TALLINN UNIVERSITY OF TECHNOLOGY
School of Information Technologies

Tarmo Prillop 178206IASM

OBSERVER BASED SENSORLESS CONTROL OF SMALL BLDC MOTORS

Master's Thesis

Supervisor: Eduard Petlenkov

PhD

Tallinn 2019

TALLINNA TEHNIKAÜLIKOOL
Infotehnoloogia teaduskond

Tarmo Prillop 178206IASM

OLEKUTAASTAJAL PÕHINEV ANDURITETA JUHTIMINE VÄIKESTELE BLDC MOOTORITELE

Magistritöö

Juhendaja: Eduard Petlenkov

PhD

Tallinn 2019

Author's declaration of originality

I hereby certify that I am the sole author of this thesis. All the used materials, references to the literature and the work of others have been referred to. This thesis has not been presented for examination anywhere else.

Author: Tarmo Prillop

06.05.2019

Abstract

The aim of this thesis is to create an implementation of an observer based sensorless speed controller for brushless DC motors. Initially theoretical aspects of three phase motors, inverters and observer based control are described. Additionally an overview of state of the art in sensorless observer based control is given.

Further sections give overview of the physical hardware designed and used during the course of the work. The main practical part of the work was divided into two. At first measurement procedures and results are presented for motor parameters as they are essential in model based control approaches, which observer based control is. Secondly a suitable observer based algorithm is chosen from the available research and implemented as a program on the microcontroller.

The performance analysis of the final controller showed that the chosen algorithm performs well even though not specifically designed for brushless DC motors.

This thesis is written in English and is 62 pages long, including 5 chapters, 39 figures, and 5 tables.

Annotatsioon

Käesoleva töö eesmärgiks oli luua olekutaastajal põhinev kontroller väikestele harjavabadele alalisvoolumootoritele. Töö alguses tuuakse välja olulisemad teoreetilised aspektid kolmefaasiliste mootorite, nende juhtelektroonika ning levinematute juhtimisalgoritmide kohta. Seejärel kirjeldatakse süsteemide juhtimist olekutaastajaga ning tehakse ülevaade viimastest teaduspublikatsioonidest selle valdkonna kohta.

Töö praktilise osa kirjeldamine algab disainitud riistvara ning kasutatud mootorite kirjeldamisest. Seejärel selgitatakse mootorite elektriliste parameetrite mõõtmiste põhimõtteid ning tuuakse välja mõõtmistulemused kasutatavate mootorite parameetrite kohta. Seejärel põhjendatakse olekutaastaja valikut ning kirjeldatakse valitud olekutaastaja struktuuri ning valemeid millel see põhineb.

Töö lõpus esitatakse mõõtmistulemused loodud kontrolleri käitumisest ning analüüsitakse neid ning nende vastavust püstitatud nõuetele. Analüüsi tulemustest lähtuvalt oli töö edukas ning mootori juhtimine vastas juhtimiskriteeriumidele ning oli stabiilne.

Lõputöö on kirjutatud inglise keeles ning sisaldab teksti 62 leheküljel, 5 peatükki, 39 joonist, 5 tabelit.

List of abbreviations and terms

AC	Alternating current
ADALINE	Adaptive linear neural network
BLDC	Brushless DC motor
CSV	Comma separated value
DAC	Digital to analog converter
DC	Direct current
DTC	Direct torque control
EMF	Electromotive force
FOC	Field oriented control
MOSFET	Metal oxide semiconductor field-effect transistor
PLL	Phase lock loop
PMSM	Permanent magnet synchronous motor
PWM	Pulse width modulation
RPM	Revolutions per minute
SPMSM	Surface permanent magnet synchronous motor
SVM	Space vector modulation

Contents

List of Figures

List of Tables

1 Introduction

Embedded computing power in the form of microcontrollers is becoming more and more cheaper as the number of intelligent embedded devices grow [1]. Also the amount of brushless direct current motors is ever increasing due to the growth of hobby and commercial sector of multirotor configuration aerial vehicles which use such motors for propulsion. The combination of those aspects makes a prospect of using complex control algorithms to be applied in controlling those motors for use in other fields like mobile ground robots and others where precise control of the motor speed is required. The usage of brushless direct current motors is appealing as their construction is simple, involving only one moving part and contain no additional components expect from bearings that are subject to wear (like brushes in a conventional direct current motor). In addition as the motors are designed for use in aerial vehicles where low weight is a crucial design criteria and the lifting power is desired to be as high as possible, the brushless direct current motors have an excellent power to weight ratio. Adding a controller with control algorithms that allow precise control over the motor speed without sensors creates a powerful combination of small sized, high performance and low maintenance solution. This thesis will focus on developing a practical solution in the form of the controller to evaluate the control quality that can be achieved by using sensorless control algorithms from research.

1.1 Problem and end goal definition

The main objective of this thesis is to create a practical implementation of a brushless direct current (DC) motor speed controller using an observer based control technique together with field oriented control. The observer structure used will be selected amongst the ones published in research and the control quality will be evaluated experimentally. The end goal would be a hardware and software combination that would demonstrate the practical use of advanced control principles applied to small hobby grade brushless DC motor (BLDC) motors. In addition to implementing the actual control part of the algorithm, additional functionality will be developed to use the controller hardware to measure most of the motor electrical parameters that are necessary for model based control methods.

1.2 Methodology

At first a theoretical introduction to synchronous alternating current motors, control methods and observe based control is given. Then an overview of the designed hardware will be given with the working principles of the parts essential to the control algorithm implementation.

The practical algorithm implementation part will be divided into two. The first part will concentrate on the motor parameter measurement methods to allow model based control approaches to be implemented. It will consist of motor electrical parameter measurement algorithms so that minimal amount of external information would be needed by the controller to successfully perform the control tasks. In the second part an observer structure is selected from the available research done in the field and implemented on the microcontroller. In addition to the pure implementation of the observer, practical implementation problems will be resolved so that the result will be a fully functional speed control approach to brushless DC motors.

Lastly the performance of the final controller will be evaluated against the requirements and conclusions are made about the observer operation and control quality achieved.

The controller hardware design was done using Altium Designer software suite, which is a professional printed circuit board schematic and layout software. The source code was created using Visual Studio 2017 and VisualGDB extension from Sysprogs OÜ. The flashing and debugging was done using Lauterbach PowerDebug II and PowerTrace II adapters and accompanying Trace32 software, which enabled a lot more insight into the run-time code execution and helped to solve a lot of problems related to interrupt and direct-memory access functionality very quickly.

2 Theoretical overview

This section gives a general overview of three phase motors and how used brushless direct current DC motors are situated amongst them. Then an overview of three phase motor control methods is given, with focus on field oriented control and space vector modulation. Lastly an introduction to observer based control is made and an overview of state of the art observer based control of synchronous three phase motors is given.

2.1 Three phase synchronous motors

Three phase synchronous motors are a form of alternating current motors, where the stationary part of the motor consists of field windings and the rotating part of the motor is composed of permanent magnets that rotate in the magnetic field created by stator windings when an appropriate three phase current is applied. Synchronous motors operate at a constant speed with absolute synchronism with the input current frequency. The absolute synchronism means, that the input alternating current creates a rotating magnetic field by the stator windings. The permanent magnets mounted on the rotor will follow that rotating field precisely, given that the magnetic field created by the stator is strong enough.Synchronous motors are classified according to their rotor design, construction and operation into four groups [2]:

- electromagnetically-excited motors
- permanent magnet motors
- reluctance motors
- hysteresis motors

Brushless DC motors are classified as a type of permanent magnet synchronous motors. The notion of being a DC motor comes from the fact that these types of motors are typically driven using rectangular current pulses, even though they are AC motors [3]. The difference from conventional permanent magnet synchronous motors (PMSM) comes from the shape of the electromotive force generated by the motor. Usually it's been the way that BLDC motors have trapezoidal shaped back-EMF waveform and PM synchronous motors have sinusoidal back-electromotive force electromotive force (EMF) [3]. In this thesis BLDC motors are considered as general alternating current (AC) machines

that are driven with three phase alternating currents.

BLDC motors can have different configurations regarding the construction. A common aspect between those configurations is that the permanent magnets are mounted on the rotor and the field windings are contained in the stator. There exists two main configurations that BLDC motors are produced: inrunner configuration where the rotor resides inside the stator and outrunner configuration where the stator is inside the rotor. The motors used in this work are of outrunner type with permanent magnets mounted to the surface of the rotor, classifying them as surface-permanent magnet motors (SPMSM) and such configuration is visible in figure 1. The aspect that magnets are mounted on the surface of the rotor define an important property of the motor that will reduce the complexity of control implementation. Motor with magnets on the surface of the rotor are considered to be magnetically round (non-salient). This implies that the inductances in the direct and quadrature axes will be equal so we can consider that $L_d = L_q = L_s$. Synchronous motor with permanent magnet mounted on the surface of the rotor can be described by the following equations from [4]:

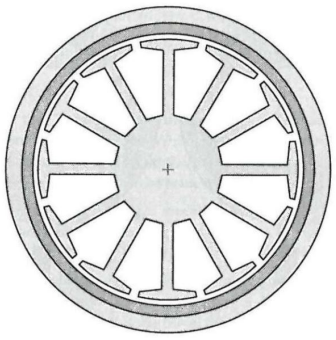


Figure 1. BLDC motor with outside rotor [3].

Classical mathematical model of a surface-PMSM motor in the stator fixed reference frame is the following [5]:

$$L_s \cdot \frac{di_{\alpha\beta}}{dt} = -R_s \cdot i_{\alpha\beta} - \omega \cdot \lambda \cdot \begin{bmatrix} -sin(\theta) \\ cos(\theta) \end{bmatrix} + u_{\alpha\beta} \tag{1}$$

$$\dot{\theta} = n_p \cdot \omega$$

where L_s is the stator inductance, $i_{\alpha\beta} = [i_\alpha \; i_\beta]^T$ stator currents in the stationary reference

frame, R_s resistance of stator phase winding, ω electrical angular velocity, λ flux linkage, $u_{\alpha\beta} = [u_\alpha \; u_\beta]^T$ stator voltages in the stationary reference frame and n_p is the number of pole pairs the motor has.

Electrical torque produced by the motor is given by

$$T_e = K_t \cdot (i_\beta \cdot cos(\theta) - i_\alpha \cdot sin(\theta)) \tag{2}$$

where K_t is the torque constant of the motor given by

$$K_t = \frac{3}{2} \cdot \lambda \cdot n_p \tag{3}$$

As can be seen from equation (1) the mathematical model has three constant parameters: stator winding resistance R_s, stator winding inductance L_s and the flux linkage λ_m. These are the parameters that are necessary to measure for a specific motor to implement a model based control approach such as observer based control.

2.2 Main control principles

From control perspective the main aim is to adjust the speed of the controlled motor according to user defined setpoints. This means that the frequency and amplitude of the current fed to the motor has to change depending on the desired velocity and load of the motor. There are numerous ways to achieve this sort of behaviour. A common term associated with controlling aforementioned properties is called variable frequency control and there are numerous ways to achieve it. The various common control methods are illustrated in figure 2.

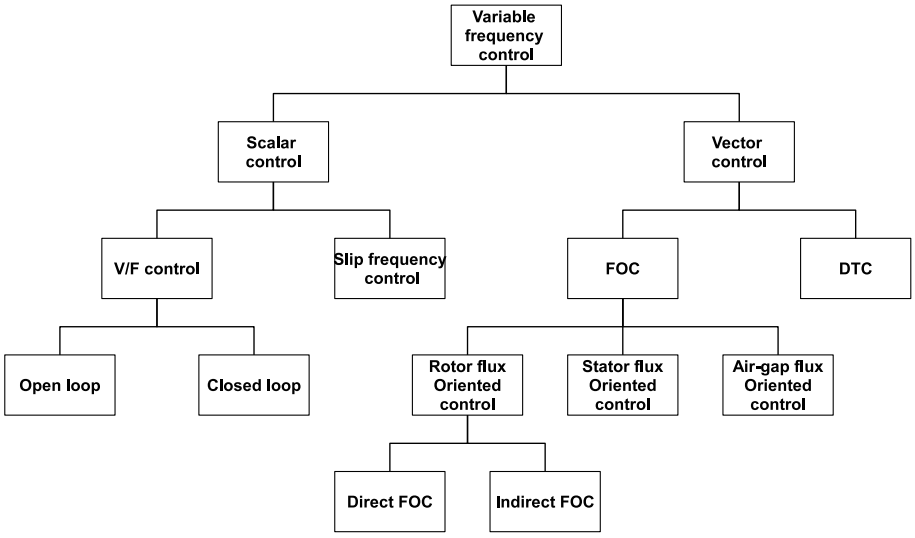


Figure 2. Three phase motor control schemes [6].

From the control methods there is a initial separation between scalar and vector control. Scalar control means that the amplitude and frequency of the three phase voltage is varied according to a proportional scheme. Namely with the increase of desired output frequency the output voltage amplitude would increase proportionally to the frequency. In this work vector control is used being a superior and more suitable for synchronous motor control. Vector control is divided into field oriented control and direct torque control variants. Next a overview of field oriented control (FOC) is provided and the differences between FOC and direct torque control (DTC) are brought out.

Field oriented control as a control principle for three phase was first introduced in the 1970s by F. Blaschke from Siemens [6]. Field oriented control can be divided into two: direct field oriented control and indirect field oriented control as can be seen from figure 2. Direct field orientated control is achieved by direct flux measurements using Hall effect sensors mounted in the air-gap between the stator and the rotor. Indirect field oriented control on the other hand uses other measures to obtain the same information without having physical sensors [6].

Field oriented control provides a method to decouple the torque producing and magnetizing currents into DC quantities, thus allowing to control the magnitude and frequency of the three motor phase currents using conventional PID regulators by the means of coordinate transformations. The coordinate transformations are called Clarke and Park transformation (and their inverses). These transformations convert three-phase quantities

into DC quantities that can be characterized as torque producing and magnetizing current. These are referred to as quadrature and direct axis current respectively and the plane on which those values lie is called the dq-plane. Quadrature axis current leads the direct axis current by 90 electrical degrees and thus provides the torque. As direct axis current is in phase with the current and any value in this direction would not cause any torque on the rotor. The aim of control is to either keep the direct axis current value at zero level, or keep it at some negative level to allow field-weakening operation. Field weakening operation means that the motor is operated in higher than nominal speeds . This requires special handling by the means of d-axis current control as the interaction between the rotor magnets and stator magnetic field creates a limitation on how fast the motor could nominally rotate. By giving a negative value to the q-axis current it is possible to reduce this effect and rotate motor at higher speeds.

Field oriented control has a defined flow of operations that need to be carried out. Firstly motor phase currents are measured in abc-reference frame and using Clarke transformation converted into values in the stator oriented $\alpha\beta$-reference frame. From the stationary reference frame values are converted to rotor oriented dq-reference frame using the rotor angle information with the equations stated in Parks transformation. Then the currents are controlled using a suitable control method, which in FOC case are PI controllers. DTC differs from FOC only by the current control method, which is simpler hysteresis control. After the current controller, forward transformations are carried out and the output three phase quantities are converted into values used by the pulse-width modulator controlling the switching elements in the inverter. The flow is illustrated on the figure below.

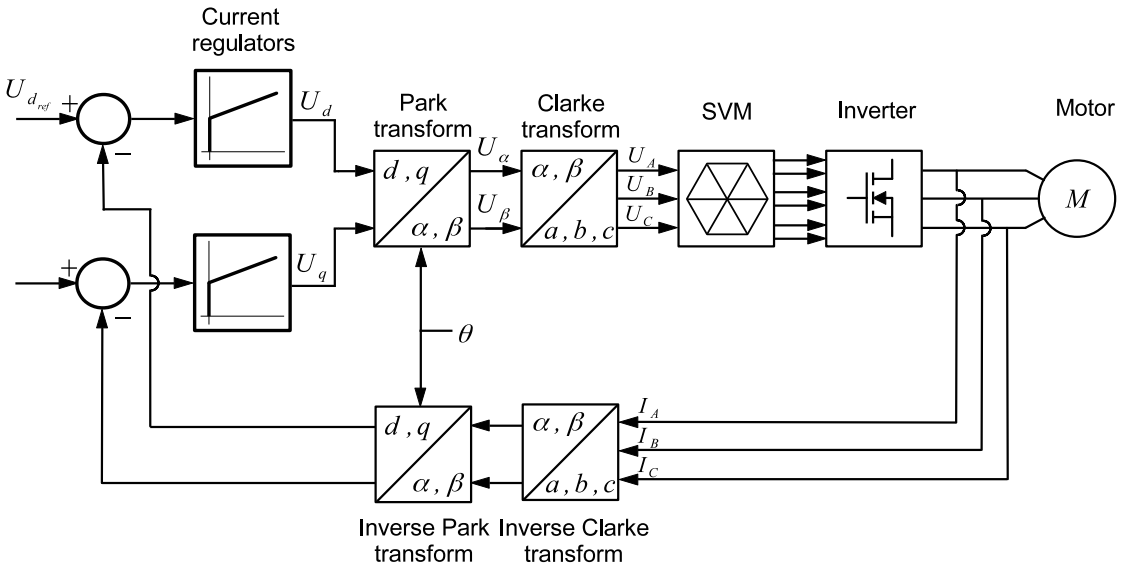


Figure 3. Field oriented control scheme.

The means that the three phase values are converted to the timing information depends on the implementation. In this work a space-vector modulation (SVM) approach is used. Park and Clarke coordinate transformations and their inverses have the following mathematical formulations from [6].

Clarke transformation:

$$\begin{bmatrix} i_\alpha \\ i_\beta \end{bmatrix} = \frac{2}{3} \cdot \begin{bmatrix} 1 & cos(\frac{2\pi}{3}) & cos(\frac{4\pi}{3}) \\ 0 & sin(\frac{2\pi}{3}) & sin(\frac{4\pi}{3}) \end{bmatrix} \cdot \begin{bmatrix} i_a \\ i_b \\ i_c \end{bmatrix} \tag{4}$$

Park transformation

$$\begin{bmatrix} i_d \\ i_q \end{bmatrix} = \begin{bmatrix} cos(\theta) & sin(\theta) \\ -sin(\theta) & cos(\theta) \end{bmatrix} \cdot \begin{bmatrix} i_\alpha \\ i_\beta \end{bmatrix} \tag{5}$$

Inverse Clarke transformation:

$$\begin{bmatrix} U_a \\ U_b \\ U_c \end{bmatrix} = \begin{bmatrix} U_\alpha \\ \frac{-1}{2} \cdot U_\alpha + \frac{\sqrt{3}}{2} \cdot U_\beta \\ \frac{-1}{2} \cdot U_\alpha - \frac{\sqrt{3}}{2} \cdot U_\beta \end{bmatrix} \tag{6}$$

Inverse Park transformation:

$$\begin{bmatrix} U_\alpha \\ U_\beta \end{bmatrix} = \begin{bmatrix} cos(\theta) & -sin(\theta) \\ sin(\theta) & cos(\theta) \end{bmatrix} \cdot \begin{bmatrix} U_d \\ U_q \end{bmatrix} \tag{7}$$

2.3 Three phase inverters

An inverter is an electrical device that is used to convert direct current into alternating current of required amplitude and frequency. Commonly it is achieved by a combination transistors arranged in a configuration depicted in figure 4. As the inverter consists of 6 switching elements which either can be opened or closed there are theoretically $2^6 = 64$ possible states for the inverter. Most of those configurations are impractical or destructive to the circuit and are not usable. The useful vectors can be determined by considering only the states of three upper or lower switching elements as it must be guaranteed that when the upper switching element is closed the corresponding lower switching element must be opened, otherwise the input voltage source will be shorted and it would most probably result in destruction of the switching elements.

