University of Southern Denmark

MSc in Engineering - Electronics
1. Semester Project

Control of PMAC Motor for an Electric Go-Kart

Authors:
230390 Martin Brøchner Andersen
240289 Morten Tholstrup Pedersen
030192 Mikkel Skaarup Jaedicke
211185 Erlingur Ívar Jóhannsson
100589 Thomas S. Christensen

Supervisors:
Jacob Lykke Pedersen
Karsten Holm Andersen

Date: 31-05-2016

Preface

This report is written in partial fulfilment of the first semester of the masters programme in electrical engineering at the University of Southern Denmark. Due to time constraints it was decided to collaborate across all the students in the class. For this reason most of the electrical circuitry was developed by the authors of this report whereas the power electronics was mostly devised by the other group. Regardless, all aspects of the project will be discussed throughout the report.

Acknowledgment

We would like to thank our supervisors, Karsten Holm Andersen and Jacob Lykke Pedersen for the invaluable assistance and patience they have exerted throughout the semester. A word of thanks goes to our lecturers; Morten Nymand who has, on numerous occasions, provided assistance in the technical aspects of board layout and power electronics, and Jørgen Christian Larsen for saving countless hours of frustration with his assistance on Xilinx and all of the problems that follow. Additionally, for helping with practical aspects of the project including layout and sourcing of components, Thanks to Jesper Nielsen.

Martin Brøchner Andersen

Morten Tholstrup Pedersen

Mikkel Skaarup Jaedicke

Erlingur Ívar Jóhannsson

Thomas Søndergaard Christensen

The report, source code, data, plotting script and simulations can be found at:

`github.com/mikkeljae/SEM1PRO_ELECTRONICS`

Abstract

The topic of motor control for electric vehicles is a fast growing industry both in terms of academic studies and in terms of economy. This report describes the process of creating a three phase inverter and a controller for an electrical go-kart. This process requires work in a variety of fields, including: electronics design, embedded design, control theory and power electronics. Extensive simulation as well as calculations on the system was done in order to more accurately determine the necessary components to be used in the inverter. The embedded system containing the controller, PWM generator and various other essential components were developed using Xilinx tools, such as System Generator. It was made possible to drive a smaller PMAC motor using the controller and inverter developed, however certain parts of the project remain unsolved. Making the electronics function properly was abandoned as there was insufficient time to spin another board. Much of the testing and verification of the system has also not been sufficiently examined. This, unfortunately, includes testing the inverter on the go-kart.

Contents

1 Introduction

Power electronics is currently a thriving field in engineering. With more and more of the energy used coming from electric sources, it becomes increasingly important to be able to convert and transmit electricity as efficiently as possible. This process involves not only highly efficient power inverters, but also sophisticated control of these components. Throughout this report the process of making an inverter and control system for an electric go-kart is described. This involves the development of an embedded system using the Zybo board as a platform as well as developing electronics for interfacing sensor equipment, manage safety procedures and supply of power to various parts of the system. Driving a PMAC motor is a high-current task which will require the use of novel circuitry-techniques in order to minimize parasitics caused by switching throughout the inverter. Minimizing parasitics is a crucial part of making any inverter efficient as any uncontrolled harmonics in the system will likely cause losses, thereby creating heat, increasing the size of the cooling required and, by increasing cost. Additionally, several methods for determining the transfer function of the system in order to properly design a controller are explored. Correctly modelling the system is an incredible advantage if high efficiency is the goal as a proper model allows the engineer to quickly, easily and perhaps most important, cheaply, test different hypothesis without relying on a physical model. In an effort to create as accurate a model as possible, an in depth analysis of the system is done using simulations of the various parts of the system, as well as a simulation of the complete system.

2 Requirements

The project description given by the supervisors outlines the requirements that are set for this project. They are defined to utilize the given hardware, and to limit the project in scope. The requirements are as follows:

- Design and implement a motor controller to replace the current Sevcon gen4 [16] motor controller used in the SDU electric go-kart.
- The motor controller must use the same connection types for easy replacement of the control and driver solutions.
- The motor controller is required to be able to control the PMAC motor in the forward direction.
- The digital part of the control of the PMAC motor must be handled using the Zynq-7010 present on the Zybo board.
- The design of the three phase inverter necessary to control the PMAC motor is part of the project
- It must be possible to set and get relevant parameters through a communication interface by a computer.
- All sensors and switches available on the go-kart must be interfaced and usable to their desired effect.
- It is not allowed to change any mechanical or electrical parts on the go-kart, except replacing the Sevcon gen4 controller.

These requirements will be the base of the solution presented in this report.

3 Hardware Description

Many parts are involved in making the go-kart drive. This section will give a short description of each part in order to make an overview of the system.

3.1 Go kart Frame

The Go-Kart itself consists of a pre assembled aluminium frame with wheels, steering column, break pedal and seat all pre-attached to the frame. A plate is attached behind the seat to fit all of the power electronics and drive systems that are designed throughout this project. To the right of the driver a mount is positioned for the motor. The mount is protected by a metal cover. In order to transfer power to the rear wheels, a gearing is placed between the motor and the rear axle.
A conventional disc brake comes pre-installed on the go-kart with the break pedal connected using hydraulics. The speeder pedal is also installed, but with no mechanical effect on the system.

Figure 3.1: The frame of the Go-kart used in the project.

3.2 Battery Supply

The batteries provided for this project are of the type SB12V20P-FC [17]. Four of these batteries are put in series, yielding a combined nominal voltage of 52.8V and a discharge current of 560A. The discharge current can go as high as 1200A for one second. The batteries will be providing all the required power for the go-kart, including the motor and all control and drive circuits.

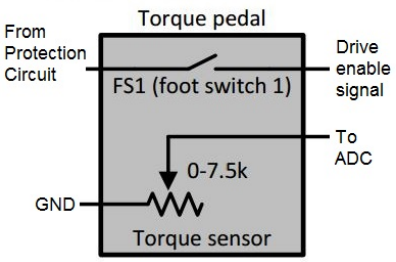
3.3 PMAC Motor

The PMAC motor provided is a three phase motor capable of producing 6hp continuously and a peak power of 19hp for a minute, at a maximum rotational speed of 5000rpm. It has an efficiency of 90% and a stall torque of 38Nm. Its typical uses are, among other things, marine, robotics and go-karts.

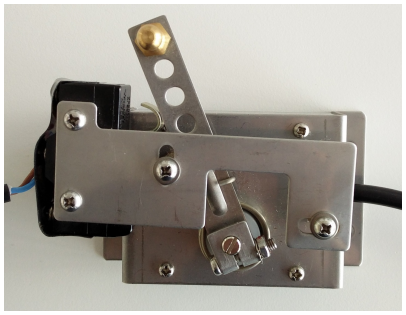
3.4 Torque Pedal

The foot pedal or "speeder" will be used to control the torque produced by the motor and will henceforth be referred to as a torque pedal. It consists of a levered variable resistor with the

electrical diagram seen in figure 3.2a.
The torque pedal will be used to control the torque by altering the resistance, resulting in a change in voltage on the return wire. By measurement, it is found to have a variable resistance of 0kΩ to 7.5kΩ. By nature a variable resistor is noisy therefore filtering the signal will be necessary.



(a) Diagram of the torque pedal.



(b) Picture of the actual torque pedal.

Figure 3.2: Diagram and picture of the torque pedal.

3.5 Switches and Wiring

The wiring of the go-kart frame is not pre-assembled, but has been provided by the supervisors of the project. A diagram of the wiring can be seen on figure 3.3.

The diagram shows the Sevcon Gen4 controller as the centerpiece surrounded by the other active parts. The motor to the right. The torque pedal, battery, undervoltage protection and multiple switches to the left. The switches include a key switch to turn the system on, an emergency switch, and a drive enable switch with three positions to change between driving forwards, reversing or disabling drive at the neutral position.

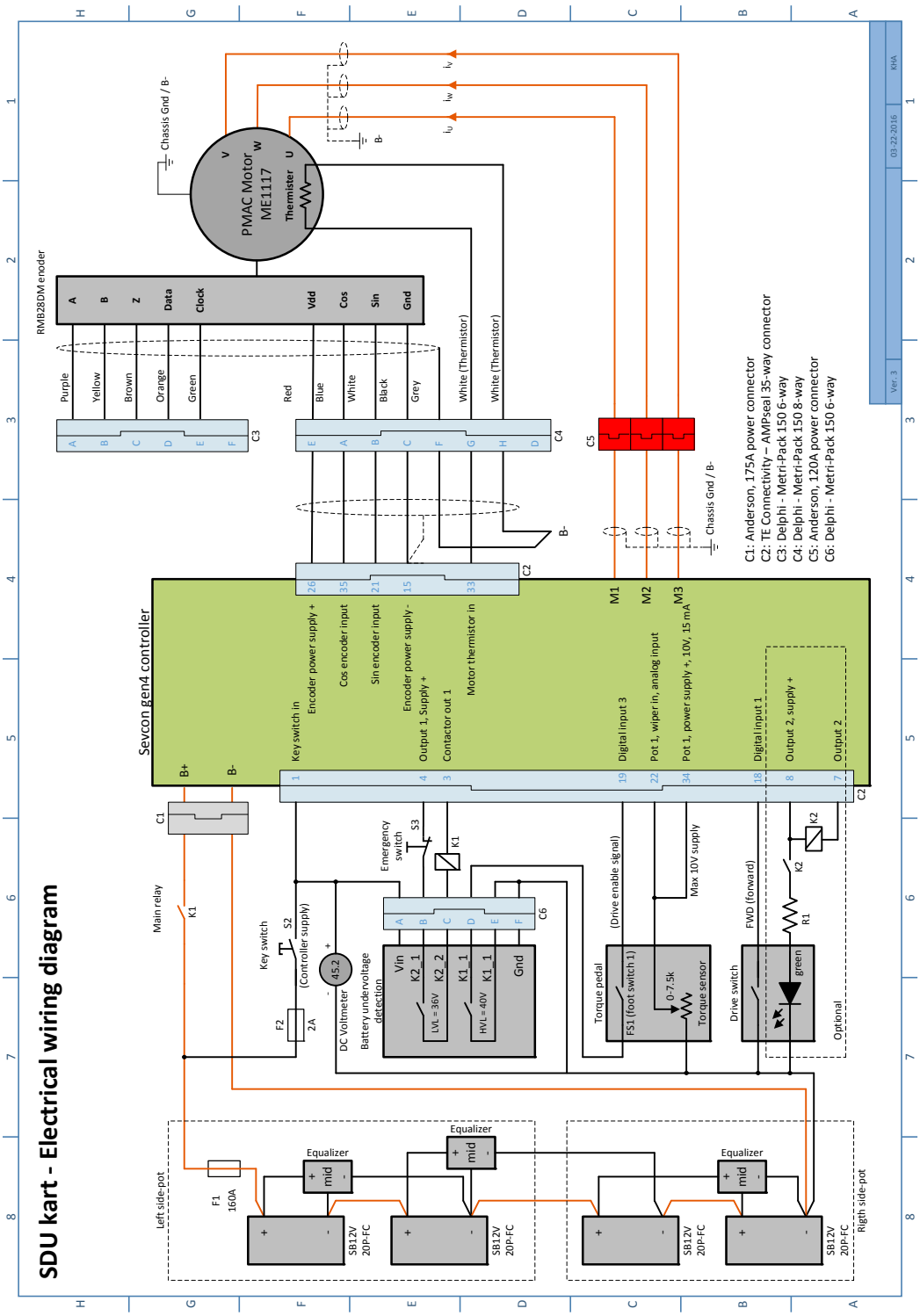


Figure 3.3: Go-Kart wiring diagram.

4 System Analysis

To get an overview of the system connections and all the major components needed to make the go-kart run, a system diagram is made based on the wiring diagram seen in figure 3.3. Figure 4.1 shows a simplified diagram of the wiring, where the Zybo board has replaced the Sevcon Gen4 controller.

The green component represents the Zybo board. The PMAC motor, the driver and the inverter are all placed to the right of the controller board. Both the driver and inverter are parts of the Sevcon Gen4 controller and as such, they must be developed. The two blocks next to the motor are LEM current sensors. Two of these are sufficient as the third current can be calculated using Kirchoff's Current Law. The encoder is measuring the rotational position of the rotor and feeding it to the controller.

To the left of the Zybo the different safety systems are placed. These will ensure that starting, stopping and emergency procedures are handled safely, preferably without any burned components. On top are the different switches, the key switch, the emergency switch and the enable switch. Below these is the undervoltage protection circuit which ensures that the go-kart is only driving when sufficient power is supplied. The final parts to the left are the torque pedal for the go-kart and the battery supply.

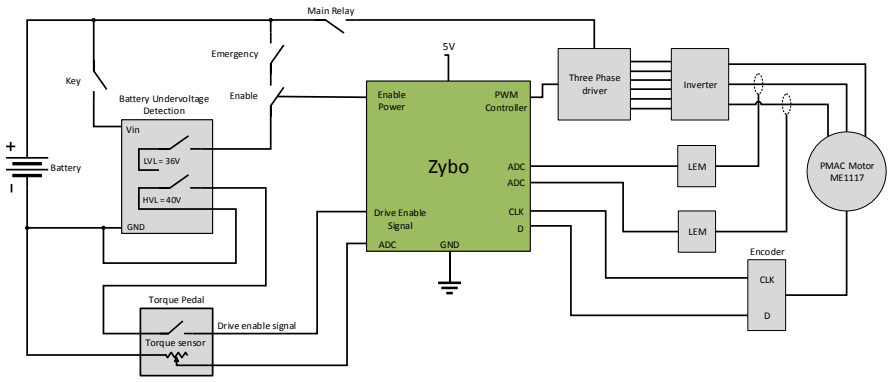


Figure 4.1: Wiring diagram with the Zybo board.

A diagram of the functionality of the wanted control process can be seen in figure 4.2. The different blocks of the figure will be explained in depth in the coming sections, but are presented here to give an overview of the complete system.

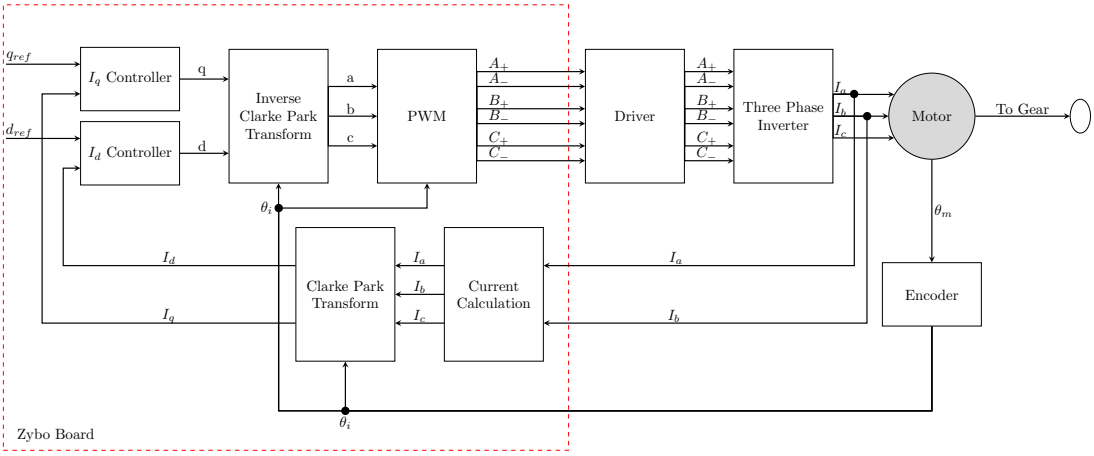


Figure 4.2: Diagram of the drive control system.

The following sections will explain what needs to be done in different areas to achieve the functionality of figure 4.1 and 4.2

4.1 Electronics

In order to fulfil the requirements of the project, electric circuits will be developed to perform certain tasks. The tasks at hand for the electronic circuits were found to be:

- Overcurrent protection.
- Circuitry for LEM current transducer output.
- Power supplies for the Zybo board, current transducers and other parts of the circuit.
- Torque pedal downscale. The torque pedal output must be scaled to comply with the input characteristics of the Zybo board.
- Various logic to control different parts of the system, including logic level shifting for the encoders as well as circuitry for a drive enable signal.
- Driver circuit. A driver is needed to control the inverter.

All of the above points will be explored further in section 5.

4.2 Embedded System

An embedded system is needed in the project in order to perform the tasks shown in the red box in figure 4.2. These tasks include:

- Creation of PWM signals. The motor will require a three phase sinusoidal signal in order to run, this is created using PWM.
- Controller. The digital controller used to control the entire system will be implemented using the embedded system.

- Interface for peripherals. A number of switches, LED's, enable and fault signals are used in the control of the motor.
- Conversion of the analog signal created by the current transducers and the torque pedal into a digital signal which can be used in controlling the motor.
- Decoding the input from the encoder.
- Interface to a computer in order to set and get relevant parameters.

The overall software design and development will be explored further in section 6.

4.3 Power Electronics

A three phase inverter will be developed throughout this project. Specifications for the inverter are:

- Must be able to generate three AC phases from one common DC supply using PWM.
- It should be able to handle 300A amplitude
- It must be able to handle 65V amplitude.
- Noise from the inverter must not disturb the other electronics.
- The inverter must not become too hot during prolonged use.

The development of the power electronics is discussed in detail in section 7.

4.4 PMAC Motor

The PMAC motor provided needs to be analysed in order to drive it optimally and simulate the performance of it. Parameterisation of the motor will be done as a part of the analysis. This analysis as well as a general explanation of the functionality of a PMAC motor will be presented throughout section 8.

4.5 Control Theory

To make the go-kart drive optimally a controller needs to be developed. A three phase PMAC motor needs a three phase sinosoidal signal to drive and as this is problematic to control directly some transformation will be needed. As the control algorihm will need to be run on an embedded system a discrete controller is needed. The development of the controller will be described in detail in section 9.

5 Electronics Design

This section is dedicated to describing the electronics designed throughout this report. This includes an analog, a digital and a driver board. Each of these boards, the choice of components and the layout will be discussed in appropriate detail in the following sections. While the power board is referenced throughout this section, the discussion of the power board can be found in section 7. One aspect of the layout specifically is important; the grounding layout. As this discussion provides a convenient means of giving an overview of the various components, the grounding will be discussed initially:

5.1 Electronics Components and Ground Layout

As previously mentioned, there are four boards in this design. Across these four boards are three different ground planes: digital ground (GNDD), analog ground (GNDA) and power ground (GNDP). Keeping these seperate is done mainly to avoid noise propagating throughout the circuits. For instance, GNDP is used as the return path of the high currents flowing through the motor. The voltage on this rail is likely to be fluctuating significantly. This will surely compromise the integrity of more crucial signals such as the current measurements used to determine the position of the rotor. GNDD will also carry significant noise from the digital circuitry and should also be kept separate from GNDA. The only crossover between the GNDD and GNDA will happen in the DRV8301 (see section 5.4) and in the NOR circuitry described in section 5.3.2. The latter occurs at the over-current event signal which is generated on the analog board but used in the digital system. An analog signal is fed to the SN74LVC1G17, a buffer with built-in Schmitt-trigger circuitry which is supplied from the digital 3.3V rail, thereby connecting the grounds.
A simplified overview of the grounding planes can be seen on figure 5.1. As it is undesirable to have the potentially noisy GNDP present on the boards, it is constrained to one corner of the digital board. Here it is used to supply the isolated DC-DC converters that generate the ±15V and 5V rails respectively. The previously mentioned 3.3V rail is generated on the Zybo board. The driver chip (The DRV8301 [4] is discussed in more detail in section 5.4), is also supplied from the power rail, and as such GNDP is extended to a corner of the driver board. Limiting the extent of GNDP is also the reasoning behind generating the $\pm 15V$ rail on the digital board, even though it is used exclusively on the analog board.

5.2 Analog Board

The analog board houses, as the name implies, all of the analog signal processing. Mainly the over-current protection, but also the torque pedal scaling and the current sensing is present on this board.

5.2.1 Over-Current Protection

In order to protect the system, it is necessary to create some form of over-current protection, OCP. Two choices immediately present themselves:

1. The DRV8301 is equipped with current sensing circuitry that allows the chip to automatically shut down the gate drivers when an over-current event, OCE, occurs.

2. External circuitry that, based on the measurements done by the current transducers, detects an OCE and turns off the drive signals by driving the EN_GATE signal of the DRV8301 low.

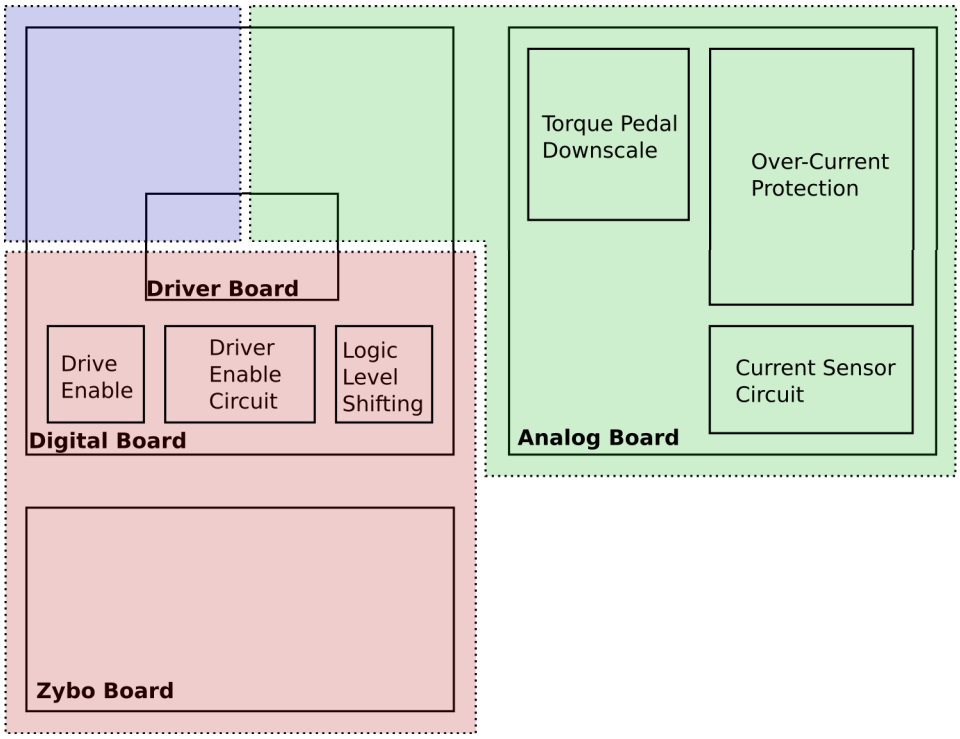


Figure 5.1: Simplified view of the electronics with the ground planes marked. Red is digital ground, green is analog ground and blue is power ground.

Initially 1 seemed like a sensible choice, however, at the time it was not possible to deduce the functionality of the OCP in the DRV8301 based on the datasheet. Therefore it was decided to use 2 since the function of the OCP will be well known and can be easily adjusted to meet any altered requirements, should the need arise.

In testing, the OCP function of the DRV8301 repeatedly shut down the drive circuitry and it was necessary to determine how to disable this feature. During this research the functionality of the OCP was more well understood. Due to difficulties with the debugging of the custom OCP circuitry presented below, it was decided to use the built-in OCP function of the DRV8301. This is discussed in more detail in section 5.4.

Regardless of this change, it was decided to keep the description of the original OCP circuitry as this constitutes a significant portion of the work done on the project. This description continues here: Since only two phases are measured directly it is necessary to calculate the third. This is done using a non-inverting summing amplifier. See figure 5.2. The output of this amplifier is given by:

$$V_{out} = \left(1 + \frac{R_4}{R_3}\right)\left(V_1 \frac{R_2}{R_1 + R_2} + V_2 \frac{R_1}{R_2 + R_1}\right) \tag{5.1}$$

No amplification of the signals is wanted, thus the resistances are all set to $R = 10\text{k}\Omega$. With all three phase signals available it is possible to process them to detect any OCE's.
The entire protection circuit for one phase can be seen in figure 5.3. Here the circuit has been split in accordance with the functionality of its four main parts: ADC isolation, full wave rectifier,

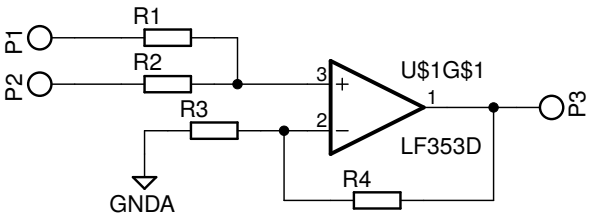


Figure 5.2: The non-inverting summing amplifier used to calculate the current in the third phase.

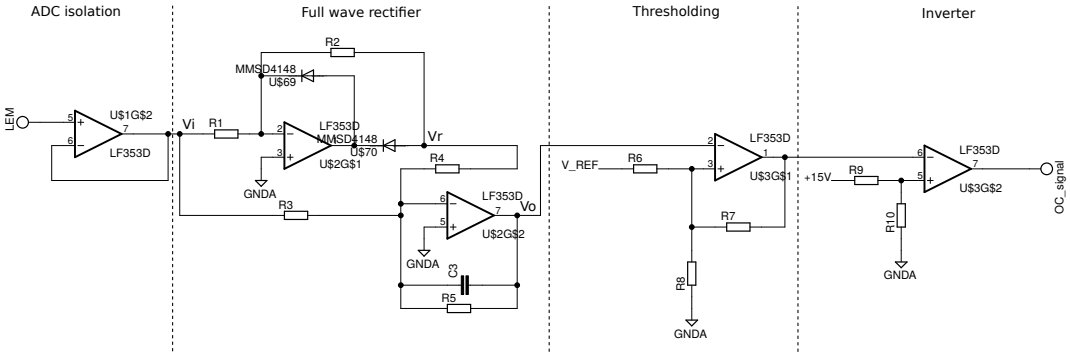


Figure 5.3: One phase of the over-current protection circuit.

thresholding circuit and an inverter. Each part will be explained in appropriate detail in the following paragraphs.

ADC Isolation: The current measured in each phase is used in the control of the motor and as such it is important to maintain the integrity of the signal. Therefore, after routing the signal to the ADC it is applied to the OCP circuit through a voltage follower. Since the op amp has extremely high input resistance ($10^{12}\Omega$ in the case of the LF353 used in this circuit) what happens on the output-side of the voltage follower has no meaningful impact on the input-side.

Full Wave Rectifier: The phase signals are sinusoids with an expected peak amplitude of ± 0.5V. For simplicity, it is desired to detect only the positive OCE. This requires rectifying the signal. Due to the small magnitude of the signals it is necessary to use a precision full wave rectifier. The rectifier used here can be divided into two subcircuits; a half-wave rectifier and a summing amplifier. In order to show the functionality of the rectifier, three test points are marked on the circuit, V_i, V_r, and V_o. V_i is the input voltage, V_r is the rectification stage and V_o is the rectified signal. Figure 5.4 shows the voltages at these points.

