TALLINN UNIVERSITY OF TECHNOLOGY
Faculty of Information Technology

IA40LT

Tarmo Prillop 103762IASB

FIELD ORIENTED CONTROLLER FOR BRUSHLESS DC MOTORS

Bachelor's thesis

Supervisor: Eduard Petlenkov
PhD

Tallinn 2017

TALLINNA TEHNIKAÜLIKOOL
Infotehnoloogia teaduskond

IA40LT

Tarmo Prillop 103762IASB

VÄLJAORIENTEERITUD JUHTIMISEGA HARJAVABA ALALISVOOLUMOOTORI KONTROLLER

Bakalaureusetöö

Juhendaja: Eduard Petlenkov
PhD

Tallinn 2017

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

[22.05.2017]

Abstract

The aim of this thesis is to describe the principles behind field orientated control method for brushless direct current motors using an absolute position sensor and to implement a working field orientated algorithm base motor controller using a microcontroller and complementing hardware.

The work consists of text, pictures and tables describing the theoretical background, implementation and system verification.

This thesis is written in English and is 43 pages long, including 5 chapters, 24 figures and 4 tables.

Abstract

Väljaorienteeritud juhtimisega harjavaba alaslisvoolumootori kontroller

Käesoleva töö eesmärgiks on kirjeldada väljarienteeritud juhtimise põhimõtteid ning rakendada neid harjavabal alalisvoolu mootoril kasutades absoluutset asendiandurit. Töö tulemuseks on mikrokontrolleril implementeeritud väljaorienteeritud juhtimismetoodikal põhinev tarkvara harjavaba alalisvoolumootori juhtimiseks.

Töö koosneb tektist, piltidest ja tabelitest mis kirjeldavad töö teoreetilisi aluseid, implementeerimist ning töö tulemuste analüüsi.

Lõputöö on kirjutatud inglise keeles ning sisaldab teksti 43 leheküljel, 5 peatükki, 24 joonist, 4 tabelit.

List of abbreviations and terms

AC	Alternating current
ADC	Analog to digital converter
BLDC	Brushless direct current
DAC	Digital to analog converter
DC	Direct current
EMF	Electromotive force
FOC	Field oriented control
GPIO	General purpose input output
IC	Integrated circuit
LSB	Least significant bit
MCU	Microcontroller
PID	Proportional, integral, derivative
PMSM	Permanent magnet synchronous motor
PWM	Pulse width modulation
QEI	Quadrature encoder interface
RPM	Revolutions per minute
SPI	Serial peripheral interface
SVM	Space vector modulation
TMU	Trigonometric mathematics unit

Table of Contents

List of Figures

List of Tables

1 Introduction

Due to a number of factors price of commercially available brushless DC motors has dropped and the availability has increased. On the other hand electronics and software used to control those motors has remained relatively primitive. The motors are ran in open-loop scalar control mode without any feedback. This is understandable from the point of view where those types of motors are used: driving propellers of multirotored copters and model aeroplanes. These sorts of application do not require precise speed or position control in order for them to function.

These brushless DC motors for model aircraft's are designed so that their power output per mass would be as high as possible. The reason behind this is to increase the flight time. An aspect that hasn't been considered very much is that those kinds of motors could be used in a variety of other fields like robotics and automation by using more sophisticated control methods. The benefits for using brushless DC motors in those applications are substantial: size of existing systems could be reduced, maintenance and system reliability would increase as bearings are the only part of the motor that is subject to wear.

The main motivation of this thesis is to leverage the usage of low cost but high power to weight ratio motors by controlling them using field oriented control to provide precise speed control.

The aim of this thesis is to create embedded solution on a microcontroller using readily available development kits to implement field oriented control for brushless DC motors with absolute encoder as feedback, in order to extend the potential usage of these kinds of motors in other applications than model aircraft.

2 Theoretical background

In this chapter an overview of brushless DC motors and field oriented control is given. In addition a practical method for implementing field oriented control is described.

2.1 Brushless DC motor

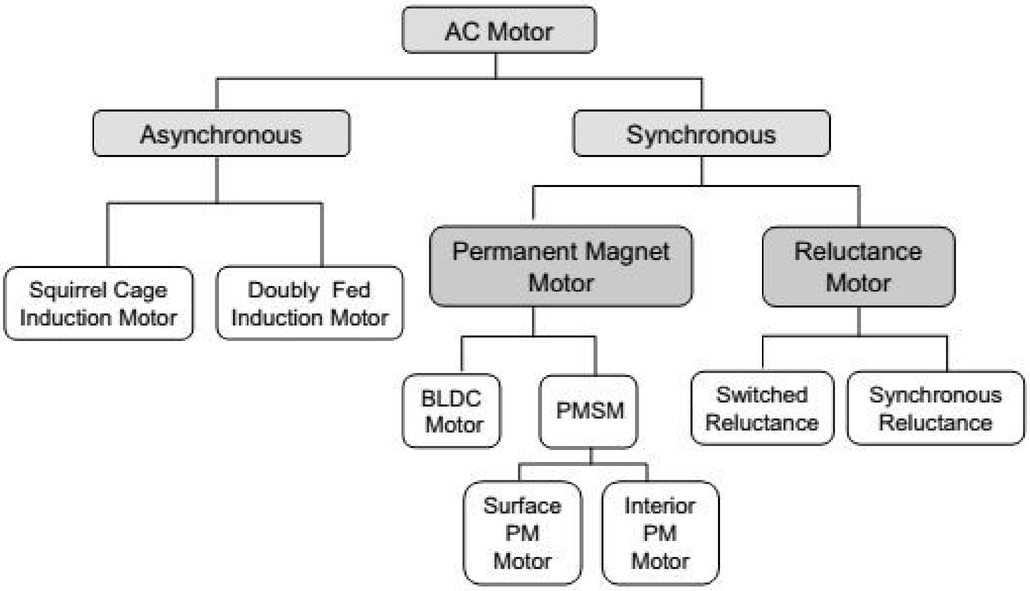


Illustration 1: Taxonomy of AC motors [1]