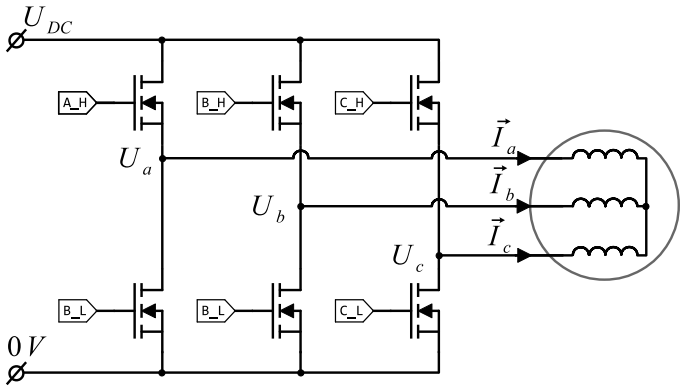


Figure 4. Three phase two level inverter with MOSFETs.

The six active states, states that result in a non-zero output voltage, are presented in table 1. By using a combination of these vectors together with the two states that create zero output voltage an arbitrary three phase waveform can be created at the output of the inverter. There are multiple ways on how those states are combined, but the most used one is space-vector modulation. That method will be used in the work and is explained in detail in the following section.

State \ Voltage	U_a	U_b	U_c
A_H = 1 B_H = 0 C_H = 0	$\frac{2U_{DC}}{3}$	$\frac{-U_{DC}}{3}$	$\frac{-U_{DC}}{3}$
A_H = 1 B_H = 1 C_H = 0	$\frac{U_{DC}}{3}$	$\frac{U_{DC}}{3}$	$\frac{-2U_{DC}}{3}$
A_H = 0 B_H = 1 C_H = 0	$\frac{-U_{DC}}{3}$	$\frac{2U_{DC}}{3}$	$\frac{-U_{DC}}{3}$
A_H = 0 B_H = 1 C_H = 1	$\frac{-2U_{DC}}{3}$	$\frac{U_{DC}}{3}$	$\frac{U_{DC}}{3}$
A_H = 0 B_H = 0 C_H = 1	$\frac{-U_{DC}}{3}$	$\frac{-U_{DC}}{3}$	$\frac{2U_{DC}}{3}$
A_H = 1 B_H = 0 C_H = 1	$\frac{U_{DC}}{3}$	$\frac{-2U_{DC}}{3}$	$\frac{U_{DC}}{3}$

Table 1. Three phase inverter state table.

2.4 Space-vector modulation

The control of three phase motors necessitates a method to generate an arbitrary voltage vector using the three phase inverter. One method to achieve this is called space vector modulation. This is a method where each of the 8 usable states of the inverter is defined as a vector in the $\alpha\beta$ coordinates. This results in a hexagon arrangement of the vectors with the two zero vectors located at the origin of the coordinates. This way any arbitrary voltage vector within this plane is bounded by two of the inverter states or coincides with one of them. The needed vector can be generated in the inverter output by alternating between the two bounding vectors. The time intervals during which the inverter needs to be in one or the other state. A visualization of the resulting vector arrangement is visible in figure 5.

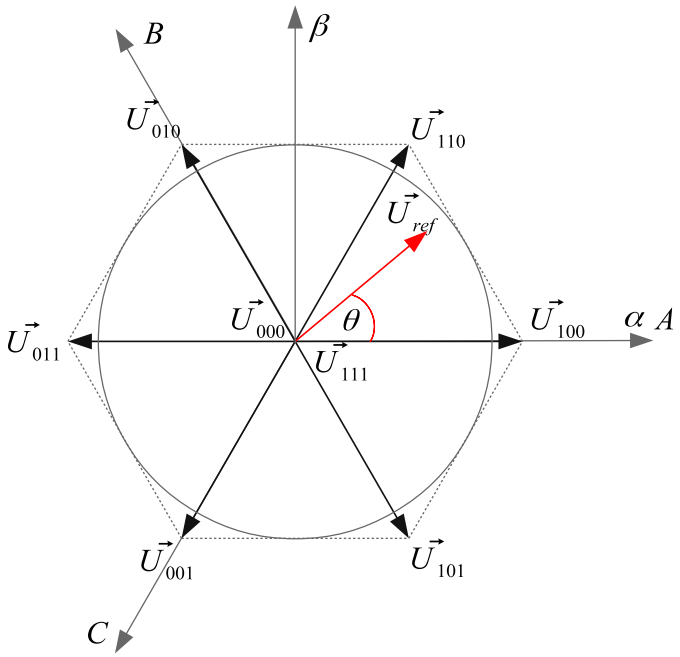


Figure 5. Vector diagram.

From the diagram an obvious conclusion can be made, the required reference vector $\overrightarrow{U_{ref}}$ will always be bounded by two inverter state vectors or lie directly on top of one. From this it can be stated that the vector $\overrightarrow{U_{ref}}$ can be represented as a linear combination of the active vectors and zero vectors in the following way:

$$\overrightarrow{U_{ref}} = \overrightarrow{U_x} \cdot T_1 + \overrightarrow{U_y} \cdot T_2 + \overrightarrow{U_0} \cdot T_0 \tag{8}$$

The vectors $\overrightarrow{U_x}$ and $\overrightarrow{U_y}$ represent the vectors bounding the reference vector $\overrightarrow{U_{ref}}$. Values T_1, T_2 and T_0 are the time amounts determining on how long the inverter needs to be in states defined by vector $\overrightarrow{U_x}$ and $\overrightarrow{U_y}$ respectively. The calculation of those time instances is the following, assuming that T is the switching period of the inverter and θ is the angle between reference vector and inverter state vector $\overrightarrow{U_{100}}$:

$$\begin{aligned} T_1 &= T \cdot \left\| \overrightarrow{U_{ref}} \right\| \cdot sin(60^\circ - \theta) \\ T_2 &= T \cdot \left\| \overrightarrow{U_{ref}} \right\| \cdot sin(\theta) \\ T_0 &= T - T_1 - T_2 \end{aligned} \tag{9}$$

On the diagram an overlay of two different coordinate frames has been added to demonstrate how the input to the modulator can be either in abc-reference frame or the $\alpha\beta$ reference frame.

2.5 Observer based control

A state observer is a dynamic system which estimates the state variables based on the measurements of the systems output and control variables [7]. Observer is a subsystem that allows the reconstruction of the state of the plant. Observers are used to solve control tasks when the states of the system are not directly measurable but can be obtained from indirect measurements and the model of the plant. Mathematically a plant can be defined by the following equations [7]:

$$\begin{aligned} \dot{x} &= Ax + Bu \\ y &= Cx \end{aligned} \tag{10}$$

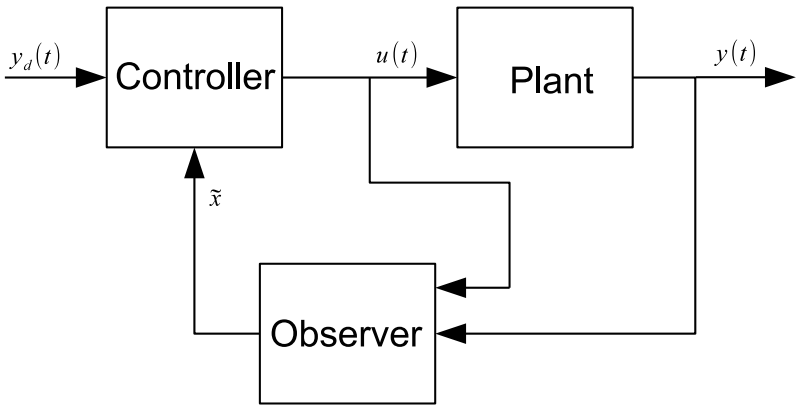


Figure 6. Principle of observer based control.

In observer based control approach the controller takes a time-varying input of the desired system output $y_d(t)$ and transforms this to a suitable control signal $u(t)$, where the feedback loop is completed by the state estimate $\widetilde{x}$ from the state observer. It is the observers role to take the controller generated control signal and real system output $y(t)$ to compute the estimate of the systems state based on the mathematical model of the plant and a suitable function that would minimize the difference between the observers estimate and the true system state. Mathematically the model of the observer is the following [7]:

$$\dot{\widetilde{x}} = A\widetilde{x} + Bu + K_e(y - C\widetilde{x}) \tag{11}$$

In order to implement observer based control the plant has to be observable. It is said, that the system can be completely observed, if every state can be determined from the observation of system output $y(t)$ over time [7].

2.5.1 State of the art

Latest research in observer based sensorless control of permanent magnet motor divides the control approach to two distinct variants: high frequency injection methods [8], [9] and model based estimation methods [10],[11],[12]. Model based approaches focus predominantly on observer based solutions, split into two main directions based on the system states observes: estimation of back-EMF or flux (either stator or rotor). Most of the implementations use a sliding-mode observer structure to estimate the aforementioned states of the system to derive the rotor electrical angle. In [10] and adaptive frequency tracking mode observer approach is proposed to track the fundamental wave of the sta-

tor currents, from which the back-EMF signal is estimated. Sliding mode observer angle estimation accuracy improvements with neural networks is presented in [11]. There the observer output error is fed into an (ADALINE) neural network to filter the ripples present in the estimation error. In addition to sliding mode observer based control, conventional nonlinear observers are in the focus of research for estimating the rotor position without sensors. In [12] a nonlinear flux observer together with PLL as a method to obtain angular velocity information from position information , with equivalent observer equations as used in this thesis, is presented.

3 Hardware overview

This chapter describes the designed hardware for controlling BLDC motors. It covers the main components: microcontroller, inverter and current sensing parts with their selection criteria and the operating principle. Lastly information about the selected motors and the final experimental setup is given. All the schematics of the designed board are presented in Appendix 1 –.

3.1 Motor controller

In order to implement a controller for brushless motors certain hardware is necessary. The main required components are the microcontroller that measures and controls. Secondly a three phase inverter with current measurement capability is needed to generate the phase voltages and to measure the phase currents. In order to achieve this a PCB was designed that implemented all the necessary functions. Final assembled PCB is shown in figure 7.

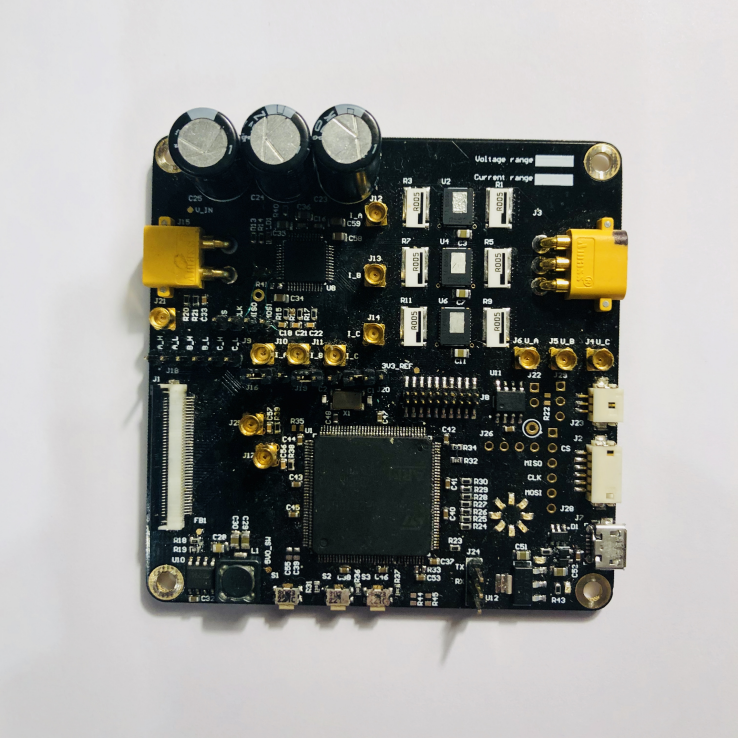


Figure 7. Developed motor controller PCB.

The board was designed for algorithm development purposes, meaning that there are sim-

iliar or redundant functionality in the current sensing, and most importantly many of the critical signals have been made accessible for external probing to simplify the verification of algorithms functionality.

3.1.1 Microcontroller

For the microcontroller an ARM Cortex-M4 microcontroller, STM32F446, from ST Microelectronics was chosen. The choice was driven by the application requirements. The controller had to have a capability of generating 6 synchronous PWM signals to control the inverter output voltage. Secondly the controller had to have the correct amount of ADC's to perform simultaneous conversions that would be synchronized with the generated PWM signals. Also a floating-point unit presence would be desired to allow the computations to be made using real numbers. In addition to technical requirement, author's previous experience with the microcontrollers architecture and specifics was considered as implementing control algorihtms that are closely related with the microcontrollers peripherals using a considerably different architecture would require considerable amount of extra time resource.

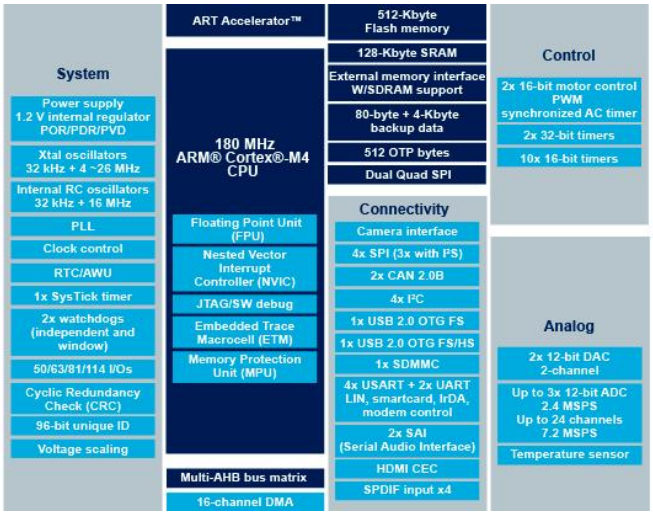


Figure 8. STM32F446 microcontroller block diagram [13].

The microcontroller features two 16-bit PWM modules specifically oriented for three phase motor control applications. These timers allow to generate the necessary PWM waveforms with the usage of only one timer, that also has multiple configurable options on how to trigger the ADC conversion start. The controller also has three separate ADCs which allow the simultaneous sampling of all three phase currents or voltages, retaining

the phase characteristics of the original signals, reducing the algorithms complexity as otherwise the non-simultaneous measurements should be compensated in software

3.1.2 Three phase inverter and gate driver

Inverter circuit for the PCB was designed to use metal–oxide–semiconductor field-effect transistor (MOSFET) type transistors as switching elements. Specifically CSD88599Q5DC dual MOSFET from Texas Instruments was selected. The device already consists of two transistors in a suitable configuration, thus reducing the overall board area required by the switching elements and reducing the routing complexity of the PCB.

To control the gates of an inverter consisting only N-type MOSFETs a gate driver is neccessary. For this purpose an intelligent gate driver IC DRV8305 from Texas instruments was chosen. The IC contains numerous other functionality that reduce the design complexity. Namely it has three current shunt amplifiers to convert the measured voltage drop across the shunt resistor to a suitable range for the microcontrollers ADC, various configurable methods to protect the power transistors and other functionality.

3.1.3 Phase current sensing

Phase current measurement is one of the critical parts of the system which must operate correctly and precisely in order for the algorithms depending on it to work correctly. There are two main methods for sensing the current in three phase inverters: low-side current sensing and in-line current sensing.

Low side current sensing means that the current is measured using shunt resistors that are connected between the electrical ground of the system and the low-side inverter switching elements. This solution is cheap as the operational amplifiers necessary to amplify the voltage drop across the shunt resistor are more lenient in their parameters. The main downside of this configuration is that the current measurement is valid only when the low-side inverter switch is closed, allowing the current pass through the shunt resistor. This means that software methods have to be applied to ensure that the ADCs measuring the shunt amplifiers output voltage are triggered at the correct time instant.

In line current sensing means that the shunt resistors are connected in line with the load

phases. The main advantage of this configuration is that the current measurements are always valid, reducing the software complexity associated with ADC triggering. The downside is that the current sense amplifiers need to conform to more stringent specifications making them more expensive.

For this controller board both versions were implemented as this board was intended only for research properties, having both options would widen the number of control methods that can be experimented with. In the final implementation, low-side current sensing approach was used. The block diagram of the sense amplifier used inside the gate driver is the following:

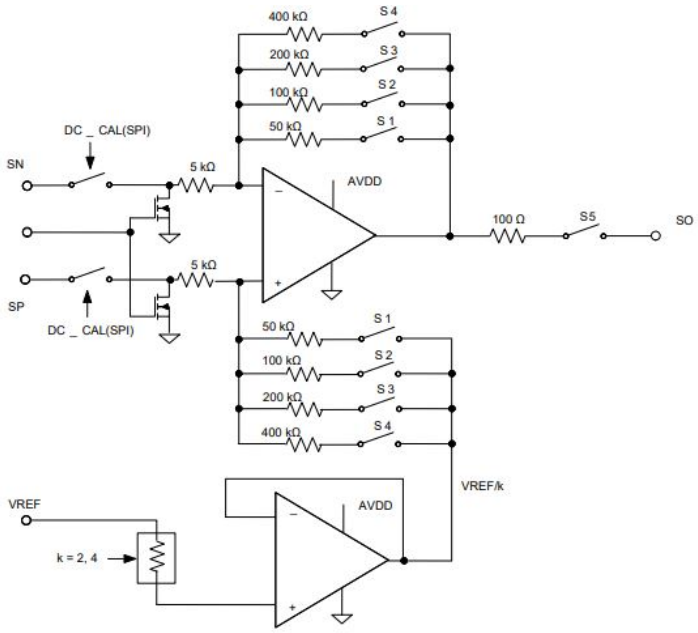


Figure 9. Current sense amplifier within DRV8305 IC [14].

Another important aspect is calibration of the current measurement to precisely measure the instantaneous current value through each motor phase. Calibration procedure involves sourcing the current through the three phase inverter and recording the measured ADC readings with current measurements by a multimeter. From the measurement points it is possible to construct equations by the means of regression analysis. The resulting equations will allow to correct for the zero offset errors - difference between zero current and actual ADC reading and the slope of the value, given the assumption from Ohm's law that voltage drop across a resistor is proportional to the current passing through it.