Thresholding: As explained in section 5.2.2, the input signal to the OCP circuit is dimensioned such that the maximum current in either of the phases will result in a voltage of 0.5V. Also mentioned in this section is the desire to leave some headroom in case of an OCE. This is due to the response time of the system. 50mV is estimated to be sufficient. An OCE is therefore defined

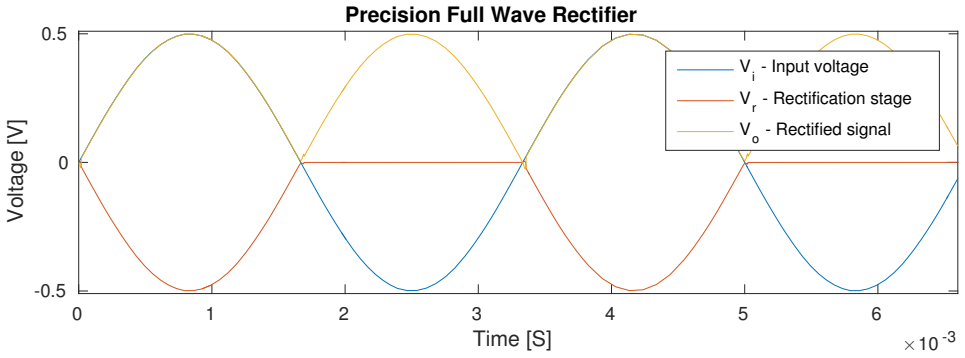


Figure 5.4: The condition of the signal at points V_i, V_r and V_o in the full wave rectifier.

as a time where the voltage exceeds 0.45V. In order to avoid repeatedly switching the signal on and off, it was decided to use a Schmitt trigger. Determining the component values is done using the following equations:

$$\frac{V_{cc} - V_h}{R_7} = \frac{V_h}{R_8} + \frac{V_h - V_{ref}}{R_6} \tag{5.2}$$

$$\frac{V_{ee} - V_l}{R_7} = \frac{V_l}{R_8} + \frac{V_l - V_{ref}}{R_6} \tag{5.3}$$

Where $V_{ref} = 2.5$V as set by the LM4040, a precision 2.5V voltage reference, the supply voltages $V_{cc} = 15$V and $V_{ee} =$ ground. The desired hysteresis voltages V_h and V_l are 0.475V and 0.25V respectively. This ensures that the enable signal will be pulled low when the current through any of the phases exceeds 285A, and will not return until the current is below 150A. The LM4040 supplies a voltage in, effectively the same manner as a linear voltage regulator. It is therefore desirable to draw as little current through R_6, as possible. As per the datasheet it can operate from 60µA $\rightarrow$ 15mA. It is decided to set the maximum current through R_6, $\frac{V_{ref} - V_l}{R_6} = I_{R_6} = 100$µA. This sets $R_6 = 22.5$kΩ. R_7 and R_8 can now be isolated using 5.2 and 5.3:

$$R_7 = \frac{(V_{cc}V_l - V_{ee}V_h)(V_l - V_{ref})}{I_{R_6}(V_h - V_l)V_{ref}} \tag{5.4}$$

$$R_8 = \frac{(V_{cc}V_l - V_{ee}V_h)(V_l - V_{ref})}{I_{R_6}(V_{cc}(V_l - V_{ref}) - V_{ee}(V_h - V_{ref}) + (V_h - V_l)V_{ref})} \tag{5.5}$$

Resulting in $R_7 = 150$kΩ and $R_8 = 2542.37$Ω. Since 2542.37Ω is hardly a standard value, the nearest standard value is chosen such that $R_8 = 2550$Ω.

Inverter: Due to the NOR-stage described in section 5.3.2, it is necessary to invert the signal. A simple comparator is used to achieve this effect. The input voltage to this stage is either V_{cc} or GNDD, the voltage on the non-inverting terminal is therefore not critical. Setting $R_9 = R_{10} =$ 10kΩ is sufficient.
The circuitry discussed above will ensure that in case of an OCE, a signal is sent to the digital board where it will be used to safely handle the event. This is discussed in further detail in later sections.

5.2.2 Current Transducer - LF 205-S

While these current sensors are not strictly part of the analog board, the circuitry related to them is and therefore it was chosen to include their description here. The LF 205-S utilises the Hall effect in order to measure the current flowing through a wire. As per the datasheet the internal transformer has a 1:2000 turns ratio, therefore the secondary current, I_S can be found as:

$$\frac{I_S}{I_P} = \frac{N_P}{N_S} \quad \Rightarrow \quad I_S = \frac{I_P}{N_S} \tag{5.6}$$

Essentially, the LF 205-S generates a current proportional to the current flowing through the device, the primary current; I_P. By changing the value of the resistor used to convert the current generated by the LF 205-S to voltage, R_m, the resulting voltage drop can be dictated.
In section 7 the maximum current to be expected in each phase is found to be 300A. As will be described in section 6, the ADC on the Zynq chip can work on signals in the ranges 0V-1V or $\pm$0.5V. Since the current is sinusoidal it can be negative. Choosing the range $\pm$0.5V therefore simplifies the circuitry needed to read the value correctly. Consequently R_m needs to be set such that the maximum expected current results in the maximum allowed voltage. However, it is desirable to leave some headroom in case of an OCE, 50mV should be sufficient. Thus:

$$R_m = \frac{0.45}{I_{Pmax}/N_S} = 3\Omega \tag{5.7}$$

It has later been discovered that bi-polar does not allow for voltages to be below the digital ground. For this reason unipolar mode is used instead. In order to properly read the values, 0.5V is added to the signal using a summing amplifier.

5.2.3 Torque Pedal Downscale

The torque pedal is simply a variable resistor, $R_{tp} = 7.5\text{k}\Omega$. As the pedal is actuated the resistance increases. Figure 5.5 is a depiction of the circuit used to read the position of the torque pedal.
R_1 and R_{tp} form a variable voltage divider of the 15V rail. The analog input of the ADC in the FPGA, however, is limited to 0-1V. Therefore it is necessary to downscale the voltage.

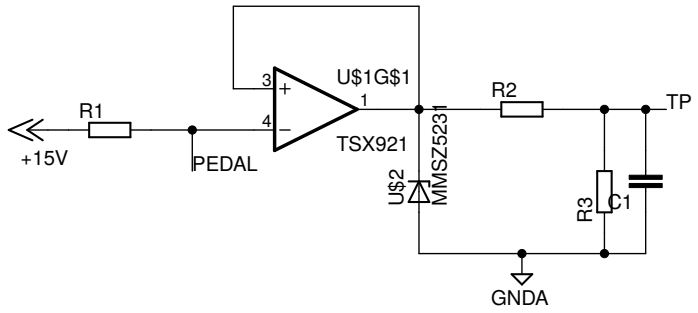


Figure 5.5: Circuit used to scale the voltage from the torque pedal from 0V $\rightarrow$ 5V to 0V $\rightarrow$ 1V.

Intuitively this could be done using simply a voltage divider, but since the torque pedal functions as a voltage divider, the varying resistance distribution of the torque pedal would influence the voltage divider. For this reason a voltage follower is used to isolate the two circuits. In addition, a 5.1V zener diode is clamping the signal to ground in order to protect the ADC of the Zybo. Due

to the mechanical nature of this signal it is expected to be reasonably noisy, therefore a filtering capacitor is placed on the output. This will act as a low-pass filter, stabilising the output signal. The resistance of the pedal is assumed to change linearly, which means the voltage division does not. Voltage at the point TP as a function of pedal is:

$$V_{TP}(R_P) = V_{CC}\frac{R_P}{R1 + R_P} \cdot \frac{R3}{R3 + R2} \tag{5.8}$$

By isolating R_P, it's possible to calculate what its resistance is. The resistance of R_P varies from 0 to 7.5kΩ, as the pedal is pressed down. Hence, equation 5.9 will give a value proportional to the torque requested by the driver

$$R_P = \frac{V_{TP} \cdot R1 \cdot (R2 + R3)}{R3 \cdot (V_{CC} - T_{TP}) - V_{TP} \cdot R2} \tag{5.9}$$

5.3 Digital Board

The digital board is used mostly as an interconnection board between the various electronic components in the system. The logic levels of the different systems are managed here along with the enable signal for the DRV8301. Additionally this board houses the in-rush relay.

5.3.1 Drive Enable Signal

This signal is produced by the torque pedal. When the user actuates the pedal a switch is connected to ground. By reading the drive enable signal it is therefore possible to determine when `EN_GATE` should be active. Since this signal is left floating when the pedal is not active, it can be routed directly to the Zybo, using the internal pull-up resistors.

5.3.2 Driver Enable Circuit

The DRV8301 is equipped with a pin called `EN_GATE`. If, while running, this pin is pulled low, the gate signals are immediately shut down. This feature allows for fast and reliable shut down of the motor in case of any detected faults. There are two different types of faults detected within the system, the aforementioned OCE and an over-temperature event, OTE. In addition to these two a signal is routed from the Zybo which enables the digital part of the system to disable/enable the motor, should the need arise. As there are three different signals that all need to connect to the same pin, they are all NOR'ed, as seen in figure 5.6.

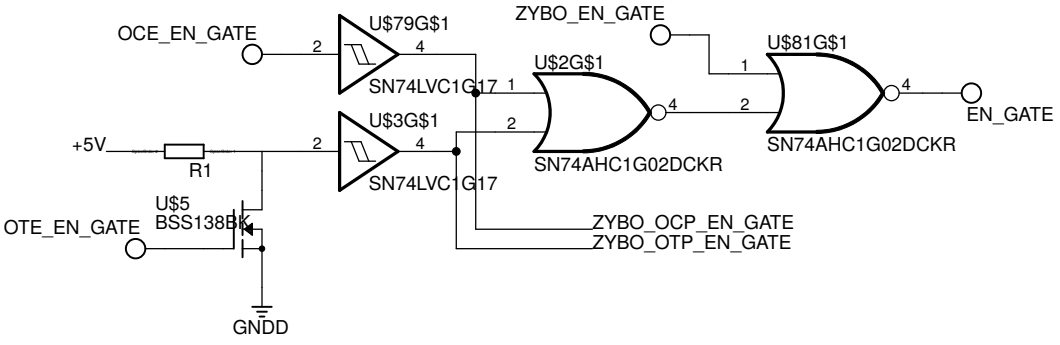


Figure 5.6: The NOR circuit that compares the three enable signals to determine whether the drive signals should be on.

The observant reader will notice that two 2-input NOR gates were used rather than one 3-input gate. Reason for this is that the OTE protection was not implemented until after the components were ordered. In order to not need additional components it was decided to cascade the available component. This cascading however does mean that it is necessary to invert the logic level of the signals OCE, and OTE. OCE is inverted on the analog board while OTE is used to pull the signal to ground. The two signals are both routed to the Zybo. This is done such that it is possible to discern which event has triggered when the motor halts.

5.3.3 Logic-Level Shifters

Tracking the angular position of the rotor is done using an encoder mounted on the motor. Upon receiving a clock signal the encoder will return the current position of the rotor. The encoder module is an AM256 which, according to the datasheet, requires at least 3.5V for digital high input. However, the Zybo uses 3.3V and the clock signal therefore needs to be shifted to the correct voltage level, 5V. A, B, Z and Data generated by the AM256 all need to be shifted down to 3.3V in order to protect the Zybo. The circuit for these four signals can be seen in figure 5.7. The buffer used for these signals is, as can be seen from the figure, the 74AHC125 which is a quad-buffer. V_{out} of this device is equal to V_{cc}, while V_{in} can be any voltage from $0 \rightarrow 5$V, by supplying it from the 3.3V rail it will therefore act as a logic level shifter.

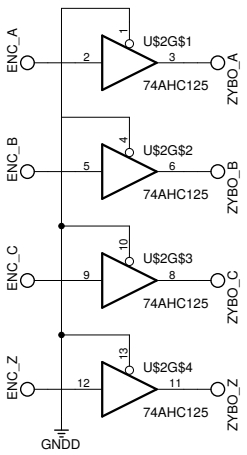


Figure 5.7: The 74AHC125 quad-buffer used to shift the logic level of the encoder from 5V down to the 3.3V required by the Zybo. Due to an oversight, a buffer with an enable on each port was bought. This feature is unnecessary and they are always enabled.

5.3.4 Inrush Relay

When first switching on the entire system, a considerable amount of current is required to charge the capacitors used on the inverter. In order to protect the system, it is necessary to limit the amount of current that is flowing to the capacitors until the voltage drop across them is reasonably small. The main relay, when closed, acts as a short circuit between the batteries and the inverter. Instead of closing this relay immediately, an additional relay is set in parallel. This is the in-rush relay. In series with the inrush relay a resistor is placed which is dimensioned to limit the current sufficiently. This circuit can be seen in figure 5.8. It has been decided that charging the capacitors to 90% in two seconds is acceptable. In order to do this, the series resistor should be 50Ω. This results in up to approximately 1A current.

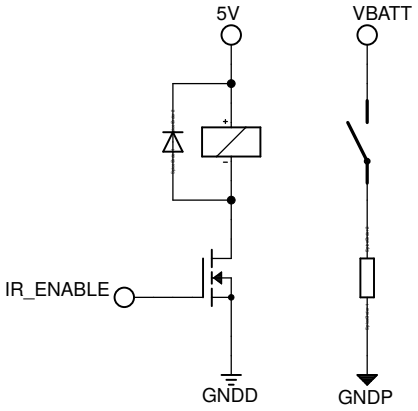


Figure 5.8: Inrush relay circuit. A small relay is driven from the Zybo in order to pre-charge the capacitors before switching the main relay.

5.4 Driver Board

This driver board is made exclusively to house the DRV8301, the driver chip chosen for this project. The reason for the seemingly superfluous board, is that there is no equipment available to the group capable of creating the small footprint of the chip. Therefore this board is made by an external fabricator.
The DRV8301 is a three phase gate driver with a built-in buck converter. Additionally it features both dead time control as well as over-current protection. This chip is made specifically with driving PMAC motors in mind, making it an ideal candidate for this project. Figure 5.9 shows the pinout of the DRV8301. This will be used as a basis for a brief discussion of the features provided by the chip.

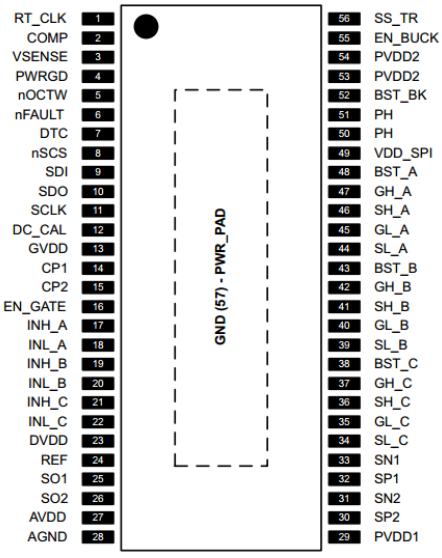


Figure 5.9: Overview of the pinout of the DRV8301.

5.4.1 Buck Converter

The DRV8301 is equipped with a buck converter capable of supplying up to 1.5A. It contains only the switching circuitry and the characteristics of the converter can be designed using external components. For instance, the voltage can be adjusted using the VSENSE pin. According to the datasheet [4] the buck converter is isolated from the remaining circuitry. These aspects makes it an ideal candidate for creating the 5V rail needed to supply the Zybo board. Initially this was the intention, however, this introduced significant complications in terms of the grounding layout and it was decided to use an isolated DC-DC converter, as described previously. This decision led to a simpler grounding plane as well as a simpler driver board, both desirable outcomes.

5.4.2 Over-Current Protection

As has already been discussed, the functionality of the over-current protection of the DRV8301 was initially decided against, mainly due to the apparent complexity of the feature. It was believed that this feature relied on the use of shunt resistors on the source pin of the low-side of two of the half-bridges. Since this feature was not to be used the, believed to be, associated pins SN1, SP1, SN2 and SP2 were pulled to ground. This was believed to disable the effects of the OCP. In testing the driver board the nOCTW pin was repeatedly pulled low, indicating that an OCE had occured. One feature of the OCP is the ability to disable it using SPI. In attempting this, much was learned about its functionality.

The OCP detects an OCE by measuring V_{ds}. V_{ds} for the high side of either channel is measured between pin SH_x and PVDD1 while the low side V_{ds} is measured between SL_x and SH_x. $R_{ds(on)}$ is known from the MOSFETs the threshold V_{ds} for any I_{ds} can be determined:

$$V_{ds} = R_{ds(on)} \cdot I_{ds} \tag{5.10}$$

Since the maximum $I_{ds} = 300\text{A}$ and $R_{ds(on)} = 3.2\text{m}\Omega$ for the IRFP4468PBF (the MOSFET chosen for the inverter, see section 7.2):

$$V_{ds} = 0.96\text{V} \tag{5.11}$$

Clearly, it is necessary to set the correct threshold of V_{ds}. This parameter as well as several others can be set using SPI.

5.4.3 SPI

The DRV8301 is capable of performing SPI communication as a slave. As mentioned above, this can be used to change various parameters concerning the functionality of the chip. When sending a message to the driver, it will respond in the following communication cycle with the appropriate information. Figure 5.10 is a depiction the format of an SPI input data packet. Any message sent to the driver must adhere to this format. As can be seen, the first 11 bits are data, followed by 4 address bits and finally, a read/write bit.

	R/W	ADDRESS				DATA										
Word Bit	B15	B14	B13	B12	B11	B10	B9	B8	B7	B6	B5	B4	B3	B2	B1	B0
Command	W0	A3	A2	A1	A0	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0

Figure 5.10: SPI input data packet format of the DRV8301.

	R/W	DATA														
Word Bit	B15	B14	B13	B12	B11	B10	B9	B8	B7	B6	B5	B4	B3	B2	B1	B0
Command	F0	A3	A2	A1	A0	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0

Figure 5.11: SPI output data packet format of the DRV8301.

Figure 5.11 is a depiction of the format of an SPI output data packet. Upon receiving a message on SDI a response is prepared in this format. In a response packet, the R/W bit is used as a fault bit, indicating whether the received packet is valid.
For a data packet to be considered valid, the DRV8301 requires the following:

- SCLK must be low when $\overline{\text{SCS}}$ goes low.
- Packet must have 16 full clock cycles.
- SCLK must be low when $\overline{\text{SCS}}$ goes high.

If the received packet is of type WRITE, the data in the response will contain the contents of status register 1, see figure 5.12. Using a READ message the contents of any of the four registers can be read at any time.

ADDRESS	REGISTER NAME	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
0x00	Status Register 1	FAULT	GVDD_UV	PVDD_UV	OTSD	OTW	FETHA_OC	FETLA_OC	FETHB_OC	FETLB_OC	FETHC_OC	FETLC_OC

Figure 5.12: Status register 1 of the DRV8301.

All of the control of the functionality of the DRV8301 can be done through the control registers as seen in figure 5.13. Of interest to this project are OCP_MODE and OC_ADJ_SET. OCP_MODE allows the user to set the mode of operation of the OCP. One of four options can be chosen: current limit mode, OC latch shut down mode, report only mode and OC disabled mode. During testing the driver was being run in OC disabled mode, however, during this time it was decided to attempt using the feature for detection of OCE's. The OC_ADJ_SET allows setting the value of V_{ds} appropriate for the type of MOSFET chosen. As previously mentioned, this value is equal $V_{ds} = 0.96\text{V}$. Figure 5.14 shows the different values that can be set as the threshold. Since 0.96V is not available, the nearest value below is chosen. Choosing a value above would result in an OC limit higher than the 300A specified. For this reason, the threshold is set to $V_{ds} = 0.926\text{V}$ by setting OC_ADJ_SET in control register 1 equal to 23.

ADDRESS	NAME	DESCRIPTION	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
0x02	GATE_CURRENT	Gate drive peak current 1.7 A										0(1)	0(1)
		Gate drive peak current 0.7 A										0	1
		Gate drive peak current 0.25 A										1	0
		Reserved										1	1
	GATE_RESET	Normal mode									0(1)		
		Reset gate driver latched faults (reverts to 0)									1		
	PWM_MODE	6 PWM inputs (see Table 1)								0(1)			
		3 PWM inputs (see Table 2)								1			
	OCP_MODE	Current limit						0(1)	0(1)				
		OC latch shut down						0	1				
		Report only						1	0				
		OC disabled						1	1				
	OC_ADJ_SET	See OC_ADJ_SET table	X	X	X	X	X						

(a) Control register 1 of the DRV8301.

ADDRESS	NAME	DESCRIPTION	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
0x03	OCTW_MODE	Report both OT and OC at nOCTW pin										0(1)	0(1)
		Report OT only										0	1
		Report OC only										1	0
		Report OC only (reserved)										1	1
	GAIN	Gain of shunt amplifier: 10 V/V								0(1)	0(1)		
		Gain of shunt amplifier: 20 V/V								0	1		
		Gain of shunt amplifier: 40 V/V								1	0		
		Gain of shunt amplifier: 80 V/V								1	1		
	DC_CAL_CH1	Shunt amplifier 1 connects to load through input pins							0(1)				
		Shunt amplifier 1 shorts input pins and disconnects from load for external calibration							1				
	DC_CAL_CH2	Shunt amplifier 2 connects to load through input pins						0(1)					
		Shunt amplifier 2 shorts input pins and disconnects from load for external calibration						1					
	OC_TOFF	Cycle by cycle					0(1)						
		Off-time control					1						
	Reserved												

(b) Control register 2 of the DRV8301.

Figure 5.13: These registers are used to control the functionality of the DRV8301.

Control Bit (D6–D10) (0xH)	0	1	2	3	4	5	6	7
Vds (V)	0.060	0.068	0.076	0.086	0.097	0.109	0.123	0.138
Control Bit (D6–D10) (0xH)	8	9	10	11	12	13	14	15
Vds (V)	0.155	0.175	0.197	0.222	0.250	0.282	0.317	0.358
Control Bit (D6–D10) (0xH)	16	17	18	19	20	21	22	23
Vds (V)	0.403	0.454	0.511	0.576	0.648	0.730	0.822	0.926
Code Number (0xH)	24	25	26	27	28	29	30	31
Vds (V)	1.043	1.175	1.324	1.491	1.679(1)	1.892(1)	2.131(1)	2.400(1)

Figure 5.14: The threshold values for V_{ds} and the associated code to be set in control register 1.

6 Embedded Design

An embedded system is needed to handle a wide range of task. It is a requirement for the project that the Zybo board should be used as the embedded system in the project. The Zybo board features a Zynq Z-7010 chip from Xilinx, which has an integrated dual-core ARM Cortex-A9 processor and a Xilinx 7-series field programmable gate array (FPGA). The Zybo board itself consists, among other, of several buttons, switches, LEDs and connections for USB, Ethernet, HDMI and several PMOD connectors. Throughout the remainder of the report the FPGA part will be referred to as the Programmable Logic (PL) and the ARM processor as the Processing System (PS).

6.1 Functionality

An analysis of the complete system yielded the functionalities listed below for the Zybo board.

- Digital inputs and outputs.
- Generating PWM.
- Measuring voltage output from LEM sensors and torque pedal.
- UART communication with external PC.
- Control algorithms.
- Read position from encoder.
- SPI communication as master.

As can be seen on the list there are multiple tasks that needs to be handled. Optimally, these tasks should be run at different speeds. For instance, the PWM switching frequency should be 20kHz as described in section 7. As this is quite fast it should be handled in PL. The rest can be handled in PS. In order to more easily run tasks at different frequencies it was chosen to use khaOS, a Run To Complete Scheduler (RTCS) system made by Karsten Holm Andersen. Using an RTCS simplifies scheduling tasks to run at different frequencies.
As an RTCS has no way to preempt tasks it leaves the responsibility of preserving real time performance of the system to the programmer. khaOS was chosen before other RTCS as the group had prior knowledge about the system from an Embedded Systems course.

6.2 Architecture

The software functionalities are grouped into tasks based on functionality and desired frequency. A task diagram of the complete system on the Zybo can be seen in figure 6.1.
As can be seen, various signals are coming from the outside world through the PL into the PS. All tasks, IP cores and I/O peripherals will be described in the coming sections.

6.2.1 Three Phase PWM Generator

The PWM generator is placed in the PL and is developed using Xilinx System Generator in Matlab Simulink. The generator is made to have adjustable frequency and duty cycle. It was chosen to generate center aligned PWM as this introduces less harmonic distortion than edge aligned PWM [6]. Furthermore center aligned PWM has the benefit of producing symmetrical PWM.

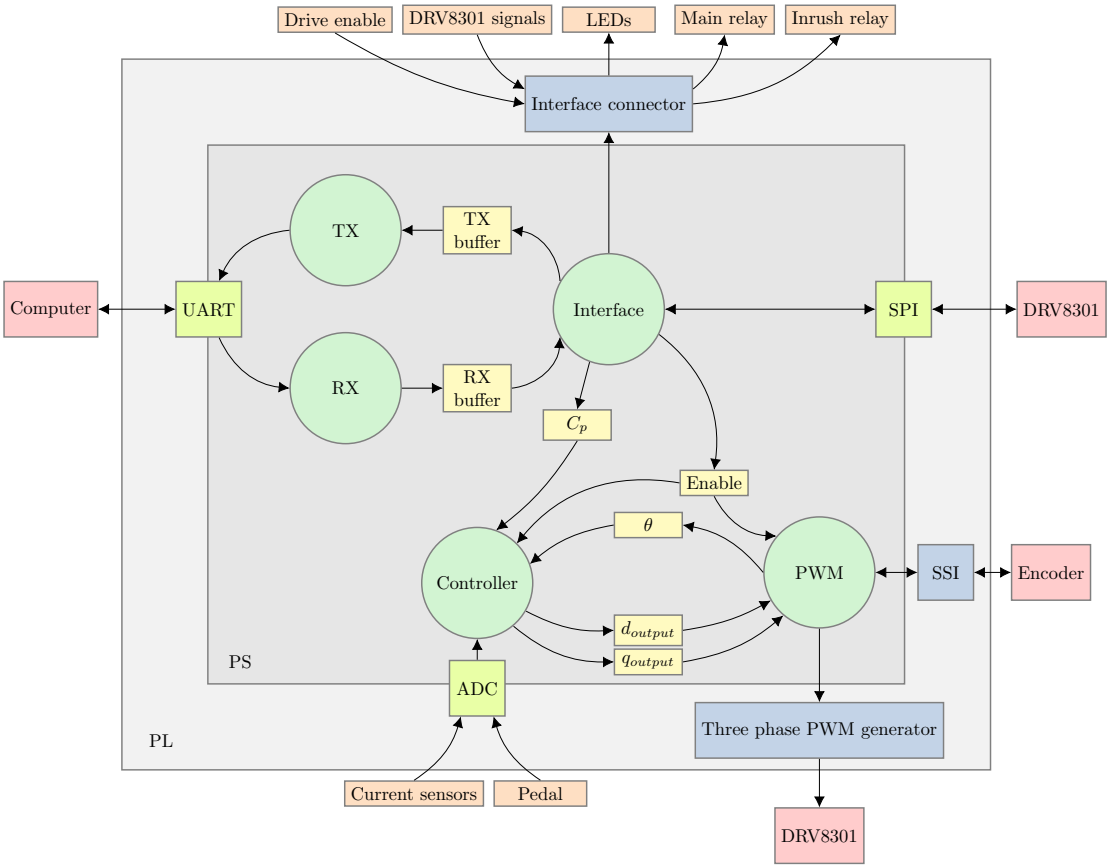


Figure 6.1: Task diagram showing software tasks as green circles, shared variables as yellow boxes, IP cores as blue boxes, I/O peripherals as lime green boxes, external signals as orange boxes and external components as red boxes. The PS and PL areas are marked accordingly and arrows show the data and signal direction. C_p consists of various controller parameters.

Symmetrical PWM has a well defined midpoint where the current can be sampled correctly. The high limit for the counter can be calculated for a given frequency, f, by the following:

$$H = L + \frac{f_{Zybo}}{f \cdot 2}$$

Where H is the high limit, L is the low limit and f_{zybo} is the Zybo clock frequency. The PWM counter system can be seen in figure 6.2. The counter block will count up or down depending on the input. The output from the counter value will go into to two compare blocks along with the high and low limits. The compare blocks will produce a high signal when the counter value is equal either the high or low limit. These signals are then fed to an m-code block which contains a simple state machine which determines the direction of the counter. The counter out signal can be seen in figure 6.4. This signal is then fed into the PWM generator shown in figure 6.3. The duty cycle is given to the PWM generator as data type u32, an integer type. Therefore the duty cycle is given in a range of 0 to 1000, giving a resolution of 0.1%. The switching limit is the value, where when the counter value equals, switches the polarity of the PWM output. The switching

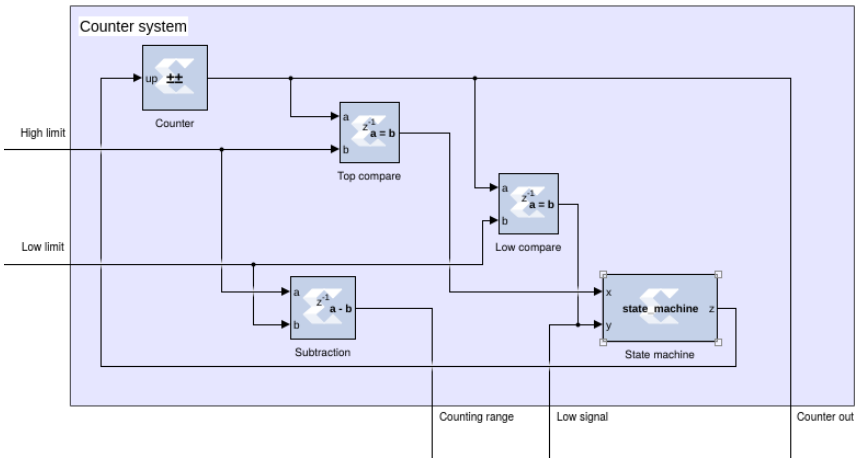


Figure 6.2: Simulink block diagram of the counter in the PWM generator.

limit is calculated from the following:

$$l = (1 - \frac{d}{1000}) \cdot r$$

Where l is the switching limit, d is the dutycycle in the range 0 - 1000 and $r = H - L$ is the counter range. A register will then make sure that the switching limit is only applied when the counter is lowest. This is important as otherwise the dutycycle may be corrupted by misaligning the PWM. The compare block will make the PWM signal by comparing the switcing limit to the counter value. The counter signal and output of the PWM generator can be seen in figure 6.4. As the PWM generator should be able to do three independent PWM signal there are three of the systems shown in figure 6.3, one for each phase.