Brushless DC motors are categorized as synchronous AC motors. BLDC motor are permanent magnet motors, which are typically classified as having trapezoidal back electromotive force (EMF). All brushless permanent magnet motors are constructed with electrical windings on the stator and permanent magnets on the rotor. This construction is why those type of motors are popular, the windings remain stationary and there are no commutators that could wear out [2].

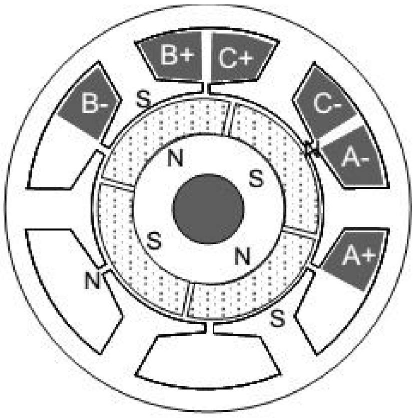


Figure 1: Cross section of BLDC motor [2]

Commonly BLDC motors have three phases and any even number of rotor magnet poles. The number of pole pairs determines the number of electrical revolutions within one mechanical revolution. This notion becomes important when measuring rotor flux angle using an absolute encoder on the motor shaft. Typically motors have 2 or 3 poles but more poles are preferable as more magnet poles creates greater torque output for the same current [2].

Torque in a BLDC motor, as in every PMSM, is generated by the interaction of two magnetic fields: magnetic field created by the current in stator windings and magnetic field created by the rotors permanent magnets.

2.2 Three phase inverter

Three phase inverter consists of six switching elements arranged into pairs, know as inverter legs. All of the inverter legs being equal, means that output voltage of the inverter leg depends on the DC bus voltage and the switch status [3].

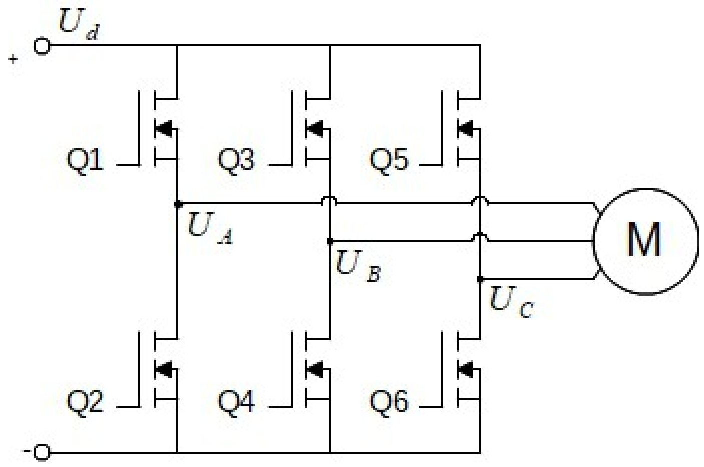


Figure 2: Three phase inverter schematic with connected motor

Theoretically the inverter can have $2^6=64$ different states as any of the six switching elements can either be opened or closed. Most of them are unpractical, for example combinations where both switches on one inverter leg are closed. This would create a short-circuit across DC bus and consequently thermal damage to the switching elements. A better way to define the inverter states is with respect to the state of the top switches and condition that whenever the top switch within an inverter leg is opened then the lower switch within the inverter leg must be closed. Using that methodology the inverter output voltages for each motor phase can be calculated and are given in the table below.

Table 1: 3-phase inverter switching table [3]

U	1 0 0	1 1 0	0 1 0	0 1 1	0 0 1	1 0 1
	Q1 Q3 Q5	Q1 Q3 Q5	Q1 Q3 Q5	Q1 Q3 Q5	Q1 Q3 Q5	Q1 Q3 Q5
U_A	$\frac{2U_d}{3}$	$\frac{U_d}{3}$	$\frac{-U_d}{3}$	$\frac{-2U_d}{3}$	$\frac{-U_d}{3}$	$\frac{U_d}{3}$
U_B	$\frac{-U_d}{3}$	$\frac{U_d}{3}$	$\frac{2U_d}{3}$	$\frac{U_d}{3}$	$\frac{-U_d}{3}$	$\frac{-2U_d}{3}$
U_C	$\frac{-U_d}{3}$	$\frac{-2U_d}{3}$	$\frac{-U_d}{3}$	$\frac{U_d}{3}$	$\frac{2U_d}{3}$	$\frac{U_d}{3}$

2.3 Field oriented control

Chronologically field oriented control (FOC) was the first vector control method developed for controlling induction motors. The principle of this method was proposed in the early 1970s by F. Blaschke of Siemens, who used physical analysis to show that two components of the stator current space vector projected along two rectangular axes, to be defined later, called the direct and quadrature axes, play the same roles as field and armature windings in a DC motor. The direct axis was found to be oriented along the axis of rotor magnetic field, which is why this approach has been called field orientation [4].

