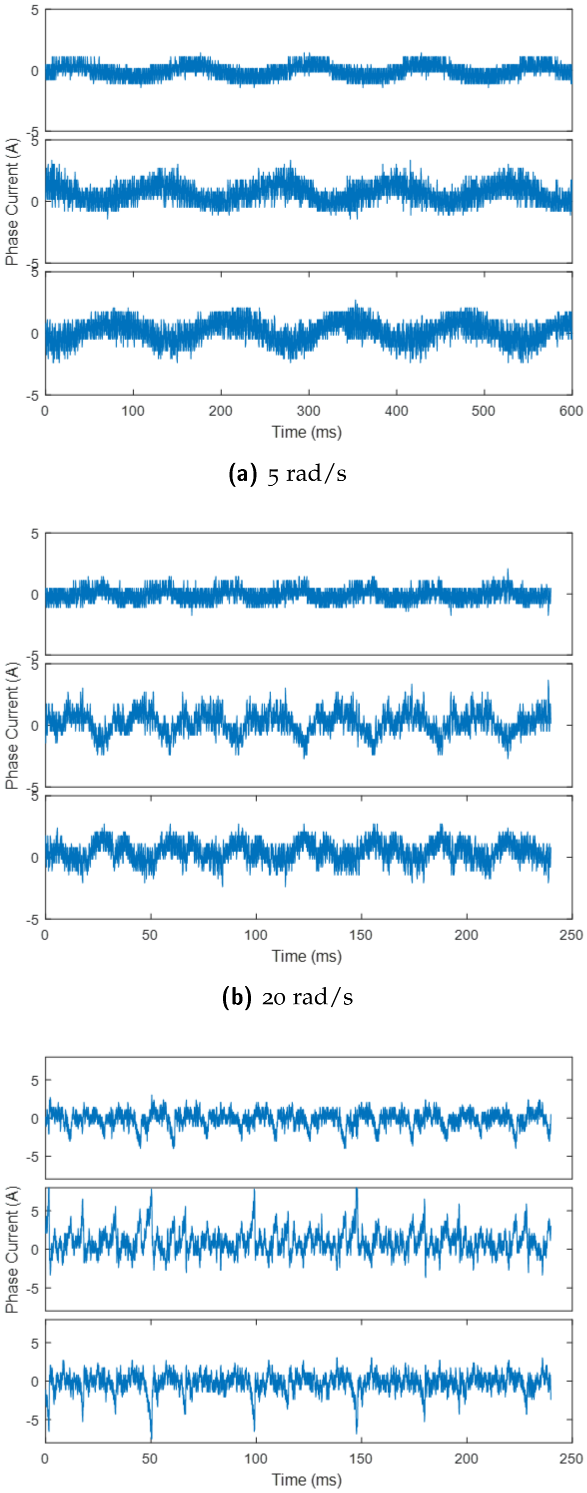
By analyzing the frequency of the waveform in Figure 6.19a we can see the ability of the controller to move the motor at a low speed with a load applied.

It can be appreciated in Figures 6.19c, 6.20c and 6.20d that the signal gets distorted when the speed reaches approximately $20 rad/s$. This is due to the maximum frequency achievable by the Orbis encoder. Currently it represents a trade-off between the choosing the trapezoidal driven speed control, which works fine at high speeds since the angular position feedback is given by the hall effect sensors, and the FOC driven speed control, which works fine at low speeds but at high speeds becomes unpredictable.

In order to have an idea of the linear speed achievable by the in-wheel motor, $20 rad/s$ is equivalent to $9.1 km/h$, which is approximately the human walking speed.

In Figure 6.22 we can see the comparison of the behavior of the controller with the same step signal (0 to $10 rad/s$) but with different load. When the load is $3Nm$, the slope of the signal becomes limited by the controller parameter `I_MAX`, which defines the saturation of the integral part of the controller to avoid current peaks. This parameter can be modified accordingly to the capacities of the system, but caution must be taken since this can provoke malfunctioning of the circuit.

In figure 6.24 we can appreciate the three phase currents settling into steady state after a step signal is commanded to the controller.

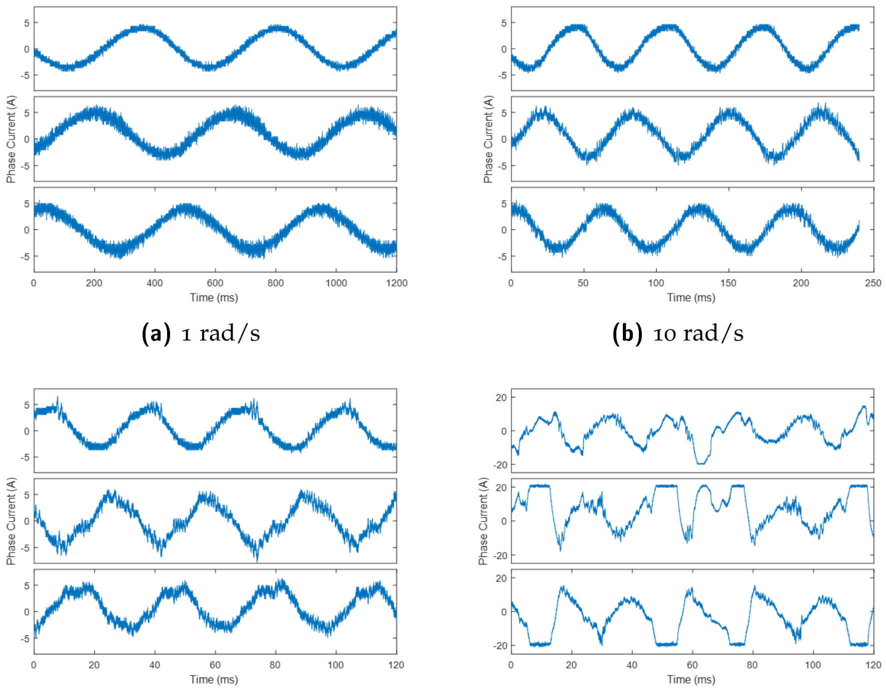


(a) 5 rad/s

(b) 20 rad/s

(c) 40 rad/s

Figure 6.19: FOC speed control loop without load applied



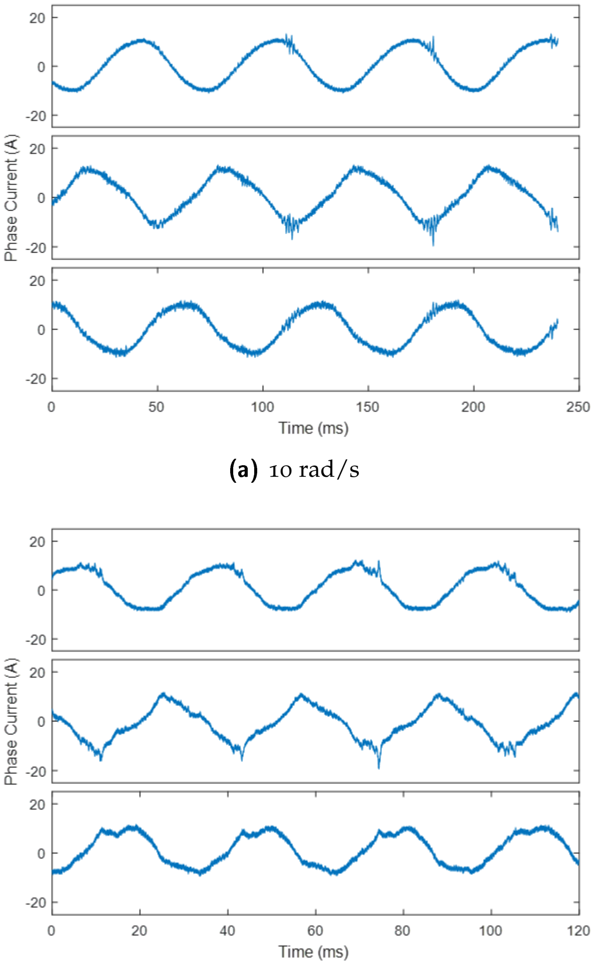
(a) 1 rad/s

(b) 10 rad/s

(c) 20 rad/s

(d) 30 rad/s

Figure 6.20: FOC speed control loop with 1 Nm load



(a) 10 rad/s

(b) 20 rad/s

Figure 6.21: FOC speed control loop with 3 Nm load

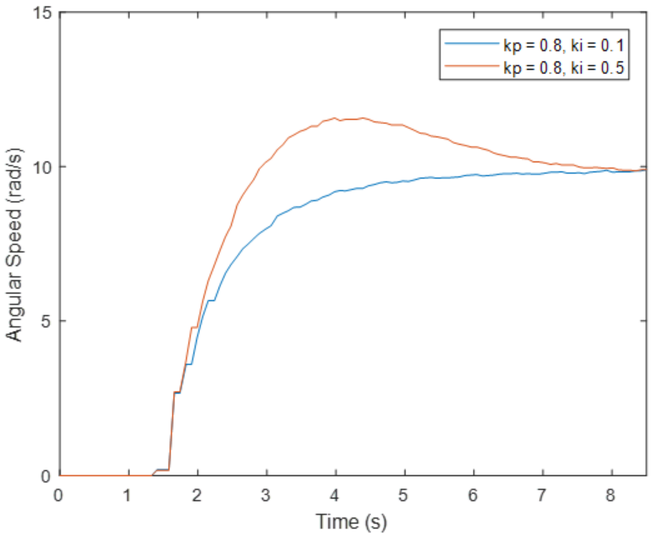


Figure 6.22: Step response to a 10 rad/s signal and a load of 1 Nm with different controller parameters