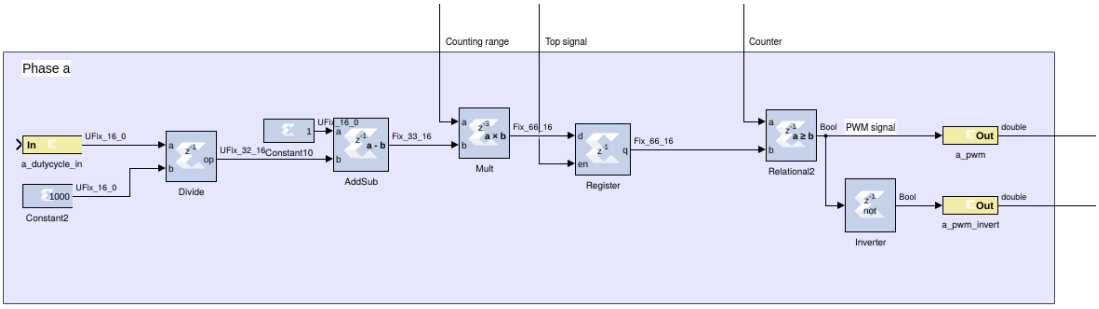


Figure 6.3: Simulink block diagram of the PWM mechanism in the PWM generator.

6.2.2 Interface Task

The interface task reads external signals, outputs digital signals and controls the other tasks. This includes ensuring that the state of the enable variable, read by the controller and the PWM task, is correct. The basic functionality of the interface task can be seen in the flowchart of figure 6.5.

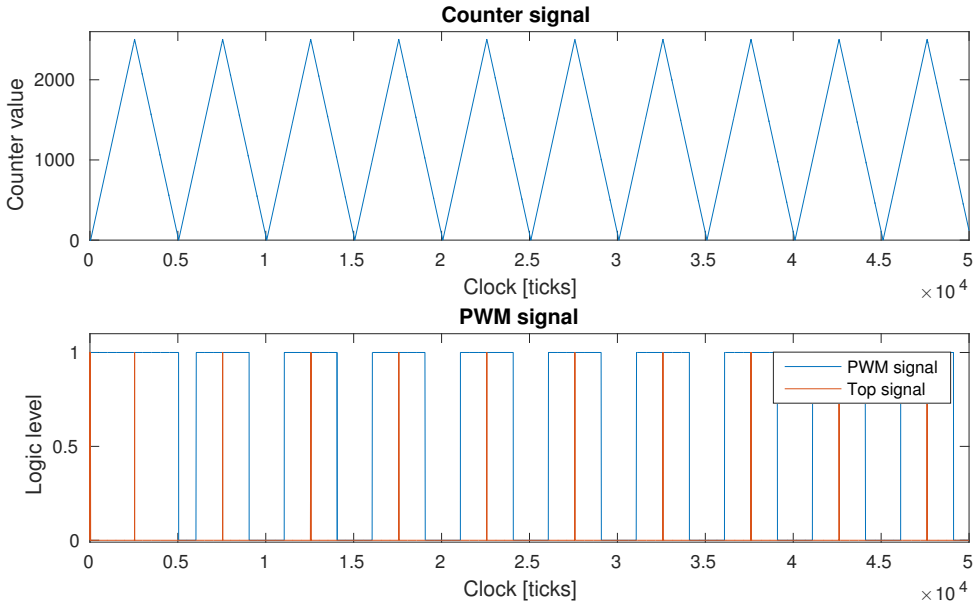


Figure 6.4: Counter and PWM signal.

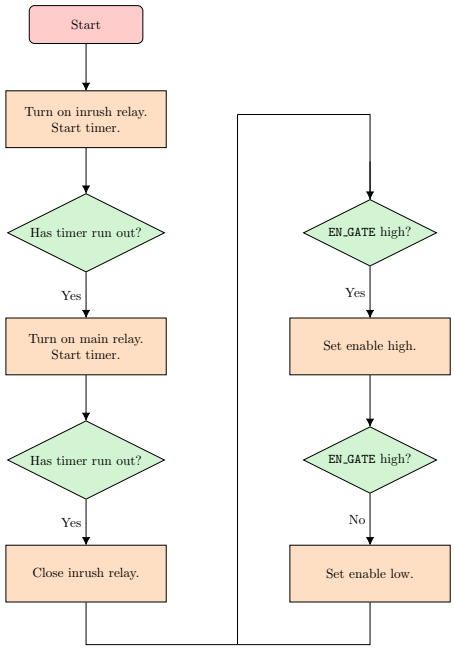


Figure 6.5: Flow chart illustrating the basic functionality of the interface task.

At startup the interface task switches on the inrush relay for a certain time. Afterwards the main relay is switched on and the inrush relay is switched off after some time to ensure that the main

relay is conducting. Then the task will read all signals from the digital board and, if `EN_GATE` is high, the global enable variable will be set high. If `EN_GATE` is low, indicating a fault, the task will change the enable variable to low. Alongside the functionality shown in figure 6.5, the interface will act as master for the two UART tasks. This enables the setting of values within the system as well as potentially reading the current status of the system. All communication through the UART interface is done through the `rx_buffer` and `tx_buffer` arrays shown in listing 6.4. Currently three parameters can be set within the system:

- `kp/ki`: These are the values of the PI controller used.
- `qmax`: The limit of the current that the controller is allowed to set as reference.

Setting any of the above is done by:

```
set <parameter> <value>
```

As can be seen in listing 6.1 the command is split into parts using `strtok()`. This is a non-reentrant function that will return all the characters until the first appearance of delimiter. Every consecutive call to the function with `NULL` as the first parameter will continue the search through the same array. With the command split into tokens it is parsed using a simple if-structure, comparing each argument in the command using `strcmp()` until a value can be set.

```
1  if(rx_flag)
2  {
3    char* cmd = strtok(rx_buffer, DELIMITER);
4    char* var = strtok(NULL, DELIMITER);
5    char* val = strtok(NULL, DELIMITER);
6
7    if(!strcmp(cmd,"set"))
8    {
9      if(!strcmp(var,"kp"))
10     {
11       scanf(val, "%f", &kp);
12       sprintf(tx_buffer, "kp was set: %f", kp);
13       tx_flag = true;
14       tx_tail = strlen(tx_buffer);
15     }
16           ⋮
17
18   }
19   rx_flag = false;
20   rx_tail = 0x00;
21 }
```

Listing 6.1: Interface task, the code responsible for decoding commands.

Once the value is set a message is returned to the user, informing of the successful setting. `rx_flag` is cleared and the buffer is emptied by setting `rx_tail` to zero. The system is made with expandability in mind, currently only a `set` command is implemented. This could easily be extended to a `read` or perhaps a `shutdown` command.

6.2.3 Controller Task

The basic functionality of the controller task can be seen in figure 6.6. The task polls data from the ADC in order to get the raw data for the phase currents I_a, I_b and the torque pedal. The raw data from the ADC is then calculated into voltages and the two currents are calculated by:

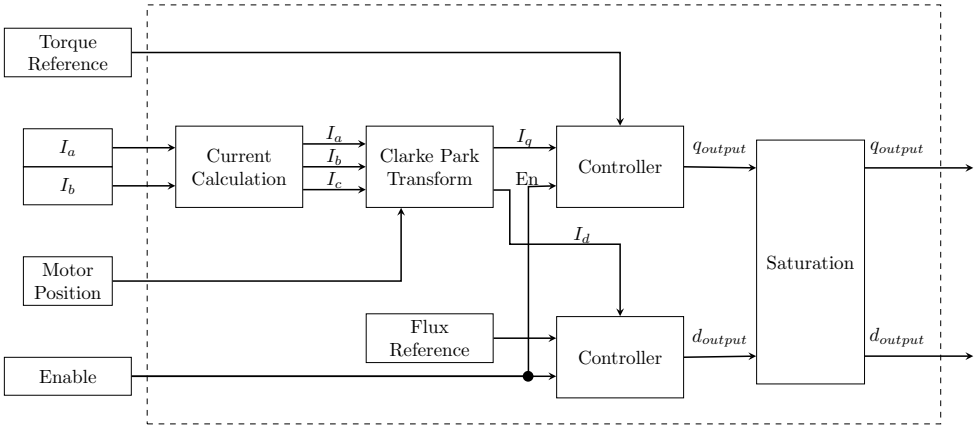


Figure 6.6: Block diagram showing the functionality of the controller task.

$$I_{measured} = \frac{V_{measured} - 0.5\text{V}}{R_m} \cdot N_S \tag{6.1}$$

This was discussed previously in section 5. I_c is calculated based on the assumption that the three currents form a balanced three phase system. The resistance of the torque pedal potentiometer is calculated from the measured voltage and equation 5.9.
I_a, I_b, I_c and θ are then used to perform a Clarke Park transformation and obtaining the values of I_d and I_q. I_d and I_q should be controlled towards a set value by two separate controllers. The value for I_d is 0 as will be explained in section 9. The value, q_{set}, for I_q should be related to the maximum allowed current, q_{max}, and the position of the torque pedal in percent, p:

$$q_{set} = p \cdot q_{max} \tag{6.2}$$

It was chosen to use two PI controllers as will be described in section 9. The code for one of the controllers can be seen in listing 6.2. As can been seen in line 2 trapezoidal integration is used to perform the integration. Lines 4-8 performs anti-windup and lines 9-14 calculates the output if enable is high and performs integrator reset if low.

```
1  d_error = d_set - I_d_measured;
2  d_integral = d_integral + (d_previous_error + d_error)*0.5*dt;
3
4  if(d_integral > MAX_TOTAL_OUTPUT){
5          d_integral = MAX_TOTAL_OUTPUT;
6  }else if (d_integral < -MAX_TOTAL_OUTPUT){
7          d_integral = -MAX_TOTAL_OUTPUT;
8  }
9  if(enable){
10         d_output = kp * d_error + ki * d_integral;
11 }else{
12         d_output = 0;
13         d_integral = 0;
14 }
15 d_previous_error = d_error;
```

Listing 6.2: C code constituting a PI controller with trapezoidal integration, anti windup and integrator reset.

Afterwards `d_output` and `q_output` are downscaled if they exceed the maximum limits. The code for downscaling can be seen in 6.3.

```
total_output_magnitude = sqrt(d_output*d_output + q_output*q_output);

if(total_output_magnitude > MAX_TOTAL_OUTPUT){
        d_output = (d_output* MAX_TOTAL_OUTPUT)/total_output_magnitude;
        q_output = (q_output* MAX_TOTAL_OUTPUT)/total_output_magnitude;
}
```

Listing 6.3: C code for detecting and downscaling output values that exceed the maximum limits.

6.2.4 PWM Task

The basic functionality of the PWM task can be seen in the block diagram of figure 6.7. The task needs to take the outputs from the controller task and transform it into duty cycles that can be feed to the PWM generator in the PL area of the Zybo. The PWM task is separate from the controller task as it might be preferable to have the two tasks run at different frequencies. The task reads the rotor position from the encoder and access the global variables `d_output` and `q_output`. This information is used to perform inverse Clarke Park transformation. Afterwards third harmonic injection is performed and if the values are higher or lower than the maximum bounds they are limited. The global enable variable is checked and passed to the enable pin.

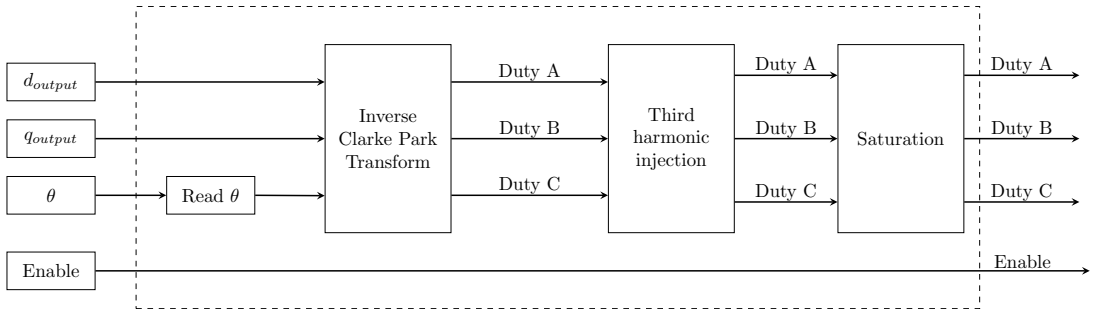


Figure 6.7: Block diagram showing the functionality of the PWM task.

6.2.5 ADC

The Zynq chip contains a dual 12-bit, 1 Mega sample per second (MSPS) Analog-to-Digital Converter (ADC) [7]. The ADC is utilized through the XADC Wizard in Vivado. The ADC measures the difference between differential analog input pins V_p and V_n. The IP core is configured to use AXI4LITE connection, simultaneous selection, unipolar mode and event mode. Simultaneous selection mode is used as it allows simultaneous sampling on the dual ADC. When using simultaneous sampling the phase relationship is preserved. When unipolar mode is enabled the input range between V_p and V_n is 1V and the input voltage must always be positive with respect to ground. Setting the ADC in event mode, means that the ADC will only sample when there is a rising edge on its CONVST pin. The CONVST pin is connected to the top signal of the PWM generator meaning that the ADC will sample in the center of the PWM pulse. This is done to sample the current correctly. Channels 6 and 14 are paired together in simultaneous mode, meaning that they will be sampled simultaneously. These are the channels that the outputs from the two LEM sensors are connected to. The torque pedal signal will be routed to channel 7 which will be sampled after channel 6 and 14. The top signal from the PWM generator will start the sampling

and therefore it will also set the sampling frequency. As the PWM frequency is 20kHz the ADC sampling frequency will be as well.

6.2.6 UART RX/TX

In order to facilitate communication between the Zybo board and a PC, a small UART connection is established. The connection consists of two tasks, UART_RX and UART_TX. They handle receiving and transmitting, respectively. As shown in listing 6.4 each of the tasks are given a buffer, a flag and a tail variable.

```
char rx_buffer[0xff];
char tx_buffer[0xff];
bool rx_flag = false, tx_flag = false;
uint8_t rx_tail = 0x00, tx_tail = 0x00;
```

Listing 6.4: UART_RX/TX buffers and variables.

```
void uart_rx_task()
{
  if (XUartPs_IsReceiveData(UART_BASEADDR))
  {
    rx_buffer[rx_tail] = XUartPs_ReadReg(UART_BASEADDR, XUARTPS_FIFO_OFFSET);
    rx_tail++;
  }
  if(rx_buffer[rx_tail-1] == '\r')
  {
    rx_flag = true;
  }
  _wait(MILLI_SEC(1));
}
```

Listing 6.5: UART_RX task.

Looking at listing 6.5 the function of these in the UART_RX task can be seen. If there is data on the line it is pushed onto the `rx_buffer` and `rx_tail` is incremented. This is to keep track of the number of characters that have been received. Finally, if the last character received is a carriage return, the `rx_flag` is set high. This indicates to the interface task that a full command has been received and is ready for execution. Listing 6.6 shows the UART_TX task. Every time it is run it will check `tx_flag`. If `tx_flag` is high, this indicates that data has been put into the `tx_buffer`. All `tx_tail` characters in the buffer are transmitted character by character. Hereafter `tx_buffer` is "emptied" by setting `tx_tail` to zero and `tx_flag` is set low to indicate that transmission is complete.

```
void uart_tx_task()
{
  if(tx_flag)
  {
    uint8_t i;
    for(i = 0; i < tx_tail; i++)
    {
      xil_printf("%c",tx_buffer[i]);
    }
    tx_tail = 0x00;
    tx_flag = false;
  }
  _wait(MILLI_SEC(250));
}
```

Listing 6.6: UART_TX task.

6.3 Encoder

The Zybo board needs to interface the RMB28MD encoder that is mounted on the motor. The RMB28MD is an absolute encoder with sine/cosine, SSI and incremental output. It was chosen to use the SSI output as an IP core capable of reading the position by SSI communication was made available by the supervisors of the group. Reading the position through SSI yields a resolution of 8 bit per mechanical revolution. The physical RMB28MD encoder signals are interfaced by an IP Core. The IP core outputs a clock to the RMB28MD chip and reads data from a data pin. It then makes the data available on the AXI4LITE bus, where it can be read from software by the PWM task.

6.4 SPI Communication

As described earlier the Zybo board will need to set certain parameters on the DRV8301 chip using SPI communication. The Zynq chip has two SPI controllers in the PS area of the chip, which can be setup to communicate with external devices. By investigation of the Zynq datasheet [9] it was found that it is not possible to configure the SPI controller to meet the requirements of the DRV8301. More specifically, it is not possible to configure the clock signal to go low for a minimum of 40 ns [4], before Slave Select, `SCS` goes low.
Two solutions are proposed to solve to this problem:

1. Develop a simple SPI driver in VHDL or System Generator to send the 16 bits according to the SPI requirements of the DRV8301.

2. Use the SPI controller in PS and delay the Slave Select signal physically.

The correct solution to the problem is to develop an SPI driver, but due to time considarations it was decided to use the SPI controller of the Zynq.
SPI 1 on the Zynq is used as it can be ported to MIO channels 10 to 15, which can be accessed through the PMOD connectors of the Zybo board. The SPI controller is initialized in software and is configured to be in master mode, manual chip select mode and manual start mode by writing to the configuration register Config_reg0. In order to comply with the SPI specifications of the DRV8301 it was configured with the XSPIPS_CLK_PHASE_1_OPTION in order to make data valid on falling edges of the clock.
The time constant of an RC filter is used to delay the voltage Slave Select signal, according to equation 6.3

$$V_{IL} = V \cdot e^{\frac{-t}{\tau}} \tag{6.3}$$

Where V_{IL} is the low input threshold for digital pins on the driver, V is the supply voltage of the driver and $\tau = R \cdot C$. The voltage drops exponentially, and the time it takes to reach V_{IL} should be at least 40ns, so to be safe, the filter will be designed with 80ns in mind. A resistor of 1kΩ, yields a capacitor of 56pF. A capacitor of 33pF was chosen. The transmission of `0b0001000000110000` was measured using an oscilloscope and can be seen in figure 6.8.

6.5 Task Frequencies

Table 1 shows the frequencies at which the task are run. The PWM task will be run at 20kHz as this allows for updates of duty cycles at the same frequency as the PWM generator. The controller task will run at 5kHz as will be explained in section 9. The most essential functionality of the interface task is to update the enable variable. As the driver chip shuts down the inverter in case

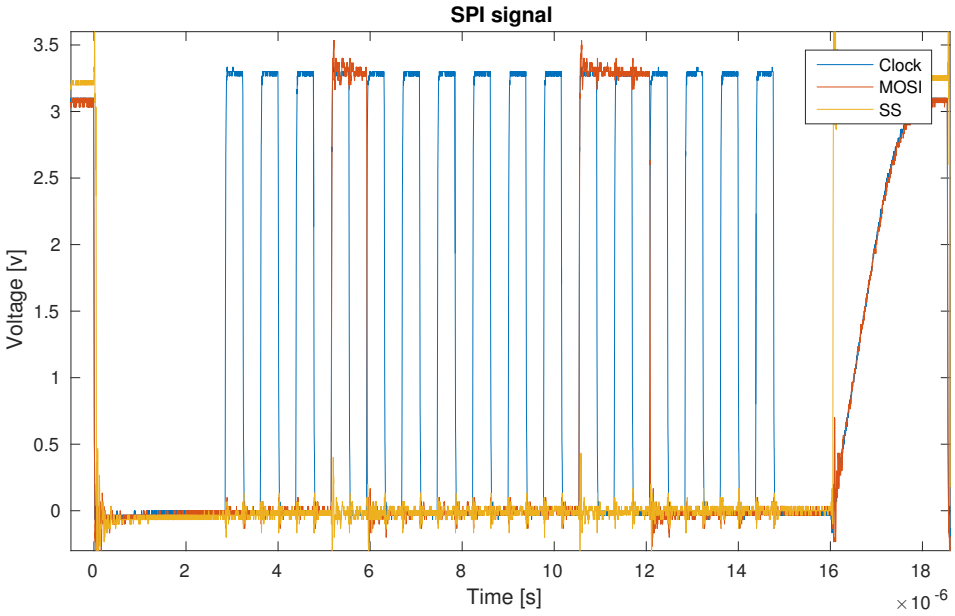


Figure 6.8: SPI transmission of two data bytes.

of an overcurrent event it is not very time critical to do so. Therefore it was chosen to run the interface task at 1kHz. The UART tasks are not in any way crucial and can therefore be run at a low frequency, though it has to be fast enough for the user to feel that the system is responding, 100Hz was chosen.

Task	Frequency
PWM	20kHz
Controller	5kHz
Interface	1kHz
UART	100Hz

Table 1: Table showing the frequency of each task.

6.6 Timing

To ensure the real time performance of the system a timing test was performed. An output pin was set high when entering a task and low, when exiting the task. PWM, controller and interface were connected to separate pins while the two UART tasks were connected to the same pin. The voltage on the pins were measured by an oscilloscope and the measurement can be seen in figure 6.9. In the first graph the tasks are in normal operation and it can be seen that the real time performance is maintained and that there is time left for expanding the tasks. In the lower graph, the interface task is programmed to output a string to the UART transmit task, giving extra work to both tasks. It is seen that the real time performance is still maintained even in, what is believed to be, a worst case operation. The delay between the PWM task and the controller task can, at time of writing, not be explained. However, it does not interfere with the functionality of the

program and will, due to time constraints, not be investigated further.

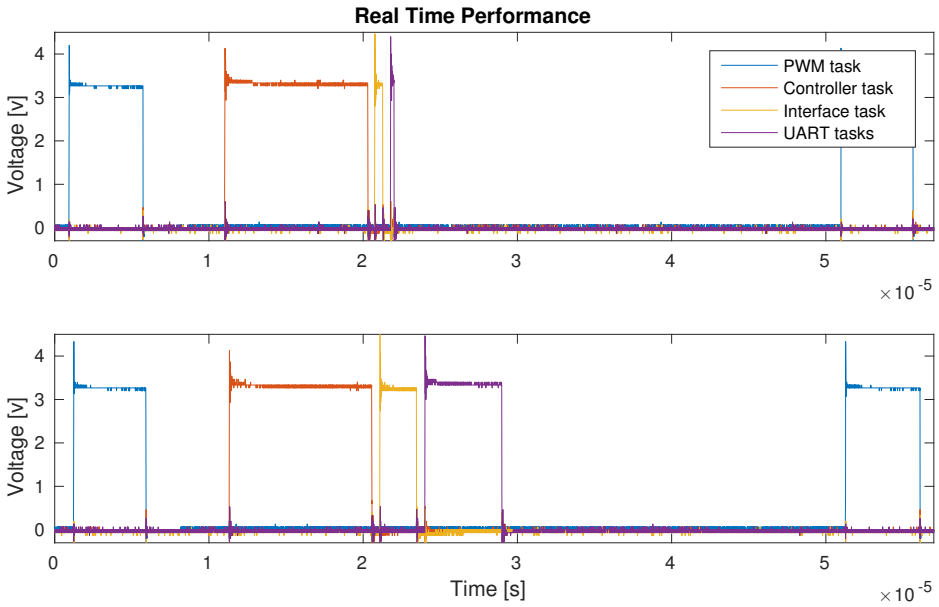


Figure 6.9: Timing test of the task running in khaOS.

7 Three Phase Inverter

This section will describe the design of the three phase inverter used to drive the PMAC motor in this project. This includes the calculations and considerations done. Initially, a short overview of the principle of operation of a three phase inverter is given.

7.1 Operation of a Three Phase Inverter

A three phase inverter enables the use of a DC supply in variable frequency control of a three phase load. In this project the DC supply is a battery and the load, a PMAC motor. Figure 7.1 is a Simulink model showing the most basic three phase inverter. As can be seen, the inverter consists of three half-bridges for a total of six switches. Each pair of switches can never be on at the same time as this would result in shorting the battery.

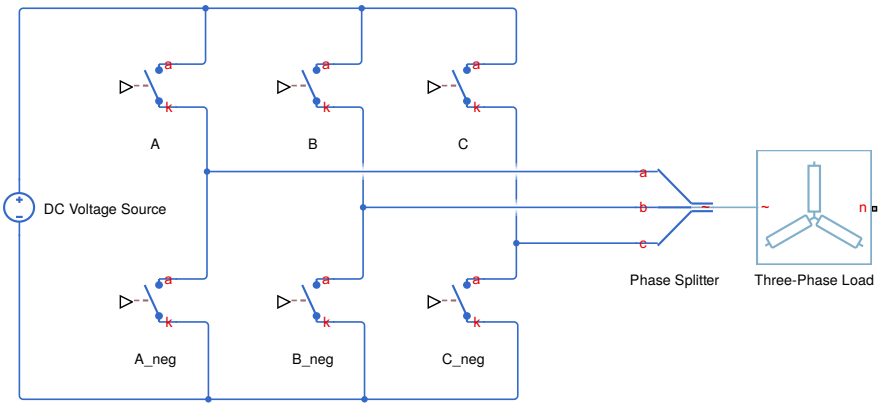


Figure 7.1: Simulink model of a simple three phase inverter for a balanced load.

This type of inverter only works with a balanced load, such as the PMAC motor. A balanced three phase load implies, that the sum of the currents in the three phases is always equal tp zero, and no current is running through a neutral wire. As the inverter does not have a connection to the neutral point of the load, it could not supply an unbalanced load, such as those in uninterruptible power supplies (UPS's) and solar cells.
Several schemes exist to control the switches. Throughout this project a PWM modulation technique using sinusoids as reference is used. Figure 7.2 is a depiction of the modulation technique. A carrier wave is compared to a sinusoidal reference signal. By low-pass filtering the output voltage, the fundamental frequency can be revealed. Whenever the reference crosses the carrier wave, the gate voltages of the relevant half-bridge invert, of course the necessary dead time has to be added to avoid short circuiting the power rail. This technique produces naturally sampled, center aligned PWM, a desirable trait due to the lower harmonic distortion when compared to other types of PWM.

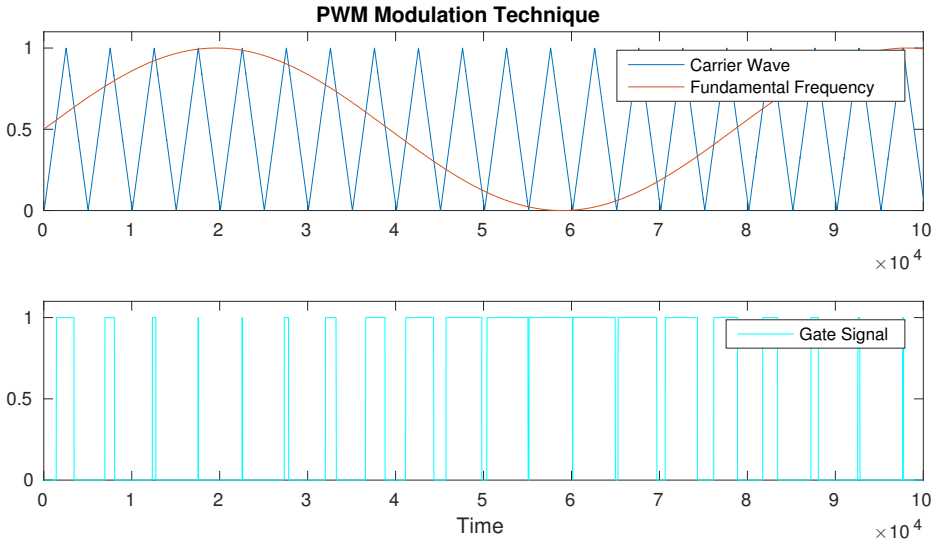


Figure 7.2: Naturally sampled PWM using a sinusoid as fundamental frequency.

7.2 Choosing the MOSFETs

Before choosing MOSFETs for the three phase inverter one will have to keep in mind the application parameters. Firstly, the batteries[8] will have an average nominal voltage of 52.8V and end of charge voltage at 60.8V. The MOSFETs will therefor need to be able to handle a minimum drain to source voltage, V_{DS}, of 60.8V. Secondly, the current drawn by the motor can reach a maximum peak of 300A. Thirdly, the motor driver[4] will supply a gate voltage, V_{GS}, of 9.5V to 11.5V. The MOSFETs will therefore need to be fully conducting at $V_{GS} = 9.5V$ and be able to handle the driver supplied range. The three main requirements are listed in table 2

	Requirement	
1	Can handle minimum voltage of	60.8V
2	Can handle minimum current of	300A
3	Operating at V_{GS} of	9.5V

Table 2: Three main requirements as a starting point for choosing the MOSFETs.