Field oriented control can be divided into two: direct field oriented control and indirect field oriented control. Direct field orientated control is based on flux vector identification which is determined either by direct measurement or estimation from other parameters. By direct measurement it is meant that there are Hall effect sensors in the air gap between rotor and stator. Indirect field orientated control means that the rotor flux vector is determined indirectly by measuring the mechanical angle of the rotor by a shaft position sensor [5]. This thesis focuses on the indirect variant of the control method.

Field oriented control for BLDC motors has six basic steps as described in [6].

1. Two phase currents are measured and the third can be calculated using Kirchoffs current law.

2. Measured phase currents are converted into a two axis system by using Clarkes transformation.

3. The two axis coordinate system which was generated in the previous step is transformed to align with the rotor magnetic flux by using Park's transformation.

4. Calculated d and q axis currents are used as feedback for PI controllers, where the direct axis current reference value is zero for permanent magnet motor and direct axis current reference is generated from the torque command generated by the speed control loop.

5. The direct and quadrature axis voltages from PI controllers are translated back to the stationary two axis reference frame using inverse Park's transformation.

6. The alpha-beta reference frame values are converted back to the 3-phase quantities and are inputs to the pulse width modulation.

2.4 Space vector modulation

Space vector modulation is a digital technique, which purpose is to generate an arbitrary voltage vector by using the eight possible states of the 3-phase inverter. This is one of possible methods to take results from FOC and apply them to the BLDC motor via 3-phase inverter. The other alternative being sinusoidal modulation either with or without third harmonic injection.

If with every binary word from Table 14 a space vector is associated we can create a space vector diagram, also known as Concordia diagram. This diagram shows six active voltage space vectors U_1 to U_6 , which correspond to the switching states . In addition there are two zero voltage space vectors U_0 and U_7 , which correspond to binary words 000 and 111 respectively. This diagram is shown in Figure 3 . In the diagram voltage vectors $\vec{U}_{100}$, $\vec{U}_{010}$ and $\vec{U}_{001}$ are aligned with the motor phases A, B and C [2].

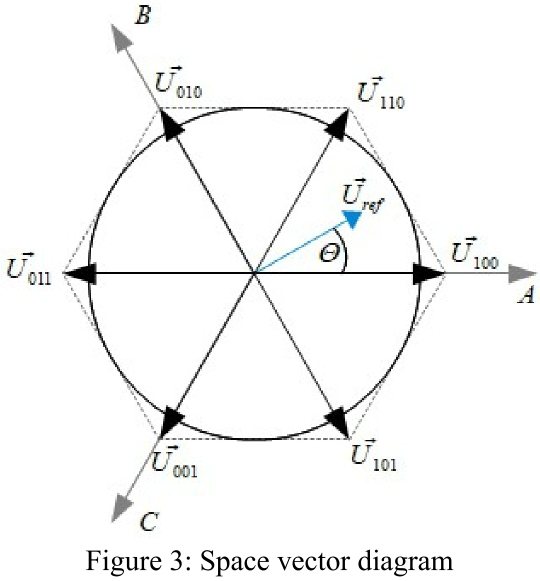


Figure 3: Space vector diagram

The space vectors form six distinctive sectors and $\vec{U}_{ref}$ can either be aligned with one of the space vectors or between two of them. From the space vector diagram it is fairly easy to see how an arbitrary space vector can be represented using active and zero voltage vectors.

$$\vec{U}_{ref}=\vec{U}_x\cdot T_1+\vec{U}_y\cdot T_2+\vec{U}_0\cdot T_0 \tag{2.1}$$

In equation (2.1) vector $\vec{U}_x$ is the most clockwise vector and $\vec{U}_y$ is the most counter-clockwise vector for any given sector. Values T_1 , T_2 and T_0 can be determined by the following equations if vector $\vec{U}_{ref}$ is represented in polar notation:

$$T_1=T\cdot m\cdot\sin(60^{\circ}-\Theta) \tag{2.2}$$

$$T_2=T\cdot m\cdot\sin(\Theta) \tag{2.3}$$

$$T_0=T-T_1-T_2 \tag{2.4}$$

In equations (2.2), (2.3) and (2.4) symbol T represents the switching period, m is the magnitude of vector $\vec{U}_{ref}$ and α is the angle between vectors $\vec{U}_{100}$ and $\vec{U}_{ref}$. These equations are necessary to calculate switch on/off times during modulation cycle. The procedure of using these values will be explained in paragraph describing the implementation, as the equations (2.3), (2.4) and (2.2) are tightly coupled with the hardware level implementation.

3 Overview of used components

The hardware platform 4 main components: microcontroller development board, three phase inverter and its control circuitry, absolute magnetic encoder and the motor to be controlled. Brief description about those used components are given in the subsections below.