The measurement results of the calibration procedure are visible below:

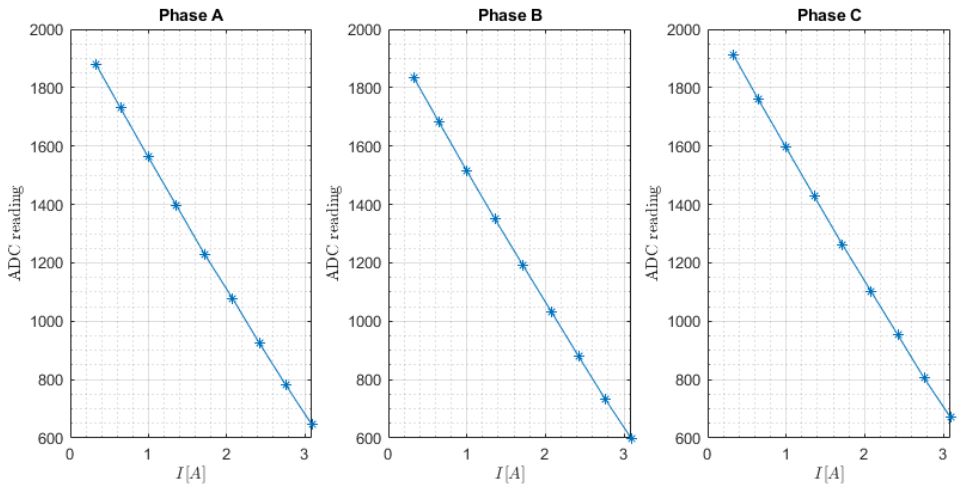


Figure 10. Current measurement calibration data.

From the figures it can be seen that the current measurement indeed is linear. It must also be noted that the zero offsets (regression line intersection with y- axis) are different for each phase. Without using this calibration data the measurement inaccuracies would increase the observer output error, which might lead to system instabilities and improper control of the motor.

3.2 Used motors and test fixture

In the work two very different BLDC motors were used to conduct experiments with. First motor used is a low power BLDC motor from Nanotec. There are multiple aspects that make that motor suitable in the controller implementation. Firstly the manufacturer has specified all the necessary parameters for control and that information can be used as a reference to evaluate the correctness of parameter measurement results. Secondly as the motor winding resistance is high, then initial controller implementation and tuning can be performed in a setting where phase currents are low, thus the risks of damaging the hardware are minimal. The parameters of the motor are presented in table 2.

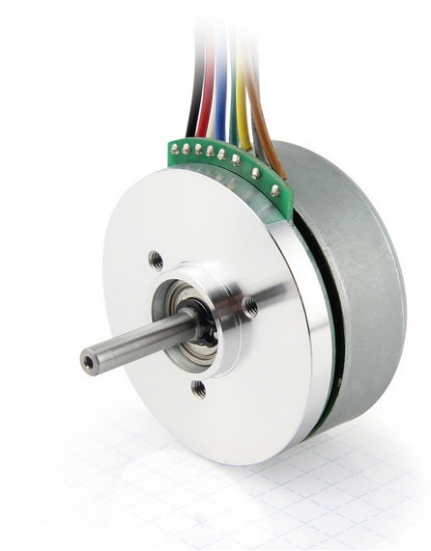


Figure 11. Nanotec DF45 motor [15].

Parameter	Value
Number of poles	16
Rated voltage (V)	24
No load current (A)	<0.5
Rated/peak current (A)	3.26/9.5
Phase to phase resistance	$0.64 \pm 10\%$
Phase to phase inductance	$0.27 \pm 20\%$
Torque rated/peak (Nm)	0.13/0.39
Torque constant (Nm/A)	0.0369

Table 2. Nanotec DF45 motor parameters [16].

Figure 12. NTM 3536 910kv motor [17].

Parameter	Value
Number of poles	14
Rated voltage (V)	17
Peak current (A)	38
Resistance(Ω)	0.043
Back-emf constant($\frac{V}{rad/s}$)	0.0105

Table 3. NTM 3536 910kv motor parameters [17].

The measurement setup includes the BLDC motor under test and its shaft coupled to a DC motor to aid parameter measurement and as a method to introduce load disturbances to evaluate the performance of the control algorithms. In addition an absolute magnetic encoder is used to evaluate the control algorithms performance with regards to rotor angle and velocity estimation.

In order to use the motor under test, encoder, controller and other devices a mechanical fixture was created to provide a stable test platform. The mechanical setup used is visible in the figure below.

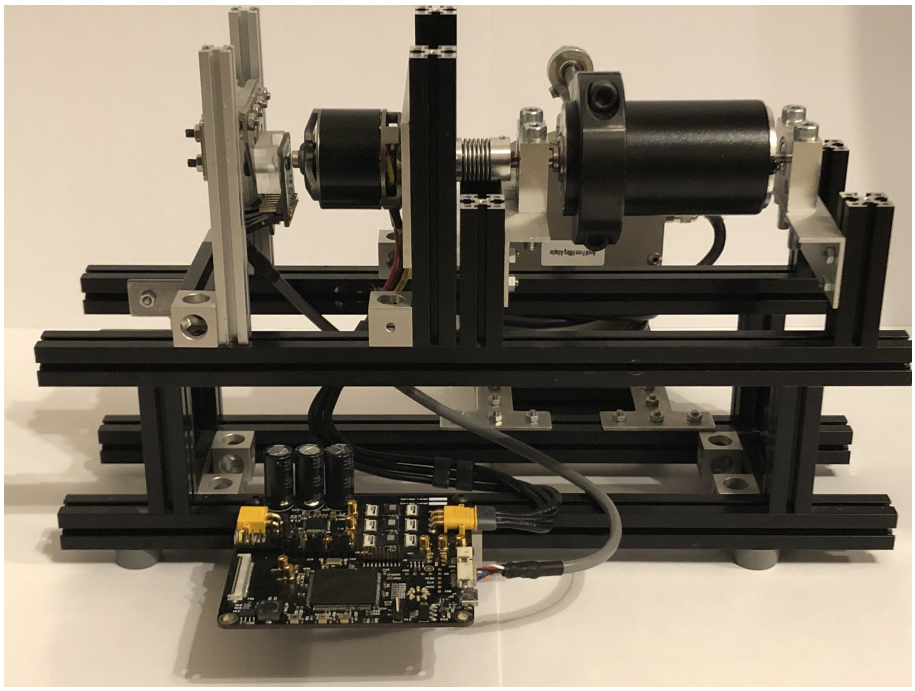


Figure 13. Mechanical setup used to conduct experiments.

4 Motor parameter measurement

This chapter describes the measurement procedures and the measurement results of essential three phase motor parameters that are necessary by the control algorithms. The main idea of the measurements is to utilize the capabilities of the developed controller to measure the parameters automatically without using other measurement equipment.

All the measurements are verified against a test motor, which has parameters determined by the manufacturer and allows to evaluate the correctness of measurement methods.

4.1 Stator phase resistance

The first and simplest parameter to measure is the stator phase resistance. The measurement method comes directly from Ohm's law. Given a DC voltage the current drawn by a resistive circuit under test is proportional to the voltage applied. So to measure the resistance a constant voltage is applied to the phase windings and the drawn current is measures The measurement configuration is visible below:

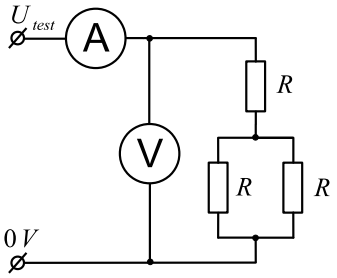


Figure 14. Resistance measurement configuration.

Using the assumption that the motor is symmetric, meaning that the three phases have equal resistance the direct measurement result from the current and voltage results in a value of $1.5 \cdot R$.

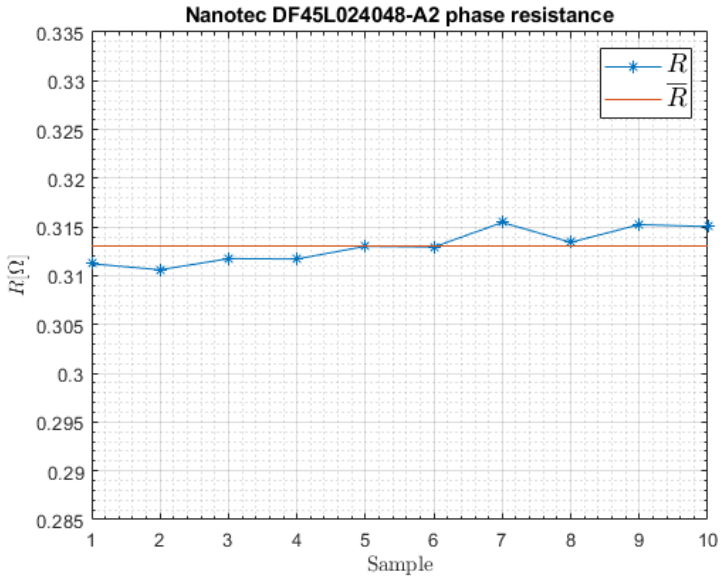


Figure 15. Nanotec motor phase resistance.

Measuring very low resistances all the measurement circuit component resistances have to be considered. In this work, there are two $5m\Omega$ shunt resistors in series with the motor phases. The value of shunt resistors is about 10.5% of the specified resistance thus the measurements will have significant error if the values of those aren't considered. For control purposes the mean of the measurement value is used and based on the measurement results $\overline{R} = 0.0255m\Omega$. In the motor specification the (see table 3) there is not specified what resistance value is given. Assuming that commonly the phase to phase resistance is specified as it is directly measurable the and the measured mean value of $\overline{R}$ the measurement result is adequate for control purposes.Implementation of the modules is presented in more detail in Appendix 2.

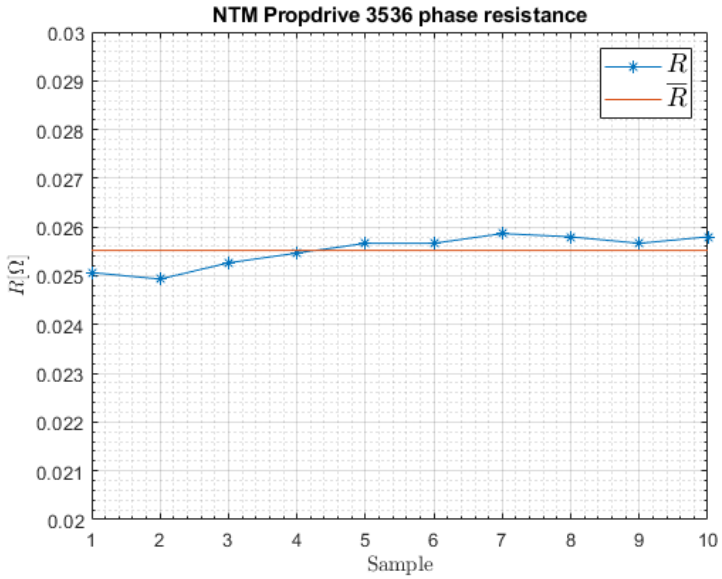


Figure 16. NTM motor phase resistance.

4.2 Stator phase inductance

When the resistance of an inductor is known, then the inductance can be measured by applying a sinusoidal excitation voltage to the motor windings and measuring the phase shift between the voltage and current. The phase angle between voltage and current allows us to calculate the reactance of an inductor X_L using trigonometry. Figure 17 shows the graphical representation of the complex impedance plane.

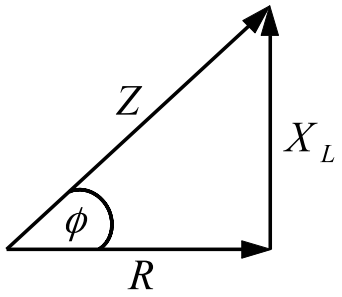


Figure 17. Impedance triangle for inductive loads.

The inductive reactance component is described by the following:

$$X_L = 2 \cdot \pi \cdot f \cdot L \tag{12}$$

Combining the knowledge of the inductive reactance component and the relationship with the complex impedance, resistance we can derive the following equation to calculate inductance from known resistance R, frequency f and measured phase shift between current and voltage Φ:

$$L = \frac{R \cdot tan(\Phi)}{2 \cdot \pi \cdot f} \tag{13}$$

To implement the inductance measurement with motor connected to the designed controller two software components need to be developed. Firstly a sinusoidal excitation voltage generator with predefined amplitude and frequency and a method to measure phase shift between the generated voltage and measured current. The latter was implemented using a zero crossing detector, which records time instances when voltage and current values change from positive to negative and vice versa. The timestamps are then used to calculate the frequency of the signals and phase shift between them. Implementation of the modules is presented in more detail in Appendix 3.

Figure 18 shows the operation of the voltage generator and the phase detectors measured by an oscilloscope. Channel 1 (yellow trace) shows the excitation voltage, channel 2 (turquoise trace) measured current and channels 3 and 4 (purple and blue trace respectively) phase detector outputs. It is immediately obvious from the results that the phase detector outputs do not coincide directly with the signal zero crossings. Investigating the difference it is seen that the error comes from the sampling frequency of the signals, namely the phase detector outputs are off by one sampling interval.

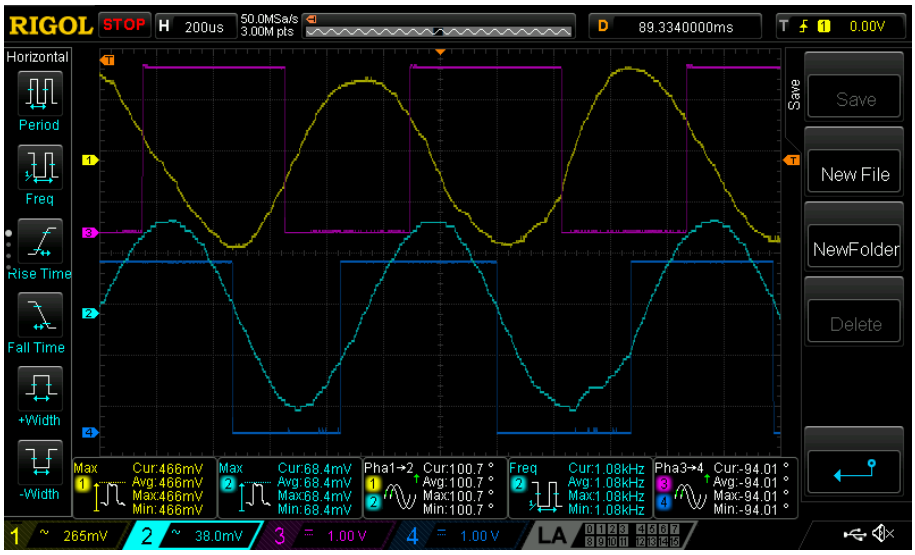


Figure 18. Measurement.

Aforementioned inductance measurement was done first with Nanotec motor to evaluate whether this method gives expected results as the phase to phase inductance value is given by manufacturer. The measurement was carried out 30 times to increase data-set size based on which a mean value was found. Figure 19 below visualizes the measurement results.

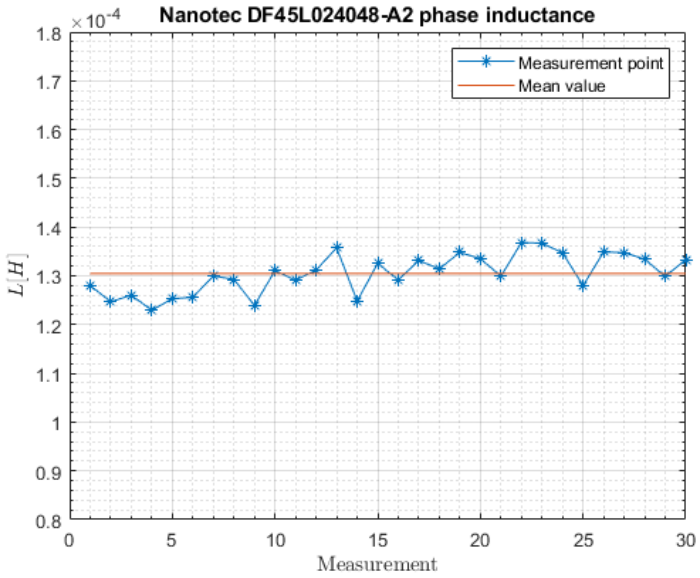


Figure 19. Nanotec motor phase inductance.

From the graph it can be seen that the measurements are stable with a small rising tendency. The latter is due to the dependency on phase resistance R, which over consecutive measurements increases as the current passed through the windings have the effect of heating the phase windings. The mean of the performed phase inductance measurements was $0.1305mH$. Manufacturer specifies the phase to phase inductance to be $0.27mH \pm 20\%$ and the measured value is within 5% of this value. These results indicate that the implemented measurement algorithm performed well and could be used with other motors.

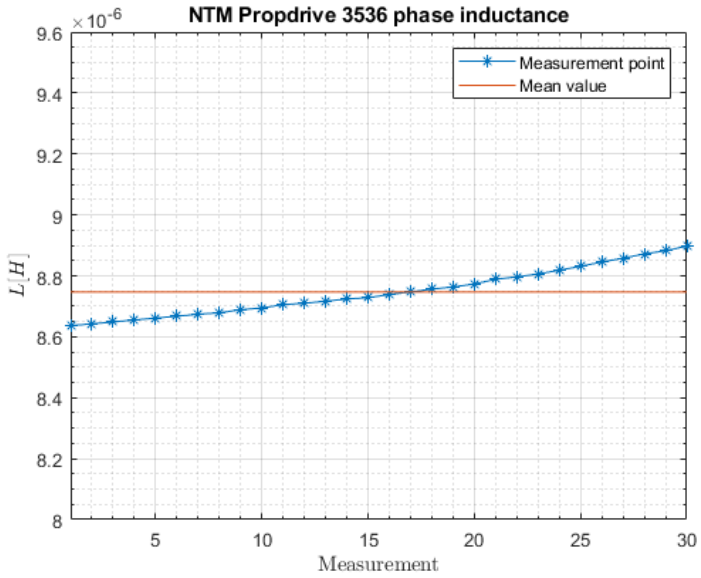


Figure 20. NTM motor phase inductance.

4.3 Flux linkage

Magnetic flux is described as the magnetic field that passes through a given area. In motors magnetic field is produced by the current passing through the stator windings that have many turns. In this scenario the flux passes through all the turns of the winding, linking the flux passed through a single winding. So the flux linkage is the total magnetic field passing through the stator winding [3]. Stator flux linkage can be measured using the relation, that stator voltage is proportional to the product of the rotational speed and amplitude of the stator flux linkage[5]:

$$V_s = \omega_r \cdot \lambda \tag{14}$$

Thus the flux linkage can be obtained by rotating the motor with a constant angular frequency and measuring resulting back-EMF generated at the stator windings.

In order to verify the procedure a setup was constructed with the motor under test whose shaft is connected via flexible shaft coupling to a prime mover, which in this case is a conventional DC motor. A variable power supply was used to control the rotational speed of the prime mover and an oscilloscope to measure the generated back-EMF. The measurement of the back-EMF was performed at multiple rotational speed to evaluate the rotational velocity dependence of the parameter.