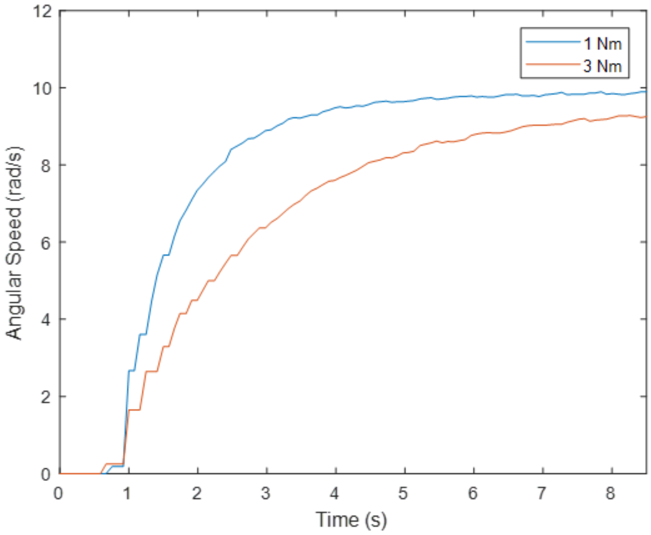
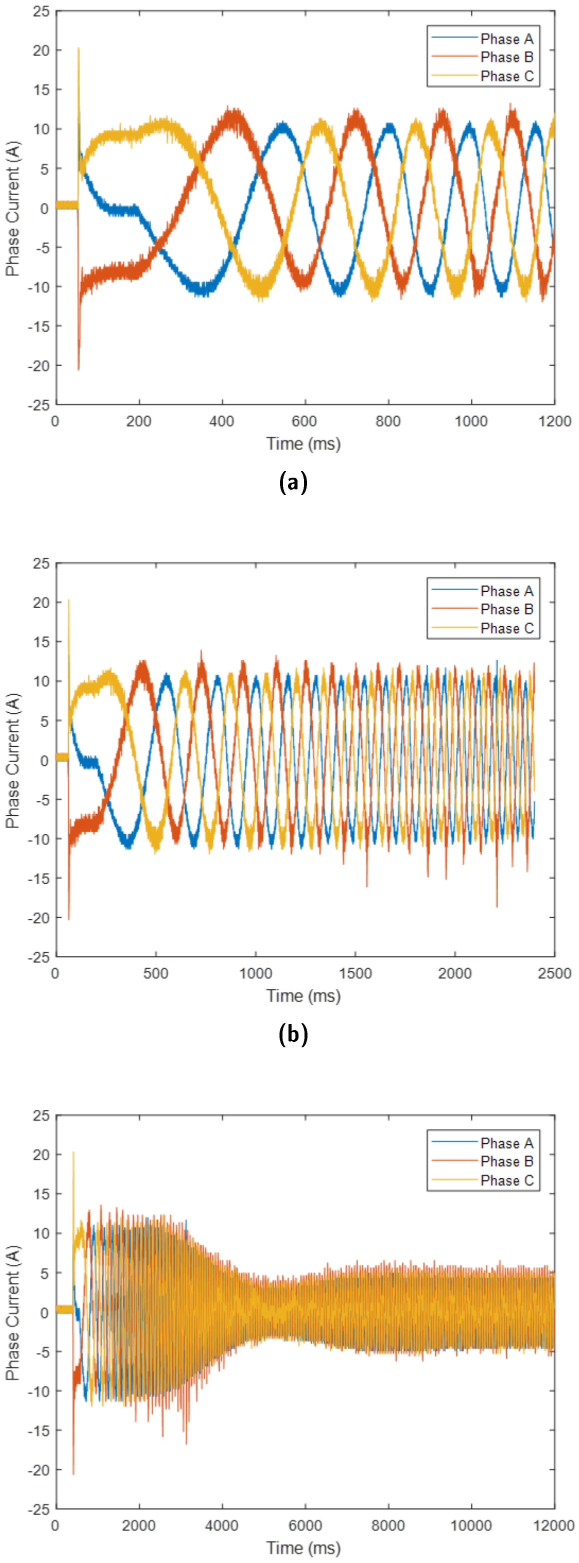


Figure 6.23: Step response to a 10 rad/s signal with different loads



(a)

(b)

(c)

Figure 6.24: FOC three-phase current waveform in a step response to a 10 rad/s signal with a load of 1 Nm

7 CONCLUSION

In this thesis we studied the feasibility of the implementation of the WMU of ROBI' based on the hardware design of the VESC board and on the implementation of the algorithms using the Miosix OS environment.

Since we were able to drive the in-wheel PMSM motor, we can conclude that the implementation of the driving algorithms on the WMU was successful. But, after analyzing the implementation procedure and the test results, we can also conclude that the overall architecture proposed is not optimal for such motor drive system.

The main positive aspects of the architecture implemented are the following:

- The driver *DRV*8302 is a useful tool since it solves all the issues related to MOSFET driving and the current provided to charge the gate is enough to have a good driving frequency, it protects the board from overcurrent and undervoltage conditions and it provides a voltage regulator, which reduces the area and the components on the board.
- The microcontroller capacity was not a limitation for the implementation of the algorithms required to drive the motor. Also, the use of Miosix as the operating system for the WMU simplified the development of those algorithms and the communication with the UI implemented in LabView.

The main issues with the implementation following this topology are the following:

- The driver *DRV*8302 provides only two shunt current amplifiers, which was one of the reasons of the generation of an offset in the driving signals in the implementation of the FOC, which reduces the efficiency of the motor.
- The model selected of the Orbis sensor becomes a limiting factor for the implementation of the FOC due to its maximum transmission frequency.

7.1 CHANGES PROPOSAL

Based on the previous analysis of the development of the WMU, we can propose the following changes for a future development:

- Since the only problem with the *DRV*8302 driver was that there are only two shunt current amplifiers, it can be replaced by a more recent version, the *DRV*8320, which includes the same functionalities but it considers three shunt current amplifiers.

- Due to the large currents flowing through the board, we need to use wires with a larger diameter than the one used in the developed system to avoid melting them. The problem with using wider wires is that it becomes mechanically unsafe to have them soldered directly to the board, so it is important to consider the use of connectors for the power supply and for the motor terminals.

- The last proposal is to change the model of the Orbis encoder to one with a higher transmission frequency to avoid the frequency limitation in the FOC algorithm.

7.2 FUTURE WORK

The implementation of some of the requirements for the WMU is still pending. One of those requirements is the communication between ECUs through the CAN communication protocol. As mentioned before, the drivers for this were provided by the Skyward Experimental Rocketry team, therefore, its implementation should be simple.

Another important point to work on, is the implementation of strategies to improve the efficiency of the motor, like the reduction of current offsets by using mean value control techniques, and the reduction of harmonic signals by the implementation of other driving methods, like the space vector modulation.

Finally, it is recommended to do a thermal analysis of the WMU, which is crucial in order to improve the robustness of the system. During the capability tests, it could be detected that many components increased their temperature, mainly the power MOSFETs. This temperature increase leads to a decrease of the power dissipation capability of the transistors, which might lead to a failure of the system. The temperature sensor of the board was enabled but no study was done regarding the change of temperature of the board in function of the operation conditions of the system. From such analysis, the need of implementation of a heat sink could be determined.

One solution to the implementation of a heat sink in a hardware layout like the one proposed in the VESC board could be the use of heat pipes. Heat pipes are devices that transfer heat by a process of evaporation and condensation of a fluid circulating within a sealed cavity (ESDU International plc, 2008). These devices are designed to remove heat from semiconductor devices and they are normally found in computing systems due to the small volume required for their implementation.

BIBLIOGRAPHY

Arrigo, Jason (2006). *Input and Output Capacitor Selection*. Texas Instruments (cit. on p. 43).