Due to high currents through the MOSFETs, the heat dissipation is assumed to be high as well. To prevent overheating and burning out the MOSFETs they will be mounted on to a heat sink. Therefore a through hole package will be required for this application. Because of its size and easy mounting, the chosen package is the TO-247 [12]. The TO-247 is the largest housing available and is not capable of handling the current. When looking for MOSFETs that add up to the requirements of table 2, the International rectifier power MOSFET IRFP4468PBF is chosen. It does not exactly meet the requirements, but paralleling at least two of these will.
The TO-247 package has a continuous drain current limit of 195A that is only around two third of the minimum current requirement. With that in mind, the MOSFET parameters in table 3 seem to be a suitable choice for the application if more than one MOSFET is used for each switching side. Using a parallel setup, the MOSFETs are assumed to share the current equally. If one would

Symbol	Parameter	Min.	Typ.	Max.	Units
I_D	Continuous Drain Current (Silicon Limited)	-	-	290	A
I_D	Continuous Drain Current (Package Limited)	-	-	195	
V_{GS}	Gate to Source Voltage	-	-	± 20	V
$V_{GS(th)}$	Gate Threshold Voltage	2.0	-	4.0	
$V_{(BR)DSS}$	Drain to Source Breakdown Voltage	100	-	-	
$R_{DS(on)}$	Static Drain to Source On Resistance	-	2.0	2.6	mΩ

Table 3: Some important parameters from the IRFP4468PBF MOSFET datasheet[5]. All parameters are found at junction temperature $T_J = 25°\text{C}$.

draw more current than the others, the temperature would rise causing $R_{DS(on)}$ to rise as seen in figure 7.4. Which will result in reduced current. As will be later discussed, parallel setup will also have an effect on the total power loss.

7.3 Power Loss

The total power loss, P_{loss} of a switching device can be estimated using equation 7.1 where P_c and P_{sw} are conduction- and switching losses respectively.

$$P_{loss} = P_c + P_{sw} \tag{7.1}$$

For simplicity the power loss will be estimated from a single phase of the inverter circuit seen in figure 7.3. The current I_{out} is drawn by some load R_{load} that will absorb the load power P_{load} which is given by equation 7.2. Since there are two switches, Q_1 and Q_2, it is assumed that the load power is delivered equally from both. The output power, P_Q, of each switch is given by equation 7.3

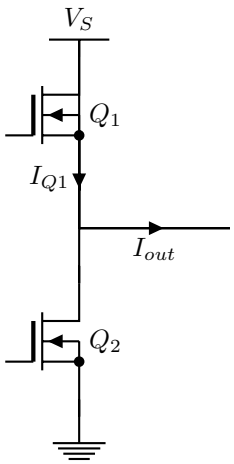


Figure 7.3: Single phase from where the MOSFET current is calculated.

$$P_{load} = R_{load} I^2_{out,RMS} \tag{7.2}$$

$$P_{Qi} = \frac{P_{load}}{2} = R_{load} I^2_{Qi,RMS} \tag{7.3}$$

Inserting equation 7.2 into 7.3 and solving for $I_{Qi,RMS}$, equation 7.4 is formed.

$$\frac{R_{load}I_{out,RMS}^2}{2} = R_{load}I_{Qi,RMS}^2 \rightarrow I_{Qi,RMS} = \frac{I_{out,RMS}}{\sqrt{2}} \tag{7.4}$$

Which then again can be expressed as in equation 7.5.

$$I_{Qi,RMS} = \frac{I_{out,RMS}}{\sqrt{2}} = \frac{I_{out}}{\sqrt{2}\sqrt{2}} = \frac{I_{out}}{2} \tag{7.5}$$

As mentioned earlier each switching setup will consist of n number of MOSFETs in parallel. Therefore the drain current of each parallel MOSFET becomes $I_{Pi,RMS}$ in equation 7.6.

$$I_{Pi,RMS} = \frac{I_{Qi,RMS}}{n} = \frac{I_{out}}{2n} \tag{7.6}$$

7.3.1 Conduction loss

Conduction loss of each MOSFET can be calculated using equation 7.7

$$P_c = R_{DS(on)} \cdot I_{Pi,RMS}^2 \tag{7.7}$$

Although listed in the datasheet, the drain to source resistance $R_{DS(on)}$ only applies to the $T_J = 25°C$ condition. The resistance will vary with the junction temperature T_J. The relation between T_J and $R_{DS(on)}$ can be seen in figure 7.4 from where the multiplication factor α in equation 7.8 can be estimated.

$$R_{DS(on),@T_J} = \alpha R_{DS(on),@25°C} \tag{7.8}$$

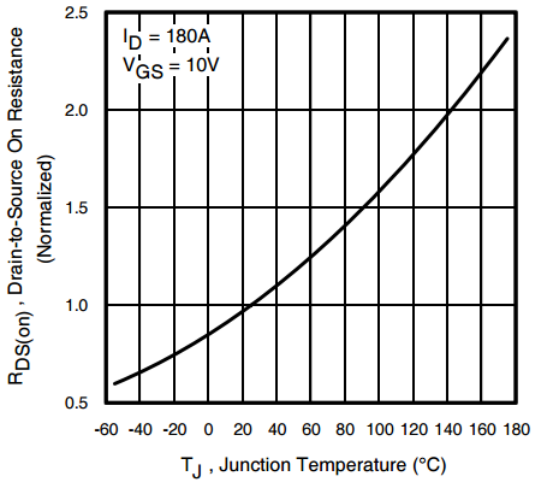


Figure 7.4: The $R_{DS(on)}$ and T_J relation at $I_D = 180A$ and $V_{GS} = 10V$ [5].

If assumed that the junction temperature will be around 100°C one can estimate the factor being approximately $\alpha = 1.6$. The instantaneous conduction loss for each MOSFET in parallel can then be estimated using equations 7.7, 7.6 and 7.8 to form equation 7.9.

$$P_c = R_{DS(on),@100°C}I_{Pi,RMS}^2 = R_{DS(on),@100°C}\left(\frac{I_{out}}{2n}\right)^2 \tag{7.9}$$

7.3.2 Switching Loss

Whenever any MOSFET is switching from its on state to its off state or vice versa, a small amount of energy is lost, as the MOSFET passes through its active region. This loss can be found with experiments or estimated from calculations. An experiment will not be carried out at this point but the loss will be estimated with calculations and assumptions using equation 7.10.

$$P_{sw} = \frac{1}{2} V_{in} I_D (t_{c,on} + t_{c,off}) f_s \tag{7.10}$$

where I_D is the drain current, $I_{Pi,RMS}$, of each MOSFET from equation 7.6 and V_{in} is the supply voltage for the inverter. The times $t_{c,on}$ and $t_{c,off}$ are the combined on and off time respectively and are defined in figure 7.5 as equation 7.11 and 7.12

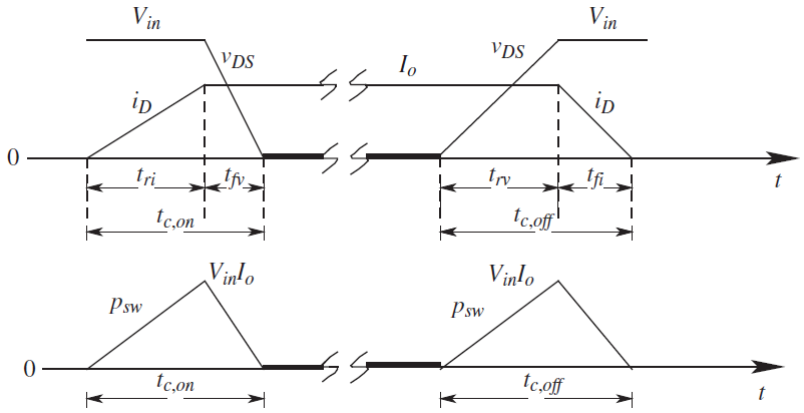


Figure 7.5: MOSFET switching losses. t_{ri} and t_{fi} are the current,I_D, rise and fall times respectively and t_{rv} and t_{fv} are the rise and fall time of D_{DS} respectively[18].

$$t_{c,on} = t_{ri} + t_{fv} \tag{7.11}$$

$$t_{c,off} = t_{rv} + t_{fi} \tag{7.12}$$

The voltage rise and fall times, t_{rv} and t_{fv} respectively, in figure 7.5 are the time it takes the drain to source voltage, V_{DS} to rise from zero to V_{in} and opposite. The values are defined in the MOSFET datasheet[5] as the rise and fall times t_r and t_f seen in figure 7.6. The times are listed in table 4. Equation 7.13 and 7.14 show the relation.

Symbol	Parameter	Typ.	Units
t_r	Rise Time	230	ns
t_f	Fall Time	260	

Table 4: V_{GS} rise and fall time taken from datasheet[5].

$$t_{fv} = t_r \tag{7.13}$$

$$t_{rv} = t_f \tag{7.14}$$

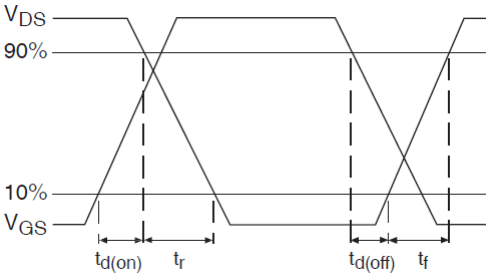
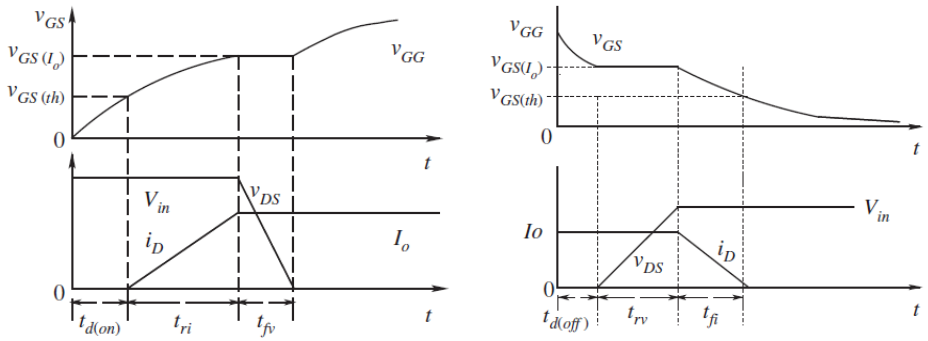


Figure 7.6: Switching Time Waveforms[5].

The turn on and off characteristics of the MOSFETs are shown in figure 7.7. From there one can see that the current rise and fall times, t_{ri} and t_{fi} respectively, are the same times as it takes the gate voltage V_{GS} to rise from $V_{GS(th)}$ to $V_{GS(I_0)}$ for the on state and opposite for the off state.



(a) Turn on characteristic.

(b) Turn off characteristic.

Figure 7.7: The turn on and off characteristics of the MOSFETs[18].

The gate to source current, I_{gs} is defined in equation 7.15 where dt is the current rise and fall times t_{ri} and t_{fi} respectively. dV is the difference between $V_{GS(th)}$ and $V_{GS(I_0)}$ as seen in equation 7.16 and n is the number of MOSFETs in parallel.

$$I_{gs} = nC\frac{dV}{dt} \tag{7.15}$$

$$dV = |V_{GS(I_0)} - V_{GS(th)}| \tag{7.16}$$

For simplicity it is assumed that the gate charge current remains constant. Then one can assume that the gate to source current is known. The motor driver sources up to 1.7A when switching to the on state and sinks up to 2.3A when switching to the off state[4]. Equation 7.15 can then be rearranged for dt as shown in equation 7.17. It is assumed that the charge current will be constant for the duration of the turn on and turn off times.

$$dt = nC\frac{dV}{I_{gs}} \tag{7.17}$$

7.4 Total Power Loss

The instantaneous power losses are calculated for up to 10 MOSFETs in parallel using equations 7.9, 7.10 and 7.1 where the switching frequency is $f_s = 20\text{kHz}$. Figures 7.8 and 7.9 shows the results.

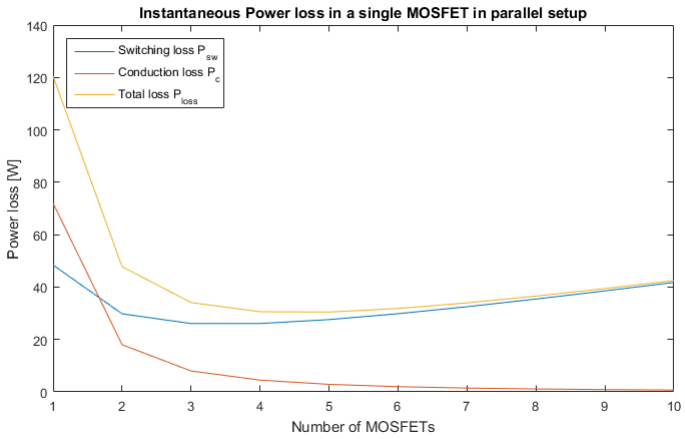


Figure 7.8: Instantaneous power losses of a single MOSFET in a parallel setup as a function of number of MOSFETs.

In figure 7.8 the losses are calculated for each MOSFET in a parallel setup of n MOSFETs. One will notice, as previously stated, that the number of MOSFETs in parallel, affects the power losses significantly. The conduction loss will decrease rapidly and will become close to nothing when n increases. This happens because the current is equally divided between the MOSFETs, thus, each MOSFET has less current flowing through. The switching loss however increases with n and will become dominant. In that case the gate capacitance will increase by a factor of n causing the switching loss to increase.

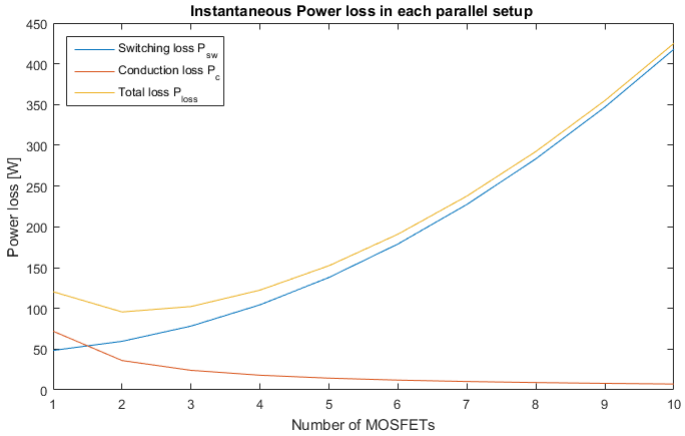


Figure 7.9: Combined instantaneous power losses of each parallel setup as a function of number of MOSFETs.

From the combined power loss shown in figure 7.9, it is clear that the total instantaneous power loss is smallest when $n = 2$ or $P_{loss} \approx 95\text{W}$. The design will have two MOSFETs in parallel

switching each phase. The average power loss at maximum RMS current will become for all three phases as seen in equation 7.18.

$$P_{loss,total} = 95 \cdot 6 = 570\text{W} \tag{7.18}$$

7.5 Temperature

The power losses defined in previous sections will cause the MOSFETs to heat up while operating. Therefore they will be mounted onto a heat sink as mentioned in section 7.2. To get an estimate on how hot the MOSFETs will be, equations 7.19, 7.20 and 7.21 are used. T_s is the heat sink temperature, T_c the case temperature and T_j is the junction temperature. The parameters can be seen in table 5. To get an estimate of the temperature it is assumed that the time it takes the go-kart to get from zero to full speed, $t = 5s$. The system will for that time be operating at maximum torque, for maximum time. In reality, other factors will effect the power dissipation but will not be discussed further.

Symbol	Parameter	Value	Units
$T_{J,max}$	MOSFET maximum junction temperature	175	°C
R_{jc}	MOSFET junction to case thermal resistance	0.29	°C/W
Re_{cs}	MOSFET case to sink thermal resistance	0.24	
R_{sa}	Heat sink thermal resistance	0.29	
C_s	Heat sink thermal capacitance	2190	J/°C
$P_{loss,total}$	Instantaneous power loss of each MOSFET	570	W

Table 5: Parameters used to calculate the temperature.

$$T_s = (R_{sa}P_{loss,total})\left(1 - exp(-\frac{t}{R_{sa}C_s})\right) + T_a \tag{7.19}$$

$$T_c = T_s + P_{loss}R_{cs} \tag{7.20}$$

$$T_j = T_c + P_{loss}R_{jc} \tag{7.21}$$

The temperatures are calculated to be:

$$\begin{aligned} T_s &= 26°\text{C} \\ T_c &= 49°\text{C} \\ T_j &= 77°\text{C} \end{aligned}$$

The thermal capacity, C_s of the heat sink is calculated using equation 7.22.

$$C_s = v \cdot \rho_m C_p \tag{7.22}$$

where C_p is the specific heat capacity of aluminium, which is 910 $\frac{\text{J K}}{\text{kg}}$, and ρ_m is the mass density of aluminium. For a heatsink of 30x30x1cm, the heat capacity is 2180 $\frac{\text{K}}{\text{J}}$.
The calculated junction temperature is well below the maximum ratings shown in table 5. If it is assumed that the go-kart will be operating with full power loss for t = 5min the temperature will reach $T_j \approx 140°\text{C}$. Thus, it is safe to assume that the MOSFETs will not reach the maximum limit.

7.6 Inverter Layout

The layout of the inverter was mostly done by the other group though discussion about design and assembly were made between the two groups. In this section the overall principles of the layout is described. As the inverter has to switch large currents, all parasitic inductances should be avoided. This is generally done by avoiding long wires as they all contribute with extra inductance. Wires are not completely avoidable, but they should be kept short, and forward and return wires should be twisted to keep them in close proximity.
The power stage of the inverter needs a higly conductive ground plane as well as a power plane, which is connected to the battery voltage. This is done with two 1mm thick plates of copper, separated by Kapton tape. This is a temperature resistant insulating tape meaning that there is no galvanic connection from the power plane on top, to the ground plane on the bottom. The developed layout can be seen in figure 7.10.

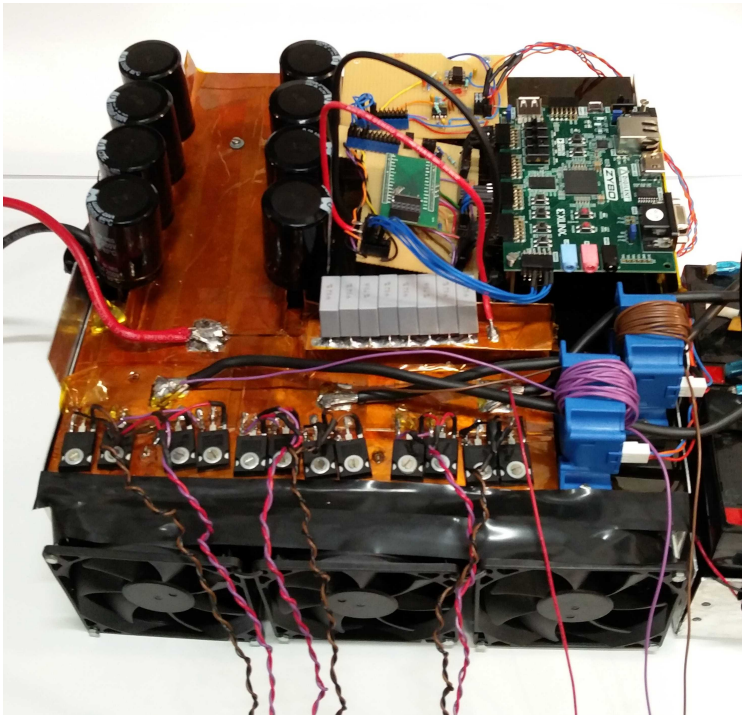


Figure 7.10: Picture of the finished three phase inverter with mounted Zybo, analogue and digital boards.

As it can be seen, the the inverter was build on a large heat sink, along with the the Zybo, digital and analogue boards. This heat sink is much larger than needed, but it was chosen for its structural integrity along with its cooling capability. The heat sink is mounted on an aluminium box, that blocks for air flow from the sides.
Below the heat sink three fans were places to produce airflow on the heat sink for better heat dissipation. This was necessary because the heat sink needs to be mounted vertically for passive convection to work properly.

The wires from the batteries will introduce inductance to the system in the order of a few µH. This will cause the current not to change instantaneously, when the upper MOSFETs are switched. Capacitors are added to filter this behavior. It is chosen to add eight Panasonic Aluminum Elec-

trolytic Capacitors , each having capacitance of 2200μF. Those are added to smoothen out low frequency voltage spikes. These each have an equivalent series resistance of 57mΩ, and can handle currents of 4.06A each. To counter for the higher frequency, another eight film capacitors with capacitance of 4.7μF are added.

8 PMAC Motor

This section focuses on the motor used for this project. Initially the functionality of the motor will be described, then certain motor parameters will be tested, as the online product description is not all that clear. Lastly a parking test will be performed to determine the angle of the encoder mounted on the motor. The motor used to drive the go-kart is an 8-pole non-salient permanent magnet synchronous motor.

8.1 Clarke Park Transform

The Clarke Park transform is integral to the type of control desired for the PMAC motor, so it must be understood. It is done by 'transforming the stator currents into a rotating reference frame that is locked to the rotation of the rotor' ([15] p.221). The transform starts by viewing the three phase star connection of the PMAC motor and displaying each phase as ABC (or W U and V as seen on figure 8.1 from the book cited [15]). This ABC current is first transform into a stationary frame of only two phases called the $\alpha\beta$ transform and then further transformed into dq0. When in dq0 form the reference frame rotates with the angular velocity ω of the motor, while being referenced to the stationary frame of the $\alpha\beta$ frame. This transform yields a d and q DC value and a zero, which are respectively referred to a torque and a flux demand in the controller.

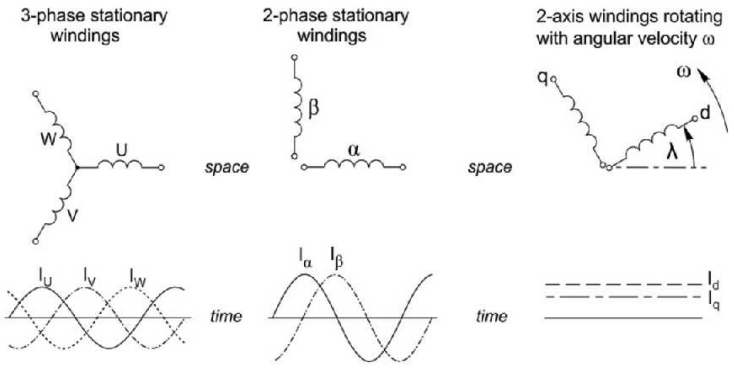


Figure 8.1: Clarke Park Transform reference frames. [15]

The Clarke Park transform is integral to the type of control desired for the PMAC motor, so it must be understood. It is done by 'transforming the stator currents into a rotating reference frame that is locked to the rotation of the rotor' ([15] p.221). The transform starts by viewing the three phase star connection of the PMAC motor and displaying each phase as ABC (or W U and V as seen on figure 8.1 from the book cited [15]). This ABC current is initially transformed into a stationary frame of only two phases, along horizontal and vertical axes of the three phase stationary windings, called the $\alpha\beta$ transformation. This is the Clarke transform. These axes are then further transformed into dq axes, that rotate along with the rotor. This is the Park transform. This allows

the controller to generate a magnetic field relative to the rotor. The q-axis current controls the torque produced in the motor, and the d-axis current controls the flux.

It is not relevant what the $\alpha\beta$ currents or voltages are, so the Clarke Park transform will be done in one step. This is described in equation 8.1.

$$\begin{bmatrix} I_d \\ I_q \end{bmatrix} = \frac{2}{3} \begin{bmatrix} \cos(\Phi) & \cos(\Phi - \gamma) & \cos(\Phi + \gamma) \\ -\sin(\Phi) & -\sin(\Phi - \gamma) & -\sin(\Phi + \gamma) \end{bmatrix} \cdot \begin{bmatrix} I_A \\ I_B \\ I_C \end{bmatrix} \tag{8.1}$$

where $\gamma = \frac{2\pi}{3}$.
This will be used to convert the controller output to reference signals for the PWM generator.

The inverse Clarke Park transform is done by equation 8.2.

$$\begin{bmatrix} I_A \\ I_B \\ I_C \end{bmatrix} = \begin{bmatrix} \cos(\Phi) & -\sin(\Phi) \\ \cos(\Phi - \gamma) & -\sin(\Phi - \gamma) \\ \cos(\Phi + \gamma) & -\sin(\Phi + \gamma) \end{bmatrix} \cdot \begin{bmatrix} I_d \\ I_q \end{bmatrix} \tag{8.2}$$

The requirement for this project outlines that the motor must be torque controlled, hence the flux demand will be zero while the torque demand will be defined by the speeder pedal. After the dq values have been compared to the reference/demand values the error is fed to a controller which will adjust the output accordingly. This output is transformed using the inverse dq transform, which yields a three phase AC signal that can be fed into the inverter and yield a new motor speed.

Illustrating this, the block diagram presented in section 4 can be retrieved. Figure 9.1. This shows how the current of the three phase AC values are retrieved from the motor input, transformed to dq and then compared to reference values. The i_d value will be zero as it is the flux reference, while the i_q value will be set by the speeder pedal as a torque reference. The controller acts upon these and the dq values are transformed back to ABC three phase AC and fed to the inverter.
In reality it is not a torque value that must be a reference for the controller, but a current value, as the current determines the torque. Taking this into account will be part of creating a functional model of the motor.

8.2 Motor Description

The motor used is a three phase Wye-connected, 8-pole non-salient permanent magnet AC motor, the Motenergy ME1117. Although there are many different types of PMAC motors, there is no physical difference between brushless DC motors (BLDC), permanent magnet AC motors (PMAC), and permanent magnet synchronous motors (PMSM). The motor will be driven as an AC or synchronous motor (since these are the same) to produce a more consistent torque. This means that the rotor consists of four pairs of magnets - one with the north side facing outward towards the stator and one with the north side facing inward to the ferromagnetic core of the rotor. The stator consists of 24 solenoids mounted inward on a ferromagnetic outer ring. Looking at the motor from the axle side, a counter clockwise rotoration results in the go-kart going forward. See figure 8.2. The solenoids are placed so that the four phase A solenoids are placed along the horizontal and vertical axes - this is not necessarily true, but a parking test will be performed in section 8.5 to determine the actual placement. The order of the other solenoids, when going counter-clockwise is then $\bar{B}$, C, $\bar{A}$, B and $\bar{C}$, as shown on figure 8.2. The solenoids A correspond to terminal M1, solenoids B correspond to M2, and solenoids C correspond to M3. Current going into terminal

M1, will then pass through all the A and $\bar{A}$ solenoids in series, before reaching the star point, and continuing through B and $\bar{B}$ and/or C and $\bar{C}$.

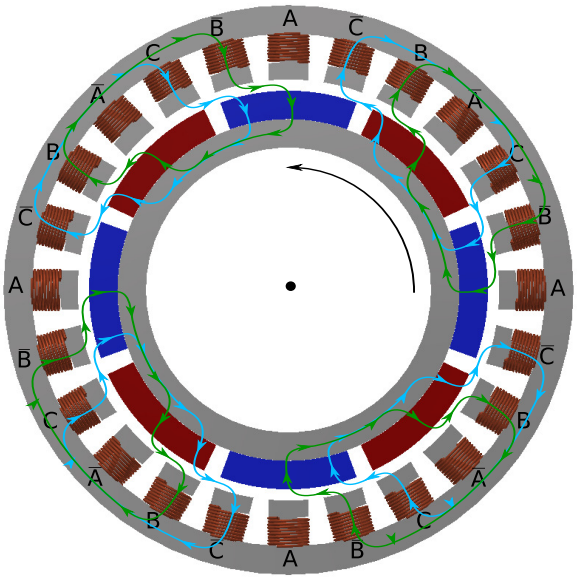


Figure 8.2: Cross section diagram of the motor. Magnetic flux lines are drawn in green and cyan for a stator field being 90 electrical degrees ahead of the rotor.