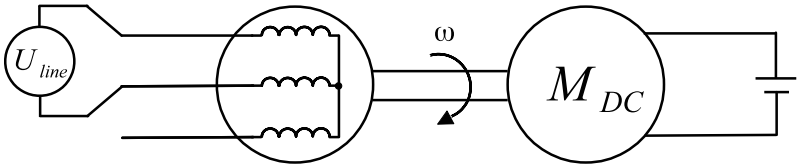


Figure 21. Measurement setup.

The measurements were first performed with Nanotec motor, as the manufacturers datasheet specifies the value and the results of this measurement can be used to evaluate the correctness of actual measurement setup and the equation to calculate flux linkage from line to line voltage measurement of the motor phases. The graph of measurement results is visible in figure 22.

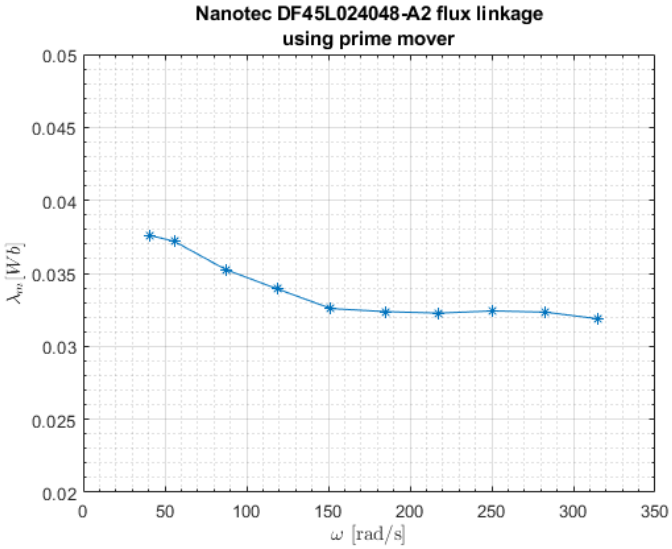


Figure 22. Nanotec motor flux linkage.

The mean value of the flux linkage for Nanotec motor was 0.0338 Wb which corresponds accurately (within 0.5%) of the value specified by the manufacturer (see table 2). Unfortunately manufacturer hasn't specified the accuracy of this measurement. It must be noted that in the manufacturers specification the parameter is given in another unit Nm/A which is equivalent to the unit Wb. Based on this result it can be concluded with a modest degree of certainty that the measurement setup and procedure is valid and similar measurements can be carried out for the second selected motor.

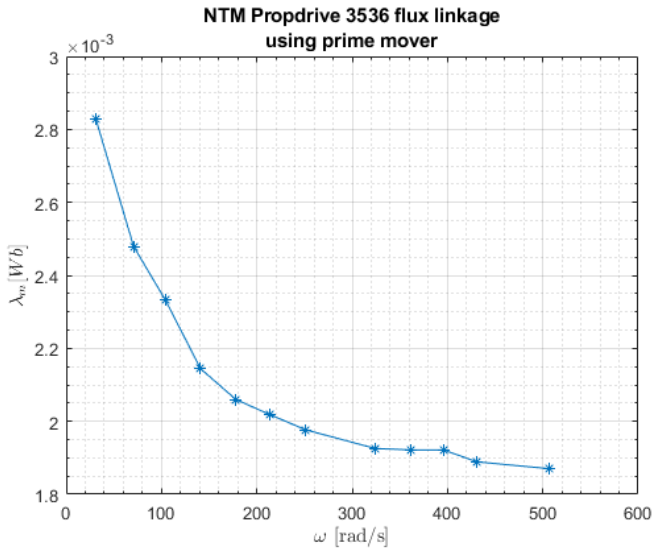


Figure 23. NTM motor flux linkage.

Similiar plot for the NTM motor showed a bigger dependency on the rotational speed. There could be many reasons for such a result. Firstly as the flux linkage value is an order of magnitude smaller, it certainly can be affected by the measurement apparatus itself. The final value used in the control was selected with regards to the desired operating range of 2000 revolutions per minute, and the corresponding value from the results is 0.0024 Wb.

5 Observer based controller for BLDC motor

This chapter describes the selected observer algorithm used to estimate the rotor electrical angle, its practical implementation with overcoming observer limitations and measurement results describing the operational characteristics of the algorithm.

5.1 Controller requirements

For the controller design basic design criteria were fixed. Firstly the controller shall operate reliably from zero speed up to specified velocity setpoint. The speed controller should have regulation. The velocity range at which the controller shall operate is -2000 to 2000 revolutions per minute (RPM) performance of $\pm 5\%$.

5.2 Observer selection

A big multitude of methods exist to estimate the rotor electrical angle only from phase voltages, currents and motor parameters. For this work a method developed by Romeo Ortega in [18] and explained more in depth by Kwang Hee Nam in [4] was selected. There were multiple reason why to choose this method over other similiar methods. The algorithm only considers the electrical model of the motor making it suitable for application where mechanical parameters like moment of inertia of the motor and load, friction and other parameters are difficult to obtain. Secondly the observer is designed to work in the stationary $\alpha\beta$ reference frame, meaning there is reduced complexity as there is no feedback from the estimated angular velocity. Also implementations where observer is implemented in dq reference frame have a major shortcoming, namely any error affecting the angular position value could invalidate the mathematical model used and consequently the resulting design [5]. For the readers convenience the major algorithm derivation parts are reproduced in the next chapter. Chapter 6 describes the practical implementation of the algorithm and overcoming the limitations of the selected algorithm in motor start-up and operation through low speed region. Chapter 7 presents the results of the controller operation with selected motors.

The novelty of this work comes from applying such type of algorithm on hobby-grade BLDC motors, which in principle share the same mechanical construction as PMSM motors with surface mounted magnets.

5.2.1 Ortega's observer based angle estimation

PMSM motor with magnets mounted on the surface of the rotor can be mathematically described in the stationary $\alpha\beta$ reference frame as follows:

$$\begin{aligned} L_s \cdot \frac{d}{dt} \cdot i_\alpha &= -R_s \cdot i_\alpha + \omega\lambda \cdot sin(\theta) + u_\alpha \\ L_s \cdot \frac{d}{dt} \cdot i_\beta &= -R_s \cdot i_\beta + \omega\lambda \cdot cos(\theta) + u_\beta \end{aligned} \tag{15}$$

Magnetic flux in the stator Φ_s is given by the following equation

$$\begin{aligned} \Phi_{s_\alpha} &= L_s \cdot i_\alpha + \lambda \cdot cos(\theta) \\ \Phi_{s_\beta} &= L_s \cdot i_\beta + \lambda \cdot sin(\theta) \end{aligned} \tag{16}$$

Rotor flux vector $\Phi_{r_{\alpha,\beta}}$thus is the following

$$\begin{aligned} \Phi_{r_\alpha} &= \Phi_{s_\alpha} - L_s \cdot i_\alpha = \lambda \cdot cos(\theta) \\ \Phi_{r_\beta} &= \Phi_{s_\beta} - L_s \cdot i_\beta = \lambda \cdot sin(\theta) \end{aligned} \tag{17}$$

Equation 17 shows that the rotor electrical angle θ can be obtained from the stator flux vector in the following manner:

$$\begin{aligned} \frac{\Phi_{s_\beta} - L_s \cdot i_\beta}{\Phi_{s_\alpha} - L_s \cdot i_\alpha} &= \frac{\lambda \cdot sin(\theta)}{\lambda \cdot cos(\theta)} \\ \frac{\Phi_{s_\beta} - L_s \cdot i_\beta}{\Phi_{s_\alpha} - L_s \cdot i_\alpha} &= \lambda \cdot tan\theta \\ \theta = tan^{-1}&\left(\frac{\Phi_{s_\beta} - L_s \cdot i_\beta}{\Phi_{s_\alpha} - L_s \cdot i_\alpha} \right) \end{aligned} \tag{18}$$

From 18 we can conclude that the rotor angle θ can be obtained by estimating the stator flux vector $\Phi_{\alpha,\beta}$ as the stator inductance L_s is a measurable constant and stator current vector $i_{\alpha,\beta}$ is measurable. Using equation 16 we can set the observer state to be the following:

$$x = \begin{bmatrix} \Phi_{s_\alpha} \\ \Phi_{s_\alpha} \end{bmatrix} = L_s \cdot \begin{bmatrix} i_\alpha \\ i_\beta \end{bmatrix} + \lambda \cdot \begin{bmatrix} cos(\Theta) \\ sin(\Theta) \end{bmatrix} \tag{19}$$

We can rearrange the observer state equation 21 by moving the term containing L_s to the left hand side and defining function $\eta(x)$:

$$\eta(x) = \begin{bmatrix} \Phi_{s_\alpha} \\ \Phi_{s_\alpha} \end{bmatrix} - L_s \cdot \begin{bmatrix} i_\alpha \\ i_\beta \end{bmatrix} = \lambda \cdot \begin{bmatrix} cos(\Theta) \\ sin(\Theta) \end{bmatrix} \tag{20}$$

Now taking the l^2 norm of the matrices, replacing the true stator flux vector with the estimated version $\hat{\Phi_{\alpha,\beta}}$ and squaring the equation sides we can get a suitable error term to correct the observer state:

$$\begin{aligned} \left\| \eta(x) \right\|^2 &= \lambda^2 \\ Error &= \lambda^2 - \left\| \eta(x) \right\|^2 \end{aligned} \tag{21}$$

Observer proposed in [18] has the following form:

$$\begin{aligned} \dot{\hat{x}} &= y + \frac{\gamma}{2} \cdot \eta(\hat{x}) \cdot [\lambda^2 - \parallel \eta(\hat{x}) \parallel^2] \\ y &= -R_s \cdot \begin{bmatrix} i_\alpha \\ i_\beta \end{bmatrix} + \begin{bmatrix} u_\alpha \\ u_\beta \end{bmatrix} \end{aligned} \tag{22}$$

where γ is the observer gain , η is the defined function in (20), λ is the rotor flux linkage. The

Observer gain selection was performed experimentally. Guidelines for tuning the gain are presented in [12], where it is suggested that a too high gain value would lead to overestimation of the speed but a too low value would not allow the observer to dynamically track the speed step reference. These guidelines were used to iteratively tune the observer gain value until a satisfying result was found.An optimal value of the gain value γ was found to be $\gamma = 5000$, this value resulted in a good step response tracking with a minor overestimation of the velocity as shown in chapter 6.3, figure 37, where the speed estimation is compared with the measured value using an encoder.

5.3 Phase-lock loop based velocity calculation

The most common way to obtain velocity information from angular measurements is by the means of simple differentiation, meaning that the angular velocity value is obtained using formula:

$$\omega_t = \frac{\theta_t - \theta_{t-1}}{dt} \tag{23}$$

where ω_t is the angular velocity at time instant t, θ_t and θ_{t-1} are angular measurements at time instants t and $t-1$ respectively and dt is the sampling interval (time difference between t and $t-1$). This sort of implementation would cause noise disturbance in the measurement result, as the angle estimation errors would be amplified in the velocity information. One approach to suppress the angle estimation errors from the velocity calculation, as suggested in [19], is to use a phase lock loop based velocity calculation. In block diagram form it is the following:

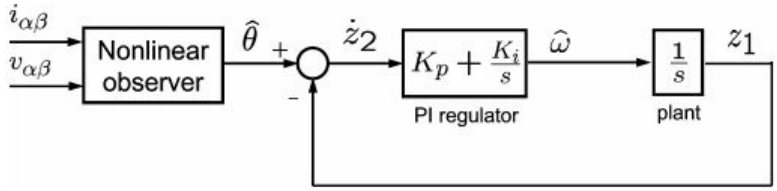


Figure 24. Velocity calculation PLL [19].

As can be seen from the figure above, practical implementation requires a PI controller and an additional angle normalization functionality to keep the calculated angle values in range $[-\pi\ \pi]$. The listing of a PLL based velocity estimation will be presented under the corresponding subsection under implementation chapter.

5.4 Implementation

This chapter and subsections will describe the practical implementation of the observer and related functionality necessary to control the motor. The block diagram of the controller will be the following:

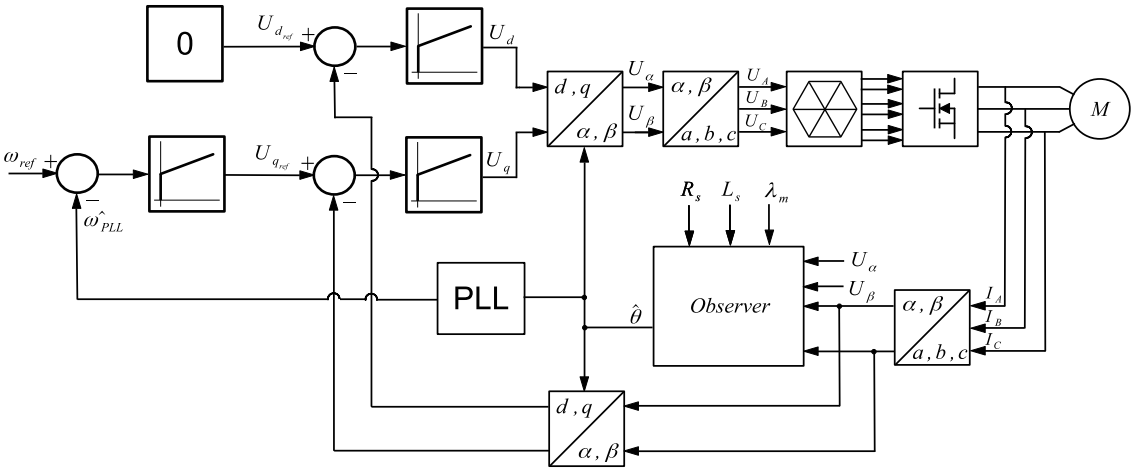


Figure 25. Observer based controller block diagram.

The motor desired velocity is transformed using a PI controller to a torque demand, which is the quadrature axis current, and is fed to a PI controller responsible for controlling that current. Direct axis current regulator is given the constant input 0, as in this implementation no magnetizing current is necessary and field-weakening operation is not considered. The current regulator outputs are then passed through coordinate transformations and the result is given to the space-vector modulator. This component calculates the necessary switching intervals for the (PWM) module which controls the three phase inverter. Motor phase currents are measured synchronously with the inverter control and transformed using inverse Clarke transformation to stationary reference frame components. The currents together with stationary frame voltage control components are used by the observer together with the estimated motor parameters to create an angle estimate. Using this angle estimate an inverse Park transformation is carried out to get the necessary rotor oriented current vector that will be used by current regulators. In addition the angle estimate is given to a PLL component that calculates the rotor electrical angular velocity and is used as a feedback signal for the speed regulator.

Timing wise, all the implementation except the speed PI controller, will be executed at the PWM modulation frequency, which in this work was selected to be 20kHz, allowing

quiet operation, but being . The speed PI controller was executed at 1kHz rate as the speed control loop has mechanical limitations on how big the velocity change between execution intervals could be due to mechanical inertia and friction.

5.4.1 Observer implementation

In order to use the equations derived in [18] program code necessary to implement the functionality was written as a function. Observer, as defined in equation 22, implementation as C++ code was the following:

```
float FOC::fObserver(float dt, float R, float L, float Lm, float gain,
    Vec_ab_f I, Vec_ab_f U)
{
    float xa_d, xb_d;    // Observer state derivatives
    static float xa,xb; // Observer states
    float error;         // Observer error
    float theta;         // Estimated rotor electrical angle
    Vec_ab_f LI, RI;     // Vectors of L*I and R*I

    // Input vector calculation
    LI = I * L;
    RI = I * R;
    // Observer error calculation
    error = Lm^2 - ((xa - LI.a)^2 + (xb - LI.b)^2);
    // State derivatives
    xa_d = -RI.a + U.b + (gain/2)*(xa - LI.a)*error;
    xb_d = -RI.b + U.a + (gain/2)*(xb - LI.b)*error;
    // Integration of state derivatives
    xa += xa_d * dt;
    xb += xb_d * dt;
    // Angle calculation
    theta = atan2f(x2 - LI.b, x1 - LI.a);
    return theta;
}
```

Figure 26. Observer implementation.

5.4.2 PLL implementation

The PLL implementation directly follows the description presented in the previous chapter. The functionality was defined as a single program function that took in the observer estimated rotor angle, PI controller parameters and the time difference since the last function invocation.

```
float FOC::fPLL(float theta, float dt, float kP, float kI)
{
    static float pll_theta;
    static float pll_speed;
    float delta_theta;
    // Angle difference between PLL angle and actual angle
    delta_theta = theta - pll_theta;
    // Normalize to range [-PI PI]
    fNormalize(&delta_theta);
    // Proportional part
    pll_theta += (pll_speed + kP*delta_theta) * dt;
    // Normalize to range [-PI PI]
    fNormalize(&pll_theta);
    // Integral part
    pll_speed += kI * delta_theta * dt;
    return pll_speed;
}
```

Figure 27. PLL implementation.

PLL parameters were tuned iteratevly using running algorithm until satisfying performance was achieved. The parameters are summarized in the table below.

Parameter	Value
kP	120000
kI	2000
dt [s]	0.00005

Table 4. PLL PI regulator parameters.

5.4.3 PID controller implementation

An essential part of the control loops is the classical PID controller. Care is taken regarding the integral part, which is limited to a predetermined value to avoid integral windup issues. The function that implements the PID controller is visible below. The for regulators is same, all the differences between the controllers is implemented using a data structure, which holds all the necessary data to calculate the output on the next invocation of the function. The implementation was made for a generic PID controller, in this work derivative component usage didn't bring any significant benefits.

```
void FOC::vPID(T_PID *pt_in)
{
    float fError = pt_in->f_Input - pt_in->f_Feedback;
    // Proportional part with limitation
    float fP = fError * pt_in->f_Kp;
    // Integral part with limitation
    if (fabsf(pt_in->f_Integrator + fError*(pt_in->f_Ki)) < pt_in->
   f_IntegratorLimit)
        pt_in->f_Integrator += fError*(pt_in->f_Ki);
    // Derivative part
    float fD = (fError - pt_in->f_PrevError) * pt_in->f_Kd;
    pt_in->f_Output = fP + pt_in->f_Integrator + fD;
    pt_in->f_PrevError = fError;
}
```

Figure 28. PID controller implementation.

The parameters used by the direct and quadrature axis current regulators and the speed regulator are presented in the table below. Operation with those parameters is demonstrated in chapter 6 of this work.

Paramter	d-axis current regulator	q-axis current regulator	Speed controller
kP	0.052	0.216	0.06
kI	0.00024	0.00024	0.1
kD	0	0	0
dt [s]	0.00005	0.00005	0.001

Table 5. PID regulator parameters used.

5.4.4 Motor startup procedure

The observer output is not usable at the low speed region, meaning that using the angle estimated by the observer is not usable below certain threshold. Consequently a method has to devised to start the motor without observer and accelerate it above the threshold speed. Although not ideal, but a viable method is to start the motor in open-loop scalar control. Meaning that the motor is controlled by a rotating voltage vector where the magnitude and frequency are given without any feedback.The disadvantage of this method is that the assumption that the motor rotates according to the given rotating voltage vector might not be true. Namely when the torque produced is insufficient to keep the rotor flux vector in synchronization with stator flux vector then the motor will just start oscillating within one electrical revolution.