Bascetta, Luca, Marco Baur, and Giambattista Gruosso (2017a). "Electrical Unmanned Vehicle Architecture for precision farming applications." In: (cit. on p. 37).

— (2017b). "ROBI': A Prototype Mobile Manipulator for Agricultural Applications." In: *MDPI Electronics* (cit. on pp. 2, 6, 37).

BASIL Networks, PLLC (2003). *Permanent Magnet Brushless Motor*. URL: `https://www.basilnetworks.com/article/motors/brushlessmotors.html` (cit. on p. 18).

Bose, B. K., T. M. Jahns, R. D. Lorenz, G. Pfaff, P. Pillay, M. H. Rashid, T. Sebastian, and G. R. Slemon (1989). *Performance and Design of Permanent Magnet AC Motor Drives*. IEEE (cit. on p. 9).

Doppelbauer, Martin (2012). *The Invention of the Electric Motor*. URL: `https://www.eti.kit.edu/english/1376.php` (visited on 02/28/2012) (cit. on p. 1).

ESDU International plc (2008). *Heat pipes – general information on their use, operation and design* (cit. on p. 98).

Ghioni, Massimo (2016a). *Introduction to Electrical Drives*. URL: `ftp://ftp.elet.polimi.it/users/Massimo.Ghioni/Power%20Electronics%20/Motor%20control/motor%20control%20overview/INTRODUCTION%20TO%20ELECTRICAL%20DRIVES.pdf` (cit. on p. 20).

— (2016b). *Lecture notes in Power Electronics* (cit. on pp. 9, 21, 48).

International Rectifier (2015). *IRFS7530-7PPbF Datasheet* (cit. on p. 45).

Jordan, Edward (1968). *Electromagnetic Waves and Radiating Systems*. Prentice-Hall (cit. on p. 11).

Lawrenz, W. and N. Obermöller (2011). *CAN: Controller Area Network: Grundlagen, Design, Anwendungen, Testtechnik*. VDE Verlag (cit. on p. 53).

Magnani, GianAntonio, Gianni Ferretti, and Paolo Rocco (2007). *Tecnologie dei sistemi di controllo*. McGraw-Hill (cit. on pp. 9, 11, 13, 64).

Maxim Integrated (2016). *Shoot-Through Current*. URL: `https://www.maximintegrated.com/en/glossary/definitions.mvp/term/Shoot-Through%20Current/gpk/1040` (cit. on p. 60).

Micro-Chip Technologies (2010). *NTC Thermistors*. URL: `http://www.microchiptechno.com/ntc_thermistors.php` (cit. on p. 54).

RS Components Ltd. (2012). *PCB Connectors Overview*. URL: `https://uk.rs-online.com/web/generalDisplay.html?id=solutions%2Fpcb-connectors` (cit. on p. 44).

Seymour Papert (1991). *Constructionism*. Ablex Publishing Corporation (cit. on p. xix).

STMicroelectronics NV (2016a). *RM0090 Reference Manual* (cit. on p. 59).

— (2016b). *STM32F PMSM single/dual FOC* (cit. on p. 61).

Terraneo, Federico (2014). *Miosix*. URL: `https://miosix.org/` (cit. on p. 58).

Tesla, Nikola (1897). *On Electricity*. Electrical Review (cit. on p. 1).

The Engineering ToolBox (2015). *Wire Gauges - Current Ratings*. URL: `https://www.engineeringtoolbox.com/wire-gauges-d_419.html` (cit. on p. 45).

Vedder, Benjamin (2015). *VESC – Open Source ESC*. URL: `http://vedder.se/2015/01/vesc-open-source-esc/` (cit. on pp. 41, 42).

Yedamale, Padmaraja (2003). *Brushless DC (BLDC) Motor Fundamentals*. Microchip Technology Inc. (cit. on pp. 17, 28, 29).