3.1 TMS320F28377S microcontroller board

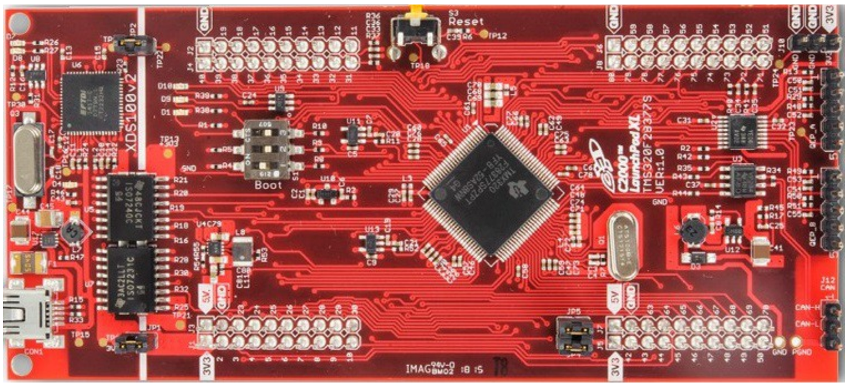


Figure 4: LAUNCHXL F28377S development board [7]

For the microcontroller an evaluation board containing the TMS320F28377 by Texas Instruments was used. The board contains the aforementioned microcontroller (MCU) and a USB to JTAG interface for program downloading and application debugging. The TMS320F28377S is a high performance DSP with hardware floating point support having maximum clock rate of 200MHz. The MCU is well suited for field oriented control applications due to the on-chip peripherals and the additional mathematical capabilities like trigonometric mathematics unit (TMU) which means that all trigonometric calculations are hardware accelerated [8].

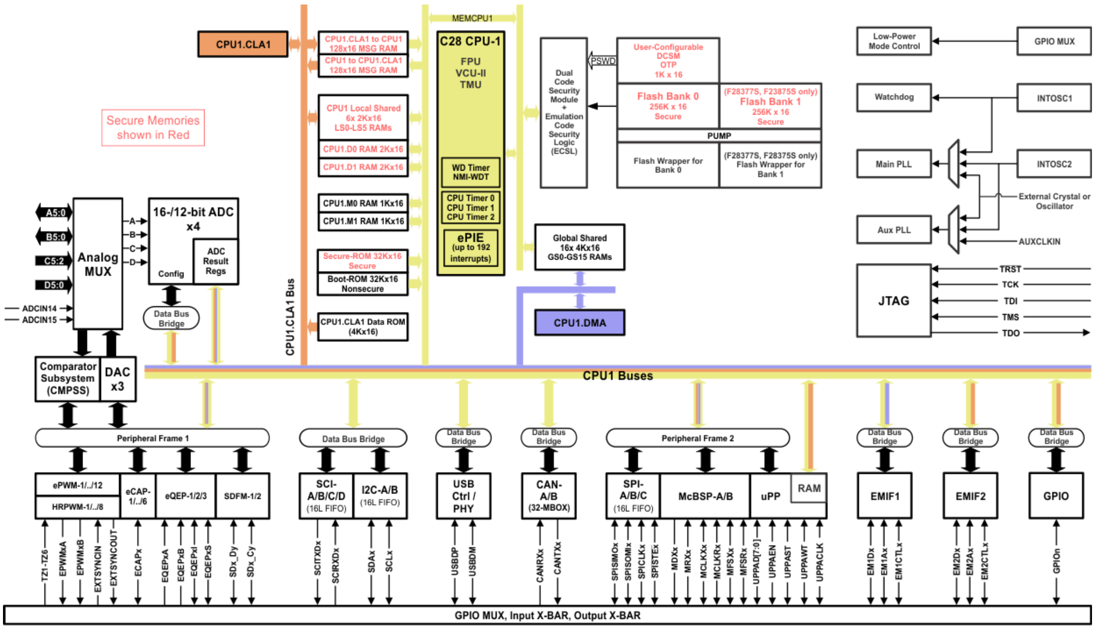


Figure 5: TMS320F28377S MCU block diagram [9]

3.2 DRV8305 based three phase inverter

BOOSTXL-DRV8305 is a three phase inverter with gate control and diagnostics integrated circuit DRV8305 from Texas Instruments. This board is designed as an add-on board for the MCU development board used. The DRV8305 IC provides a number of functionality, for example it prevents situation where two transistors on the inverter leg are opened at the same time protecting them from damage. In addition it provides current sense amplifiers required for measuring phase currents with shunt resistors [10].

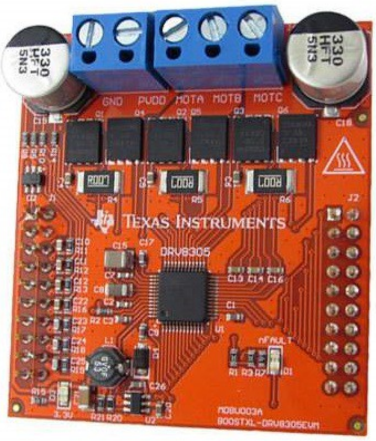


Figure 6: BOOSTXL-DRV8305 three phase inverter [11]

3.3 AS5047 encoder

AS5047D is a 14-bit absolute magnetic encoder produced by Austria Microsystems. The angular information measurement is performed using a cylindrical magnet magnetized across its diameter. Within the chip there is an array of Hall effect sensors to detect how the magnetic field changes. This sensor is well suited for motor control applications where high speed and resolution is required. For example the AS5047D encoder is capable of measuring angular velocity up to 14500 RPM [12]. Also the fact that the measurement is insensitive to external magnetic fields makes it suitable for motor control usage as the magnetic fields by stator and rotor do not affect the measurements.