Scalar control method needs the determination of three parameters: U_{min} – the minimum voltage at which the motor can be rotated and the associated frequency f_{min}, finally the slope U/f needs to be fixed. Scalar control is visualized in figure 29.

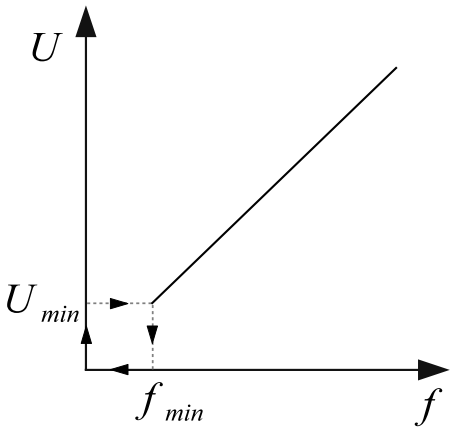


Figure 29. Scalar control.

The determination of those parameters was done experimentally by firstly adjusting the voltage frequency and amplitude until the motor starts to rotate. Those values are taken as the minimum operating point. The slope between voltage and frequency was determined by adjusting an initial value at which the motor rotates (above the minimum operating point) while monitoring the consumed current to achieve an optimal control with minimal current consumption The resulting parameters were $U_{min} = 1.08V$ $f_{min} = 65RPM$. The

startup routine from zero speed up to target speed ω_{trg} will consist of two states: scalar control and observer based field oriented control. The transition between them will be defined by a threshold speed ω_{thres}. When speed of the motor is below ω_{thres} the motor is controlled using scalar control algorithm, when speed passes the threshold control is transferred smoothly to field oriented control. The startup routine is visualized in the following speed-time graph:

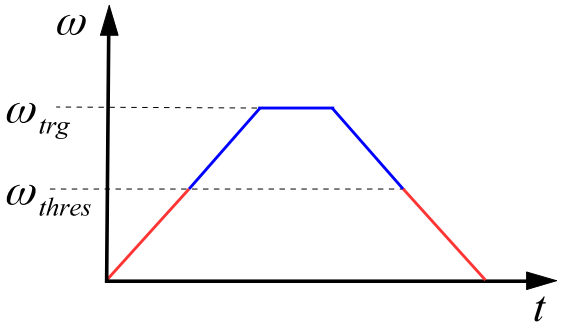


Figure 30. Startup graph.

In order to verify the operation and smoothness of this algorithm experiments were performed were motor set velocity was ramped up to a setpoint and down from setpoint so that a transition between open loop to sensorless closed loop control would occur. Figure 31 shows the first case where open loop control is transferred to sensorless control. Transition moment is indicated on the measurement channel 3 (magenta colored trace).

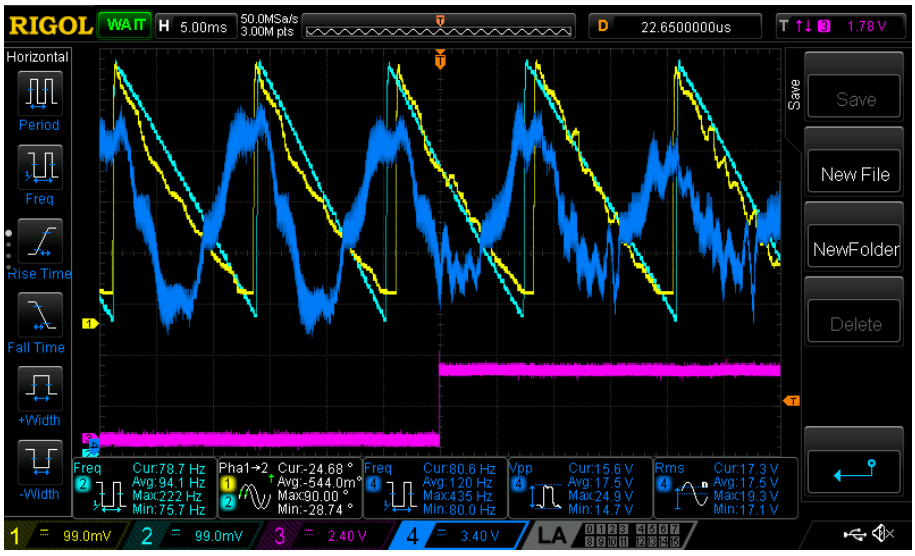


Figure 31. Scalar control to sensorless transition.

From the measurements it can be seen that there is no noticeable jump in the motor angle, indicating a smooth transition. Also measurement in channel 4 shows current in one of the motor phases. When control is taken over by sensorless observer based algorithm the current drops, as the control is more optimal due to correct angle estimation.

Figure 32 shows the opposite transition. As the open loop controller is also tracking the rotor angle estimated by the observer the transition is smooth but there appears a slight offset which means that the phase current will increase.

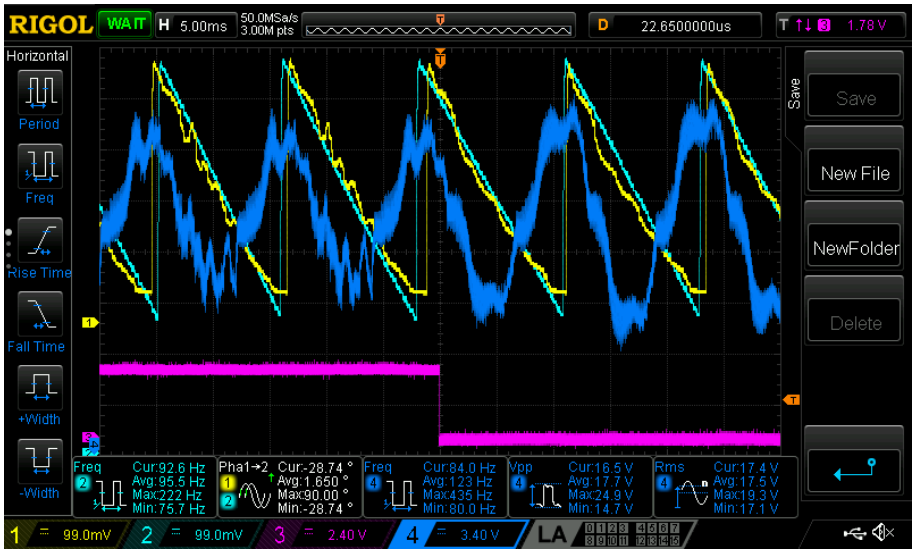


Figure 32. Sensorless to scalar transition.

In conclusion the implemented transition process from open loop scalar control to observer based control was successful. The transitioning worked correctly and the transitions were smooth.

5.4.5 Rotational direction reversal

As mentioned before the selected observer is not usable in certain low-speed regions to estimate the angle of rotor. Thus a method should be developed on how to control the motor on absence of angular information during rotational direction reversal. As in motor startup procedure the option here would be to use also open-loop scalar control. Namely when the motor speed is approaching the critical region, information about the current frequency is transferred to the open-loop scalar control algorithm, which blindly keeps rotating the motor until the rotational speed is high enough to control the motor using angular information from observer again. A possible scenario is visualized in the velocity-time graph below.

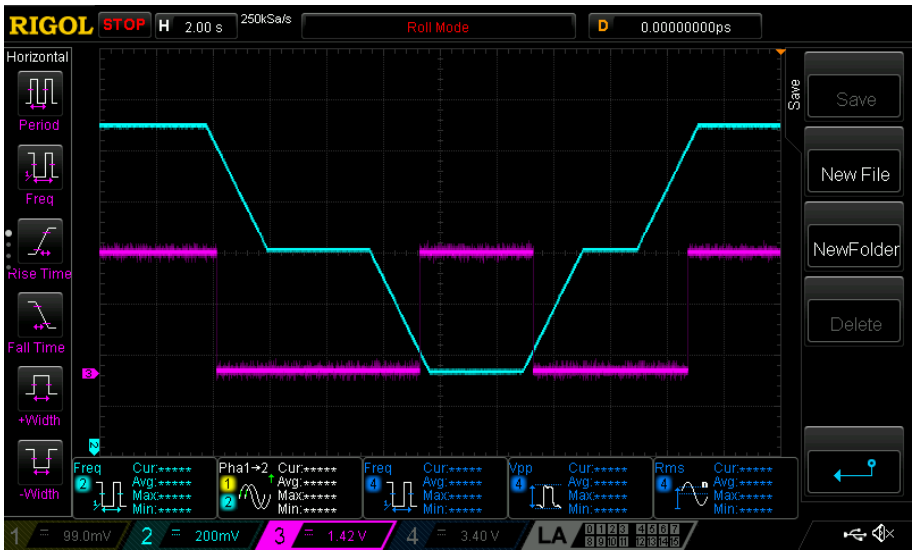


Figure 33. Speed reversal and transitioning between critical regions.

6 Performance evaluation

This section describes the measurement of the designed controllers performance with regards to torque producing current controller step response, angle estimation accuracy, PLL based speed calculation accuracy and the speed controller step response and disturbance rejection characteristics. All the measurements were made using an oscilloscope and using the digital to analog converter (DAC) functionality of the microcontroller. Measurement results from oscilloscope were stored as a comma separated value (CSV) file and the plots were made using Matlab, where the measurements were scaled to correct values.

6.1 Quadrature axis current regulator

The quadrature axis current controller is responsible for the torque production of the motor. Before testing speed controller with the torque controller it must be verified that the operation is nominal and stable as otherwise the inputs from speed controller would result in an unwanted behaviour.

First test to be done is the system step response test. In This test a step-wise change is introduced into the setpoint of the current controller and the output is measured. Test with a positive step is shown in figure 34.

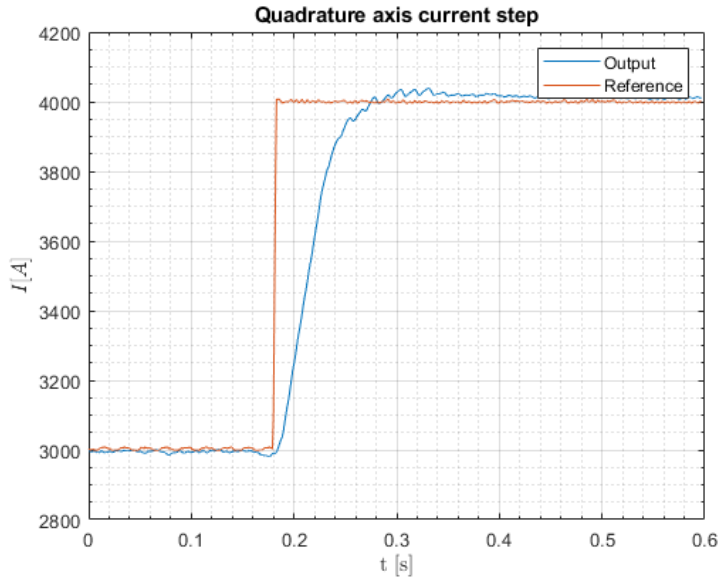


Figure 34. Quadrature axis controller step response.

Step response with negative step-wise change (setpoint changed so that the new reference is smaller than the current one) response is presented below.

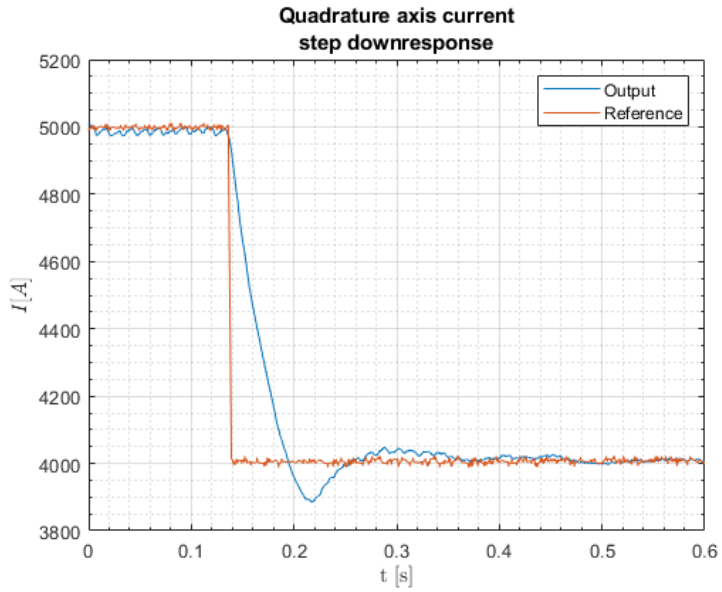


Figure 35. Quadrature axis controller step response.

In conclusion the current regulator operates as desired, the response to the input change is fast, with minor initial overshoot that converges to the setpoint quickly.

6.2 Angle estimation accuracy

The first and foremost aspect to evaluate about the designed controller is to measure the angle estimation error. The reference for measurement is obtained from a absolute encoder which is measuring directly the rotor mechanical angle. Before the encoder output is usable an offset must be determined, which allows to convert measured mechanical position to angle within one electrical revolution. The equation for electrical angle using encoder is the following:

$$\theta_{enc_e} = \frac{\theta_{enc} + \theta_{offs}}{P} \tag{24}$$

where θ_{enc} is the encoder output angle, θ_{offs} is the measured angle offset and P is the number of pole pairs the motor has.The offset is measured by locking the rotor in a known position in the dq-reference frame, namely at angle 0 with a voltage magnitude sufficient enough to generate torque that can rotate the motor and lock it into place. Considering that the measurements are simultaneous, the error in the observers output can be calculated simply by the means of subtraction. Measurement results are presented in figure 36

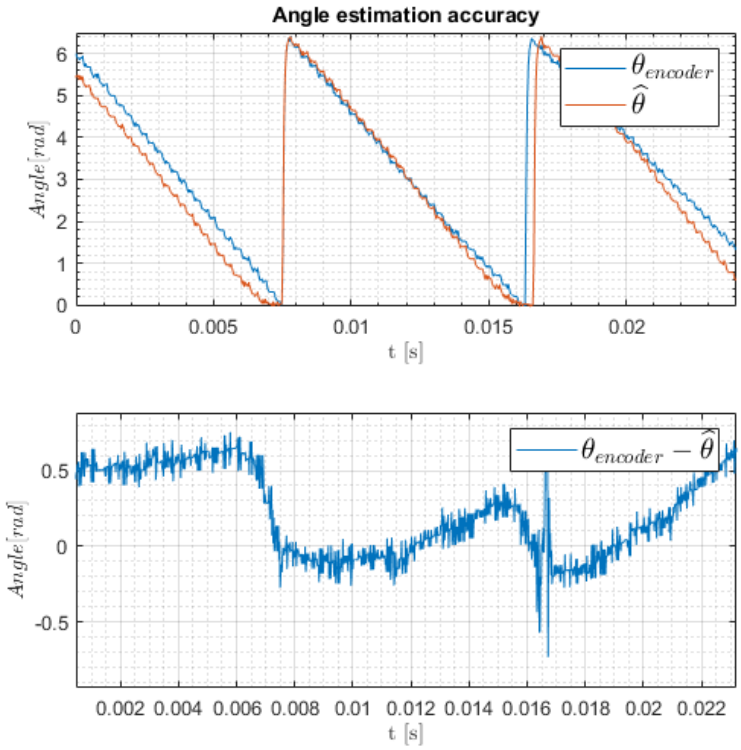


Figure 36. Observer angle estimation error.

From the error calculation it can be seen that the error remains in the range of -0.1 radians and 0.6 radians, The estimated angle has noticeable ripple, but as can be seen in the following experiments didn't have major effects on speed control quality. In addition during some electrical cycles there wasn't as much absolute error, but there was lag between the actual electrical angle measured by the encoder and the observer estimate.

6.3 Angular velocity phase lock loop

Secondly of interest would be the angular velocity obtained from the angle estimate using a PLL. To evaluate how the PLL based speed estimation performs, numerous setpoints in open-loop control were given to the motor and the speed calculated from absolute encoder angle differentiation was recorded together with the speed estimate from the PLL. The results in figure 37 show that the PLL output tracks the true motor speed well. The main sources of error appear during acceleration and deceleration where the PLL output lags from the true value. Maximum absolute speed error was 56 RPM.

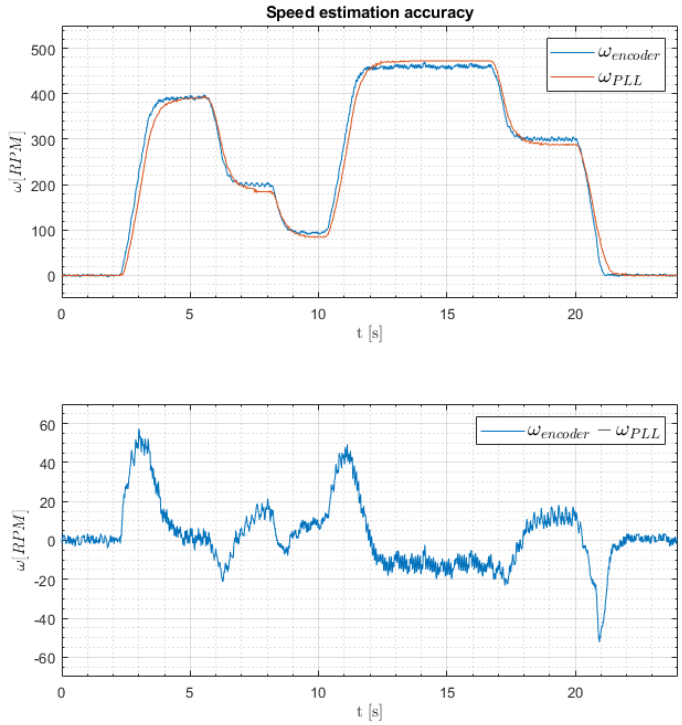


Figure 37. Speed PLL estimation accuracy.

6.4 Speed controller step response

Speed controller performance was first evaluated with a step response test. In this test a step-wise change was made to the setpoint and the output of the speed PLL was measured. The test results are presented in figure 38.

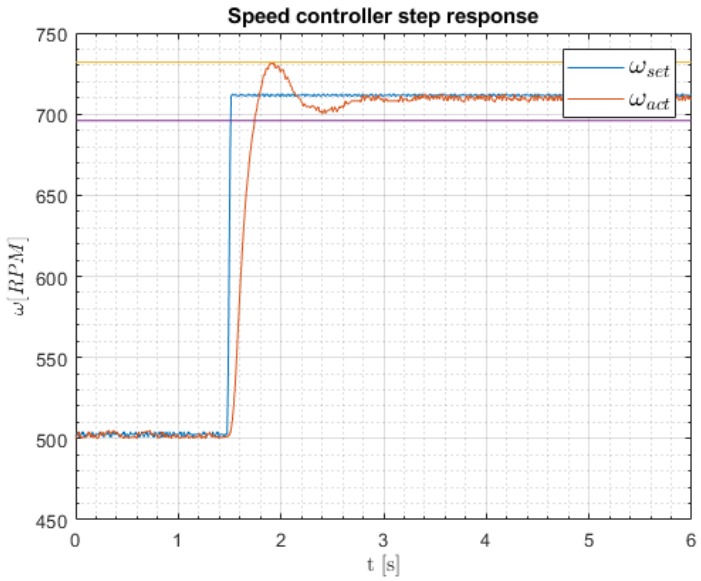


Figure 38. Speed controller step response.