A

APPENDIX A: VESC BOARD SCHEMATIC FILES

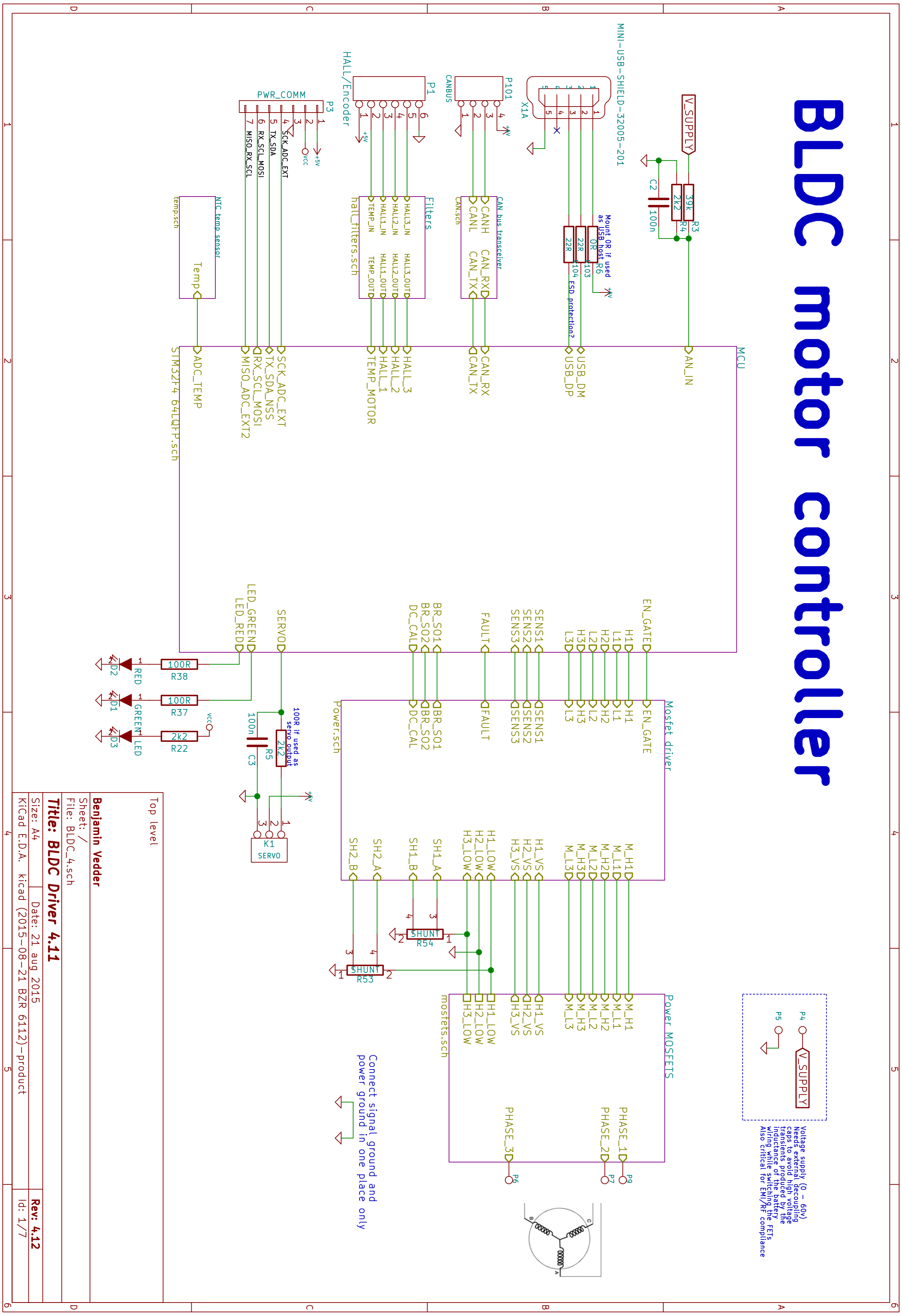


Figure A.1: Schematic Overview

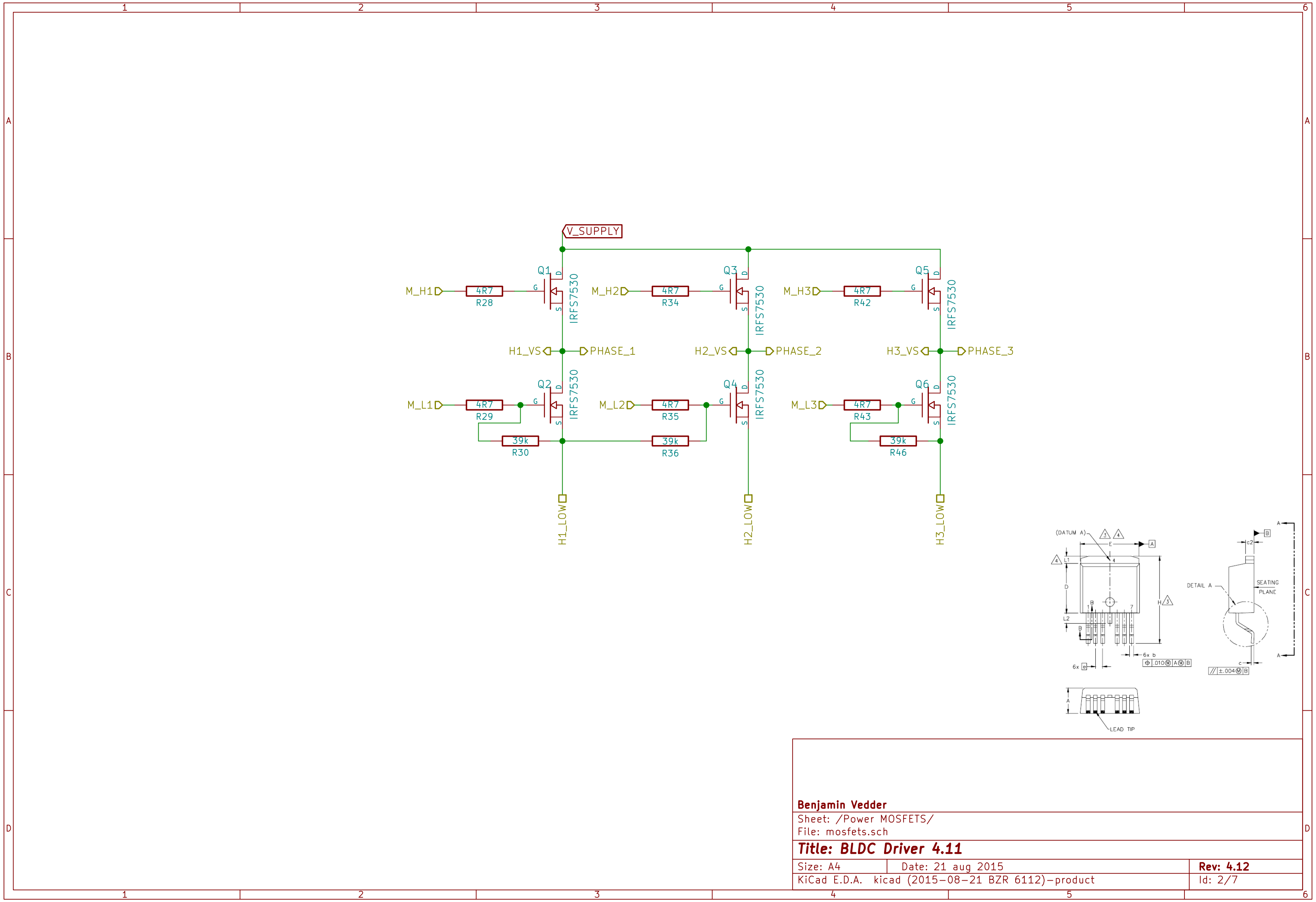


Figure A.2: Three-Phase MOSFET Inverter

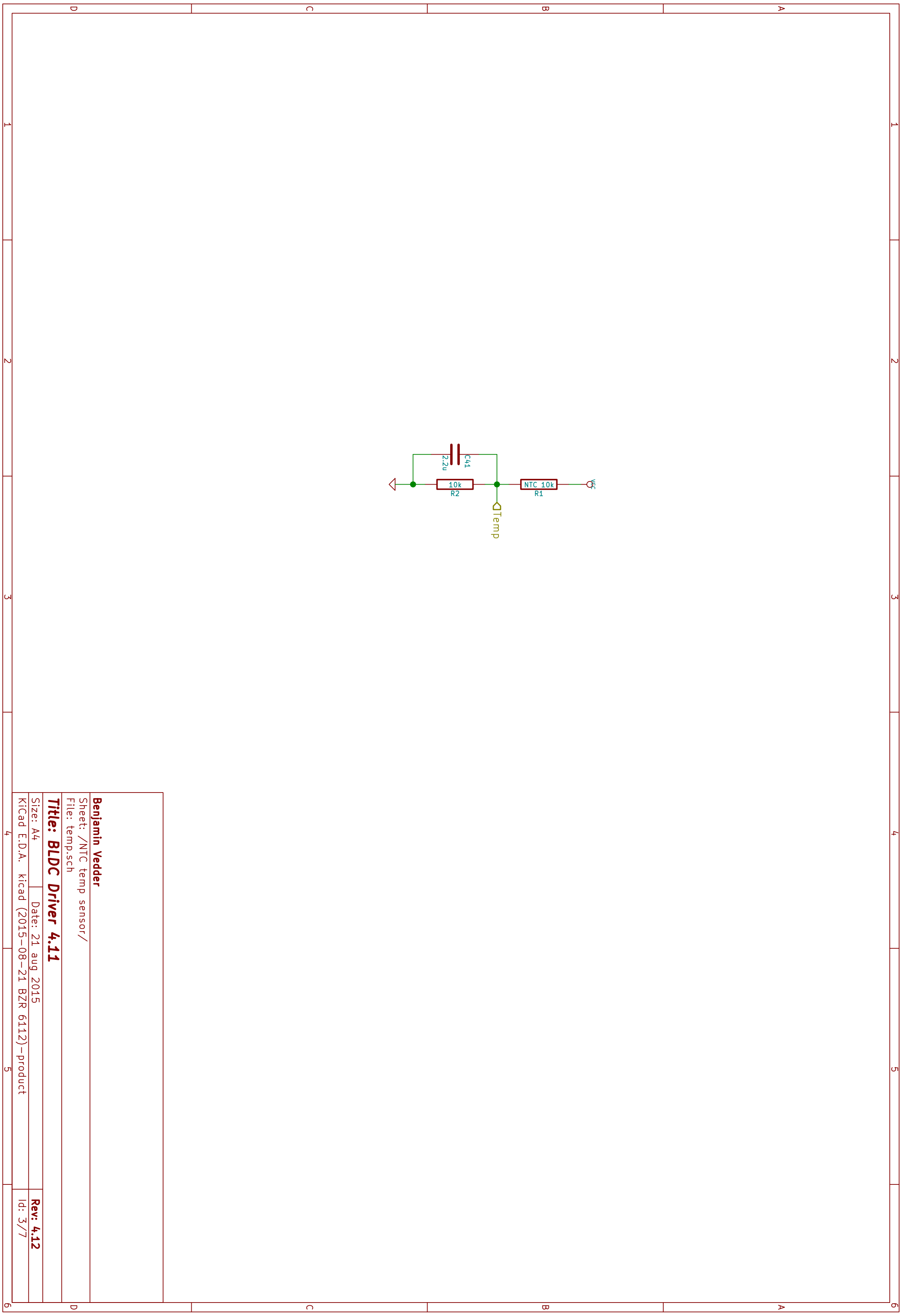


Figure A.3: In-Board Temperature Sensor

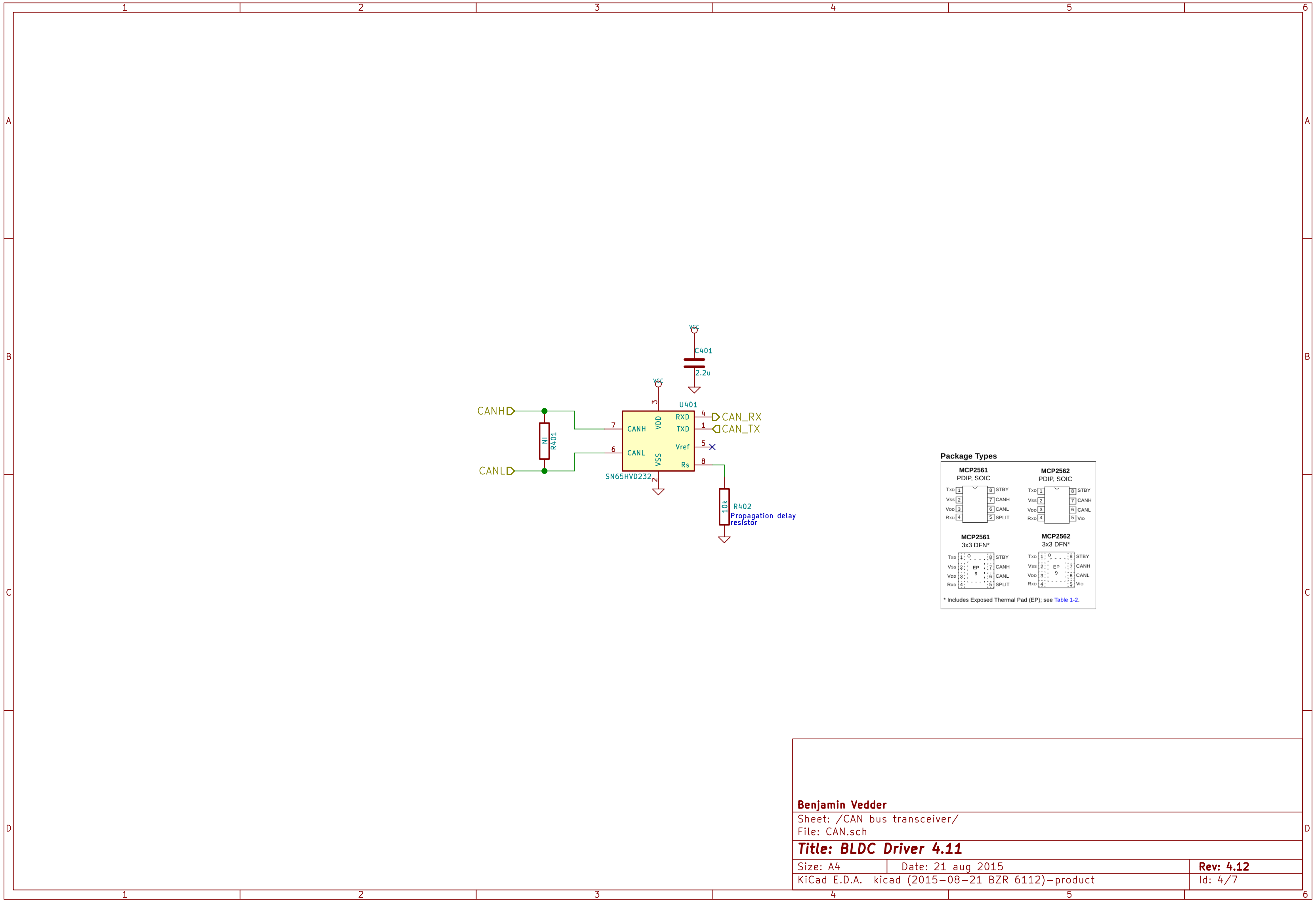


Figure A.4: CAN Bus Transceiver