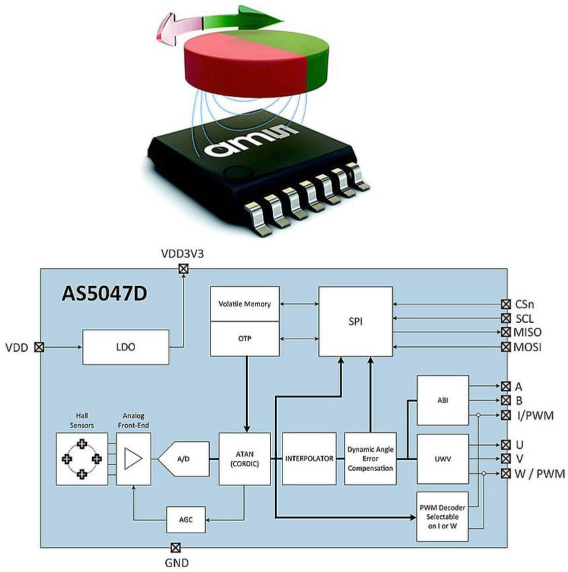


Figure 7: AS5047D encoder block diagram [13]

3.4 BLDC motor

For the motor a 24 volt nominal voltage, star connected, 16-pole BLDC motor from German manufacturer Nanotec was chosen. It is a small 65 watt motor that is suitable for using with the BOOSTXL-DRV8305 3-phase power stage. The motor is also equipped with Hall effect sensors but those were not used in the implementation.

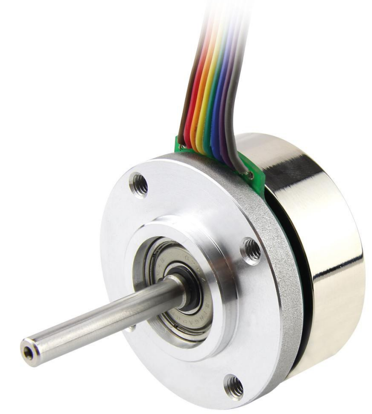


Figure 8: BLDC motor used [14]

Main specifications of the motor are given in Table 21 below.

Table 2: Nanotec DF45L024048-A2 motor specification [15]

Poles / Phases	16 / 3
Voltage rated	24
No load current	< 0.5
Current rated/peak	3.26/9.5
Resistance (phase to phase) (Ohms) @ 20C	0.64 +- 10%
Inductance (phase to phase) (mH) @1kHz	0.27 +- 20%
Torque Rated/Peak (Nm)	0.13/0.39
Torque constant (Nm/A)	0,04
Power rated (W)	65
Speed rated/no load (RPM)	4840/6100
Rotor inertia (Kg * m2)	$1.81\cdot10^{-5}$
Weight (Kg)	0,15

4 Implementation

In this paragraph the design and implementation of field oriented control is presented.

4.1 System architecture and requirements

Illustrative overview, in block diagram form, of the closed-loop speed controller with FOC is shown on the figure below.

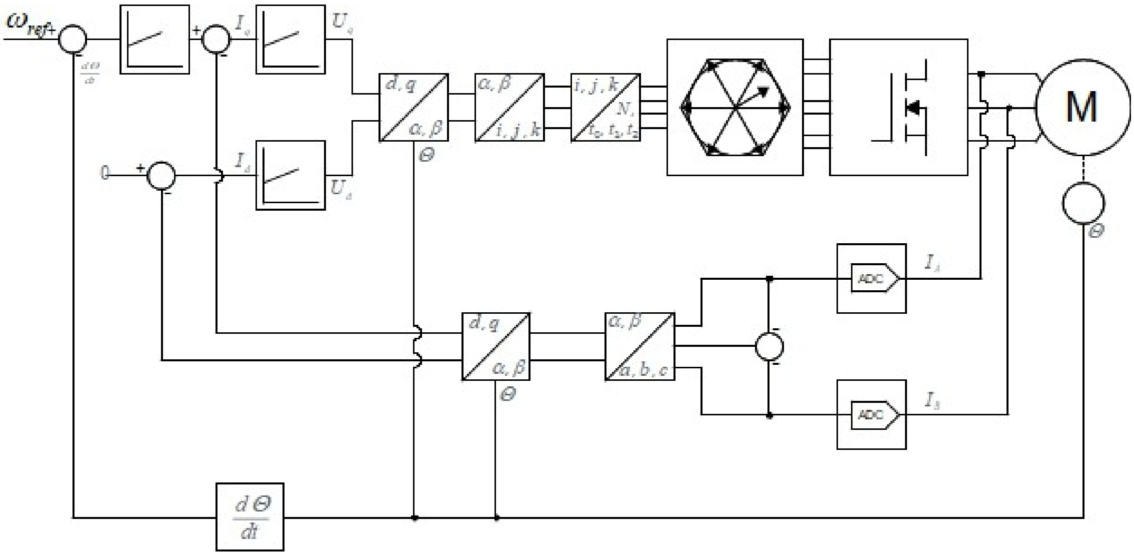


Figure 9: Overall system block diagram

The system main components of the implemented system are PID regulator for rotor speed controlling, 2 PID controllers for direct and quadrature axis current control. coordinate system transformations.

The system consists of a fast and slow control loop. Fast control loop is used to do the phase current measurements, angle measurement and FOC calculations. This control loop runs at modulation frequency. Slow control loop is used for the speed controller and this runs with 1kHz frequency.