The controller exceeded the control criteria of $\pm 5\%$ error band by two times, the maximum absolute error in the test was 2.5%. The settling time to this error window was measured to be $270ms$. From the response shape, it can be seen that control is tuned to be fast, with minor underdamping. Measured speed error reaches the upper error bound. At the same time control is stable having only one oscillation around the setpoint and then settling with no error, indicating a good control performance.

6.5 Speed controller disturbance rejection

Another quality of the system, that was evaluated, was the ability of the system to reject external disturbances. In the case of a rotating motor this would be the load change. In the test a step-wise load change was introduced and the speed response and the phase current of the motor was recorded. When the disturbance was introduced the actual motor speed dropped slightly but the regulator managed to restore nominal operation. Upon the removal of the disturbance actual motor speed raised above the setpoint because the integral part of the regulator had been increased to compensate for the load and the load removal meant that the output current produced The results are presented in figure 39.

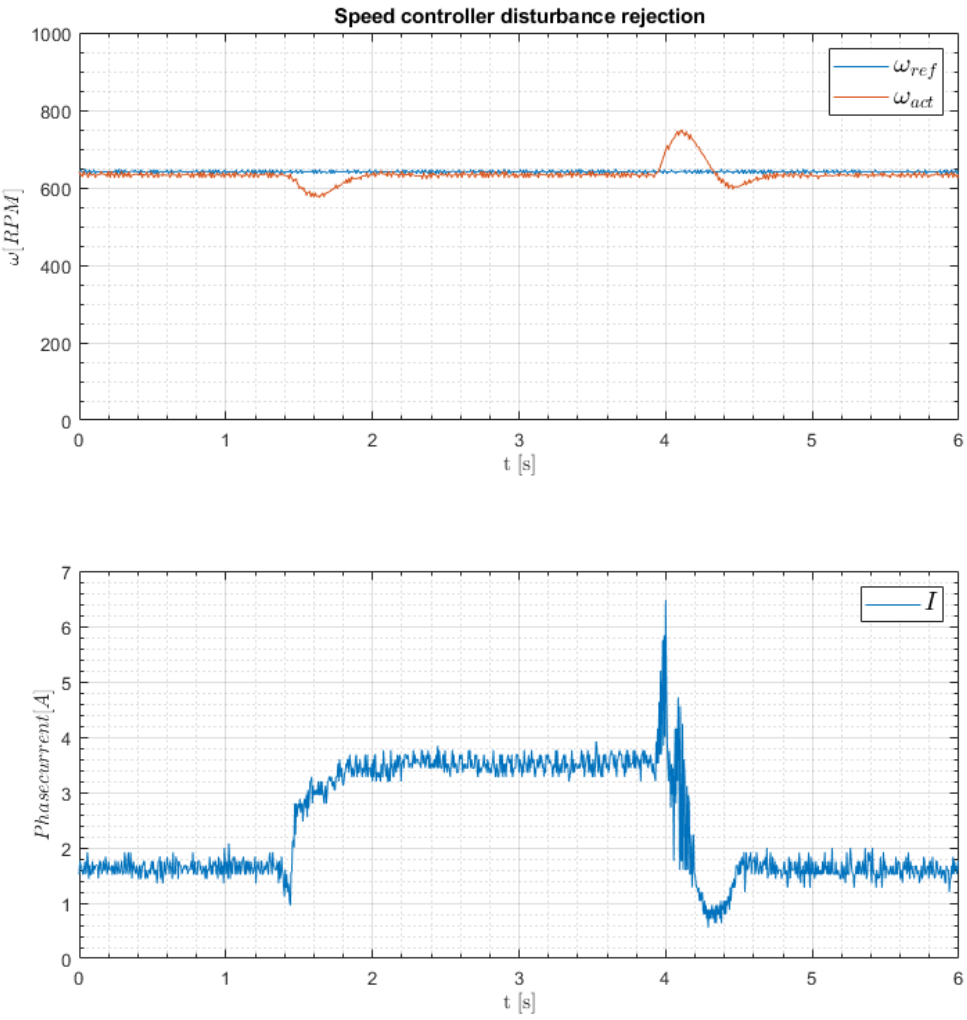


Figure 39. Speed controller disturbance rejection.

7 Summary

As a result of this work a brushless DC motor controller hardware with an observer based sensorless control algorithm was developed. The algorithm was selected from research, investigated and was successfully implemented on a real system, showing that algorithms developed for permanent magnet synchronous machines can be applicable to small hobby-grade brushless DC motor. In addition methods were developed to measure the electrical parameters of a three phase motors. Phase resistance and inductance measurement was implemented using the actual controller and flux linkage measurement using an external prime mover.

In the beginning of the work problem and end-goal statement was made along with the methods and tools that were going to be used to solve the stated problems. Then a theoretical overview of three phase motors, their main control principles was made. Then observer based control methodology was described and state-of-the-art of observer based sensorless control of three phase synchronous motors was presented. The required special hardware for controlling the motor was developed and the main design principles and component choices were presented. During the course of the work this implementation of hardware proved to be reliable and no issues were encountered during implementation of algorithms. Software side of the implementation of the parameter measurements were done successfully.In addition to implementation of observer based rotor angle estimation, additional control aspects were solved due to the fact that observer based solution was not usable for control at very low speeds.

The final solution that was implemented provides a reference on how sensorless observer based algorithms could be practically implemented and used with BLDC motors used on multirotor aerial vehicles. The speed control performance achieved was excellent and can lead the way on using those motors for applications where small-sized, high performance sensorless speed control is required.

References

[1] *Research bulletin: MCUs Sales to Reach Record-High Annual Revenues Through 2022*. [Online]. Available: http://www.icinsights.com/data/articles/documents/1101.pdf (visited on 04/29/2018).

[2] J. F. Gieras, *Permanent magnet motor technology: design and applications*. CRC press, 2010.

[3] D. C. Hanselman, *Brushless permanent magnet motor design*. The Writers' Collective, 2003.

[4] K. H. Nam, *AC motor control and electrical vehicle applications*. CRC press, 2017.

[5] F. Giri, *AC electric motors control: advanced design techniques and applications*. John Wiley & Sons, 2013.

[6] P. Vas, *Sensorless vector and direct torque control*. Oxford Univ. Press, 1998.

[7] K. Ogata and Y. Yang, *Modern control engineering*. Prentice-Hall, 2002, vol. 4.

[8] Q. Tang, A. Shen, X. Luo, and J. Xu, "PMSM sensorless control by injecting HF pulsating carrier signal into ABC frame", *IEEE Transactions on Power Electronics*, vol. 32, no. 5, pp. 3767–3776, 2017.

[9] X. Luo, Q. Tang, A. Shen, and Q. Zhang, "PMSM sensorless control by injecting hf pulsating carrier signal into estimated fixed-frequency rotating reference frame", *IEEE Transactions on Industrial Electronics*, vol. 63, no. 4, pp. 2294–2303, 2016.

[10] D. Bao, X. Pan, Y. Wang, X. Wang, and K. Li, "Adaptive synchronous-frequency tracking-mode observer for the sensorless control of a surface pmsm", *IEEE Transactions on Industry Applications*, vol. 54, no. 6, pp. 6460–6471, 2018.

[11] G. Zhang, G. Wang, D. Xu, and N. Zhao, "ADALINE-network-based pll for position sensorless interior permanent magnet synchronous motor drives", *IEEE Transactions on Power Electronics*, vol. 31, no. 2, pp. 1450–1460, 2016.

[12] H. Li, Z. Wang, C. Wen, and X. Wang, "Sensorless Control of Surface-mounted Permanent Magnet Synchronous Motor Drives Using Nonlinear Optimization", *IEEE Transactions on Power Electronics*, 2018.

[13] *STM32F446ZE High-performance foundation line, ARM Cortex-M4 core with DSP and FPU, 512 Kbytes Flash, 180 MHz CPU, ART Accelerator, Dual QSPI*. [Online]. Available: https://www.st.com/content/st_com/en/products/microcontrollers-microprocessors/stm32-32-bit-arm-cortex-mcus/stm32-high-performance-mcus/stm32f4-series/stm32f446/stm32f446ze.html (visited on 04/29/2018).

[14] *DRV8305 Three Phase Gate Driver With Current Shunt Amplifiers and Voltage Regulator datasheet (Rev. B)*. [Online]. Available: http://www.ti.com/lit/ds/symlink/drv8305.pdf (visited on 04/29/2018).

[15] *Nanotec motor image*. [Online]. Available: https://en.nanotec.com/fileadmin/_processed_/8/5/csm_DF45M_A2_2fc381809f.jpg (visited on 04/29/2018).

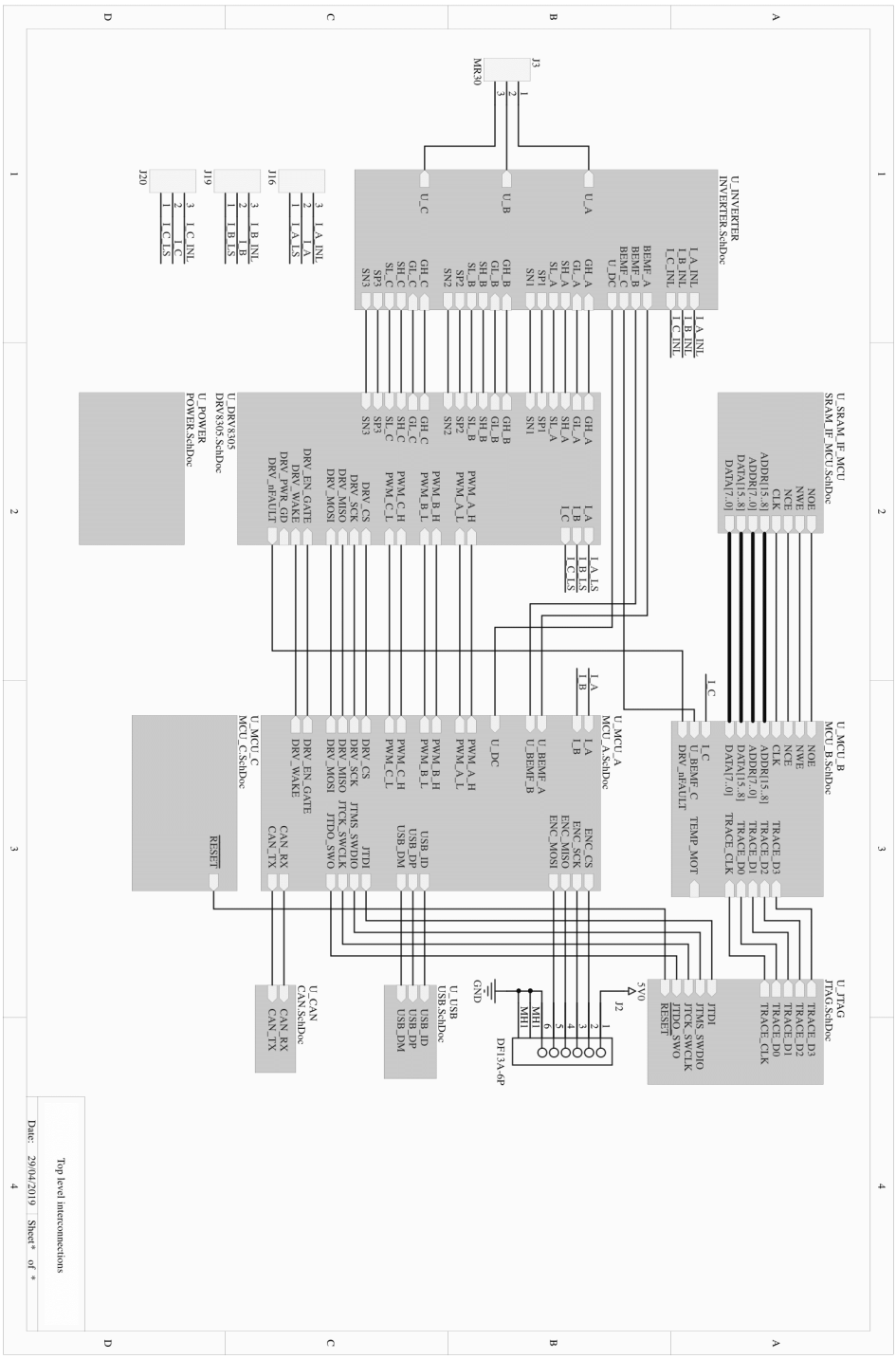
[16] *Nanotec motor datasheet*. [Online]. Available: https://en.nanotec.com/fileadmin/files/Datenblaetter/BLDC/DF45/DF45L024048-A2.pdf (visited on 04/29/2018).

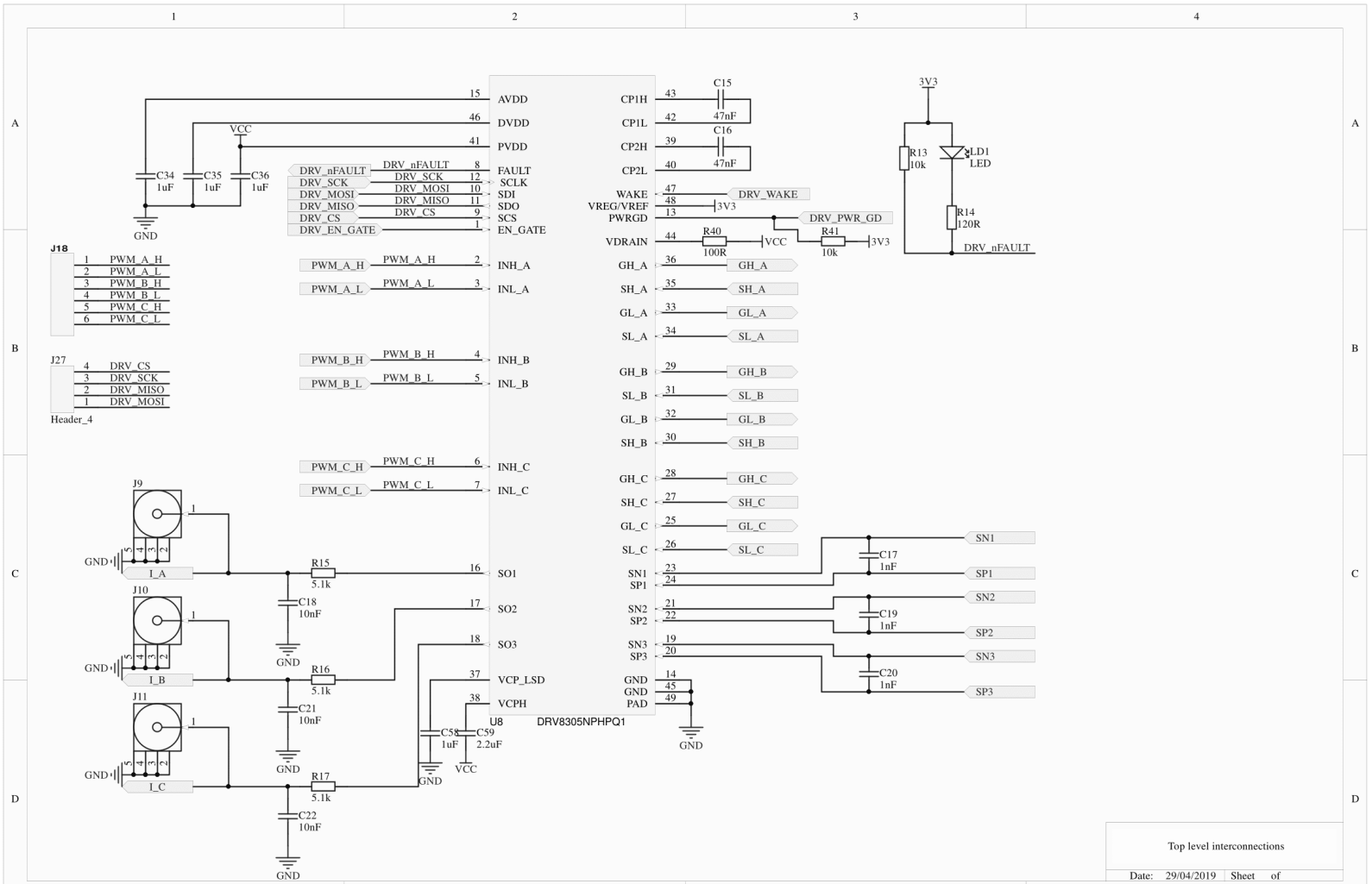
[17] *NTM PROPDRIVE v2 3536 910KV Brushless Outrunner Motor*. [Online]. Available: https://hobbyking.com/en_us/propdrive-v2-3536a-910kv-brushless-outrunner-motor.html (visited on 04/29/2018).