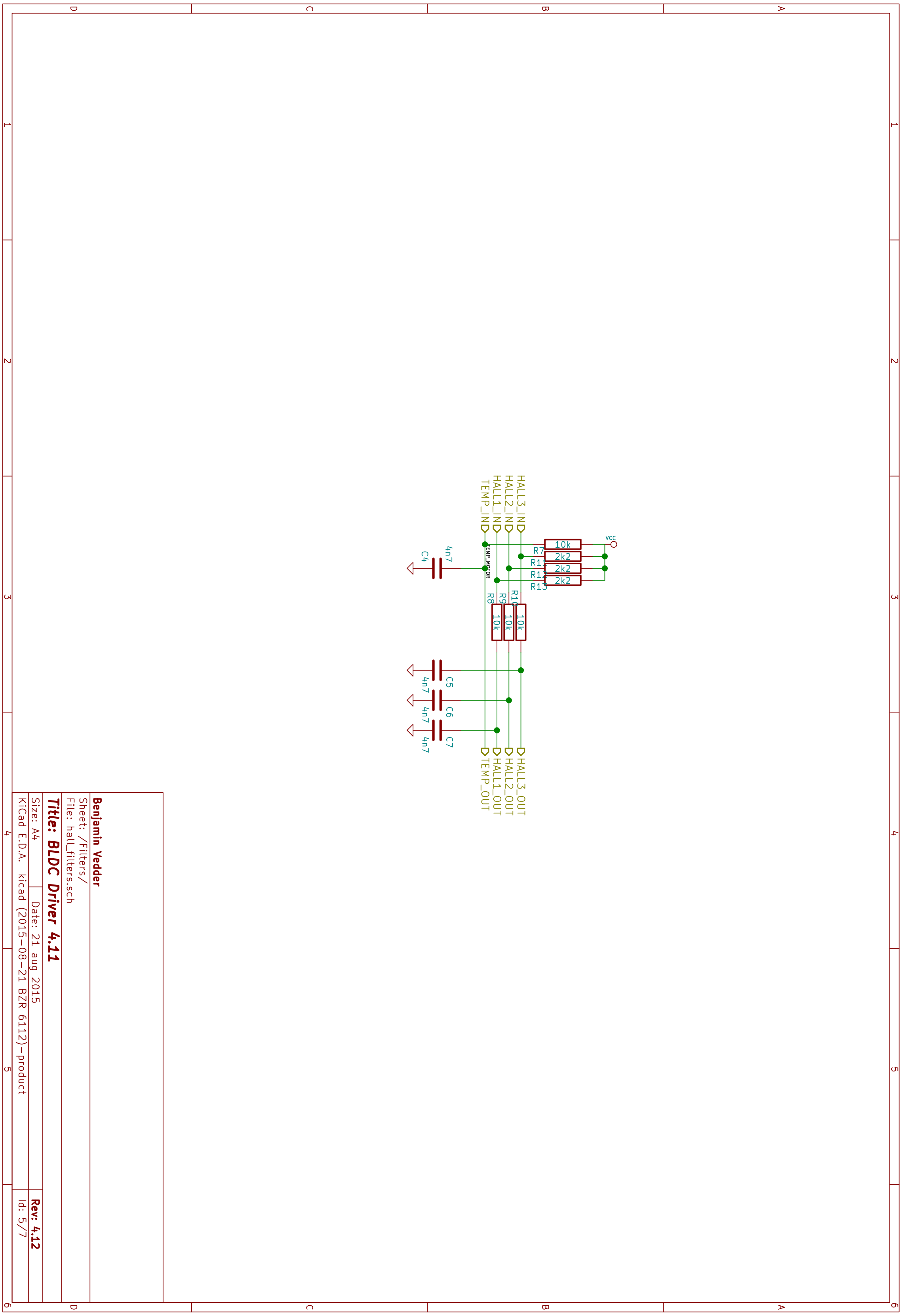


Figure A.5: Hall Effect Sensors Signal Conditioning

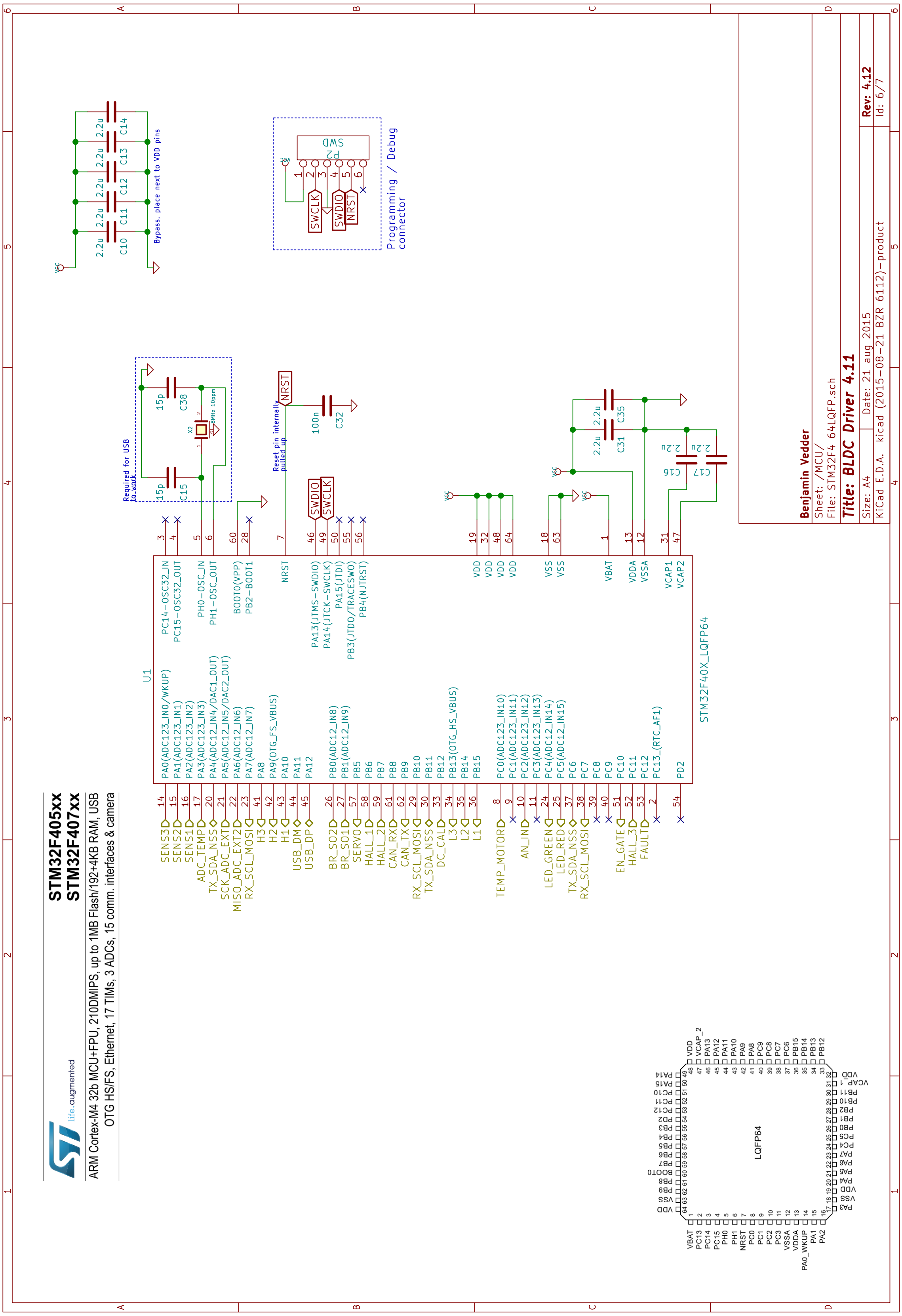
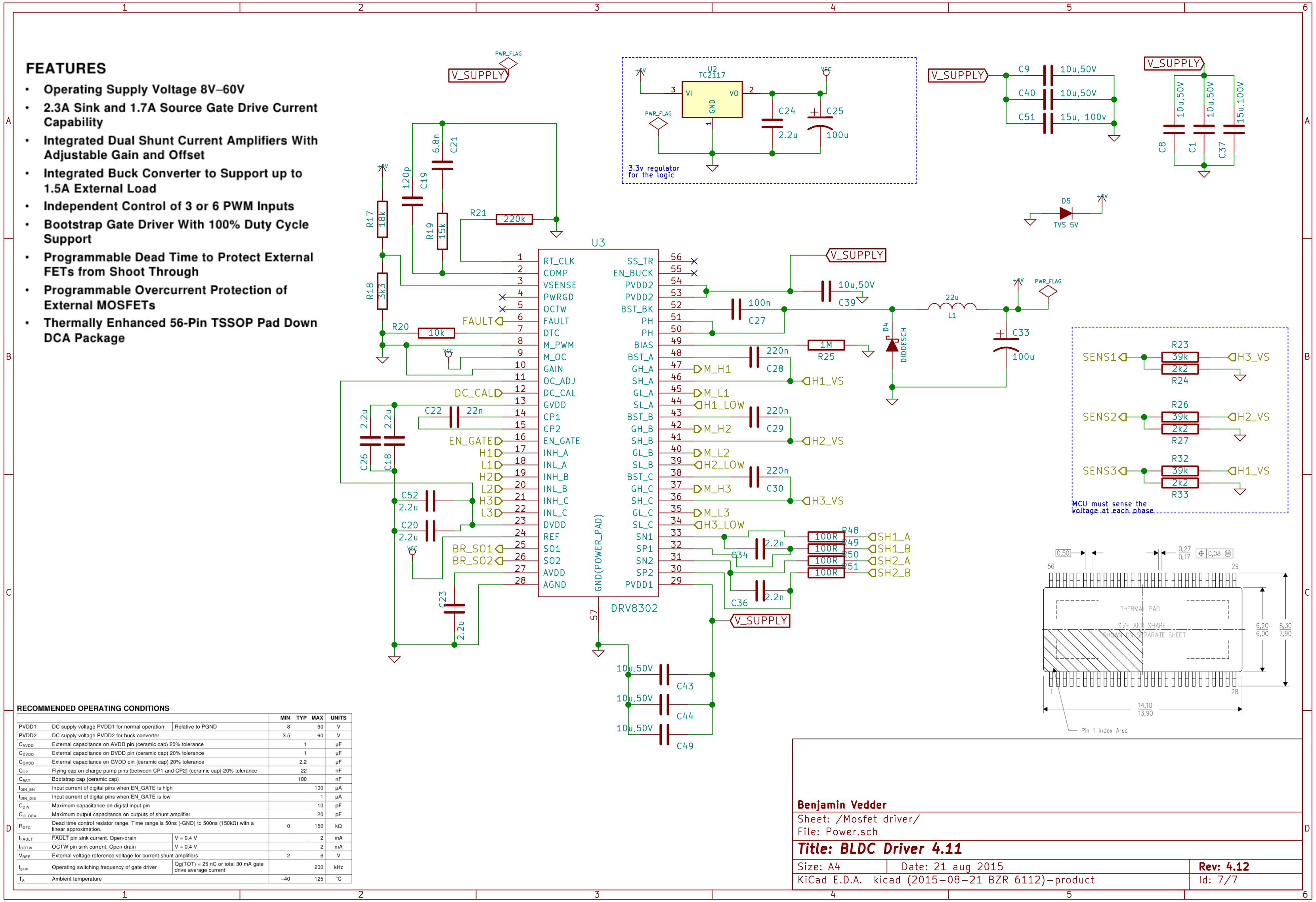


Figure A.6: STM32F405 Hardware Setup



FEATURES

- Operating Supply Voltage 8V–60V
- 2.3A Sink and 1.7A Source Gate Drive Current Capability
- Integrated Dual Shunt Current Amplifiers With Adjustable Gain and Offset
- Integrated Buck Converter to Support up to 1.5A External Load
- Independent Control of 3 or 6 PWM Inputs
- Bootstrap Gate Driver With 100% Duty Cycle Support
- Programmable Dead Time to Protect External FETs from Shoot Through
- Programmable Overcurrent Protection of External MOSFETs
- Thermally Enhanced 56-Pin TSSOP Pad Down DCA Package