In addition to those basic blocks a method for determining the relationship between the rotor mechanical angle and rotor flux angle is presented.

Before implementing the solution, basic requirements for the system have to be established.

- Modulation frequency shall be 20kHz to be out of human hearing range.
- Motor phase currents and rotor angle shall be measured at every modulation cycle to achieve cycle-by-cycle current control.
- Execution of FOC calculations takes less than half of the pulse width modulation period. So the maximum load on the CPU for FOC can be 50%.
- Speed control shall have maximum steady state error of ± 5% .
- Minimum absolute speed above which motor shall be controlled is 100RPM.

The following subsections will now explain the operation and implementation of the major components and procedures necessary for the system functioning.

4.2 Current measurements

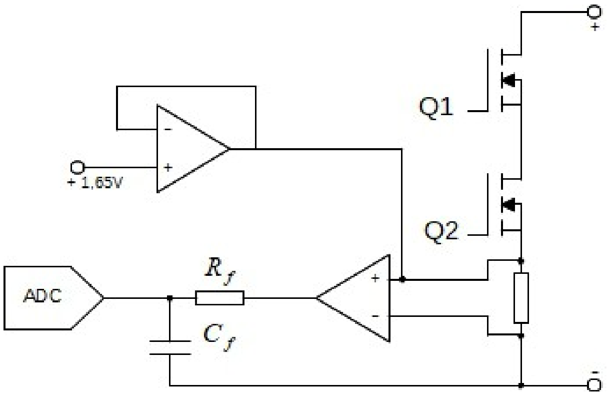


Figure 10: Current measurement schematic

Current measurement is one of the most important measurement in the system followed by the rotor angle measurement. The measurement of phase current is implemented by measuring voltage drop across shunt resistors. Due to the fact that the shunt resistors are connected between the drain and ground potential of the low-side transistor on the inverter leg, means that phase current measurement is not always possible. It can be seen on the schematic presented below, that current only flows through the shunt resistor when transistor Q2 is opened and some other inverter leg's upper transistor is opened so that current could flow through stator windings (**see** Figure 8, **page** 21).

This means that phase current measurement must be synchronised with the control of inverter transistors. Namely the current value is measurable only when the low side transistor is opened. Also the reverse applies, ADC measurements affect the control of transistors. The transistors must be controlled so, that current measurement is always possible, which means that every modulation cycle there must be time during which.

4.2.1 ADC measurement time

ADC measurement can be divided into two separate parts: sampling and conversion. Sampling time is dependent on the ADC internal sampling mechanism and ADC input circuitry. Sample and hold mechanism in TMS320F28377S MCU ADC is implemented using a switched capacitor architecture. This means that the sampling time is dependent on the RC time constant of the sample and hold circuitry, required accuracy and circuitry driving the ADC input [8]. Time-wise the conversion process has fixed time specified in microcontroller clock cycles and can be found in the data sheet.

The sample and hold operation in the other hand has configurable time. This time period must be calculated based on the equation found on the data sheet.

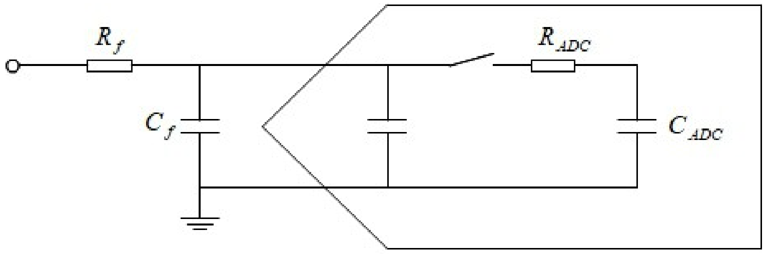


Figure 11: ADC and its input schematic overview

Above figure depicts the ADC internals and low-pass filter connected to the input. The equation to calculate the sample and hold diagram from [8] is as follows:

$$\tau_{S/H} = -\ln\left(\frac{settling\ accuracy(LSBs)}{2^N}\right) R_{ADC} C_{S/H} \tag{4.1}$$

Values for R_{ADC} and C_{ADC} can be found from the microcontroller data sheet, and they are: $R_{ADC}=425\,\Omega$, $C_{ADC}=14{,}5\,pF$. ADC resolution for single ended measurements is 12 bits, so $N=12$. It must be noted that this equation doesn't take into account that we have a first order low pass filter connected to the ADC input.

This means that the equation (4.1) must be adjusted according to the input filter values. The values for the resistor and capacitor in the filter are following: $R_f=56\,\Omega$, $C_f=2{,}2\,nF$ [10].