According to the manufacturer, the motor is rated for 48VDC and up to 300A for a minute. This corresponds well with its 19hp rating, which means, this DC voltage is the input voltage when driving it as a brushless DC motor, and thus a DC supply voltage of 52.8V does not overload the motor. However, due to limitations on the power rail the power peaks at 14.7hp. This happens when equation 8.3 is true.

$$V_{BEMF} = \frac{V_{bat}}{2} - I_{max} \cdot R_a \tag{8.3}$$

Using K_e as defined in equation 8.5 to convert the back-EMF voltage into a velocity, the maximum power is reached at 3186rpm. For controlling the motor it can be considered a 2-pole motor, which means that there are only two magnetic fields; the rotor magnetic field generated by the permanent magnets, and the stator field generated by the solenoids. This is achieved by converting the measured mechanical angle of the rotor to electrical angle, and then doing all calculations in this domain.

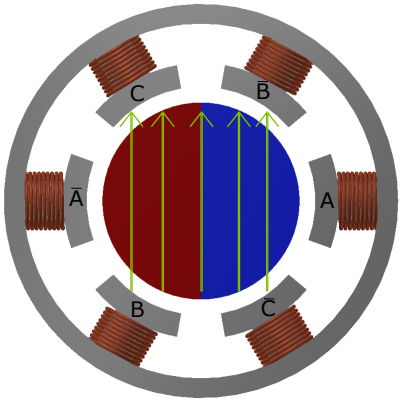


Figure 8.3: Simplified motor model showing the stator field as green arrows.

The electric angle of figures 8.2 and 8.3 are the same, and the stator field is set 90 degrees ahead. In figure 8.3, the rotor magnet will always try to line up with the magnetic field lines. In the real motor, this will cause the magnetic path to become shorter, as the rotor lines up the permanent magnet poles with the opposite pole of the solenoids. By constantly placing the stator field ahead of the rotor, the motor will start moving. Thus it is necessary to know the position of the rotor in order to determine the angle of the stator field. For that measurement, an encoder will be used.

8.3 Definition of Back-EMF Constant and Torque Constant

The parameters K_e and K_t must be the same for a PMAC motor, since they are dependent on the flux produced in the permanent magnet of the rotor. Another reason, that they are the same is, that the motor converts electrical energy to mechanical energy, and electrical energy is the product of current and voltage, and (rotational) mechanical energy is the product of torque and angular velocity. A mismatch between K_e and K_t implies, that power appears or disappears, which cannot happen.
The back-EMF constant, K_e, is defined as the amplitude of each phase-voltage divided with the mechanical angular velocity.

$$K_e = \frac{V_{BEMF}}{\omega_m} \tag{8.4}$$

$$K_t = \frac{T}{1.5 \cdot I_q} \tag{8.5}$$

This means that K_e and K_t are the same, and the torque produced is the q-current multiplied by K_t and 1.5. The mathematical reasoning behind is explained by power conservation in equations 8.6 through 8.9, and by torque contribution in equations 8.11 through 8.14.

Consider an ideal motor producing some torque, T, and running at a constant angular velocity, ω. Losses in the armature resistance and friction are zero. The motor is driven by a quadrature current, I_q, and a quadrature voltage, V_q. Currents and voltages along the d-axis are 0. The electrical power going into the machine must be equal to the mechanical power going out:

$$P_m = P_e \tag{8.6}$$

Mechanical power is the product of torque and angular velocity:

$$P_m = \omega T \tag{8.7}$$

Since there is no phase difference between voltages and currents, electrical power is the product of the RMS values of the current and voltage per phase, times 3:

$$P_e = 3\Big(\frac{1}{\sqrt{2}}V \cdot \frac{1}{\sqrt{2}}I\Big) = V \cdot \frac{3}{2}I \tag{8.8}$$

The angular velocity and torque can be calculated from the voltage and current, respectively, and put into equation 8.7:

$$P_m = \Big(\frac{V}{K_e}\Big) \cdot (\frac{3}{2}I \cdot K_t) \tag{8.9}$$

By setting $K_e = K_t$, equations 8.8 and 8.9 yield the same.

Hence, a factor of 1.5 is multiplied to $V \cdot I$ to get the mechanical power. This factor is tied to K_t: Consider a motor with with an optimal controller, so the d-current is zero, and the q-current is I. The phase current is then calculated by:

$$\begin{bmatrix} I_a \\ I_b \\ I_c \end{bmatrix} = \begin{bmatrix} I \cdot \cos(\theta_f) \\ I \cdot \cos(\theta_f - \gamma) \\ I \cdot \cos(\theta_f + \gamma) \end{bmatrix} \tag{8.10}$$

where θ_f is the electrical angle of magnetic field generated by the stator, and $\gamma = \frac{2\pi}{3}$. In this case, the motor is at constant speed, so the amplitude of all currents are the same.
Each of the three phases then contribute to the total torque, depending on the angle between the rotor and the phase, calculated by:

$$\begin{bmatrix} T_a \\ T_b \\ T_c \end{bmatrix} = \begin{bmatrix} -I_a \cdot K_T \cdot \sin(\theta_r) \\ -I_b \cdot K_T \cdot \sin(\theta_r - \gamma) \\ -I_c \cdot K_T \cdot \sin(\theta_r + \gamma) \end{bmatrix} \tag{8.11}$$

where θ_r is the electrical angle of the rotor.

The total torque is then the sum of the torque contribution of each phase:

$$T = T_a + T_b + T_c \tag{8.12}$$

The torque produced as a function of the angle difference can be calculated with equation 8.13:

$$T(d\theta) = \frac{3}{2}I \cdot K_t \cdot sin(d\theta) \tag{8.13}$$

This can be verified by inspection of figure 8.4. The rotor angle is fixed at 45°. θ_f will go from -135 degrees to 225 degrees. Figure 8.4 shows torque contribution from each of the three phases, where I and K_t are 1.

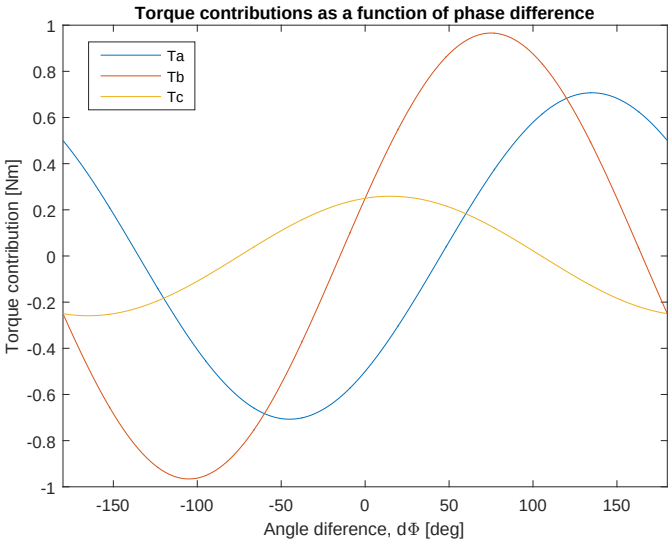


Figure 8.4: Torque contribution of each phase as a function of the phase difference.

Adding these three contributions gives equation 8.13.
If $d\theta = 90°$, equation 8.13 becomes:

$$T = \frac{3}{2} K_t \cdot I \tag{8.14}$$

It is easier to measure K_e than K_t and they are defined as the same.

8.4 Motor Parameterization

Due to the sparse information available from the manufacturer of the motor, it is necessary to measure some of the parameters of the motor. It will also be necessary to determine exactly what these parameters define, since the manufacturer has not defined certain things such as what current measurement is related to torque through K_t. Parameters to determine are the back-EMF constant, K_e, the armature resistance, R_a, and the armature inductance, L_a. Each of the following sections describe the procedure used to find each of these parameters.

8.4.1 Measuring Back-EMF Constant

To find K_e, a hand-held power drill is used to rotate the motor at a constant speed. An oscilloscope with a differential probe is used to measure the voltage between two of the phases. Because of the high input impedance of the differential probe, there is no current and thus the armature inductance and resistance can be neglected. The peak-peak voltage and the corresponding frequency is noted in the table below:
The phase amplitude of the voltage is calculated by dividing the voltages in table 6 with $2\sqrt{3}$. The mechanical angular velocity is calculated by multiplying the frequency in table 6 with 2π, and dividing with 4 pole pairs. The resultant voltages and velocities are divided to calculate K_e, and the average is calculated for these. The result is:

Frequency [Hz]	Voltage [V_{pp}]
23.49	9.7
38.32	15.3
40.97	16.5
44.9	17.7
54.7	21.8
61.81	24.2
72.13	28.3

Table 6: Measured line-line voltage at electrical frequencies.

$$K_e = K_t = 0.0733 \frac{\mathrm{V\,s}}{\mathrm{rad}} \tag{8.15}$$

with a spread of 1.3%. According to the datasheet, K_t is 0.13. This is 1.77 times higher than the measured K_e. There are two potential reasons for this. One is, that the manufacturer uses line-to-line currents and voltages instead, since the difference is close to $\sqrt{3}$. Another possible reason is, that the manufacturer defines K_t as the torque produced with the given q-current, in which case $K_t = 1.5K_e$, and the measured K_t is about 15% below the manufacturer's datasheet. Whatever the reason, equation 8.15 will be used throughout this report.

This means, the maximum speed and torque can be calculated:

$$\omega_{max} = \frac{V_{bat}}{2K_e} = 360 \frac{\mathrm{rad}}{\mathrm{s}} \tag{8.16}$$

$$T_{max} = I_{max} \cdot \frac{3}{2} K_t = 33Nm \tag{8.17}$$

8.4.2 Armature Resistance

To measure the armature resistance, a high DC current is driven through one terminal to the other. The voltage at the terminal is measured. The large current causes a significant voltage drop in the wires leading from the power supply to the motor. Based on the measured voltages and currents the resistance between the two terminals can be calculated. The results are shown in the table below:

Terminals	V [V]	I [A]	R [mΩ]
AB	0.217	17.93	12.10
AC	0.206	17.96	11.47
BC	0.235	17.93	13.11

Table 7: Measured DC resistance across terminal-pairs.

It is desirable to know the resistance from one terminal to the common star point. That can be calculated by equation 8.18

$$R_a = \frac{R_{AB} + R_{AC} - R_{BC}}{2} \tag{8.18}$$

This gives different armature resistances for different phases ranging from 5.2 to 6.9 mΩ. This is about half the line-to-line resistance listed by the manufacturer at 13mΩ. However, since the motor is a wye connection, knowing the resistance between a terminal and the star-point is more helpful when modeling the motor.

8.4.3 Armature Inductance

In order to measure the winding inductance of the motor, a full bridge driver is used to generate a square wave at different frequencies. The two terminals of the full bridge driver is connected to terminals B and C, and voltage and current is measured. The motor is not moving, so there is no back-EMF. The resistor voltage is subtracted from the voltage measurements, and what remains is the inductor voltage. Current and voltage has been plotted in figure 8.5.

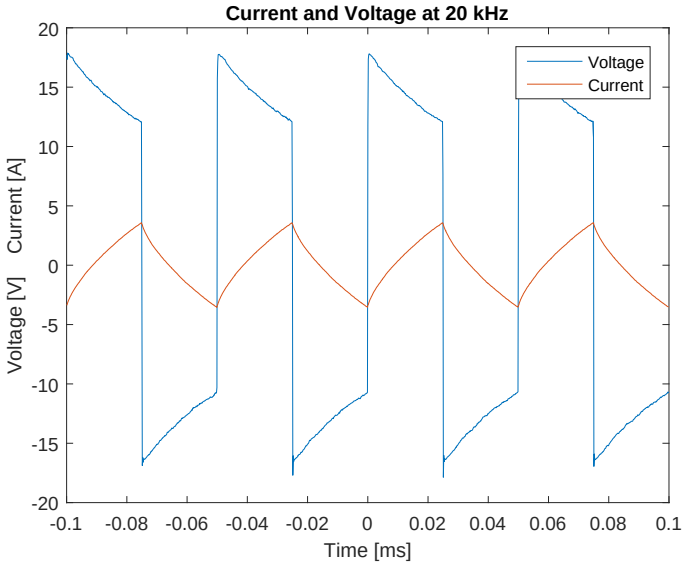


Figure 8.5: 20 kHz square wave voltage and the resulting current.

These waveforms have been measured at 5 kHz, 10 kHz, 20 kHz and 30 kHz. For this project a switching frequency of 20 kHz will be used, so only this plots will be shown. Due to the high sampling rate of the oscilloscope and low bit resolution, differentiating the current will yield very large spikes for short periods of time. Therefore, the current, time and voltage measurements will be periodically averaged and downsampled by a factor of 70. The derivative of the current is then calculated by:

$$\frac{di}{dt}[n] = \frac{i[n] - i[n-1]}{t[n] - t[n-1]} \tag{8.19}$$

This is the derivative of the current at the point immediately between two samples, so to synchronize the time and voltage arrays, each two consecutive samples are averaged. The resulting waveforms are plotted in figure 8.6
It appears, that there is some proportionality between the derivative of the current, and the voltage across this inductor.

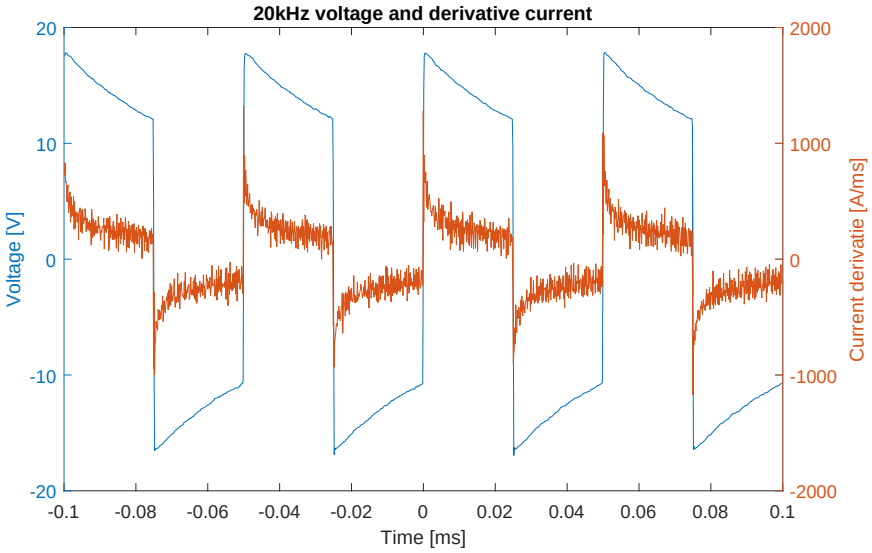


Figure 8.6: 20 kHz square wave voltage and derivative of the resulting current.

The inductance is generally lower than the terminal-to-terminal inductance provided by the manufacturer of 0.1mH. The average inductance varies a bit with different frequencies, and 20kHz is actually the lowest.

Frequency[kHz]	Average inductance[mH]	Current ripple[A]
5	0.946	16.83
10	0.844	12.10
20	0.632	7.13
30	0.879	5.25

Table 8: Average of measured inductance and current ripple at different frequencies.

The inductances in table 8 is the inductance from one terminal to another, so the per-phase inductance will be half. The inductance determines the phase shift on the sinusoidal voltages and currents driving the motor, and the current ripple at the switching frequency. The maximum frequency of these are determined by the maximum rotational speed of the motor calculated in equation 8.16, which equals 229Hz. So when modelling the motor, one must settle on a value of L_d and L_q that is close to the the low frequency inductance of 0.473mH, and the higher frequency inductance of 0.316mH. For modelling the motor, an inductance of 0.4mH will be used for both L_d and L_q.

8.5 Parking Test

The encoder is positioned on the axle of the motor at an unknown angle. Since this will not change, a one time parking test will be performed, where current will be passed into one terminal and out of the other two. This will be done with a DC power supply with current limit. The motor should position itself at one of four mechanical angles spaced 90° apart. This offset will be subtracted

from any angular readings.

The motor is parked with 30A. The test is repeated to determine all four parking spots for each phase, and the angles are shown in table 9. The phases A, B and C correspond to the terminals with the names M1, M2 and M3 etched into the case of motor.

Quadrant	Phase A	Phase B	Phase C
First	43.6°	75.9°	16.9°
Second	132.2°	167.3°	101.3°
Third	223.6°	257.3°	194.1°
Fourth	312.2°	348.8°	279.8°

Table 9: Parking angles for the encoder.

The four parking angles of each phase should be spaced 90 degrees from each other. The parking angles in one quadrant should be spaced 30 degrees apart. This does not hold true for all values - both columns A and B are spaced 90° apart within the precision of the encoder, but the difference between phase A and B in the same quadrant are spaced 34.45° apart. Column C varies from 84° to 97°, however they are on average spaced 30° from Column A. This should definitely not be the case, but the reason is currently unknown. For the controller, the Phase A readings will be used throughout the report. The angles are then wrapped down to the first quadrant by subtracting 90, 180 and 270 degrees from second, third and fourth quadrant respectively, and the average is calculated to be 42.9°, corresponding to a value of 30.5° from the encoder. Whenever the rotor position is read, 30.5 will be subtracted from the value.
It should also be noted, that the order of the phases is A, B and C, which is the opposite order of the one described in section 8.2. This means, that the encoder counts down when the go-kart is moving forward.

9 Controller Design

Controlling the PMAC motor will be done using the principles of field oriented control, which uses the Clarke Park Transform (dq0 transform) to transform three phase AC into two phase DC, allowing for easier regulation. Controlling the PMAC motor is done by varying the PWM input to the inverter. This is done by monitoring two of the three current inputs to the motor and transforming them using the dq0 transform as discussed in 8.1.

The requirement for this project outlines that the motor must be torque controlled, hence the flux demand will be zero while the torque demand will be defined by the torque pedal.
The procedure of torque control is illustrated by figure 4.2, which is repeated here for convenience.

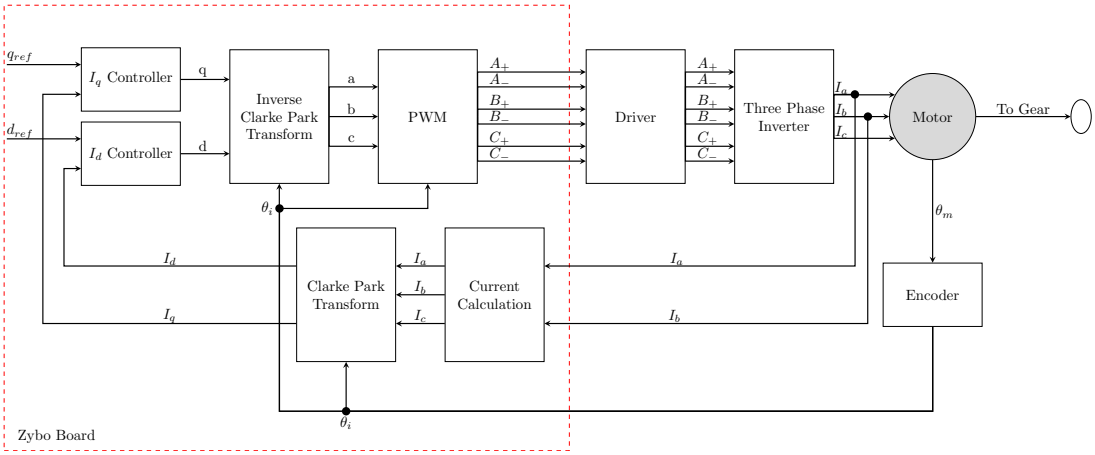


Figure 9.1: The control process as illustrated in the system overview section.

The three phase input to the motor is measured and transformed to dq. After the transformation, the two signals, I_d and I_q are compared to the reference values and fed to the controllers. The I_d reference will be zero as this is the flux reference. The I_q reference is set by the torque pedal. The controller acts upon these values ant they are transformed back to ABC three phase AC. After this PWM is adjusted and the signals are routed through the driver to the inverter.
The torque reference, I_q, is in fact a current reference. As such, torque control is equal to current control. Taking this into account will be part of creating a functional model of the motor.

To model the PMAC motor, an equivalent circuit of the motor will be used. The equivalent circuit can be represented by the diagram presented on figure 9.2.

This figure shows that the q-axis of the PMAC motor can be represented as an electrical diagram with the physical properties represented by inductors and resistors.
The components shown are: the mutual inductance l_s and L_{mq}. Armature resistance R_s and flux linkage. The most important values here are the q-axis inductor and resistance which directly affect how much torque is gained from inputting current to the PMAC motor. During this section the components above will be named as follows.

$$R_s = R_a, \quad L_{mq} = L_a \tag{9.1}$$

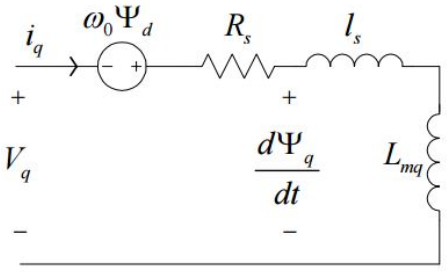


Figure 9.2: PMAC motor equivalent circuit, q-axis.

To current control the motor, the model and controller will be developed for this equivalent diagram. The q-axis by itself resembles the equivalent circuit of a DC motor and this can be used to design a mathematical model of the motor.

9.1 State Space Model

A state space model is derived from the physical representation of a PMAC motor, figure 9.3, consisting of an electrical part on the left and a mechanical part on the right side of the model. The model is made in MatLab Simulink. The model is very similar to that of a DC motor when interpreted like this.

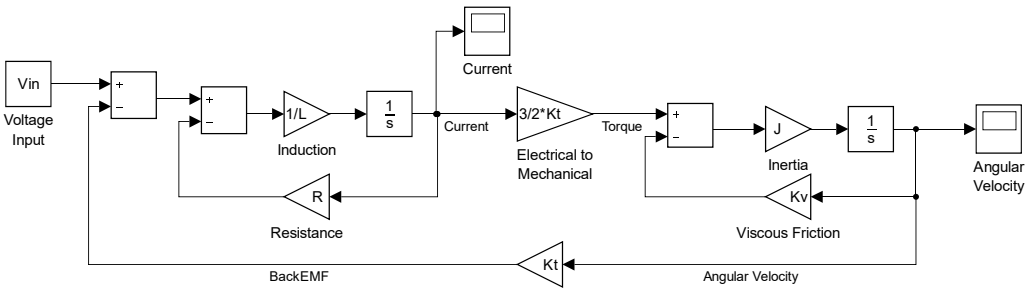


Figure 9.3: PMAC motor state space model.

Reducing the model of figure 9.3 to use transfer functions for the electrical and mechanical parts results in a more simple model with fewer loops. It is shown on figure 9.4. In the electric part of the circuit a new value TS has been added to the denominator of the transfer function, this value is the electromagnetic time constant of the stator, defined as $TS = L_a/R_a$. The constant defined J_{pro} is the projected inertia of the go-kart, based on the expected weight with a driver and also adding the inertia of each gear.

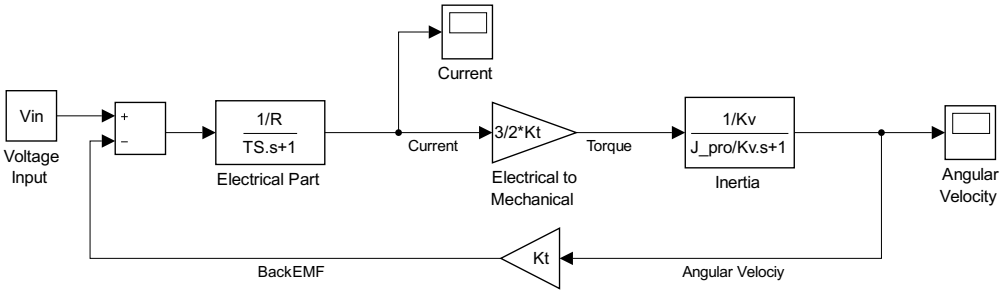


Figure 9.4: Transfer function PMAC motor model.

9.1.1 No Inductor Model

The model from figure 9.3 can be reduced by removing the inductance completely, this yields a model of reduced order. Reducing the order results in a model that can be easier to derive a controller for, since it removes a pole from the system. This reduced order model is seen on figure 9.5.

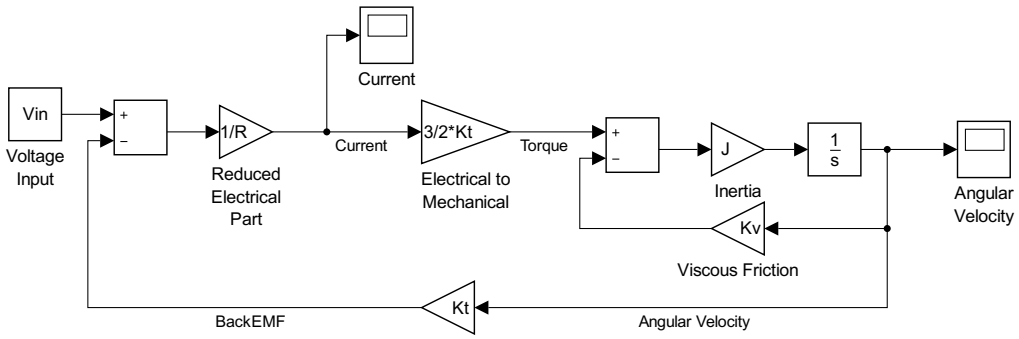
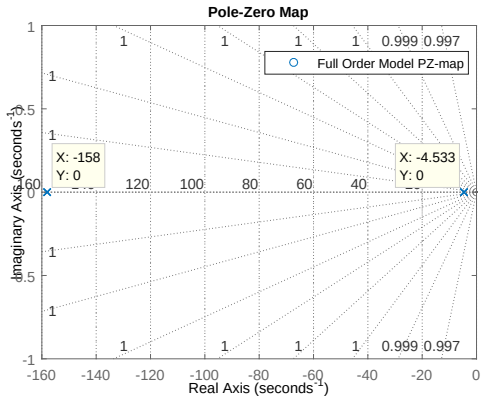
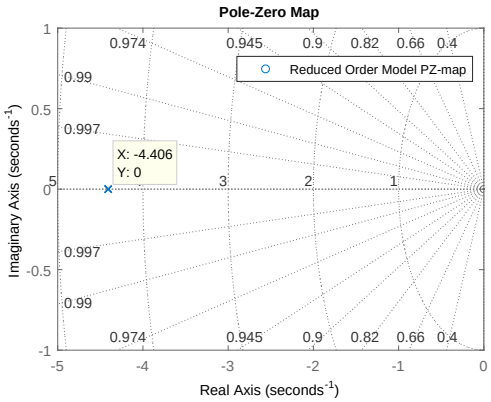


Figure 9.5: Reduced order PMAC motor model.

A pole-zero map of both models will be used to see whether the inductor can be removed without affecting the system significantly. Figure 9.6a and 9.6b show this.



(a) PZ map of full order model.

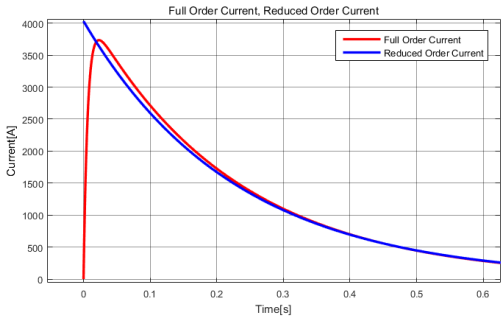


(b) PZ map of no inductor model.

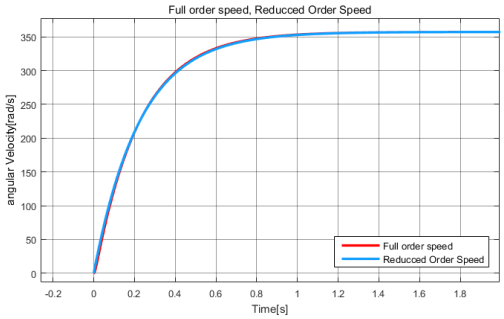
Figure 9.6: Matlab Simulink PZ Map of each model.

The pole-zero plot shows that the inductor pole is much faster than the mechanical pole, so it should be possible to remove it.

The two state space models are simulated with MatLab Simulink to see how the system is affected by the removal of the inductor. A constant voltage of 26.2V is applied to each model. Figure 9.7a and 9.7b show these simulations.



(a) Simulation of current for full and no inductor model.



(b) Simulation of speed for full and no inductor models (almost stacked on each other).

Figure 9.7: Matlab simulation of each model with the same input.

A problem shows up in these simulations. The speed behaves well for the no inductor model, but the current starts at a high value. The current starts out as a simple multiplication of the input voltage multiplied by the $1/R$ gain to equal 4000A. The amplitude of speed and current are not important at this time, as that will be controlled later, but since the model does not ramp up the current like the full order model, a controller will not be derived for this reduced order model.