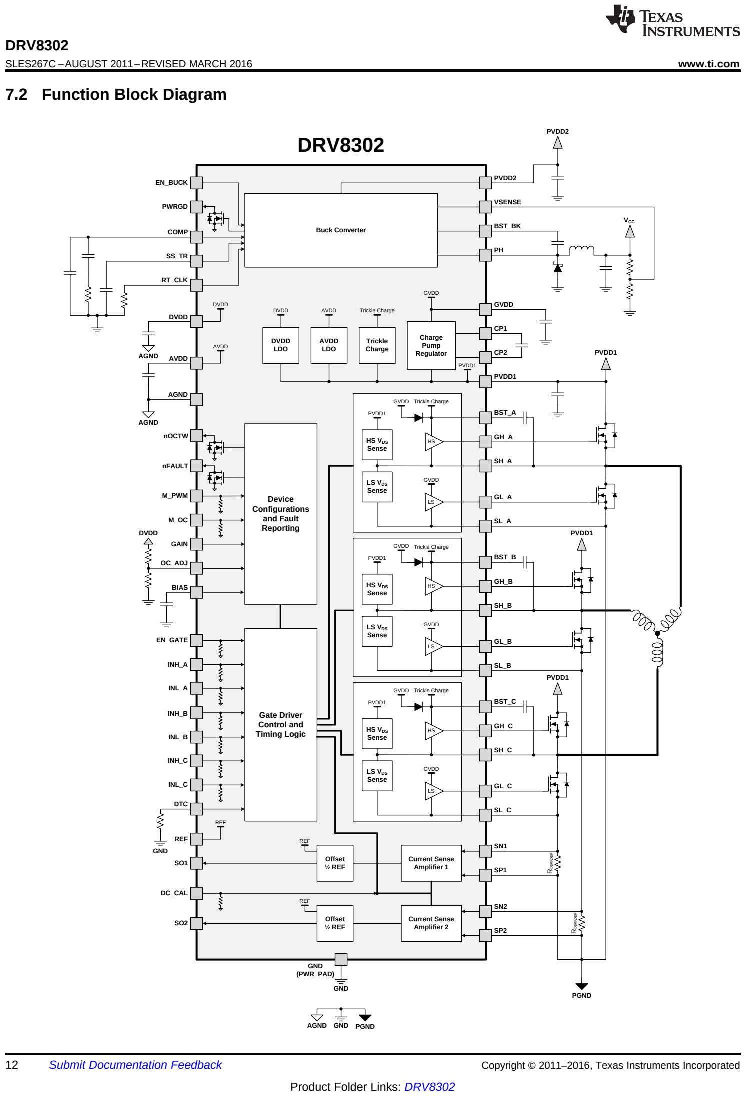
RECOMMENDED OPERATING CONDITIONS

			MIN	TYP	MAX	UNITS
PVDD1	DC supply voltage PVDD1 for normal operation	Relative to PGND	8		60	V
PVDD2	DC supply voltage PVDD2 for buck converter		3.5		60	V
C_{AVDD}	External capacitance on AVDD pin (ceramic cap) 20% tolerance			1		µF
C_{DVDD}	External capacitance on DVDD pin (ceramic cap) 20% tolerance			1		µF
C_{GVDD}	External capacitance on GVDD pin (ceramic cap) 20% tolerance			2.2		µF
C_{CP}	Flying cap on charge pump pins (between CP1 and CP2) (ceramic cap) 20% tolerance			22		nF
C_{BST}	Bootstrap cap (ceramic cap)			100		nF
I_{DIN_EN}	Input current of digital pins when EN_GATE is high				100	µA
I_{DIN_DIS}	Input current of digital pins when EN_GATE is low				1	µA
C_{DIN}	Maximum capacitance on digital input pin				10	pF
C_{O_OPA}	Maximum output capacitance on outputs of shunt amplifier				20	pF
R_{DTC}	Dead time control resistor range. Time range is 50ns (-GND) to 500ns (150kΩ) with a linear approximation.		0		150	kΩ
I_{FAULT}	$\overline{\text{FAULT}}$ pin sink current. Open-drain	V = 0.4 V			2	mA
I_{OCTW}	$\overline{\text{OCTW}}$ pin sink current. Open-drain	V = 0.4 V			2	mA
V_{REF}	External voltage reference voltage for current shunt amplifiers		2		6	V
f_{gate}	Operating switching frequency of gate driver	Qg(TOT) = 25 nC or total 30 mA gate drive average current			200	kHz
T_A	Ambient temperature		–40		125	°C