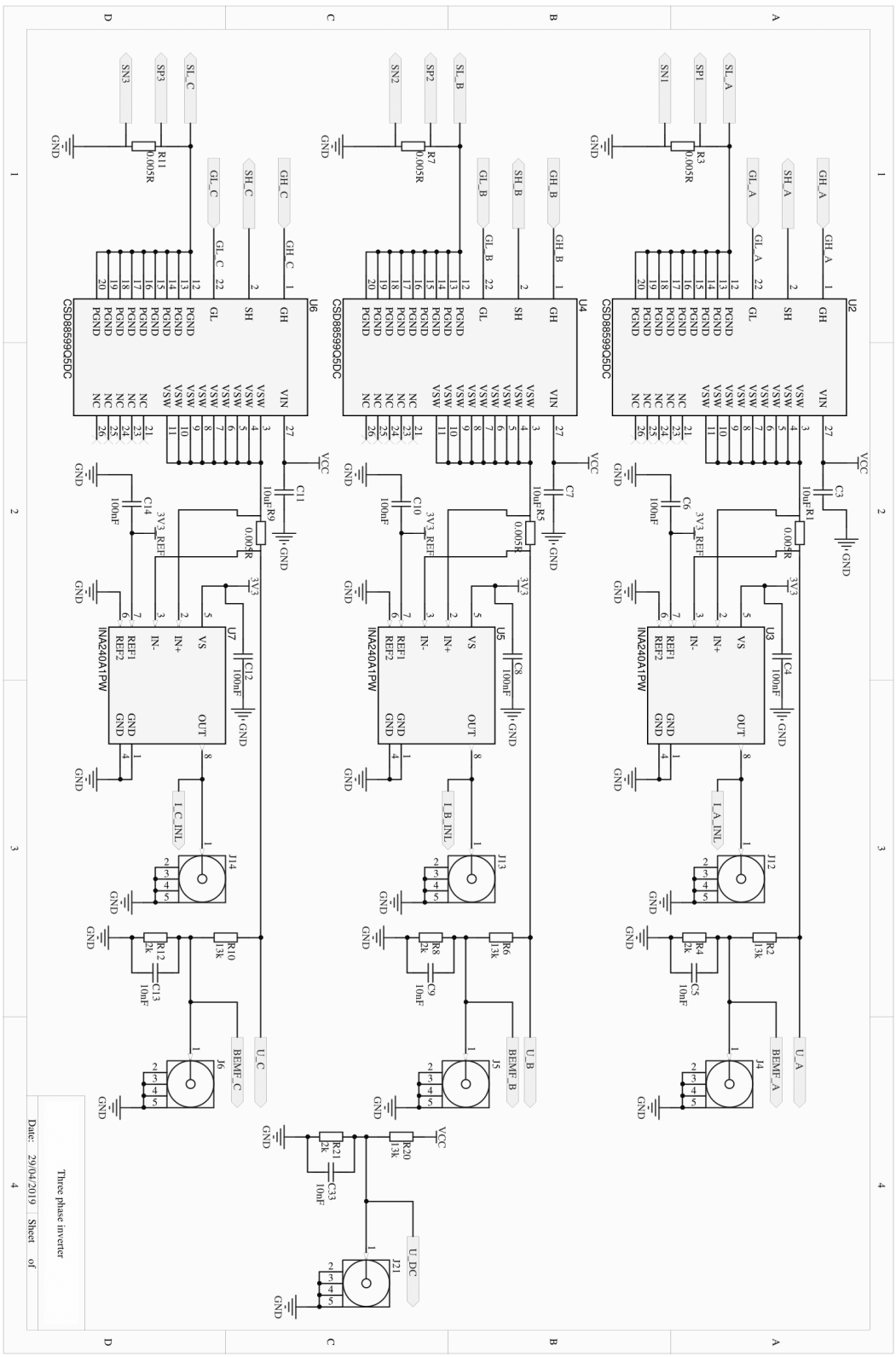
[18] R. Ortega, L. Praly, A. Astolfi, J. Lee, and K. Nam, "Estimation of rotor position and speed of permanent magnet synchronous motors with guaranteed stability", *IEEE Transactions on Control Systems Technology*, vol. 19, no. 3, pp. 601–614, 2011.

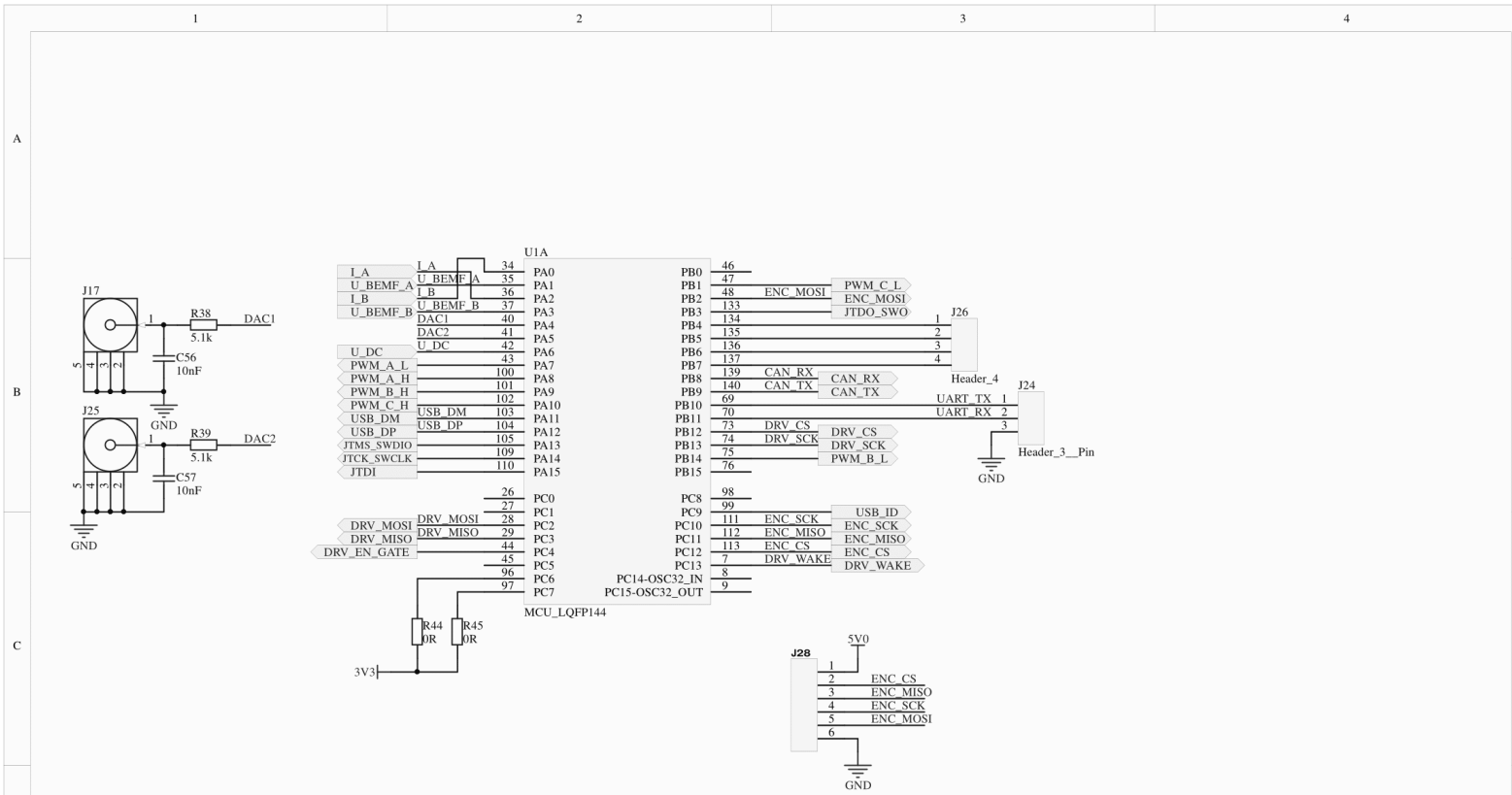
[19] J. Lee, J. Hong, K. Nam, R. Ortega, L. Praly, and A. Astolfi, "Sensorless control of surface-mount permanent-magnet synchronous motors based on a nonlinear observer", *IEEE Transactions on power electronics*, vol. 25, no. 2, pp. 290–297, 2010.

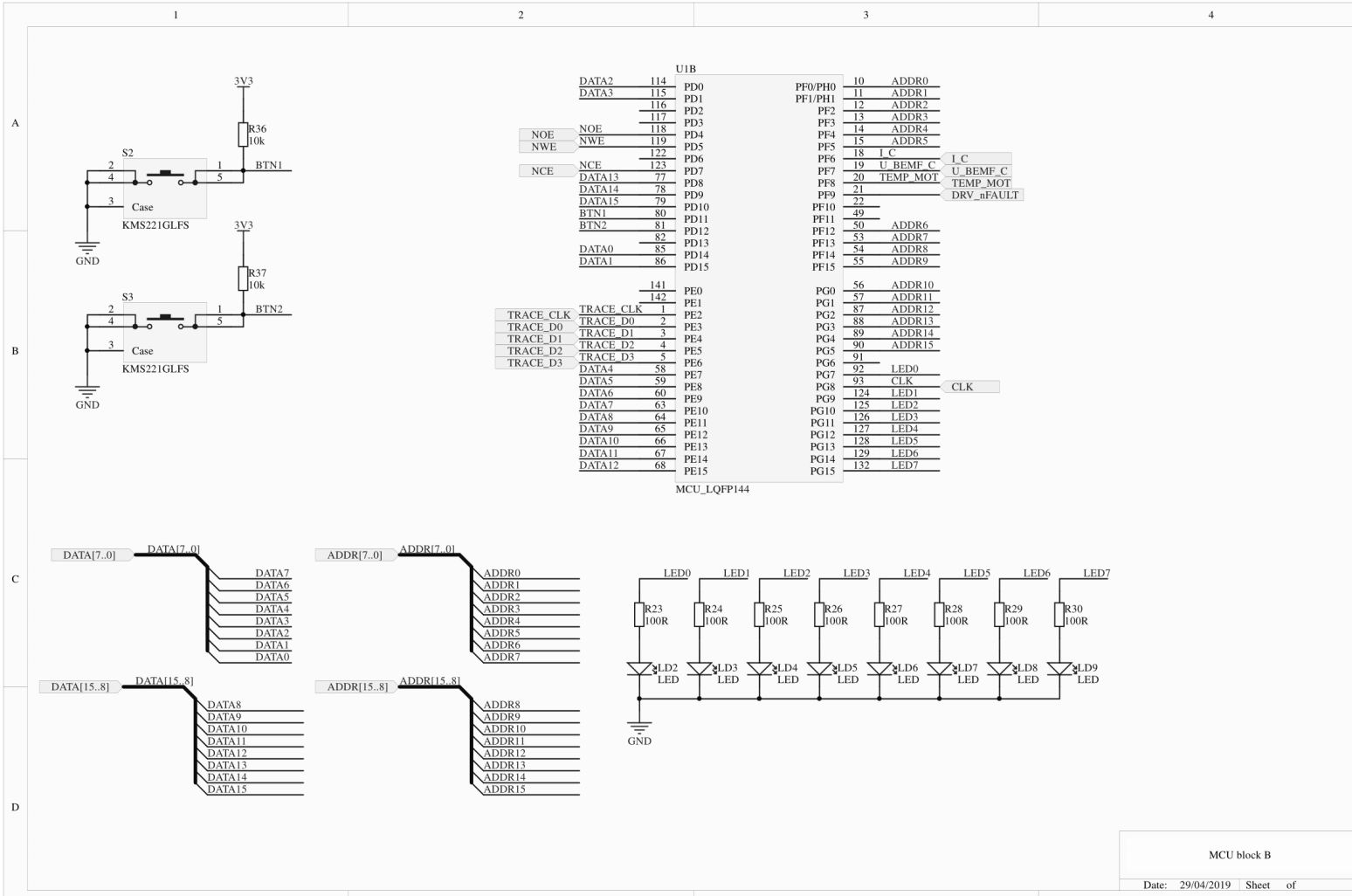
Appendix 1 – Schematic prints of developed controller

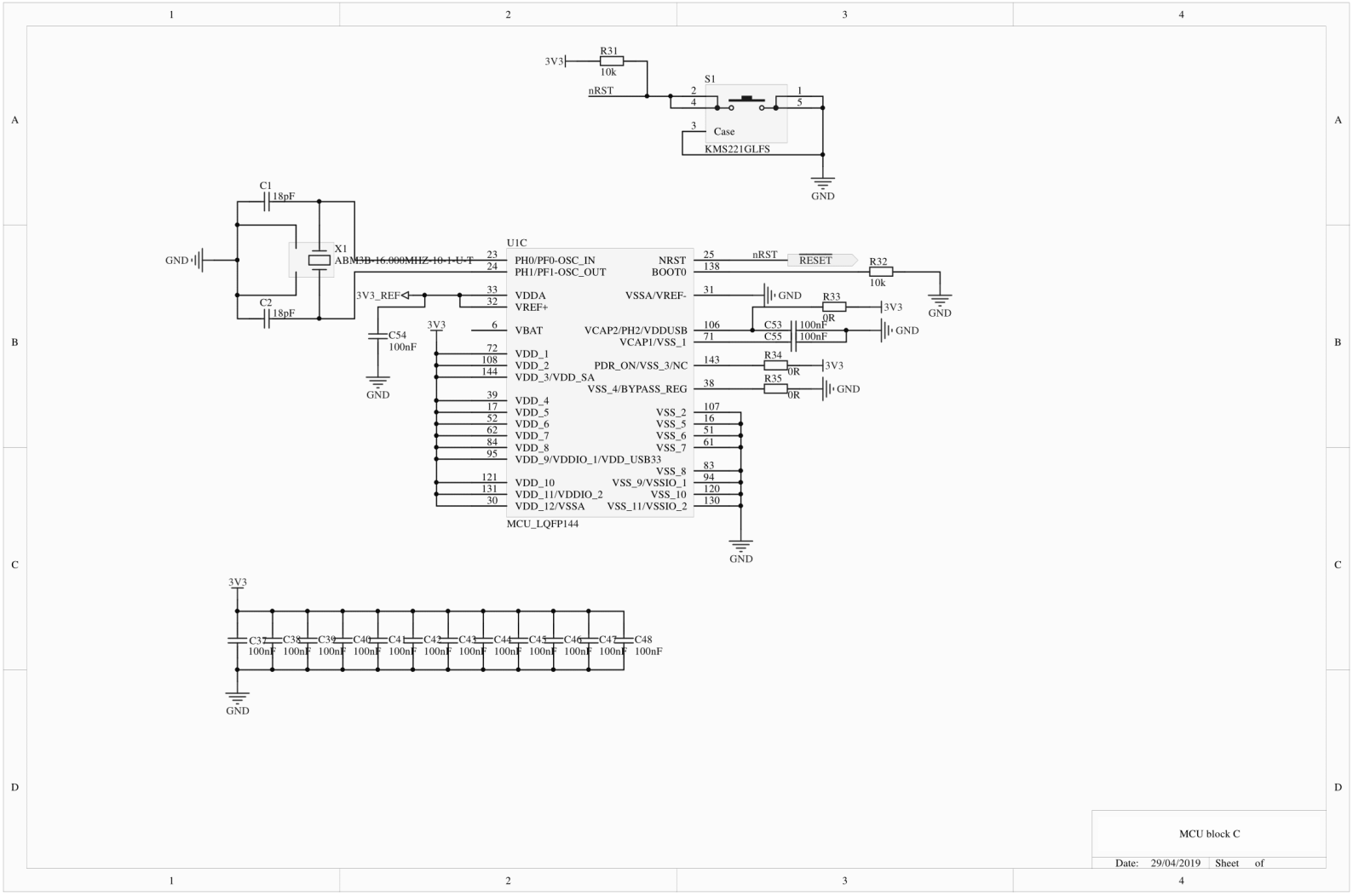



U8
DRV8305NPHPQ1
AVDD
DVDD
PVDD
FAULT
SCLK
SDI
SDO
SCS
EN_GATE
INH_A
INL_A
INH_B
INL_B
INH_C
INL_C
SO1
SO2
SO3
VCP_LSD
VCPH
CP1H
CP1L
CP2H
CP2L
WAKE
VREG/VREF
PWRGD
VDRAIN
GH_A
SH_A
GL_A
SL_A
GH_B
SL_B
GL_B
SH_B
GH_C
SH_C
GL_C
SL_C
SN1
SP1
SN2
SP2
SN3
SP3
GND
PAD
J18
J27
Header_4
J9
J10
J11
I_A
I_B
I_C
C15
C16
C17
C18
C19
C20
C21
C22
C34
C35
C36
C58
C59
R13
R14
R15
R16
R17
R40
R41
LD1
LED
3V3
VCC
DRV_nFAULT
DRV_SCK
DRV_MOSI
DRV_MISO
DRV_CS
DRV_EN_GATE
DRV_WAKE
DRV_PWR_GD
PWM_A_H
PWM_A_L
PWM_B_H
PWM_B_L
PWM_C_H
PWM_C_L
Top level interconnections
Date: 29/04/2019 Sheet of


Three phase inverter
Date: 29/04/2019 Sheet of
4

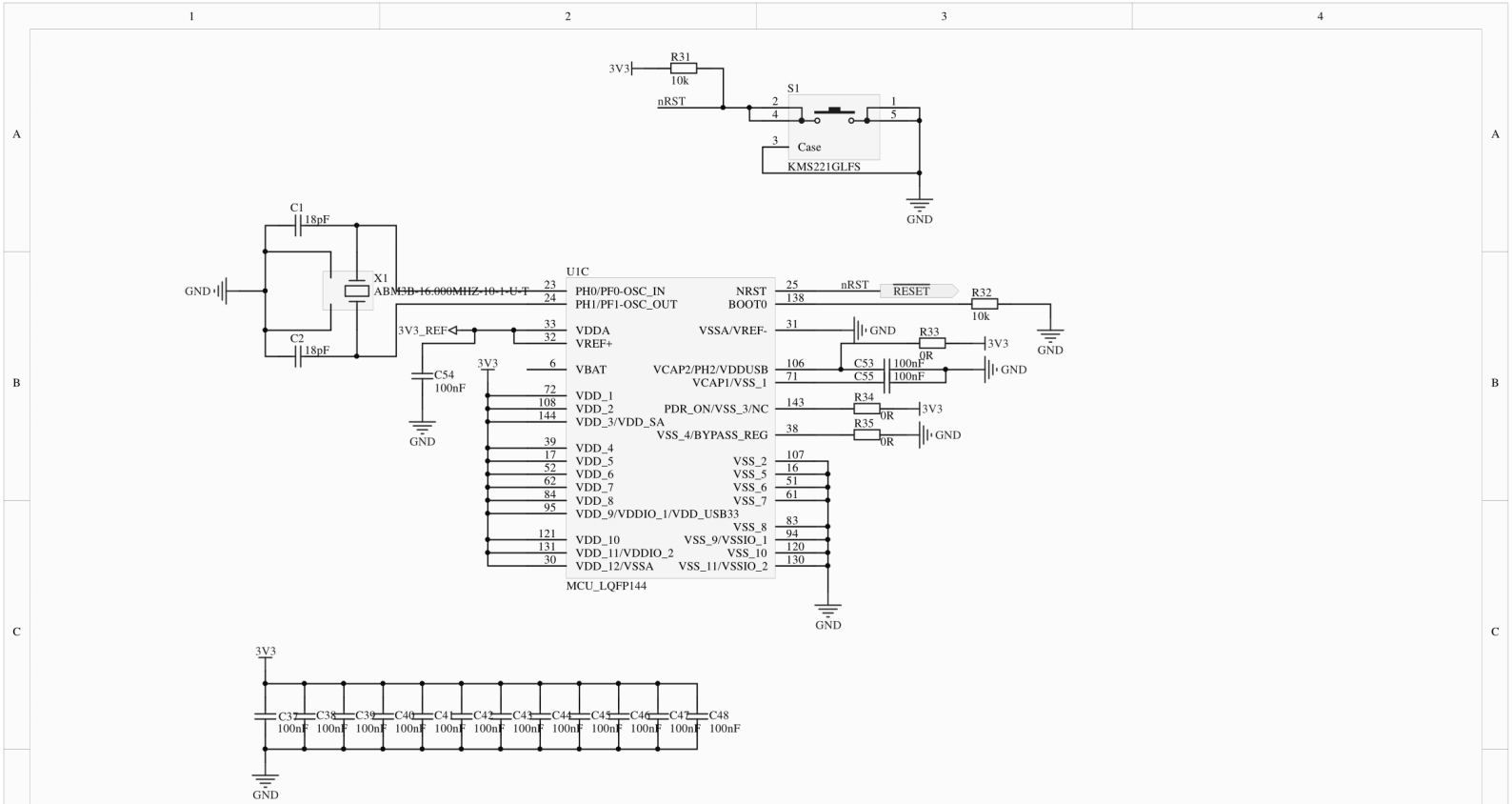

J17
R38
5.1k
DAC1
C56
10nF
GND
J25
R39
5.1k
DAC2
C57
10nF
GND
U1A
I_A
U_BEMF_A
I_B
U_BEMF_B
DAC1
DAC2
U_DC
PWM_A_L
PWM_A_H
PWM_B_H
PWM_C_H
USB_DM
USB_DP
JTMS_SWDIO
JTCK_SWCLK
JTDI
DRV_MOSI
DRV_MISO
DRV_EN_GATE
PA0
PA1
PA2
PA3
PA4
PA5
PA6
PA7
PA8
PA9
PA10
PA11
PA12
PA13
PA14
PA15
PC0
PC1
PC2
PC3
PC4
PC5
PC6
PC7
PB0
PB1
PB2
PB3
PB4
PB5
PB6
PB7
PB8
PB9
PB10
PB11
PB12
PB13
PB14
PB15
PC8
PC9
PC10
PC11
PC12
PC13
PC14-OSC32_IN
PC15-OSC32_OUT
MCU_LQFP144
PWM_C_L
ENC_MOSI
JTDO_SWO
CAN_RX
CAN_TX
DRV_CS
DRV_SCK
PWM_B_L
USB_ID
ENC_SCK
ENC_MISO
ENC_CS
DRV_WAKE
J26
Header_4
J24
UART_TX
UART_RX
Header_3__Pin
GND
R44
0R
R45
0R
3V3
5V0
J28
ENC_CS
ENC_MISO
ENC_SCK
ENC_MOSI
GND
MCU block A
Date: 29/04/2019 Sheet of