We can see that $C_f \gg C_{ADC}$, which means that we can omit it from the calculations, because time constant of the system is dominated by the ADC internal sampling circuit [16] . Based on that we can calculate the minimum required time for sampling:

$$\tau_{S/H}=-\ln\left(\frac{0.25}{2^{12}}\right)\cdot(425\,\Omega+56\,\Omega)\cdot 14{,}5\,pF=67{,}68\,nS \qquad (4.2)$$

Based on the calculated sample and hold time $\tau_{S/H}$, we can calculate the number of MCU clock cycles ADC needs :

$$N_{cycles}=\left\lceil\frac{\tau_{S/H}}{\tau_{CLK}}\right\rceil=\left\lceil\frac{67{,}68\,nS}{5\,nS}\right\rceil=14 \qquad (4.3)$$

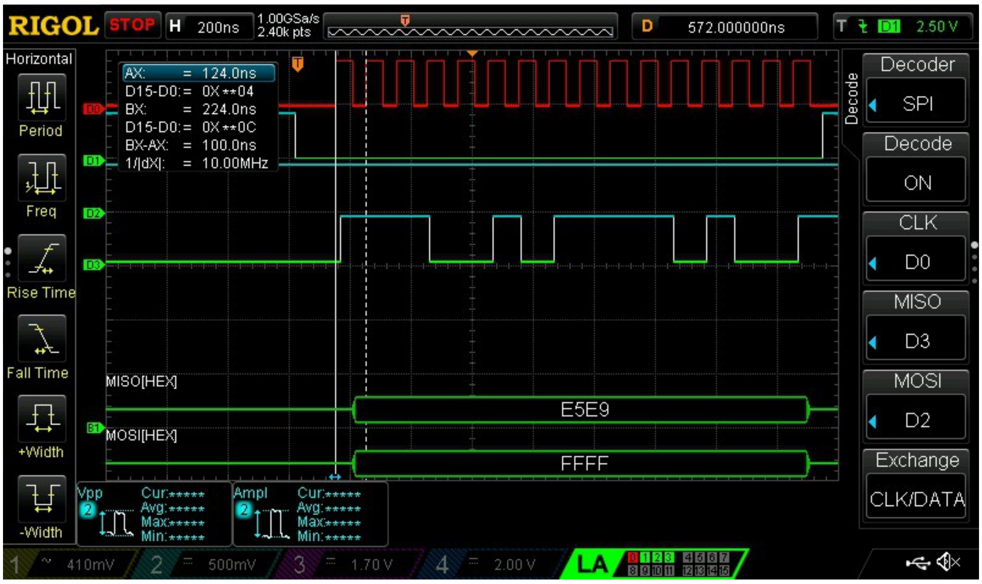
Conversion time for the sample is given in the MCU data sheet and for 12-bit mode this is 11 clock cycles [17]. In this system the ADC is clocked at its maximum frequency which is 50MHz [17]. The current value measurement thus takes time

$$\tau_{meas}=\tau_{S/H}+\left(\frac{11}{50\cdot 10^{6}}\right)=67{,}68\,nS+220\,nS=287{,}68\,nS \qquad (4.4)$$

4.3 Rotor angle and velocity measurements

Rotor angle measurement is performed using an AS5047 absolute magnetic encoder with 14 bit resolution (**see** subsection 3.3, **page** 20). The AS5047D magnetic encoder has three interfaces over which the angle could be read: traditional quadrature encoder interface, pulse width modulation interface and digital Serial Peripheral Interface communication. Out of these three the last, SPI, is chosen, because the quadrature encoder interface creates necessity to rotate the rotor until the index pulse, indicating one full revolution, is found. Also with quadrature interface the resolution is reduced to

12 bits. With PWM interface the absolute angle information is always available but like with QEI the resolution is reduced. The only option which allows full 14 bit resolution is SPI communication [12].



```
AX:      =  124.0ns
D15-D0:= 0X**04
BX:      =  224.0ns
D15-D0:= 0X**0C
BX-AX:   =  100.0ns
1/|dX|:  =  10.00MHz
```

Figure 12: Scope capture of AS5047D SPI communication

Rotor angle measurement must be synchronised with phase current measurements so that coordinate system transformations would be valid. To achieve this, rotor angle measurement is triggered right after the ADC measurements. In order to minimize the time required to read the angle value from encoder maximum clock cycle of 10MHz was used in SPI communication. An example of SPI communication with 10MHz clock rate and packet decoding is presented on the figure below.

```
void AS5x47_ReadAngle(void)
{
	SpiaRegs.SPITXBUF = 0xFFFF;

	while(SpiaRegs.SPIFFRX.bit.RXFFST == 0) { }

	u16RawAngle = SpiaRegs.SPIRXBUF & 0x3FFF;

	fRotorMechAngle = ((float)u16RawAngle * 360.0f) /
	16384.0f;
}
```

Figure 13: Reading of angle from AS5047D encoder

4.3.1 Encoder angle offset measurement

In order to use rotor angle information for FOC a relationship must be established between rotor mechanical and rotor flux angle. Implementation of this calibration procedure can be straightforward when we assume that during calibration process motor is able to rotate freely. After establishing this assumption, we can use the following procedure to measure the angle offset:

1. Command a constant zero angle voltage vector with appropriate magnitude to the stator.

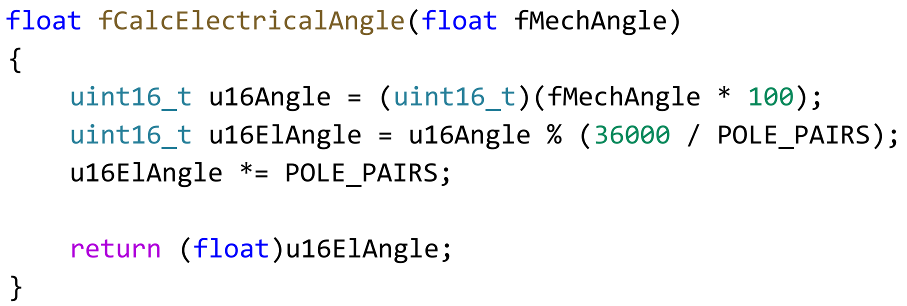
2. Measure the angle from encoder after system has stabilised after disturbance introduced by step 1.