9.1.2 Reduced Order Model - No Mechanical Part

Because of the problem with reducing the model by removing the inductor, this section will focus on reducing the order by removing the mechanical part. This model will be an experiment to

remove the slow pole from the full order model. This will affect the system significantly. The mechanical system will now be seen as a disturbance. It can be seen on figure 9.8.

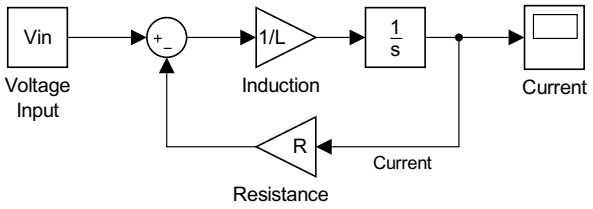


Figure 9.8: Reduced order PMAC motor model version.

The model is basically an LR circuit where the output is the current. Which will just go as high as the L and R allows and stay there, since the back-EMF is not included.

Figure 9.9 shows the current.

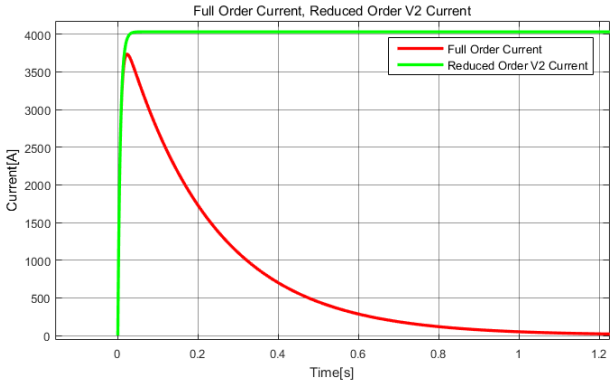


Figure 9.9: Full order vs reduced order model.

This shows that the current ramps up and stays there. The current starts at zero and ramps up. The downward slope of the full order model is caused by the back-EMF which is removed in the reduced model.

9.2 Controlling Current

The motor must be current controlled in order to control the torque. The torque pedal is used as the reference for the current controller, thus indirectly controlling the torque.

A control loop is added to the model of figure 9.4. This means that a feedback loop is added from inside the motor model and fed back to the controller. On figure 9.10 it is shown with the full order system.

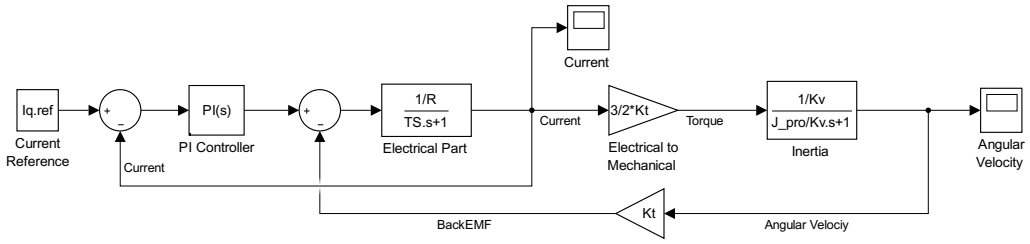


Figure 9.10: Full PMAC motor model with a current control loop.

It would be preferable to have a second order controller for a second order system but the PI itself is preferred to avoid using a derivative in the discretized controller. A derivative term can become very inaccurate at high frequency. To improve precision it must be slowed down, but then the controller might be too slow all together. Hence if the derivative term can be avoided it will be. It should be able to control the 2nd order system based on the fact that one of the poles are very fast. The following sections will determine how well the PI controller can control the system.

9.3 Controlling the Full Order System

The model of figure 9.10 can be rearranged to simplify the transfer function. This rearrangement can be seen on figure 9.11. Here the current is the output, a current reference is the input, and the mechanical part of the motor is all placed into a feedback loop. This reduces the complexity of the overall transfer function to only be a plant model with a PI controller added as shown in figure 9.12.

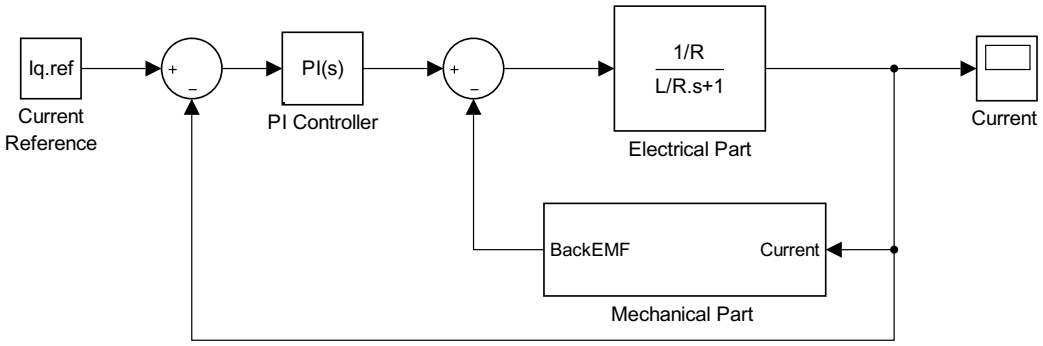


Figure 9.11: PMAC motor model rearranged to simplify the transfer function.

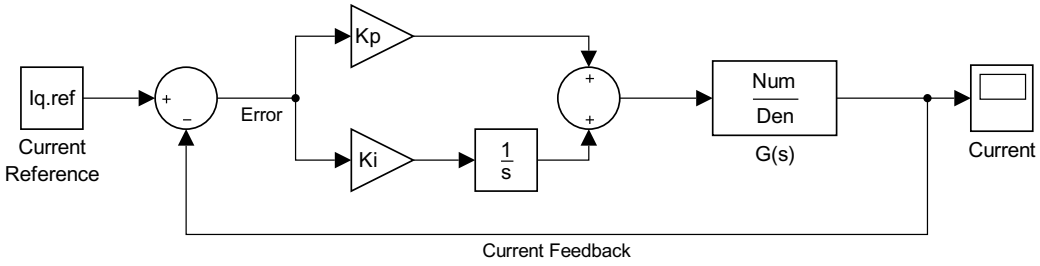


Figure 9.12: PMAC motor model as a plant with PI control.

Using 'Masons Rule'[1], the transfer function can be derived. The following equation is achieved:

$$\frac{I_q}{I_{q_{ref}}} = \frac{(K_p + K_i\frac{1}{s})G(s)}{1 + G(s)(K_p + K_i\frac{1}{s})} \tag{9.2}$$

$G(s)$ from figure 9.3 is:

$$G(s) = \frac{2J_{pro}s + 2K_v}{s^2(2J_{pro}L_a) + s(2K_vL_a + 2J_{pro}R_a) + 3K_t^2 + 2K_vR_a} \tag{9.3}$$

The denominator can be reduced by dividing everything with $2J_{pro}L_a$. yielding the transfer function in equation 9.4.

$$G(s) = \frac{\frac{1}{L_a}s + \frac{K_v}{J_{pro}}}{s^2 + s\left(\frac{K_v}{J_{pro}} + \frac{R_a}{L_a}\right) + \frac{3}{2}\frac{K_t^2}{J_{pro}L_a} + \frac{K_vR_a}{J_{pro}L_a}} \tag{9.4}$$

Inserting 9.4 in 9.2 and rearranging to get the form of a characteristics equation the transfer function is equation 9.5.

$$\frac{I_q}{I_{q_{ref}}} = \frac{s^2\frac{K_p}{L_a} + s(\frac{K_i}{L_a} + \frac{K_pK_v}{J_{pro}}) + \frac{K_iK_v}{J_{pro}}}{s^3 + s^2(\frac{R_a}{2L_a} + \frac{K_v}{J_{pro}} + \frac{K_p}{L_a}) + s(\frac{3K_t^2}{2J_{pro}L_a} + \frac{K_vR_a}{J_{pro}L_a} + \frac{K_i}{L_a} + \frac{K_pK_v}{J_{pro}L_a}) + \frac{K_iK_v}{J_{pro}L_a}} \tag{9.5}$$

This third order transfer function will be used to derive the controller values.

9.4 Full Order Controller Values

To find the controller values the Stephen Dodds Settling Time Formula for 5% settling time is used[1], The desired characteristics polynomial of a transfer function for the settling time formula to work is equation 9.6.

$$s^n + d_{n-1}s^{n-1} + \cdots + d_1s + d_0 \tag{9.6}$$

The d's of a given order can be derived by solving $(s+d)^n$ for the given order of the characteristics equation. This yields the following third order equation for a third order system.

$$s^3 + 3\alpha s^2 + 3\alpha^2 s + \alpha^3 \tag{9.7}$$

Based on the transfer function in equation 9.5 and a settling time of 0.05 seconds, α will be defined by the following equation:

$$\alpha = \frac{1.5(1+n)}{T_s^{5\%}} = \frac{6}{0.05} = 120 \tag{9.8}$$

Characteristics equation of 9.3 is equation 9.9.

$$s^3 + s^2\left(\frac{R_a}{2L_a} + \frac{K_v}{J_{pro}} + \frac{K_p}{L_a}\right) + s\left(\frac{3K_t^2}{2J_{pro}L_a} + \frac{K_vR}{J_{pro}L_a} + \frac{K_i}{L_a} + \frac{K_pK_v}{J_{pro}L_a}\right) + \frac{K_iK_v}{J_{pro}L_a} \tag{9.9}$$

Grouping constants yields the following form:

$$s^3 + s^2(a_2 + b_2K_p) + s(a_1 + b_1K_i) + a_0 + b_0 \tag{9.10}$$

Setting this equation equal to 9.7 yields:

$$s^3 + s^2(a_2 + b_2K_p) + s(a_1 + b_1K_i) + a_0 + b_0 = s^3 + 3\alpha s^2 + 3\alpha^2 s + \alpha^3 \tag{9.11}$$

Solving the above equation for K_p and K_i yields the equation two equations 9.12 and 9.13.

$$K_p = (3\alpha - \frac{K_v}{J_{pro}} - \frac{R_a}{2L_a})L_a \tag{9.12}$$

$$K_i = (3\alpha^2 - \frac{K_pK_v}{2J_{pro}} - \frac{K_vR}{J_{pro}L_a} - \frac{3K_t^2}{2J_{pro}L_a})L_a \tag{9.13}$$

These equations yield the following controller values when inserting the known motor parameters:

$$K_p = 0.0111 \tag{9.14}$$

$$K_i = 1.7566 \tag{9.15}$$

The pole-zero map for the transfer function including the controller confirms that a new pole and zero has been added. All poles and zeroes lie on the real axis, meaning they affect the system by altering the decay while no oscillation is introduced. The system is stable with these values since they are all on the left side of the imaginary axis. See figure 9.13. How the controller affects the output will be tested further on.

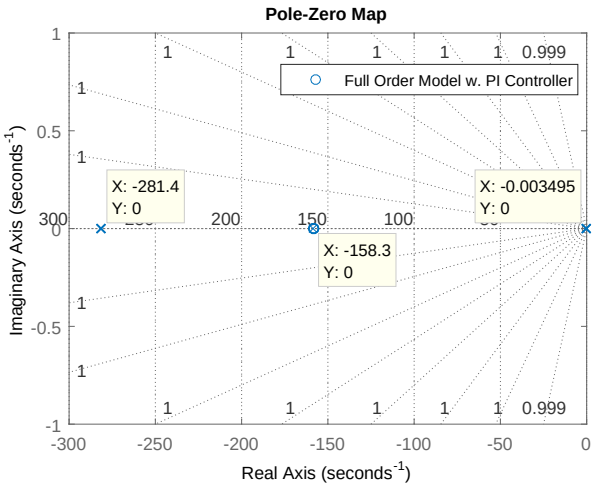


Figure 9.13: PZ map with the controller.

9.5 Controller for the Reduced Transfer Function

In section 9.1.2 a reduced order model of the PMAC motor was derived. This model removed the mechanical part of the model to reduce the order. This section will develop a controller for this model.
The same procedure as for the full order system will be used for this transfer function. The open loop transfer function is equation 9.16.

$$\frac{I_q}{V_{in}} = H(s) = \frac{1}{R_a + L_a s} \tag{9.16}$$

9.6 Reduced Order Controller Values

Finding the controller values for this model is done as it was for the full order model. However an IP controller will be used instead to avoid adding a zero to the transfer function.
The IP controller places the K_p value in a feedback loop resulting in the following diagram on figure 9.14.

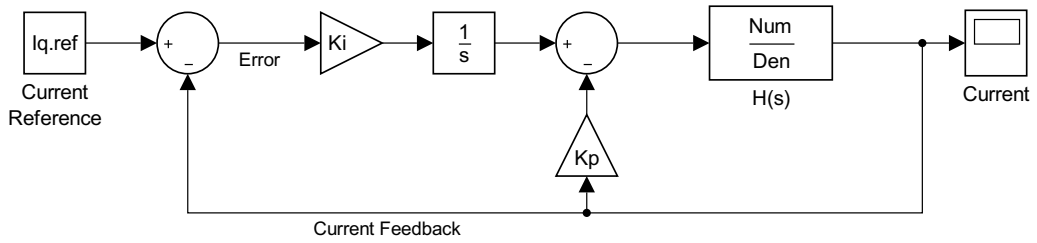


Figure 9.14: Reduced order system with an IP controller.

The closed loop transfer function of figure 9.14 is:

$$\frac{I_q}{I_{qref}} = \frac{\frac{K_t}{L_a}}{s^2 + s(\frac{R_a}{L_a} + \frac{K_p}{L_a}) + \frac{K_t}{L_a}} \tag{9.17}$$

Using Dodd's 5% settling time formula[1] and solving for a 2nd order system and a settling time of $0.05s$. The results are equation 9.18

$$\alpha = \frac{1.5(1+2)}{0.05} = 90 \tag{9.18}$$

The desired characteristics equation has the form of the following equation (9.19).

$$s^2 + sd_1 + d_0 = s^2 + 2\alpha s + \alpha^2 \tag{9.19}$$

Solving for K_p and K_p and inserting the known motor parameters yields the following controller values:

$$2\alpha = \frac{R_a}{L_a} + \frac{K_p}{L_a} \rightarrow K_p = 7e^{-4} \tag{9.20}$$

$$\alpha^2 = \frac{K_i}{L_a} \rightarrow K_i = 0.324 \tag{9.21}$$

The closed loop transfer function results in the pzmap of figure 9.15. Since there are no zero in the transfer function, there wont be one when adding an IP controller. The added pole becomes a complex conjugate pair. This means that the system is stable but might have oscillation. This will be tested in a simulation in the next section.

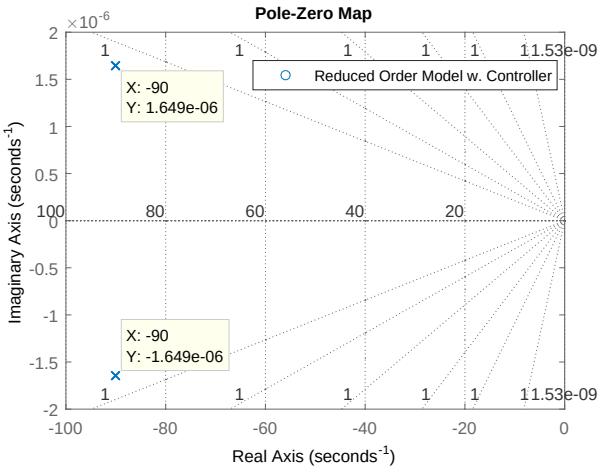


Figure 9.15: PZ map of the reduced order system with an IP controller.

9.7 Control Simulation

The derivations above have resulted in controller gain values as seen in table 10.

Model/Gain	K_p	K_i
Full Order	0.0111	1.7566
Reduced Order	$7 \cdot 10^{-4}$	0.324

Table 10: K_p and K_i derived in previous sections.

The controllers should produce a stable current output from the electrical part of the motor, which is desired to control the torque.

Figure 9.16 is the input reference signal which simulates the torque pedal being fully actuated, then held for a few seconds and then released over two seconds. Full actuation is set to 300A and the run time is set to 10s.

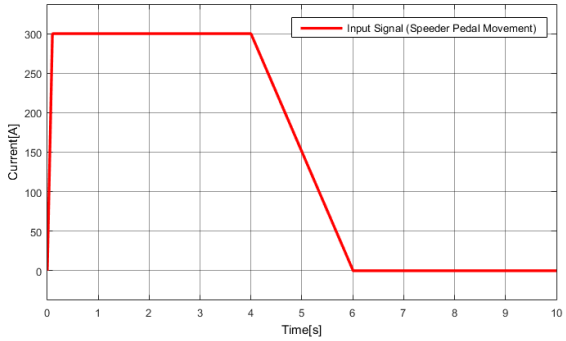
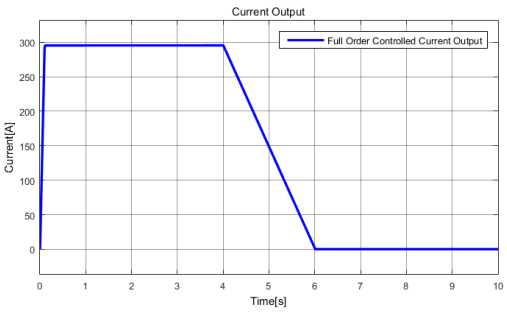
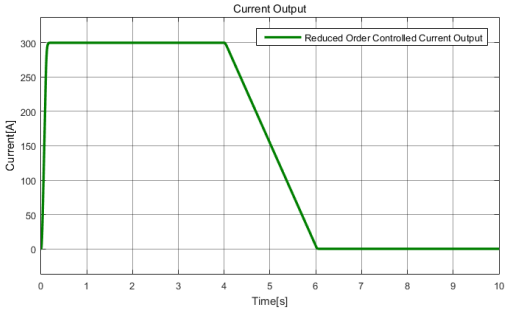


Figure 9.16: Input reference for the controller.

Figure 9.17 shows how both models perform with the controller values found above and the input signal. The full order model has a small steady state error of slightly less than $5A$ while the reduced order model is spot on. The settling time for both is higher than the 0.05 value set for the settling time formula though. The full order model settles in $0.9s$ while the reduced model settles is $0.13s$. They both handle the control as they should, with no overshoot at the start, or undershoot at the end of the breaking.



(a) Current output response of the full order model to the signal input given in figure 9.16.



(b) Current output response of LC model to the signal input given in figure 9.16.

Figure 9.17: Simulations of each model with their controllers.

The controller to be used for the go-kart will be determined by running a more in depth simulation in the next section, this simulation will include the physical systems, breaks, torque etc. A choice will be made based on controller performance in these tests.

9.8 Discretization

Discretizing the controller is required to use it in the software. To do this, the values of K_p and K_i can be used as they are, but for both a PI and IP controller the integrator must be discretized. An integrator in discrete is done by using a numerical integration method. The choice here will be the Trapezoidal method, which has the output equation given as equation 9.22.

$$y_{n+1} = y_n + \frac{1}{2}h\left(f(t_n, y_n) + f(t_{n+1}, y_{n+1})\right) \tag{9.22}$$

Here n is the time step, t is the time and y is the function to integrate. The trapezoidal method

yields a more precise method than forward or backwards Euler methods, due to it being 2nd order precise ($O(h^2)$), but it takes twice the computation power.

9.9 Third Harmonic Injection

Due to the limited battery voltage, it is not possible to drive the motor to speeds close to the maximum set by the manufacturer. Because the internal star point of the motor is not connected to the outside, it is possible to gain an additional 15% by way of third harmonic injection. The reason for this is, that the controller can add the same voltage offset to all three terminals of the motor without changing the currents, thus avoiding the peaks of the sinusoidals. There are several ways of doing this, but only the sinusoidal will be considered here. Adding a $\frac{1}{6}$ of the third harmonic will result in 15 % more line to line voltage. The resulting waveforms can be seen in section 10.10, on page 69.

The issue is that, when using field oriented control, the stator voltage does not remain at a constant angle relative to the rotor angle. That means, that calculating the third harmonic isn't as simple as multiplying the rotor position with three. Doing this will result in the the third harmonic being out of phase with the three sinusoidal voltage waveforms as speed increases. This will result in higher peak voltages at certain speeds. Instead, the third harmonic needs to be calculated from the dq output from the controller.

Equation (9) in the paper[10] defines the three phase voltage of a balances system without third harmonic injection:

$$\begin{bmatrix} V_A \\ V_B \\ V_C \end{bmatrix} = \begin{bmatrix} V\sin(\Theta) \\ V\sin(\Theta - 120°) \\ V\sin(\Theta + 120°) \end{bmatrix} \tag{9.23}$$

where Θ is the instantaneous phase, and V is the instantaneous magnitude, calculated as:

$$V = \sqrt{V_q^2 + V_d^2} \tag{9.24}$$

Note that equation 9.24 is not equal to equation 10 in the paper[10], because this dividing with 1.5 is done in the Clarke Park transform block.
From equation 9.23, we know that:

$$\Theta = \sin^{-1}\left(\frac{V_A}{V}\right) \tag{9.25}$$

The third harmonic sinusoidal is given in equation (12) of the paper[10], and shown in equation 9.26

$$V_{th} = \frac{1}{6}\sin(3\Theta) = \frac{1}{6}\sin\left(3\sin^{-1}\left(\frac{V_A}{V}\right)\right) \tag{9.26}$$

This third harmonic sinusoidal is added to each of the phases, as shown in equation 9.27

$$\begin{bmatrix} V_{Ath} \\ V_{Bth} \\ V_{Cth} \end{bmatrix} = \begin{bmatrix} V_A + V_{th} \\ V_B + V_{th} \\ V_C + V_{th} \end{bmatrix} \tag{9.27}$$

10 Simulation

Simulation tools have been a great help, not only in understanding the electromechanical system, and design a controller from it, but also in estimating requirements for the electrical components. Both Simulink and Plecs Standalone have been used, as they excel at different purposes. Simulink is used to test the controller on a system with as much realism as possible to make sure the go-kart functions, whereas Plecs is used to simulate the inverter to determine whether there will be heating issues during operation.
Simulink will be used to simulate how the controller handles certain events:

- Actuation of torque pedal.
- Speed limitation due to voltage limit.
- Release of torque pedal.
- Actuation of brake pedal - with and without wheel lock.

In all cases, the controller should ensure that the motor produces the requested torque. The go-kart does not have regenerative braking, so negative torque will be a problem. The Plecs simulations do not need these details, but the mechanical system still needs a representation of mass of the car, the gear, the wheel and wind resistance.

10.1 Mechanical System

The mechanical system consists of a mass of the go-kart with driver, turbulent air resistance, wheels and a gear.

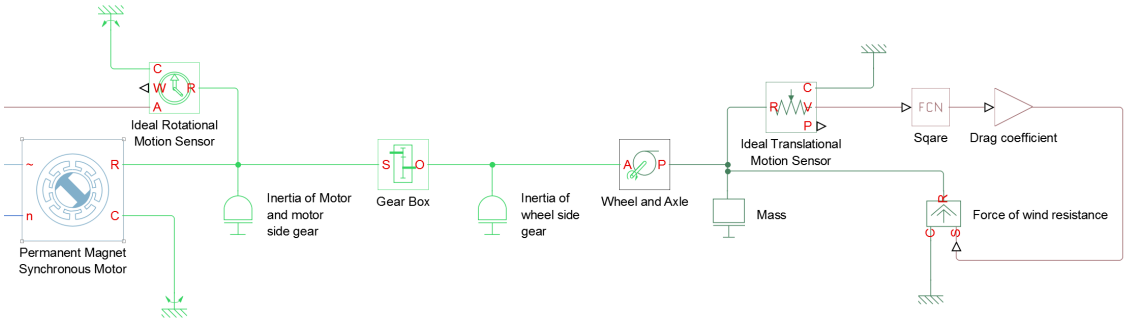


Figure 10.1: Block diagram of the mechanical system.

Figure 10.1 shows the block diagram of the mechanical part, that the motor will drive. Starting from the left, the Permanent Magnet Synchronous Motor is the motor and the Ideal Rotational Motion Sensor is used for the Clarke Park transformations. The first inertia block contains both the inertia of the motor, which is 0.0052 kg m^2 and the motor-side gear, which can be calculated by equation 10.3. Inertias on the same rod can be added together, so instead of having an inertia for the motor, and one for the motor-side gear, one inertia is enough. The inertia of the gear depends on its size. Assuming, that the gear is a disc, the mass is calculated by:

$$m_{G1} = \rho \pi r^2 \cdot h \tag{10.1}$$

where ρ is the mass density of iron of $7870\frac{kg}{m^3}$, r is the radius, and h is the thickness of 7mm. Radius is defined by the number of teeth of the gear, G, and the pitch, which is the distance between two adjacent teeth, which is 12.5mm. Hence the radius can be calculated by equation 10.2

$$r = \frac{G \cdot d_p}{2\pi} \tag{10.2}$$

where d_p is the pitch, and G is the number of teeth of the gear.

Inertia of a disc depends on mass and radius of the disc according to equation 10.3.

$$J_G = \frac{mr^2}{2} = \frac{\rho\pi r^2 \cdot h \cdot r^2}{2} = G^4 \frac{\rho\pi \frac{d_p}{2\pi} \cdot h}{2} \approx 1.36 \cdot 10^{-9} G^4 \tag{10.3}$$

This equation is used to dertermine the inertia of the two gears, the motor-side gear, $G1$, has 12 teeth, and the wheel-side gear, $G2$, has 50 teeth. This ratio, $G2/G1$ is put into the block "Gear Box" on figure 10.1. Transferring torque and angular velocity through a gear according to equation 10.4.

$$\begin{aligned} T_2 &= \frac{G_2}{G_1} T_1 \\ \omega_2 &= \frac{G_1}{G_2} \omega_1 \end{aligned} \tag{10.4}$$

Since power is the product of torque and speed, which means that power is conserved through the gear.
The mass is set to 150 - 250kg, depending on the driver, and this includes the mass of the car. This is done to test different situations – a high mass would have more stable speeds, but also cause more power loss in the inverter, whereas a lower mass would be more prone to wheel spin.

Wind resistance represents a larger friction than any other frictions, once the go-kart reach higher speeds. Wind resistance of a go-kart is primarily turbulent, and can be calculated by equation 10.5.

$$F = -\frac{1}{2} \rho A c v^2 \tag{10.5}$$

where ρ is the density of air, A is the frontal area, c is the drag coefficient and v is the speed. The frontal area has been approximated to two rectangles with the combined area of 0.6 m^2. The density of air is $1.225\frac{\text{kg}}{\text{m}^3}$, c is approximately 0.8[13]. The constants are multiplied into one constant called c_{drag}, as shown in equation 10.6

$$F = c_{drag} v^2 = -0.296 v^2 \tag{10.6}$$

These constants are put into the gain block, "Drag coefficient", and multiplied by the square of the speed. The result is put into the "force of wind resistance" block, as seen on figure 10.1.

This system can be simplified, so that all inertia and mass is combined in one block and the gear and wheel can be removed. This is done by a set of rules that apply for this mechanical circuit: This gear box reduces the speed, and increases torque, much like a transformer reduces voltage and increases current. Rotational and translational mechanics are tied together by equation 10.7.

$$\begin{aligned} T &= J \cdot r \\ v &= \omega \cdot r \end{aligned} \tag{10.7}$$

To turn the mass of the vehicle, m, into an inertia, assume that the entire mass is mounted on the surface of the driving wheels. This means, the inertia can be considered a shell with mass m, and radius r, in which case, inertia can be calculated as:

$$J = mr^2 \tag{10.8}$$

Since the mass is now transformed into an inertia, it can be added to the inertia J_{G2}.