3V3
R36
10k
S2
BTN1
Case
KMS221GLFS
GND
3V3
R37
10k
S3
BTN2
Case
KMS221GLFS
GND
U1B
MCU_LQFP144
DATA[7..0]
DATA[15..8]
ADDR[7..0]
ADDR[15..8]
LED0
LED1
LED2
LED3
LED4
LED5
LED6
LED7
R23 100R
R24 100R
R25 100R
R26 100R
R27 100R
R28 100R
R29 100R
R30 100R
LD2 LED
LD3 LED
LD4 LED
LD5 LED
LD6 LED
LD7 LED
LD8 LED
LD9 LED
GND
MCU block B
Date: 29/04/2019 Sheet of



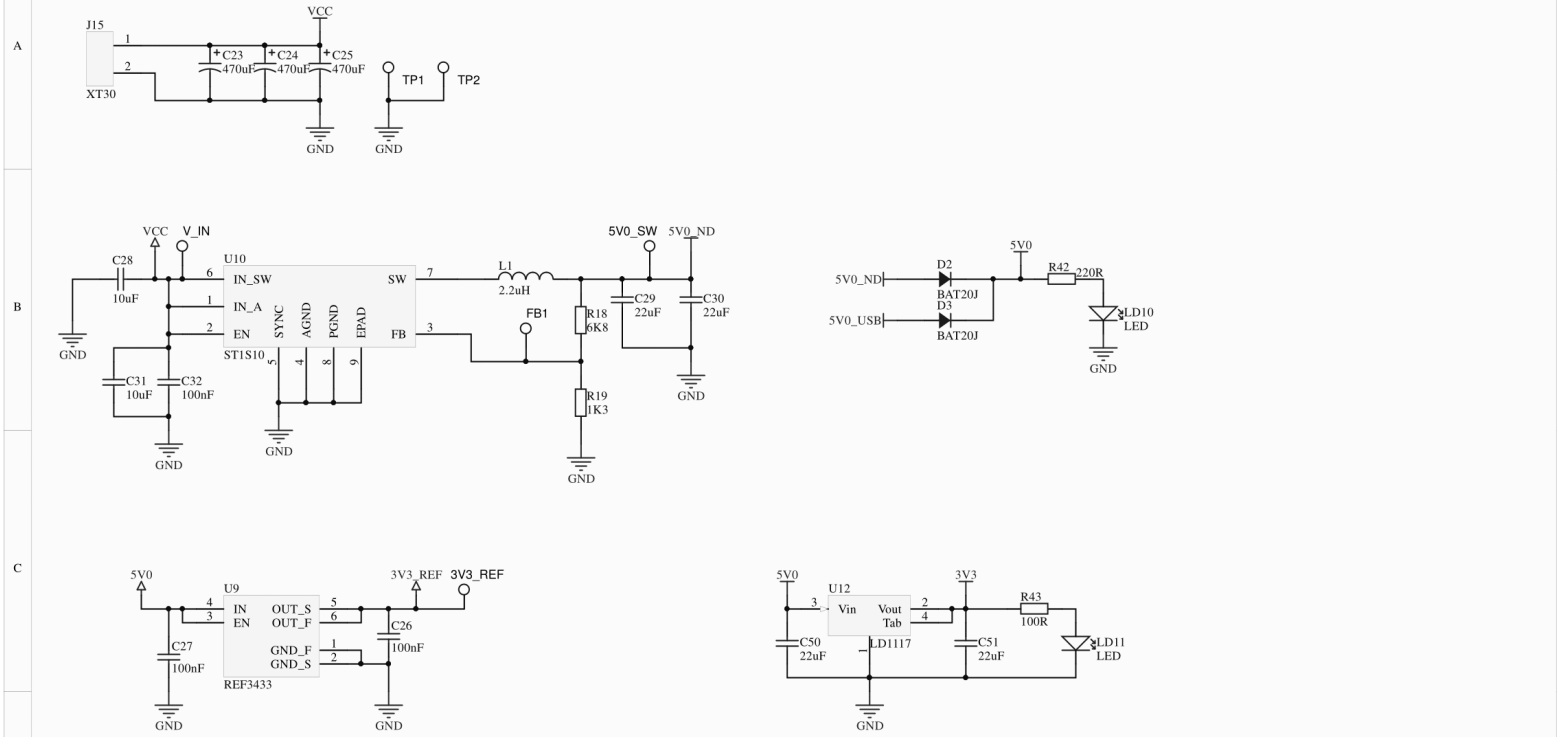
MCU block C

Date: 29/04/2019 Sheet of



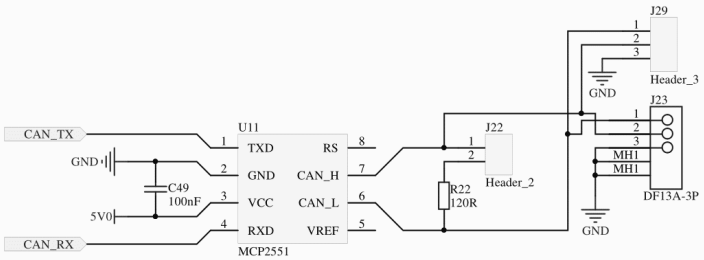
MCU block C

Date: 29/04/2019 Sheet of



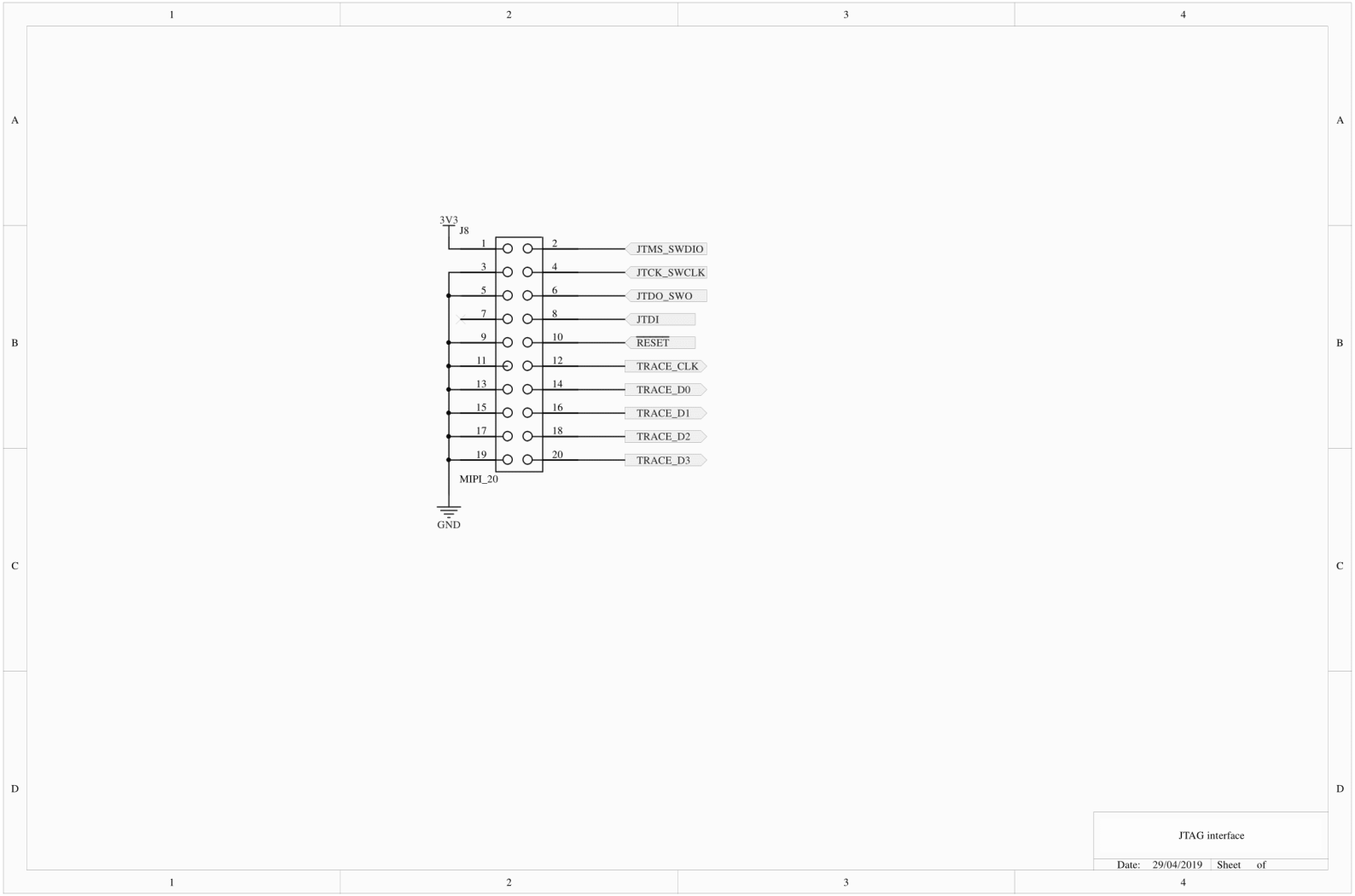
Power regulation

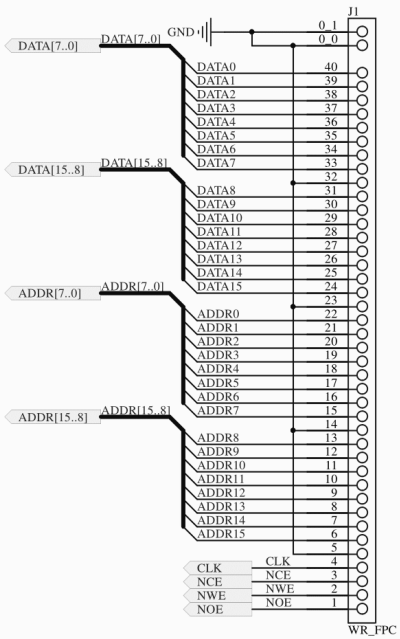
Date: 29/04/2019 Sheet of



CAN bus interface

Date: 29/04/2019 Sheet of


1
2
3
4
A
B
C
D
3V3
J8
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
JTMS_SWDIO
JTCK_SWCLK
JTDO_SWO
JTDI
RESET
TRACE_CLK
TRACE_D0
TRACE_D1
TRACE_D2
TRACE_D3
MIPI_20
GND
JTAG interface
Date: 29/04/2019 Sheet of



SRAM interface

Date: 29/04/2019 Sheet of

Appendix 2 – Resistance measurement algorithm

Following function demonstrates the resistance measurement of a motor phase.

```
float vMeasureResistance(E_STATE* eStateNext, E_RES_STATE eResEstState,
    Vec_abc_ft* vec_duty_abc, clMeas* iclmeas)
{
    static uint32_t u32StateTimer;
    static Vec_abc_ft statorRes;
    float R;
    // Switch through all the motor phases
    switch (eResEstState)
    {
    case ePhaseA:
        {
            // Set duty cycles
            vec_duty_abc->a = 0.55f;
            vec_duty_abc->b = 0.5f;
            vec_duty_abc->c = 0.5f;
            // If required time has elapsed
            if (u32StateTimer >= 40000)
            {
                // Record phase resistance value
                statorRes.a = resresAverage.Get();
                // Reset running average and state timer
                resAverage.Reset();
                u32StateTimer = 0;
                // Move to next phase
                eResEstState = ePhaseB;
            }
            else
            {
                // Let the current settle for initial 100 cycles
                if (u32StateTimer < 100)
                {
                    resAverage.Reset();
                }
                else
                {
                    // Otherwise update the resistance value according
   to Ohm's law
                    resAverage.Update(iclmeas->vecVoltageMeas.a /
   iclmeas->vecCurrentMeas.a);
                }
                u32StateTimer++;
            }
            break;
        }
    case ePhaseB:
        {
            // Set duty cycles
            vec_duty_abc->a = 0.5f;
            vec_duty_abc->b = 0.55f;
            vec_duty_abc->c = 0.5f;
            // If required time has elapsed
            if (u32StateTimer >= 40000)
            {
                // Record phase resistance value
                statorRes.b = resAverage.Get();
                // Reset running average and state timer
                resAverage.Reset();
                u32StateTimer = 0;
                // Move to next phase
                eResEstState = ePhaseC;
            }
```

```
            else
            {
                // Let the current settle for initial 100 cycles
                if (u32StateTimer < 100)
                {
                    resAverage.Reset();
                }
                else
                {
                    // Otherwise update the resistance value according
    to Ohm's law
                    resAverage.Update((iclmeas->vecVoltageMeas.b -
    iclmeas->vecVoltageMeas.a) / iclmeas->vecCurrentMeas.b);
                }
                u32StateTimer++;
            }
            break;
        }
    case ePhaseC:
        {
            // Set duty cycles
            vec_duty_abc->a = 0.5f;
            vec_duty_abc->b = 0.5f;
            vec_duty_abc->c = 0.55f;
            // If required time has elapsed
            if (u32StateTimer >= 40000)
            {
                // Record phase resistance value
                statorRes.c = resAverage.Get();
                // Reset running average and state timer
                resAverage.Reset();
                u32StateTimer = 0;
                eResEstState = ePhaseA;
                // Move outer state machine to idle state
                *eStateNext = eIdle;
                // Remove current from motor phases
                vec_duty_abc->a = 0.5f;
                vec_duty_abc->b = 0.5f;
                vec_duty_abc->c = 0.5f;
                // Calculate the average of the phase resistances
                // also compensate for shunt resistor value
                statorRes -= 0.010f;
                R = (statorRes.a + statorRes.b + statorRes.c) / 3.0f;
                R /= 1.5f;
            }
            else
            {
                // Let the current settle for initial 100 cycles
                if (u32StateTimer < 100)
                {
                    resAverage.Reset();
                }
                else
                {
                    // Otherwise update the resistance value according
    to Ohm's law
                    resAverage.Update((iclmeas->vecVoltageMeas.c) /
    iclmeas->vecCurrentMeas.c);
                }
                u32StateTimer++;
            }
            break;
        }
    }
    return R;
}
```

Appendix 3 – Inductance measurement algorithm

Following function demonstrates the inductance measurement of a motor phase.

```
float vMeasureInductance(float dt, float theta_step, float fAmpInd, \
Vec_abc_ft* vec_duty_abc, clMeas* iclmeas, uint32_t u32StateTimer)
{
    static float theta;
    static float i_prev;
    static float u_prev;
    static float curMin = 1e9;
    static float curMax = -1e9;
    static float bRisingU = false;
    static float bRisingI = false;
    float i,u;
    float mid;
    static float t0,t1,t2,t3;
    float L;
    float freq;
    float phase;
    // Sinusoidal wave generation
    if ((theta + theta_step) < (2*PI))
    {
        theta += theta_step;
    }
    else
    {
        theta = theta + theta_step - 2*PI;
    }
    float fSin = sin(theta);
    fSin *= fAmpInd;
    // Sinewave to one phase with proper offset.
    vec_duty_abc.a = 0.5f  + fAmpInd + fSin;
    vec_duty_abc.b = 0.5f;
    vec_duty_abc.c = 0.5f;
    // Record voltage and current values for the phase
    float current = iclmeas->vecCurrentMeas.a;
    float voltage = iclmeas->vecVoltageMeas.a;
    // Calculate the mean value of current for the sine wave cycle
    i = current;
    u = voltage - 12000;
    if (i < curMin)
    {
        curMin = i;
    }
    if (i > curMax)
    {
        curMax = i;
    }
    mid = (curMax -  curMin) / 2.0f;
    mid += curMin;
    i -= mid;
    // If previous edge was rising edge
    if (bRisingU == false)
    {
        // Wait until the voltage value crosses zero from positive side
        if (u > 0 && u_prev < 0)
        {
            // Mark that falling edge is detected
            bRisingU = true;
            // Record time of the falling edge
            t3 = u32StateTimer;
        }
    }
```

```
    else
    {
        // Wait until the voltage value crosses zero from negative side
        if (u < 0 && u_prev > 0)
        {
            // Mark that rising edge is detected
            bRisingU = false;
            // Record time of the rising edge
            t2 = u32StateTimer;
        }
    }
    // If previous edge was rising edge
    if (bRisingI == false)
    {
        // Wait until the current value crosses zero from positive side
        if (i > 0 && i_prev < 0)
        {
            // Mark that falling edge is detected
            bRisingI = true;
            // Record time of the falling edge
            t1 = u32StateTimer;
            // Perform phase calculation
            phase = ((t3 - t1)*dt) / (1 / freqAvg.Get());
            // Update the running average value
            phaseAvg.Update(phase);
            // Calculate the phase inductance based on phase offset
            L = fabs((R * 1.5)*tan(phaseAvg.Get()* 2.0f*PI) / (2*PI*
freqAvg.Get()));
            L /= 1.5f;
        }
    }
    else //If previous edge was falling edge
    {
        // Wait until the current value crosses zero from negative side
        if (i < 0 && i_prev > 0)
        {
            // Mark that rising edge is detected
            bRisingI = false;
            // Record the time instance of the falling edge
            t0 = u32StateTimer;
            // Calculate signal frequency and update the running
average
            freq = (1.0f/((t0 - t1) * dt))/2.0f;
            freqAvg.Update(freq);
        }
    }
    // Store values for the next cycle
    i_prev = i;
    u_prev = u;
    return L;
}
```

Appendix 4 – Source code

Complete program source code implemented on the microcontroller is attached as a CD to this work.