Figure A.7: DRV8302 MOSFET Driver

APPENDIX B: DRV8302 FUNCTION BLOCK DIAGRAM

B



DRV8302

SLES267C – AUGUST 2011 – REVISED MARCH 2016

www.ti.com

7.2 Function Block Diagram

12 Submit Documentation Feedback

Copyright © 2011–2016, Texas Instruments Incorporated

Product Folder Links: DRV8302

Figure B.1: DRV8302 Function Block Diagram

C

APPENDIX C: MEASUREMENT BOARD SCHEMATICS

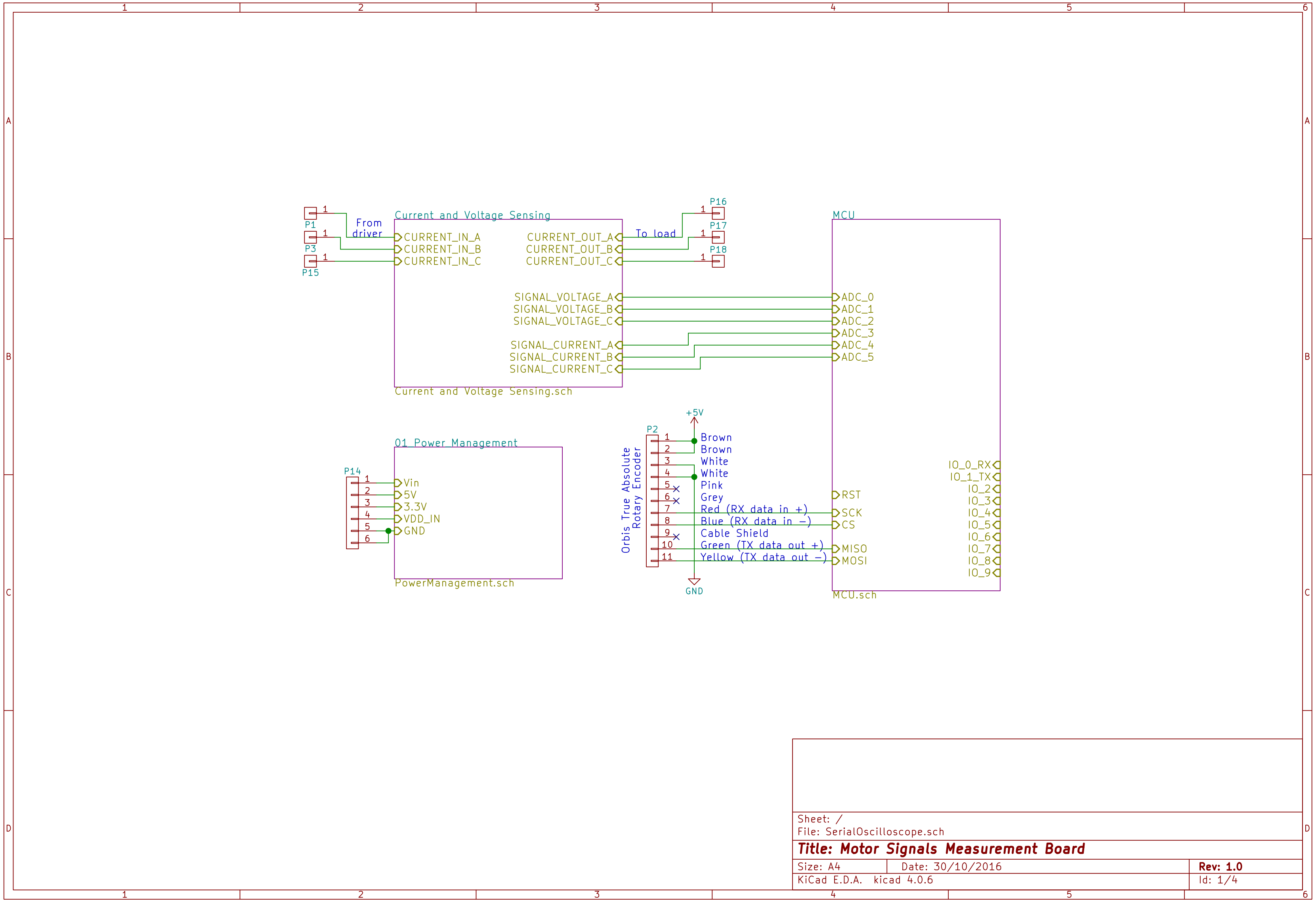


Figure C.1: Schematic Overview

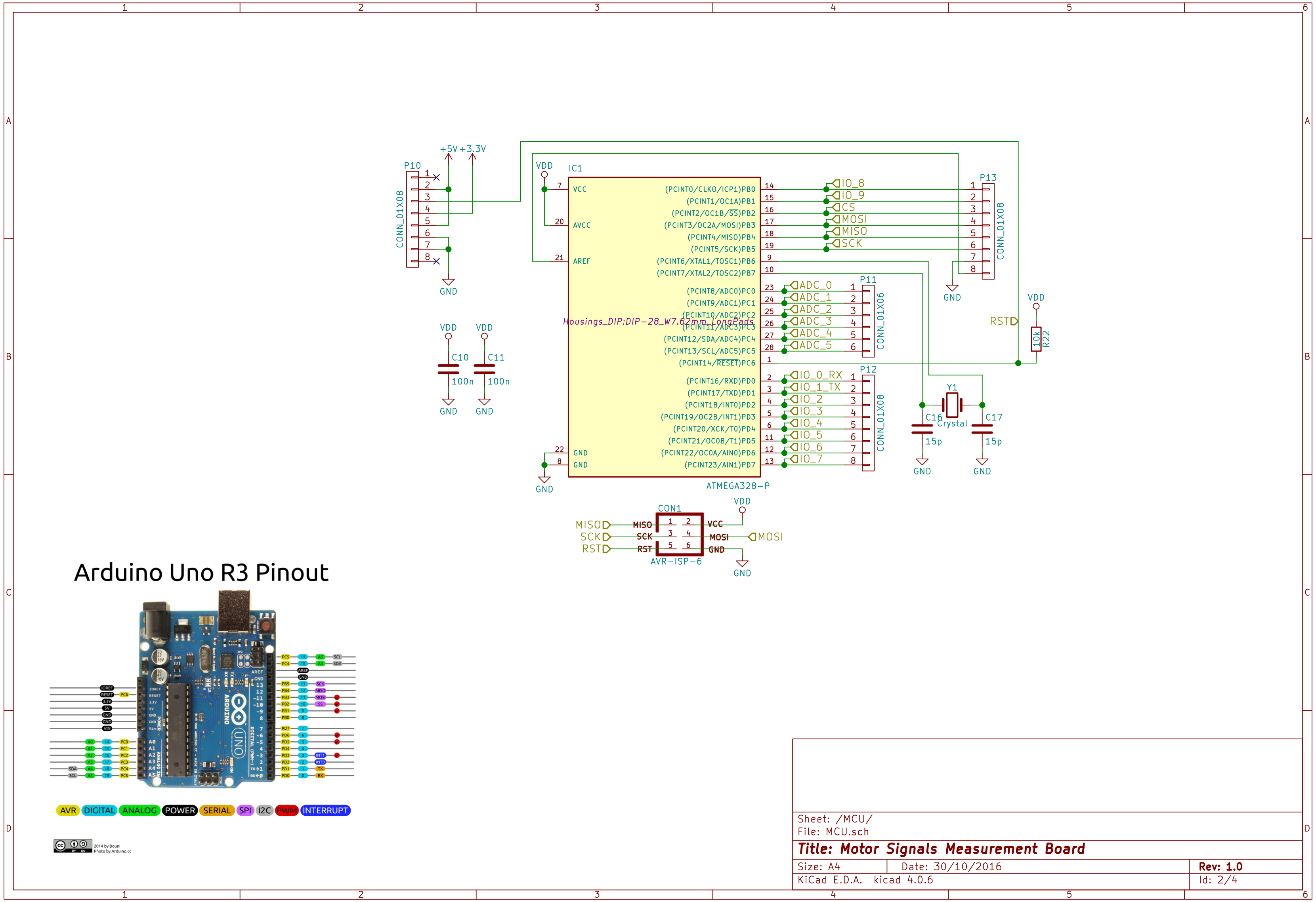


Figure C.2: Embedded system attachment pinout

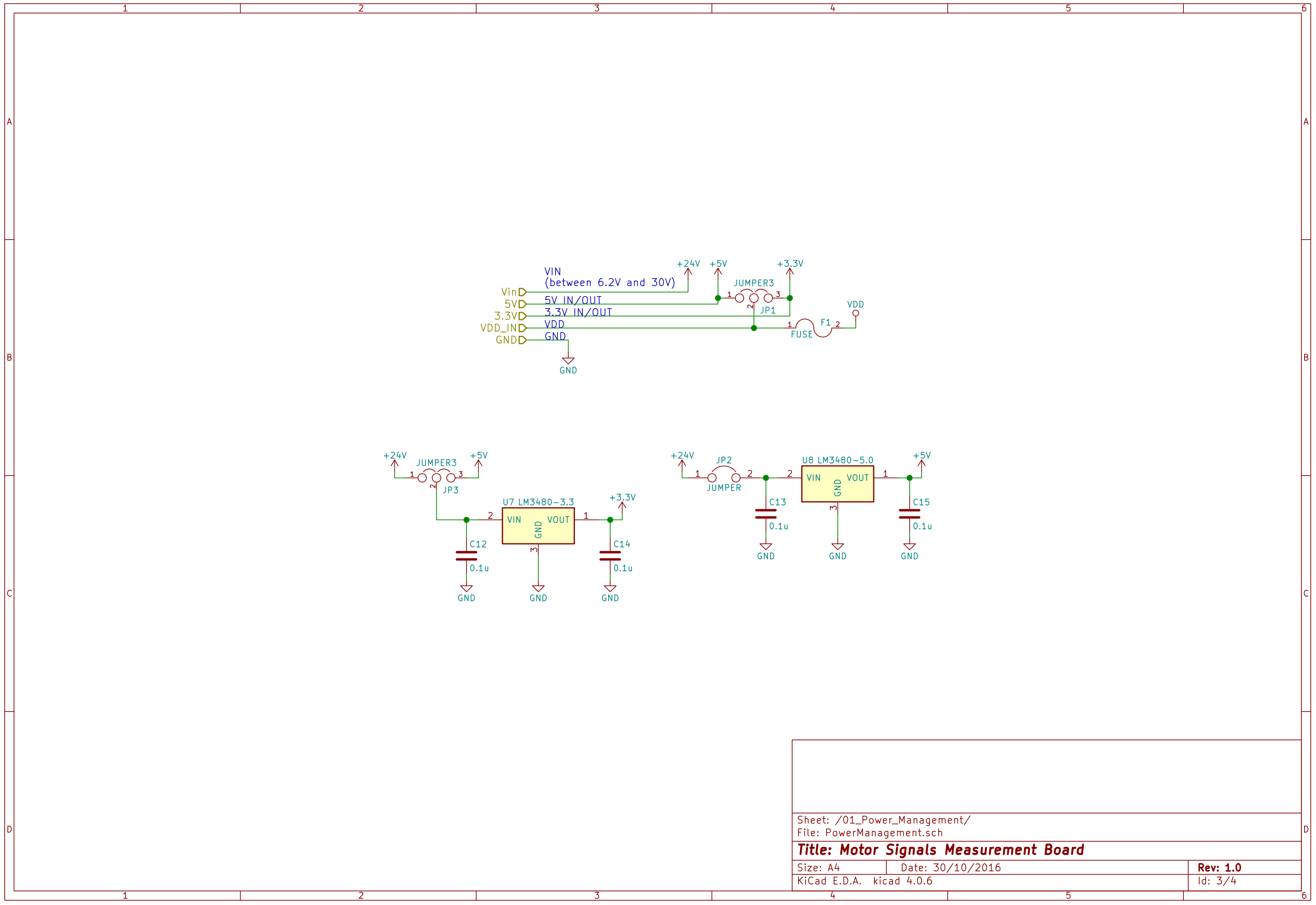


Figure C.3: Power management of the board

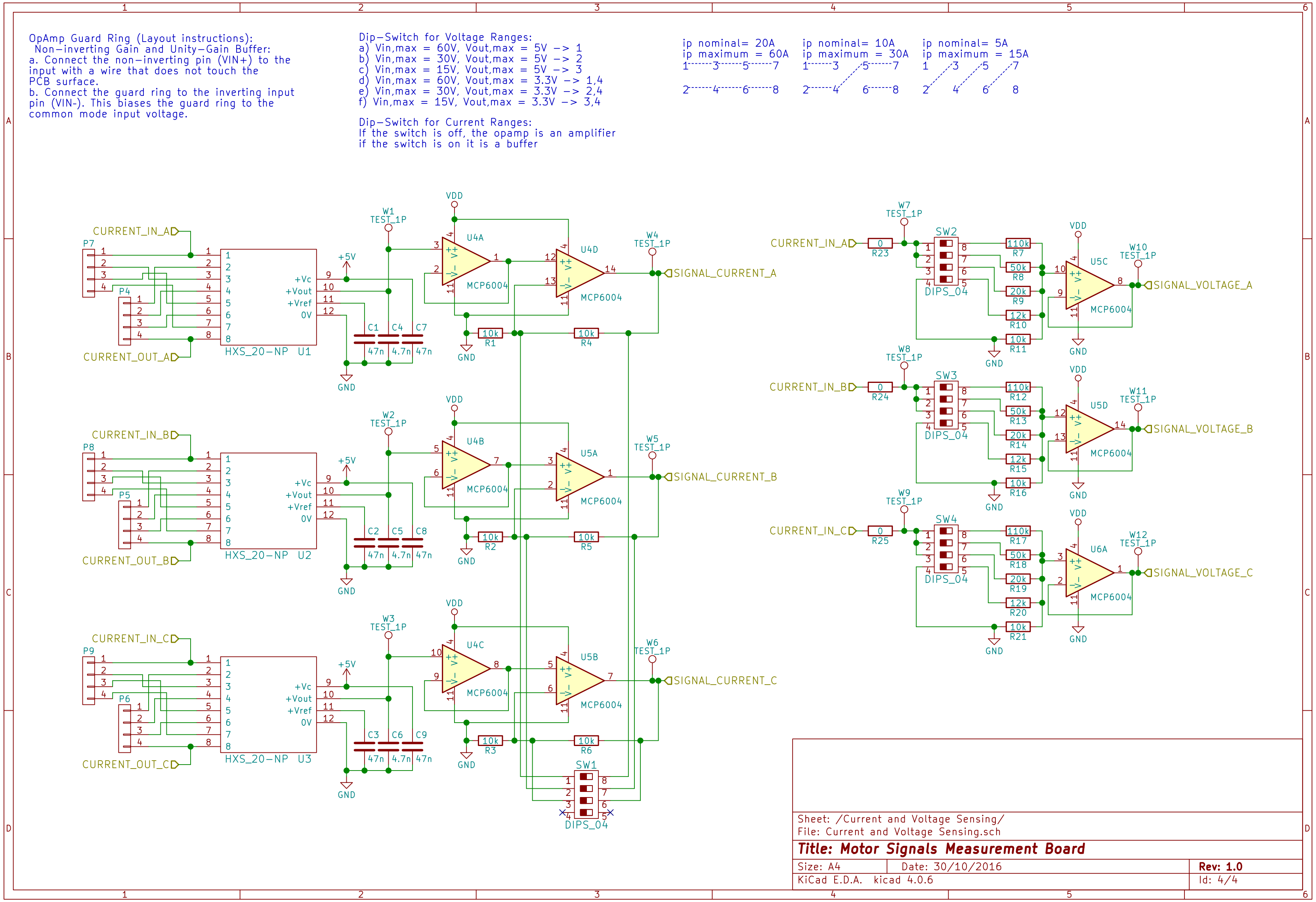


Figure C.4: Voltage and current signal conditioning circuits