Same rules apply for a gear as for a transformer when reflecting a load from one side to the other. This means, that a torque source driving an inertia through a gear, is experiencing an inertial load calculated by equation 10.9.

$$J_{ref} = \frac{G1^2}{G2^2} J \tag{10.9}$$

This inertia is then added to the inertia of the motor and J_{G1}:

$$J = (mr^2 + J_{G2}) \cdot \left(\tfrac{G1}{G2}\right)^2 + J_{G1} + J_M \tag{10.10}$$

In equation 10.6, speed can be replaced with angular velocity and a gear ratio, and force can be replaced by torque and the gear ratio, according to equation 10.7. So the equation becomes this:

$$\frac{TG2}{rG1} = c_{drag} \cdot \left(\omega r \tfrac{G1}{G2}\right)^2 \tag{10.11}$$

Isolating Tresults in:

$$T = c_{drag}\left(\frac{G1r}{G2}\right)^3 \omega^2 \approx -11.1 \cdot 10^{-6}\omega^2 \tag{10.12}$$

The mechanical diagram is reduced to figure 10.2

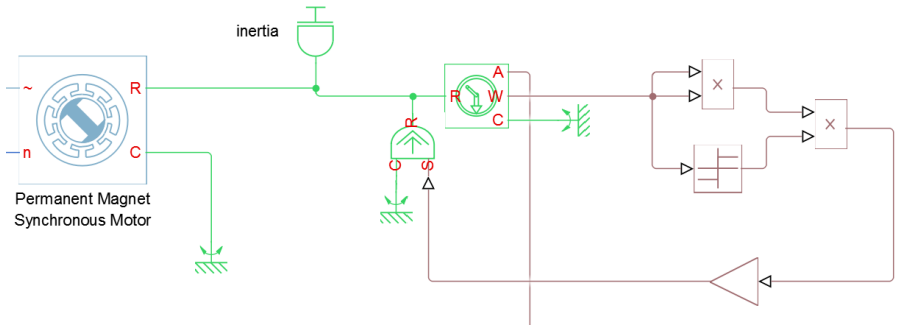


Figure 10.2: Block diagram of the reduced mechanical system.

In the Simulink model, the tyre is modelled by a friction block in series with the rotor, and the brake is modelled in a subsystem, as seen on figure 10.3.

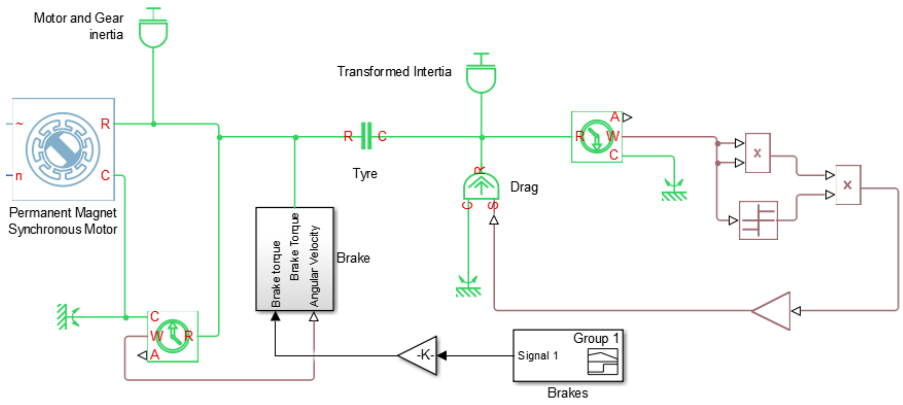


Figure 10.3: Block diagram for the mechanical system used in Simulink.

The tyre block consists of static and dynamic friction. Tyre slip is a comprehensive area of study, so the rotational friction block isn't necessarily accurate. However, it does enable sudden changes in motor speed and load, similar to spinning and locking the wheels. The tyre block consists of static, dynamic and viscous friction. The static and dynamic frictions are are calculated by equation 10.13

$$F_f = F_N \cdot c_f \cdot dm \tag{10.13}$$

where

$$F_N = mg \cdot dm \tag{10.14}$$

and c_f is either the static or dynamic friction, depending on if the wheels are slipping. The constant, dm is the distribution of mass so that $dm = 0$ means the entire mass of the go-kart is on the front wheels, and $dm = 1$ means the whole weight distribution is on the driving wheels. The viscous friction is zero. The static friction coefficient is 1, the dynamic friction is 0.7, and the mass distribution is 0.6.
The brake subsystem is a PI controlled ideal torque source, which will attempt to bring the go-kart to a halt when the brake is pressed. Without a PI controller, there is a strong possibility that the brake would suddenly make the motor go backwards instead of stopping.
Lastly the inertia is split into two inertia blocks, one containing motor and gear inertia on the left of the tyre, and the inertia due to the mass of the car on the right. The brake on the go-kart is mounted on the rear axle, which means the torque it produces must be scaled down to represent the torque seen on the motor rod.

10.2 Motor Model

Both Simulink and Plecs have built in models of PMAC motors. They have the four basic parameters: Number of pole pairs, flux linkage of the magnet, inductance and armature resistance. The parameters used for simulations are listed in table 11.
One thing to note is that the flux linkage is divided by the number of pole pairs. The reason for this is likely a matter of definitions, and the relation has been deduced using simulations. Armature resistance and inductances are per-phase, and the values used are found in section 8.4.

Parameters	Value
Flux linkage	0.0183
L_d, L_q, L_0	$4 \cdot 10^{-5}$
Rs	6.5e-3
Pole pairs	4

Table 11: Parameters used in simulations.

10.3 Electrical Network and Control

The electrical network along with the discrete controller and modulation blocks are shown in figure 10.4. This section will explore the performance of the controllers.

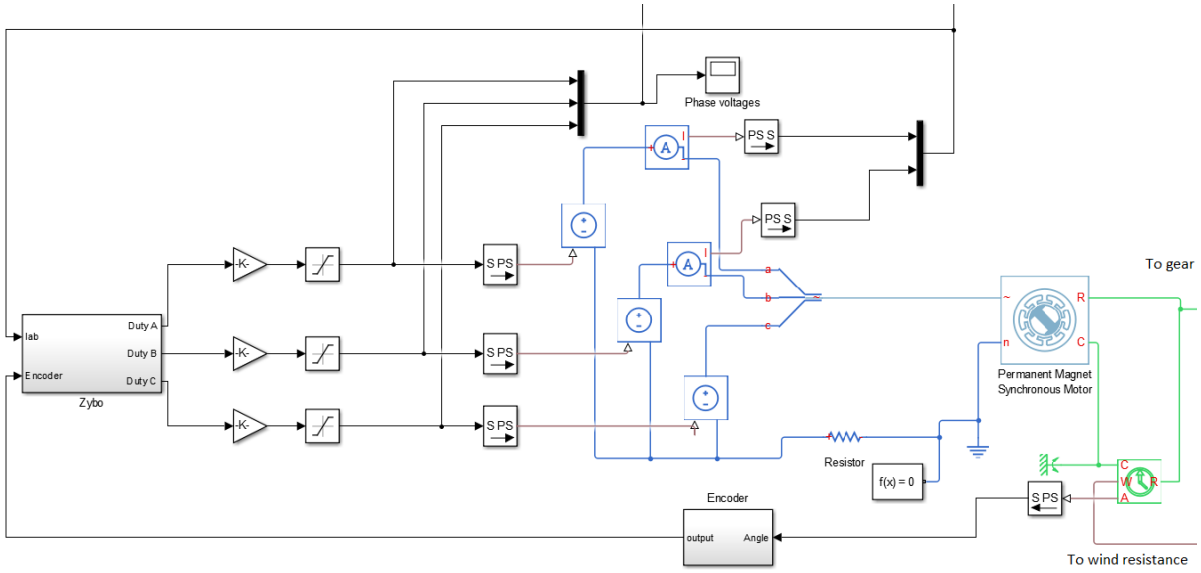


Figure 10.4: The Simulink electrical network and modulation.

The motor block has an external connection to neutral, which the real motor doesn't have. This neutral seems to need a DC path to ground, and so does the controlled voltage sources. Since the external ground cannot be connected to the internal star point of the motor, the connection is made with a very large resistor of 1GΩ. The lighter blue wire going into the "~" port of the PMSM block is a three phase electrical cable, which is used throughout the SimPowerSystem sublibrary, and the Splitter collects three wires into a cable.

Current is sensed on wires A and B, and used to calculate I_C in the Zybo block.
The angular position is measured with an ideal position sensor, and then sent to the encoder block. Here, the finite precision of the encoder is simulated by equation 10.15

$$output = \left\lfloor \left(\frac{\theta \cdot 256}{2\pi} + 0.5 \right) \%256 \right\rfloor \tag{10.15}$$

where % is the modulo function. The purpose of that is to wrap the output to a value between 0

and 255, which can be used for look-up tables. The round down function rounds a number, effectively quantizing the output. The reason for using round down rather than just round is to ensure, that the number ranges from 0 to 255. The rounding error is minimized by adding 0.5. It has been attempted to use the quantizer block, but that causes stiffness to the point where the simulation almost stalls. The output is sent to the Zybo block, which will be explained in section 10.3.1.

The Zybo subdiagram generates duty cycles for each phase ranging from -1 to 1. This value is then multiplied with 26.4V to represent half the battery voltage, and then saturated to ensure, that the ideal voltage sources do not provide more voltage than possible.

10.3.1 Zybo block

The Zybo block consists of three other subsystems, corresponding with some of the blocks on the actual Zybo; Clarke Park, Discrete controller and PWM generation. This can be seen on figure 10.5.

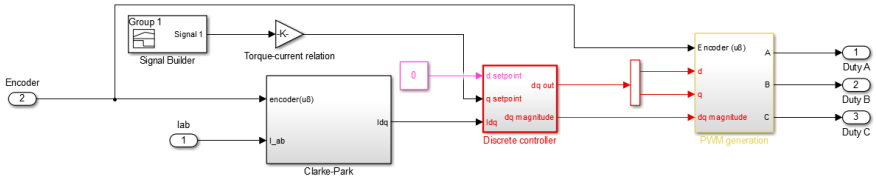


Figure 10.5: Block diagram describing the Zybo part of the system.

The Clarke Park transform and PWM generation blocks are running in variable time steps, and the discrete controller runs with a fixed step of 0.1 ms. The signal builder block is used to shape the torque requested by the pedal. This torque is converted into the current used in the controller by multiplying with $\frac{3}{2K_t}$. The Clarke Park block converts I_{AB} to I_{dq}. Due to the limitations of the encoder, the system only measures 64 different electrical angles.
The block diagram of the discrete controller is shown on figure 10.6.

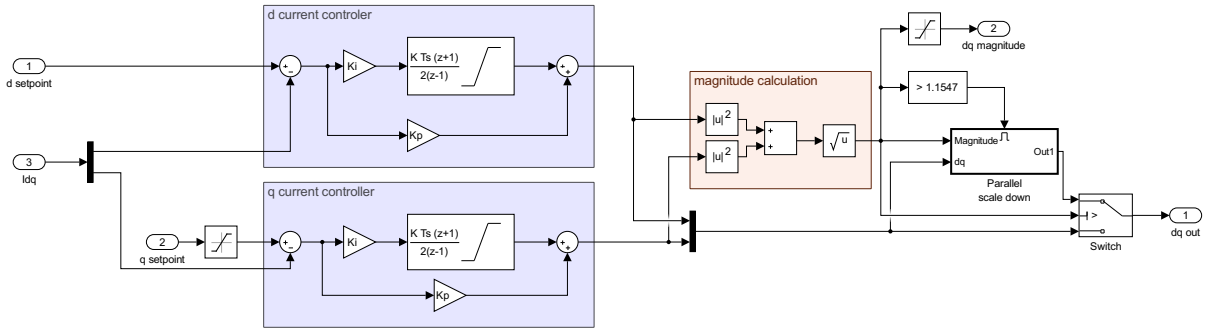


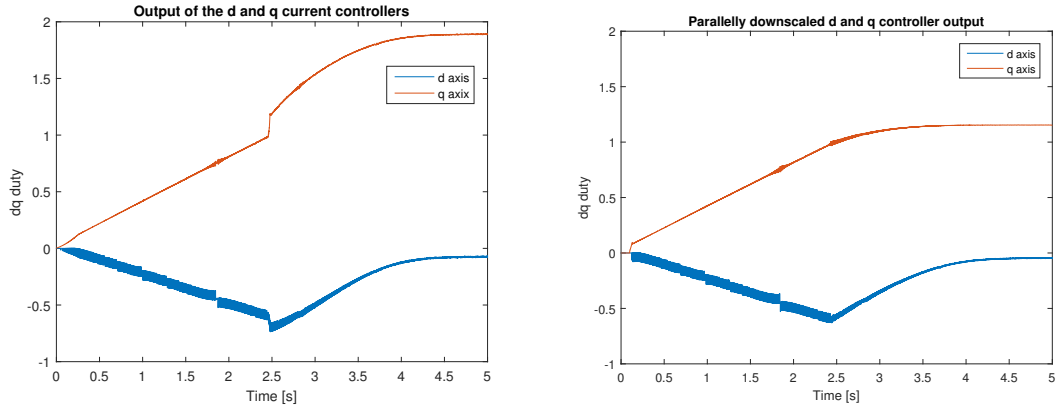
Figure 10.6: Block diagram of discrete PI controller.

The two independent current controllers are drawn in blue boxes. The feedback comes from the I_{dq} port on the left. The constants, K_p and K_i, are calculated in section 9, and divided by half the battery voltage. This is done, because the controllers designed in section 9 produce a voltage,

whereas the Simulink controller produces a percentage of half the battery voltage, i.e. between -1 and 1. Similarly the implemented PWM block in section 6.2 takes integers in the range from 0 to 1000, so the controller constants need to be scaled accordingly
The integrator block uses trapezoidal integration, and the output is limited to $\pm\frac{2}{\sqrt{(3)}}$. This is to ensure, that the integrator does not wind up. The saturation block to the left of the q current controller on figure 10.6 limits the input to between 0 and a maximum current. The PI controllers produce duty cycle values in the range of $\pm\frac{2}{\sqrt{(3)}}$, although it is possible to exceed this range at this point in the diagram.

It is necessary to ensure, that the controller does not produce values outside the range that can be generated using PWM. The obvious solution is to put a saturation block after each of the controllers, but a large value value from both controllers will still cause saturations. In order to limit both controller outputs without disturbing the controllers, both output values is reduced by the same factor. The magnitude of the dq-duty cycle vector must not exceed $\frac{2}{\sqrt{3}}$, when using third harmonic injection. For this reason, the magnitude of the two controller outputs is calculated in the red box. If the magnitude is larger than $\frac{2}{\sqrt{3}}$, the Parallel scale down subsystem is processed.

For comparison, the d and q axis duty cycles have been simulated and are shown on figure 10.7:



(a) Before parallel scale down.

(b) After parallel scale down.

Figure 10.7: dq duty cycle values before and after parallel scale down.

The graphs are simulated with a car mass of 200kg, and settling time of 0.01s. At 2.5s, the magnitude of the dq-vectors exceeds $\frac{2}{\sqrt{(3)}}$, at which point both the d and q output is scaled down. On figure 10.7b, this transition is almost invisible along the q-curve. As the torque produced by the motor is reduced due to saturation of the supply, the need for a large negative d-duty is reduced, as both d and q current is reduced.

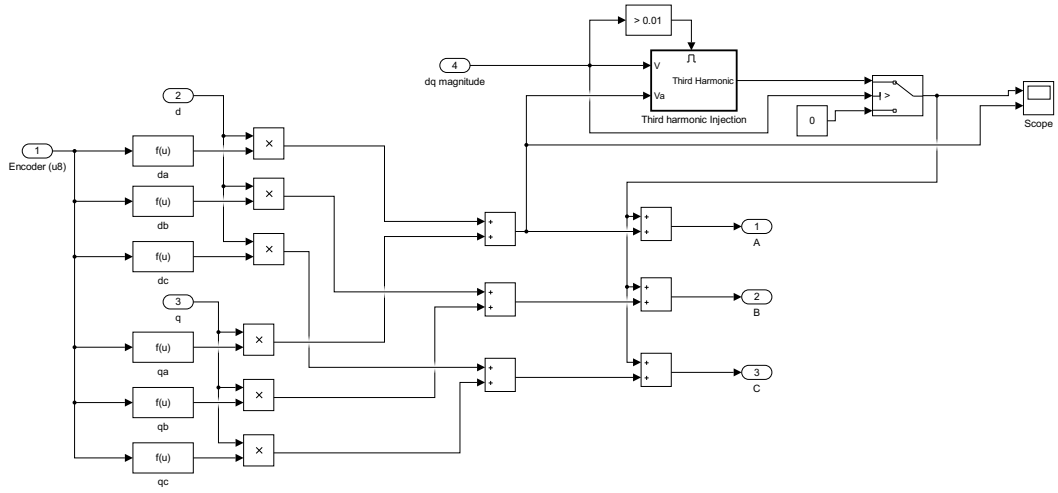


Figure 10.8: Block diagram of the PWM generator.

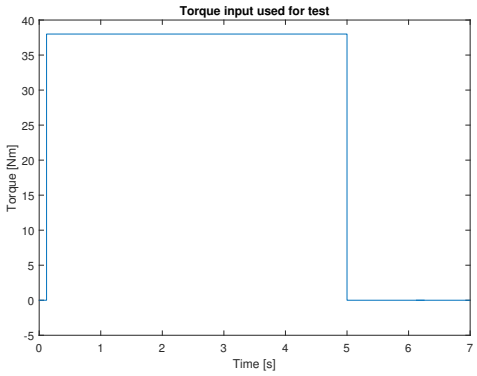
The majority of this block diagram calculates the inverse Clarke Park transformation, and in the upper right corner, the third harmonic injection is calculated using the knowledge given in 9.9.

10.4 Simulation Results

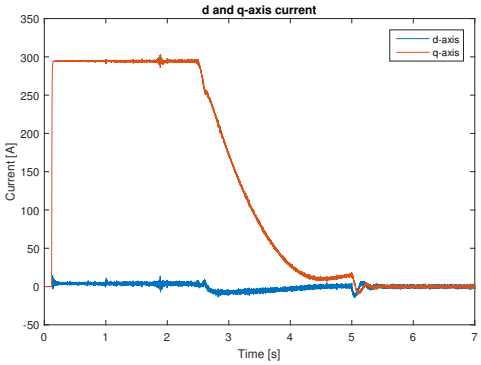
The simulation will be run with both PI and IP controllers, with values described in sections 9.4 and 9.6.

10.4.1 PI Controller

Throughout this section the PI controller developed in section 9 for a full order system will be tested. At first, a simulation will be run where the throttle is pressed fully from 0.12s to 5s, and then released. This is displayed on figure 10.9a.
This input is similar to the one used in section 9. The mass is 200kg. The resultant current along the d and q axes have been plotted in figure 10.9b.
Much like on figure 9.17a, the current steps up nicely with almost no overshoot. There is also a small steady state error of about 2 %, as was the case in section 9. There is a small current noise on I_q and especially I_d, which is caused by the quantization of the encoder. Using a higher frequency for the discrete controller reduce this ripple. For these simulations, 10kHz has been used lower frequencies can be used with acceptably low noise, up to 0.5kHz. The extra noise on both the d and q current at 1.8s is imaging error, caused by the quantization of the encoder and the frequency of the controller. For controller frequency, the error happens when the motor is running at 2250rpm. Lower frequencies mean, that the error happens at lower motor speeds.
The d-axis current is kept very close to zero, at least until 2.6 seconds, when the voltage starts saturating. A small error (of up to -20 A) is not a big issue, as this is likely to occur anyway if the position encoder is not perfectly aligned.
When saturation occurs, the current drops off almost exponentially. At 5 seconds, when the throttle is released, both the d and q current ripples a bit. This is due to the way saturation is handled, but the ripple is not a major ground for concern.



(a) Torque step input for the tests performed in this section



(b) d and q current simulated with a positive and negative step.

Figure 10.9: Torque input and simulated current output.

The voltage waveforms must not be saturated, and should remain sinusoidal with third harmonic injection. This is shown on figure 10.10

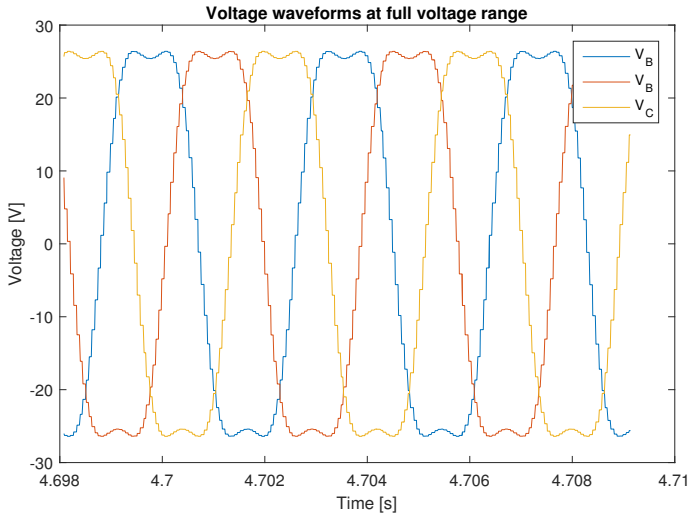
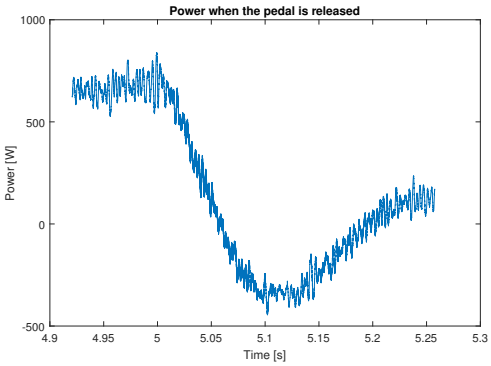


Figure 10.10: Voltage waveforms measured when the motor is running at 4000rpm.

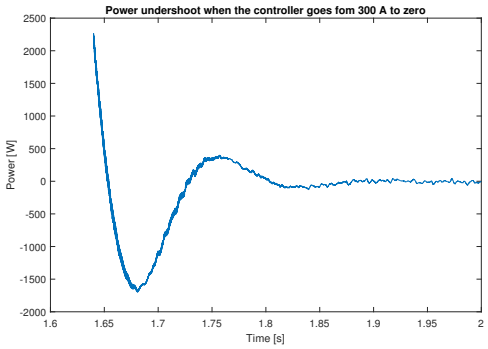
The reason for stepping voltage instead of a smooth curve is the limited resolution of the encoder. This is not an issue, because these steps will induce a high frequency noise, that the inductance of the motor will filter out. Figure 10.10 shows that the voltage waveforms span the whole range, even at speeds where the controller output is outside the ± 1.15. This works independently from the controller, and will work both for the PI and IP controller.

When the throttle is released after 5s, the power should ideally not become negative, as the inverter is not designed to handle regenerative braking. This means, that it is not known what happens if the motor tries to push power back to the battery. Figure 10.11a shows a short power undershoot

occurring when the controller input goes from full to zero.



(a) Power undershoot when throttle is released.



(b) Power undershoot crosses zero 33 ms after the throttle is released.

Figure 10.11: Undershoot when releasing torque pedal.

The power peaks at -400W for a short time. Integrating the power from 5.06 to 5.5s gives an energy of -25J. This is not likely to cause problems, as this energy easily could be lost in the inverter. The issue is larger when releasing the throttle while the go-kart is accelerating. This is shown on figure 10.11b.
Integrating this undershoot yields -62J. This is not a major concern, however it causes a ripple in the speed, that might be felt.

10.4.2 IP Controller

Throughout this section the IP controller developed in section 9 for the reduced order system is tested. In order to do that, the wire going into the Kp gain block must be connected with the feedback wire, instead of the output of the summation block

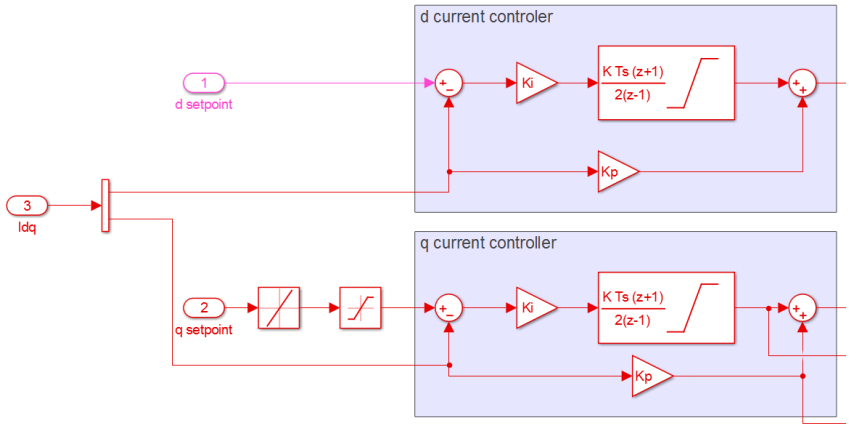
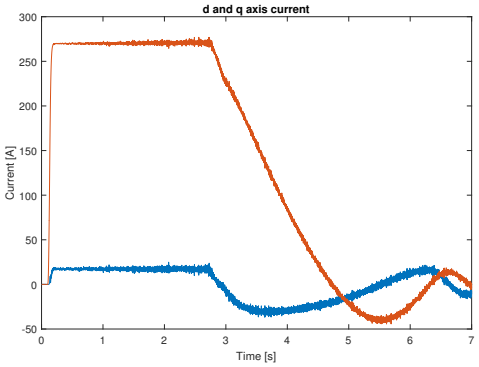


Figure 10.12: Input to K_p gain blocks are connected to the I_{dq} feedback wire instead of the output of the summation block.

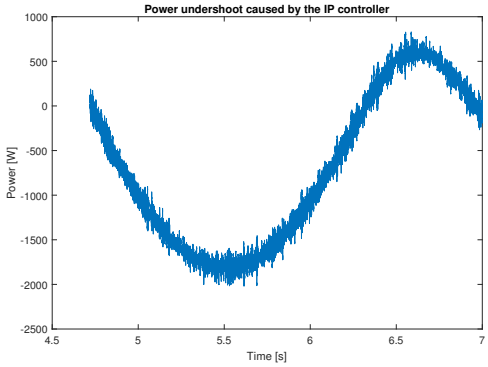
The parameters are divided by half the battery voltage, and shown in equation 10.16.

$$K_p = -2.65 \cdot 10^{-5} \quad K_i = 1.23 \cdot 10^{-2} \tag{10.16}$$

Figure 10.13a shows the d and q-axis current as the controller is given a positive and a negative step, as seen on figure 10.9a.



(a) d and q axis current with a positive and negative step with the IP controller.



(b) Negative power caused by oscillation.

Figure 10.13: Characteristics of IP controller.

By comparison to the PI controller, the IP controller does significantly worse. The steady state error of the q-controller is 10%, while the d-controller has a steady state error of 16A. What's even worse is, that the q-axis current goes below zero before the throttle is released after 5s. That means, that when driving the go-kart at full throttle, the speed of the go-kart will start oscillating quite notably. At 5 seconds, the throttle is released, at which point the current should be zero. The current is already at -22A at this point, and the negative step does not seem to alter the oscillation in any way. The negative torque causes a negative power, as shown in figure 10.13b
A power of this magnitude is likely to cause serious damage to the inverter, if it's lost in the inverter, and not used to charge the battery. The total energy being delivered back in the timespan shown on figure 10.13b to the inverter is 1614J.

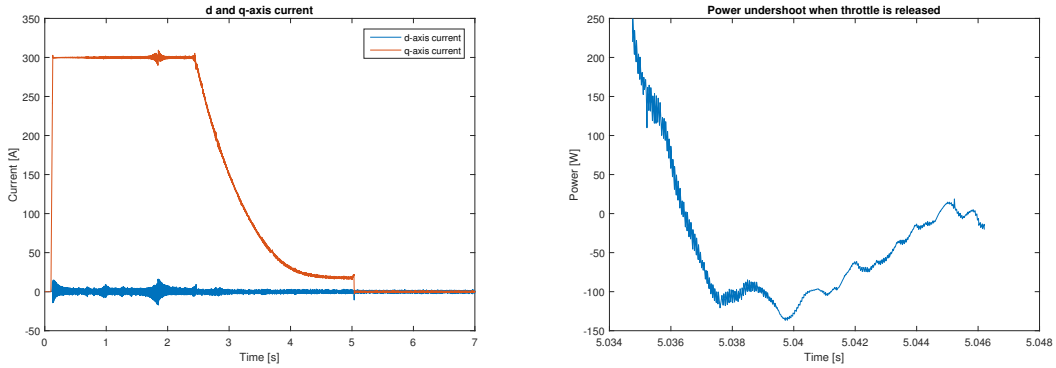
10.4.3 PI Controller with Faster Settling Time

Since neither of the two controller designs work optimally, the PI controller will be redesigned with a settling time of 10ms. The controller is designed for a load of 200kg. The control parameters in equation 10.17 are divided by half the battery voltage

$$\begin{aligned} K_p &= 1.64 \\ K_i &= 2.6 \cdot 10^{-3} \end{aligned} \tag{10.17}$$

The reason for not using the settling time of 1ms is, that it provides a large overshoot, which was also evident in this more elaborate simulator. Introducing a rate limiter block in the Discrete controller subsystem between the q setpoint port and the saturation block will reduce the overshoot to less than 1%. This block is displayed on figure 10.16. The rising and falling rates are set to $10\frac{\text{kA}}{\text{s}}$ and $-10\frac{\text{kA}}{\text{s}}$. This is fast enough, that a driver would not notice.

Figure 10.14a shows the positive and negative current step.
The faster I_q-controller is much more precise. The rate limiter ensures, that the overshoot is less than 1 percent. The steady state error negligible. However, the I_d-controller is more noisy, but still close to zero. There is a small current undershoot when the pedal is released of only -3A. The power caused by this is shown on figure 10.14b



(a) d and q axis current is a lot more precise than on figure 10.9b.

(b) A short and small current undershoot of only 0.6 J

Figure 10.14: Characteristics of IP controller.

The power undershoot is significantly smaller, and lasts for a shorter time, meaning that the energy returned to the inverter is reduced from 25J to just 600mJ.

10.4.4 Conclusion of Controller Comparison

The control parameters have been tested on a highly detailed physics simulator, with the conclusion, that neither the PI controller based on the mechanical system, nor the IP controller based on a reduced order system works satisfactorily. A faster controller was calculated with tremendous results.

Event	PI controller	IP controller	Fast PI controller
Torque step	Handles well	Slow, steady state error	Excellent
Speed limit	Great. Same for all	Great. Same for all	Great. Same for all
Torque release	Small power returned	Oscillation	Great response

Table 12: Table comparing the abilities of the three tested controllers.

Both the PI controllers could work, but the faster response gives very precise current control. The IP controller was designed in section 9 based on a reduced order system without any mechanical part. Evidently, the motor along with the mass of the go-kart has a slow pole, that the IP controller does not handle. The IP controller adds a pole to the systems, which makes the fast pole on figure 9.6a complex conjugate poles that can oscillate. This oscillation proves to be too much when adding the slow poles of the mechanical system. As it has shown, this controller is not able to account for the mechanical system as a disturbance, which is unsurprising really giving the scope of the mechanics involved. It was found that the PI controller designed for the lowest settling time showed the best performance. Further tuning of the control variables could potentially yield better results.

10.5 Plecs Model

The reason for using Plecs is that it handles semiconductors as ideal switches. This greatly reduces the time required to simulate switch mode voltages. As both the high side and low side MOSFETs of each phase switch simultaneously, each PWM period of simulation contains up to six switching events. At 20kHz, this means, the whole circuit needs to be solved 120.000 times per second, so using plecs is still not fast. The main use of this model is the simulation of heat dissipation throughout the circuit.

As previously mentioned, the Plecs model differs vastly from the Simulink model in the electrical network, as it more closely resembles the real analog circuit. It is shown on figure 10.15.

The simulation utilizes the fast PI controller from page 71. It simulates the switching inverter utilizing ideal switches as opposed to more realistic MOSFETs with limited current rate of change. Because of the ideal switching, it is not necessary to include antiparallel diodes, nor deadtime.

Figure 10.15 shows the three phase inverter. The blue rectangle around each mosfet is a heat sink, that absorbs all the power loss within it. These are used to emulate the case of the transistors. Each of these have a heat capacity of $0.01\frac{\mathrm{K}}{\mathrm{J}}$
Each transistor represent one pair of transistors, where the on-resistance is zero. The transistors have the same thermal model, with look-up tables for switch-on and switch-off energies based on equation 7.10. The energy lost per switch is calculated by equation 10.18.

$$E_{on} = \frac{1}{2} V_{DS} \cdot |I_D| \cdot t_{c,on} E_{off} = \frac{1}{2} V_{DS} \cdot |I_D| \cdot t_{c,off} \tag{10.18}$$

Because a negative current would cause the same loss, as a positive current, the absolute is taken of the drain current. All losses are calculated for two MOSFETs in parallel, conducting the combined current.

Conduction loss is dependent on the temperature of the mosfet. This however, does not change the on-resistance in the circuit, this is not possible to do in Plecs. It should be noted, that the heat sink box includes all losses inside, meaning it includes both the loss specified in the thermal model, but also the loss due to a non-zero on-resistance. This is part of the reason, the on-resistance has been set to zero, the other part being faster simulation.

All heat sink boxes are connected through a thermal resistor (the blue ones) to a shunt thermal capacitor (CthG). The thermal resistance of $0.145\frac{\mathrm{J}}{\mathrm{K}}$ represents the case-to-sink thermal resistance as read from the datasheet. The datasheet value was divided by two, because there are two TO-247 houses conducting the heat. CthG along with Heatsink Thermal Resistance models the 30x30 heat sink used to mount the inverter. CthG has a thermal capacity of 2100 $\frac{\mathrm{K}}{\mathrm{J}}$, which is based on its size and the thermal capacity of aluminium, according to 7.22.

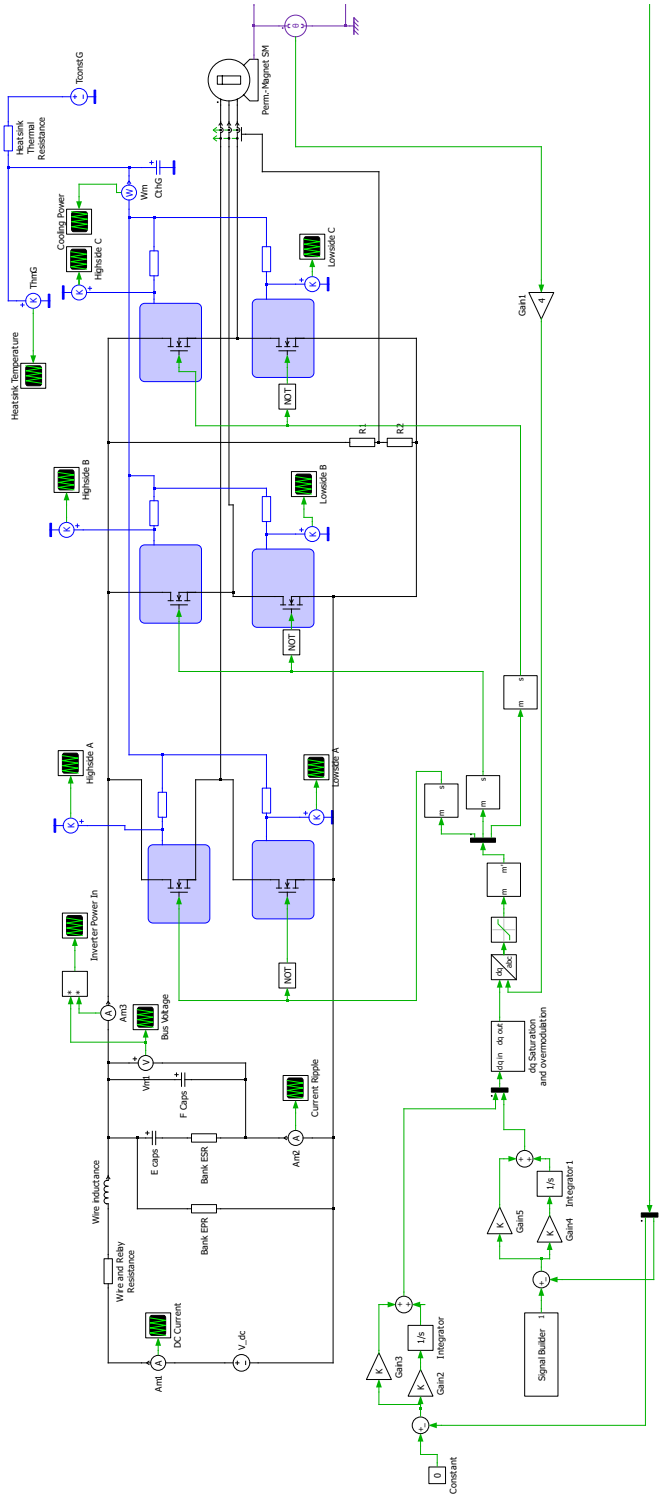


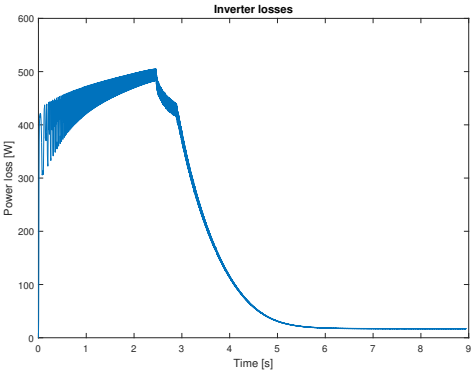
Figure 10.15: Block diagram for the svm plecs simlations.

The battery is modelled with a DC voltage source of 52.8 V. In series with this is a non-ideal wire with an inductance and resistance. Bank EPR is a gigaohm resistor needed for the simulation – it will not run without it, because the wire inductor might not have a DC path to ground. F caps represent the film capacitors. These are significantly smaller, but have no ESR, and handle higher frequencies better than the E caps.

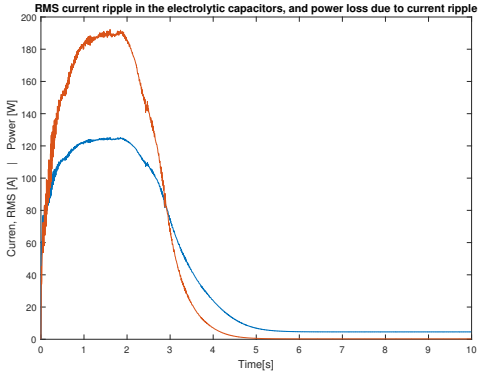
The controller is modeled on the bottom half of figure 10.15. Other than the PI controllers, it contains parallel scale down and overmodulation. However, overmodulation uses a different method than third harmonic injection, which reduces switchings.

The test will be performed with a mass of 250 kg. The current setpoint definced in the Signal Builder block goes to 300 and remains there for 10 seconds. The simulation will be used to estimate voltage ripple from the capacitor bank, the AC current going into the electrolytic capacitors and the loss in the MOSFETs.

Figure 10.16a shows the heat transfer to the heat sink. As there is a very low heat capacity in each of the MOSFETs, it's safe to assume, that this represents the total loss in the inverter



(a) Power loss in the MOSFETs



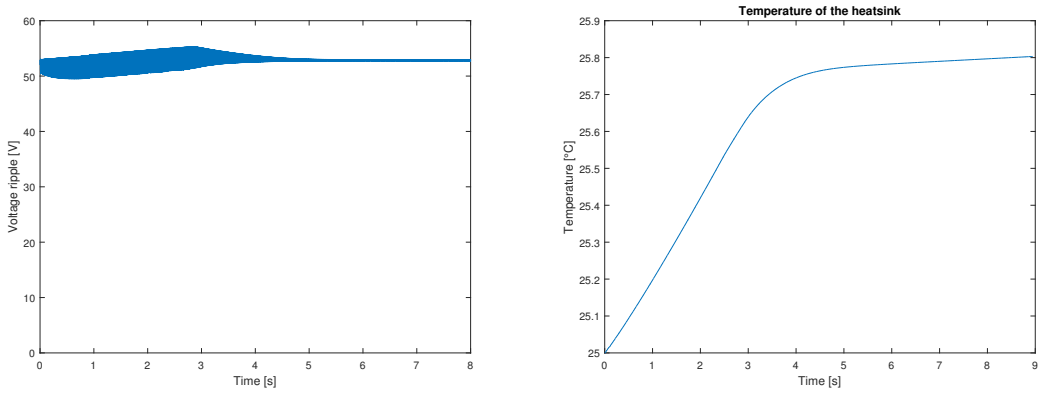
(b) Current ripple in the electrolytic capacitors, in RMS.

Figure 10.16: MOSFET and capacitor loss.

The loss reaches a maximum of 500 W, which is lower than anticipated. This might be because, the Plecs simulation takes into account for switchings that are not worst case scenario.

The current ripple is displayed on figure 10.16b

The large current ripple would result in up to 190 W of loss, which is a lot more than the capacitors can handle. Not only is this power loss damaging for the efficiency of the inverter, it also produces a lot of heat, that is not absorbed by the heat sink. A reason for this high number is likely the ideal switching used in Plecs. Because of this, the current from the supply can step from zero to 300 A instantaneously. In a more realistic simulation, stray inductances would limit the current slew rate.
Figure 10.17a displays the voltage ripple of the capacitor bus.



(a) Voltage ripple on the inverter supply.

(b) Temperature increase of the heat sink.

Figure 10.17: Voltage ripple and temperature profile.

The voltage ripple is quite low, less than 5 V, which means the inverter is supplied with an almost constant voltage, and the AC output is not begin disrupted by much.
Figure 10.17b displays the heat sink temperature during the full throttle test
The temperature only increases 0.8 °°C during the test, so it easily handles the 500 W of heat generated by the MOSFETs. The reason for this limited temperature increase is the size of the thermal capacitance, rather than the low thermal resistance.

10.5.1 Conclusion of Plecs simulations

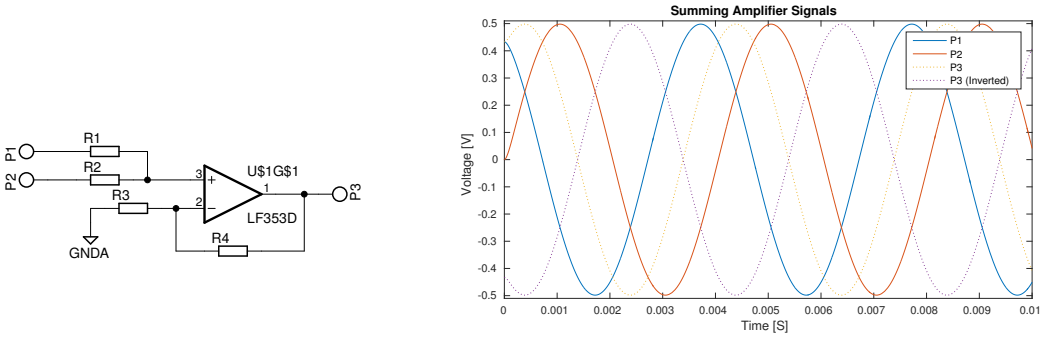
It is possible to drive a PMAC motor with three phases of naturally sampled PWM. The inverter has a large thermal capacity, meaning that it would keep cool for the duration, it is being used. However, according to Plecs, the current ripple in the electrolytic capacitors greatly exceeds what they are rated for. It was tried to add a ESL inductor to the capacitors, but that caused the simulation to become much slower to the point where it would not run at all.

10.6 Simulation of the Over-Current Protection Circuit

In order to verify the functionality of the OCP circuit discussed in section 5.2.1, simulations of the circuit were conducted in LTSpice[1]. The results of the simulation will be discussed in the order of the circuit. As mentioned, only two of the phases are measured and the third will have to be calculated. This is done using the summing amplifier circuit shown in section 5.2.1, repeated in figure 10.18a for convenience. The resulting signals can be seen on figure 10.18b. Clearly, P3 is not phase shifted 120° with respect to P1 or P2. This is a consequence of using the non-inverting summing amplifier.
Inverting P3 however, reveals the correct signal. As all three signals, P1, P2 and P3 are rectified in the next step, this discrepancy is irrelevant. For the remainder of this section only one phase is discussed as the circuitry of all three phases is identical after the initial summation.

The next step in the OCP process is the rectification of the signals. This is done using a precision full wave rectifier as shown in figure 10.19. As can be seen, the rectification error is relatively

[1]LTSpice: Free simulation software by Linear Technologies.



(a) (b)

Figure 10.18: In 10.18a is shown the summing amplifier used. In 10.18b P1 and P2 represent the two measured phases while P3 is the calculated, third phase.

minor compared to the signal size. It is mostly caused by the phase shifting done by the rectifier circuit. This is apparent when noticing that the error is mostly identical between the rectified and non-rectified parts of the signal.

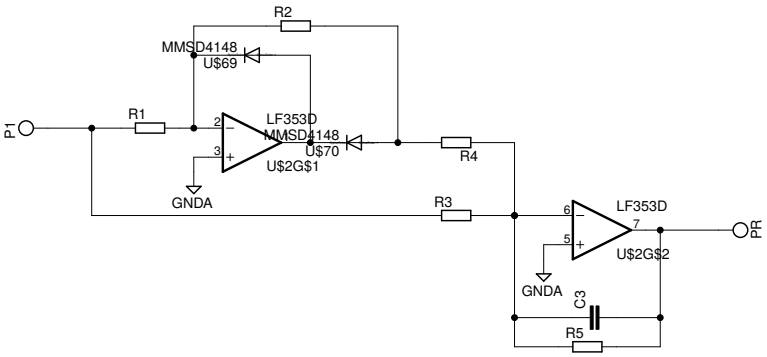
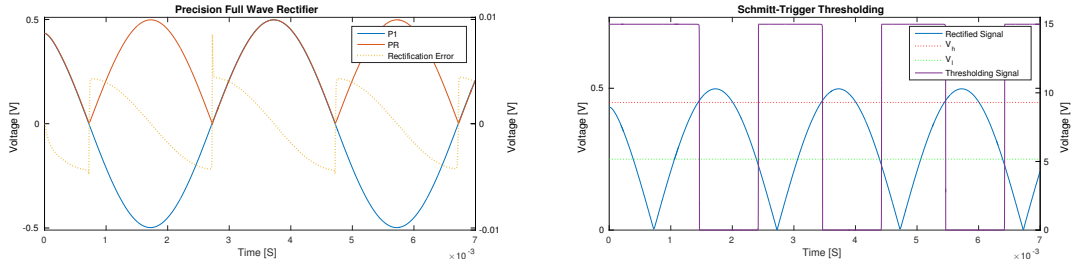


Figure 10.19: Full wave rectifier used to rectify the current signals.



(a) (b)

Figure 10.20: 10.20a shows the rectification error of the full wave rectifier. 10.20b is an overview of the Schmitt triggered signal.

After rectification the signal is fed to a Schmitt-trigger with hysteresis voltages $V_h = 0.475\text{V}$ and

$V_l = 0.25$V. These signals are shown in figure 10.20b. The hysteresis values are reasonable close to the desired values, deviating only, by inspection, by a few mV. This is expected since the resistor values calculated in section 5.2.1 are rounded to the nearest standard value. Additionally, in the real system there may be slight variations in components, these variations are not simulated however.

The remainder of the OCP circuit merely reverses the polarity of the signal. This is done such that it conforms with the logical circuitry responsible for determining the state of the EN_GATE signal discussed in section 5.

11 System Implementation and Testing

This section will describe how the system was implemented and tested on both a test setup and the real go-kart setup.

11.1 Test Setup

In order to test subparts of the complete system, a test setup was introduced. It consists of a permanent magnet motor of the type PM5113 with a RMB28MD encoder mounted on top. The setup furthermore consists of a 74HCT14 level shift interface board and a dual H-bridge and can be seen in figure 11.1. The setup was made and soldered by the supervisors of the project. The full setup with H-bridges can be used to test the functionality of the Embedded System. The motor and encoder can be used to test the full system.

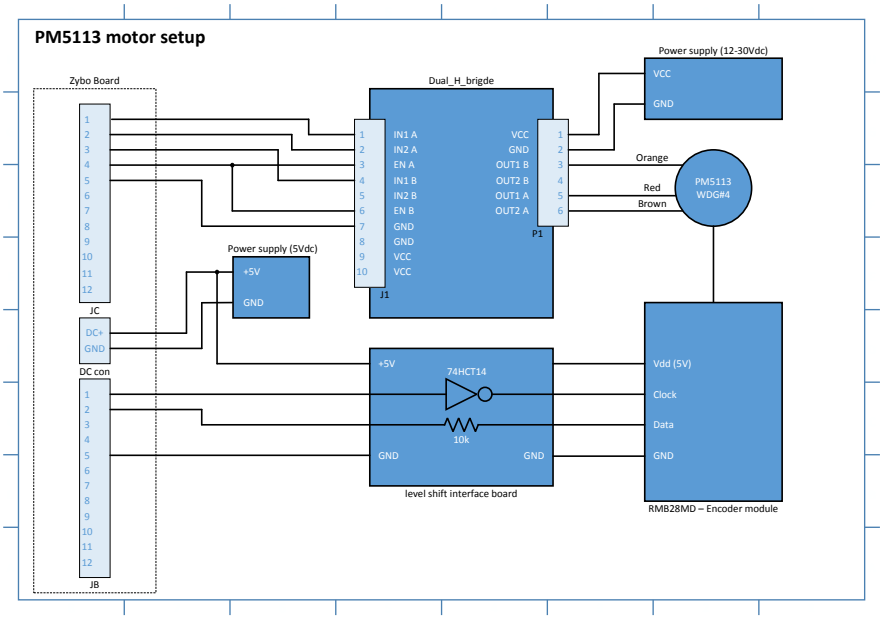


Figure 11.1: Block diagram of test setup with PM5113 motor.

11.2 PWM Generating

To test whether the Clarke Park transformation and the PWM generation on the Zybo board was working correctly a test was conducted. The test setup, described in the previous section, was used. The three output pins of the half bridges are connected to the terminals of the motor. A constant value for q was chosen and d was set to zero. The angle of the rotor was measured and used to do the Clarke Park transformation. No current measurements were done, meaning that the only feedback was the angle. Each of the signals from the output pins of the Zybo board were, in addition to the H-bridge, connected to an electric lowpass filter consisting of a 10 kΩ resistor and a 22 nF capacitor. The voltage across the capacitor was measured by an oscilloscope as the power supply for the H-bridge was turned on. The measured phase voltages can be seen in figure 11.2. As the observant reader will notice there are no third harmonic on the graphs. This is because third harmonic injection were not included at the time of the test. It can be seen that the three phase voltages are created correctly by the Zybo board. It can also be seen that the rotor is accelerated within a fraction of an electrical revolution, which is expected as there is no torque load.

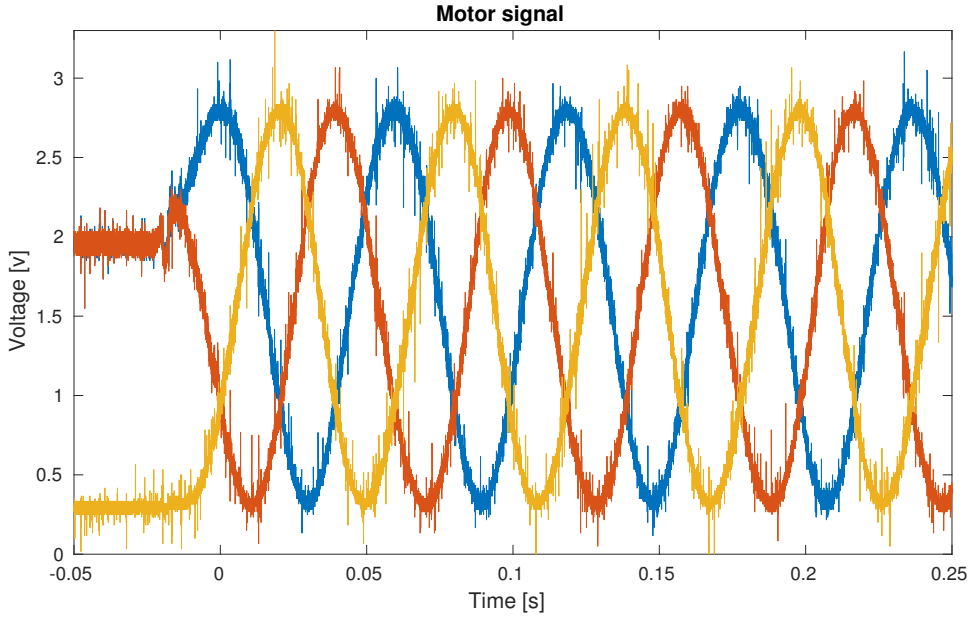


Figure 11.2: Phase voltages measured through a lowpass filter.

11.3 Analog, Digital and Driver Boards

Various issues were encountered while creating prototypes of especially the analog and digital boards. Several iterations were etched before a satisfactory result was obtained. Due to delivery issues, the components were not available until very late in the process and there was not sufficient time remaining to fully debug the boards. The driver board was made to function as was intended and is used in the test in section 11.4. As mentioned in previous sections, the strategy used for OCP changed throughout the testing phase. It became apparent that it was infeasible to spend the time necessary to have the analog board functioning in time for the deadline. In the meantime the OCP of the DRV8301 was understood, allowing to use this feature instead.

As shown in section 10.6, the general behaviour of the OCP circuit is working as intended. The errors found on the analog and digital boards are mostly related to layout and unintentional short circuits. In addition, the board were designed early in the process, without certain knowledge about the zybo and the driver, that has been acquired now. For this reason it is the conviction of the group that one or two revisions of the boards would be necessary to fully debug them.

11.4 Power Board

To ensure that the power board was working correctly, a test of it was performed using . As described in the previous section the developed digital and analog board were not made to function because of time constraints. Instead two simple boards were developed for testing purposes. They only contained the bare minimum functionality of the original boards: scaling of output from LEM sensors and torque pedal, routing of signals to/from the DRV8301 and a RC circuit for delaying the SPI slave select signal. The already tested embedded system on the Zybo was connected and programmed to run the motor. Upon powering on the system the motor starting running, verifying that the inverter is functioning correctly. Unfortunately, due to time constraints, it has not been possible to verify that it meets the requirements to run the motor on the go-kart.

12 Conclusion

Throughout this report has been described the various steps taken to create a control system for an electric go-kart. As part of this, circuit boards were developed to take care of over current protection, scaling of signals and various other tasks. The OCP was simulated in order to verify its functionality However, except for the driver board, due to time constraints it was not possible to finish the boards in time for the deadline. A driver board was made to for the DRV8301, the driver chip used in the project. This board was made to work with a test setup which includes the inverter developed, a small PMAC motor and some temporary boards made to interface the signals of the system. Using this test setup it was possible to drive the motor, verifying that the inverter is functional. A controller for the system was created using the settling time formulae. The controller was tested using the extensive simulations made of the system using Simulink. The controller was discretized in preparation for using it on the embedded system. This was unfortunately not tested. The embedded system was programmed using khaOS, a Run To Complete Scheduler. Its real time performance was verified using an oscilloscope.

Many parts of the project were not completed to the satisfaction of the group. Throughout there were problems with delivery of components as well as the quality of the boards created. This resulted in much time spent doing things not necessarily conducive to a better end product.

References

[1] Dodds, Stephen J., University of East London, 2015. Feedback Control: Linear, Nonlinear and Robust Techniques and Design with Industrial Applications.

[2] Xilinx, Nov. 2015. Zynq-7000 All Programmable SoC (Z-7010, Z-7015 and Z-7020): DC and AC Switching Characteristics.

[3] Fairchild, Dec. 2014, Design and Application Guide of Bootstrap Circuit for High-Voltage Gate-Driver IC.

[4] Texas Instruments, Aug. 2013, DRV8301 - Three Phase Pre-Driver with Dual Current Shunt Amplifiers and Buck Regulator (datasheet).

[5] International Rectifier, IRFP4468PbF Power MOSFET datasheet

[6] Neacsu, Dorin O., 2016, Power-Switching Converters: Medium and High Power-Switching

[7] Xilinx, May. 2015, 7 Series FPGAs and Zynq-7000 All Programmable SoC XADC Dual 12-Bit 1 MSPS Analog-to-Digital Converter (User Guide).

[8] Super B Lithium batteries, SB12V20P-FC Lightweight Lithium ion starter battery datasheet

[9] Xilinx, February. 2015, Zynq-7000 All Programmable SoC (Technical Reference Manual).

[10] Suda, Barath Kumar; David, James, Eaton Corporation, Mar 'A Novel Third Harmonic Injection Method for Closed Loop Control of PMSM Motors'

[11] A. B. Dehkordi, Student Member, IEEE, A. M. Gole, Senior Member, IEEE, T. L. Maguire, Senior Member, IEEE; 'Permanent Magnet Synchronous Machine Model for Real- Time Simulation'

[12] Vishay Siliconix, July. 2013, TO-247AC (High Voltage) Package Information

[13] Biancolini, Marco Evangelos, Department of Mechanical Engineering Tor Vergata University, 'Evaluation Of Aerodynamic Drag Of Go Kart By Means Of Coast Down Test And Cfd Analysis'

[14] Thomsen, Mark, 1999, IEEE, 'Tricks of the Trade: Guesstimating Inductance of Wire Loops'

[15] Austin Hughes, Bill Drury; 'Electric Motors and Drives: Fundamentals, Types and Applications - Fourth Edition'

[16] Tech/OPS Sevcon "Partner in Performance" 'Sevcon Gen4 Applications Reference Manual'

[17] Super B 'Super-B - SB12V20P-FC LiFepo4 battery.pdf - reference datasheet'.

[18] Mohan, Ned, John Wiley and Sons, 2012, Inc. Power Electronics A First Course