The appropriate magnitude in step 1 is chosen so that the voltage vector induced current generates enough torque to rotate unloaded rotor. Determining of the magnitude was done experimentally. Angle measurement is performed after delay from commanding the voltage to the motor. This delay is based on two components. Firstly a constant time delay, after which the cycle to cycle angle delay is measured. Secondly if the cycle to cycle angle difference is low enough the value is sampled and the procedure is ended. The last operation is there to avoid situations when the static time delay is not enough for the rotor to become stationary.

The result of the process is angle Θ_o which is subtracted from every measurement from angle value Θ_{meas} read from encoder. The motor has N pole pairs so the rotor electrical angle (rotor flux angle) Θ_{re} can be calculated by the following equation:

$$\theta_{re} = N \cdot \left[(\Theta_{meas} - \Theta_o) - \frac{2\pi}{N} \left\lfloor \frac{N \cdot (\Theta_{meas} - \Theta_o)}{2\pi} \right\rfloor \right] \qquad (4.5)$$

Using this mathematical formulation we can create program code to calculate the rotor flux angle. The implementation as a C program function is given in Figure 14, the argument for the function is the corrected angle read from the encoder.



```
float fCalcElectricalAngle(float fMechAngle)
{
    uint16_t u16Angle = (uint16_t)(fMechAngle * 100);
    uint16_t u16ElAngle = u16Angle % (36000 / POLE_PAIRS);
    u16ElAngle *= POLE_PAIRS;

    return (float)u16ElAngle;
}
```

Figure 14: Function to calculate rotor flux angle from mechanical angle

4.3.2 Velocity calculations

Determination of rotor mechanical velocity is rather straightforward from mathematical point of view, velocity is nothing else than the first derivative of the position with respect to time. Implementation on the other hand needs considerations, as angle measurements are done in a discrete system and the measurement has finite resolution.

The resolution of position information is 14 bits [12], meaning that angle is measurable with 0.022 degree steps. As stated previously, the angle is measured every modulation interval which is 50 microsecond. If the implementation would be that we calculate the derivative after every measurement, we have a minimum measurable velocity increment of 440deg/s = 1.22 rev/s = 73.3 revolution per minute. This is very coarse value and is not suitable to achieve speed control performance required below 1466 RPM.

A better approach is to calculate the position derivative for velocity calculation at the rate at which the speed controller runs. In this work the speed control loop runs at 1kHz which means that the velocity measurement increment is 22 deg/s = 0.061 rev/s = 3.66 revolutions per minute. This allows to achieve the required control performance above 73.2 RPM

Another aspect that has to be handled when calculating velocity are cases when $270 \leq \Theta_r[t-1] < 0$ and $0 < \Theta_r[t] \leq 90$ or vice versa. In those cases simple subtraction to obtain angle difference leads to an incorrect result. The method to correctly calculate velocity from discrete angle measurements is presented on Figure 15.

```
float fCalcRotorMechRPM(float fMechAngle, float fSamplingFreq)
{
     static float fPrevAngle = 0.0f;
     static float fAngleDiff = 0.0f;
     float fAngularVelocity = 0.0f;

     fAngleDiff = fmaxf(fMechAngle,fPrevAngle) -
                  fminf(fMechAngle,fPrevAngle);
     if(fAngleDiff > 180) fAngleDiff = 360 - fAngleDiff;
     if(fMechAngle - fPrevAngle < 0) fAngleDiff *= -1;

     fAngularVelocity = (fAngleDiff / (1/fSamplingFreq)) * 0.16667f;
     fPrevAngle = fMechAngle;
     return fAngularVelocity;
}
```

Figure 15: Function implementation for velocity calculation

4.4 Coordinate transformations

4.4.1 Forward Clarke and Park transformations

The function of forward Clarke and Park transformations is to transform a stationary 3-phase time dependent quantities into rotating reference frame that is aligned with rotor magnetic flux.

Clarke transform translates the three phase system into two orthogonal components [18]. The transform is given with equation:

$$\begin{vmatrix} i_\alpha \\ i_\beta \end{vmatrix} = \frac{2}{3} \begin{vmatrix} 1 & \cos(\frac{2\pi}{3}) & \cos(\frac{4\pi}{3}) \\ 0 & \sin(\frac{2\pi}{3}) & \sin(\frac{4\pi}{3}) \end{vmatrix} \cdot \begin{vmatrix} i_A \\ i_B \\ i_C \end{vmatrix} \qquad (4.6)$$

Based on Kirchoff's current law, we know that the sum of the currents flowing into and out of the motor must be zero. With this knowledge we can simplify the equation (4.6) to the following form:

$$\begin{vmatrix} i_\alpha \\ i_\beta \end{vmatrix} = \frac{2}{3} \begin{vmatrix} 1 & \cos\left(\frac{2\pi}{3}\right) & \cos\left(\frac{4\pi}{3}\right) \\ 0 & \sin\left(\frac{2\pi}{3}\right) & \sin\left(\frac{4\pi}{3}\right) \end{vmatrix} \cdot \begin{vmatrix} i_A \\ i_B \\ i_C \end{vmatrix}$$

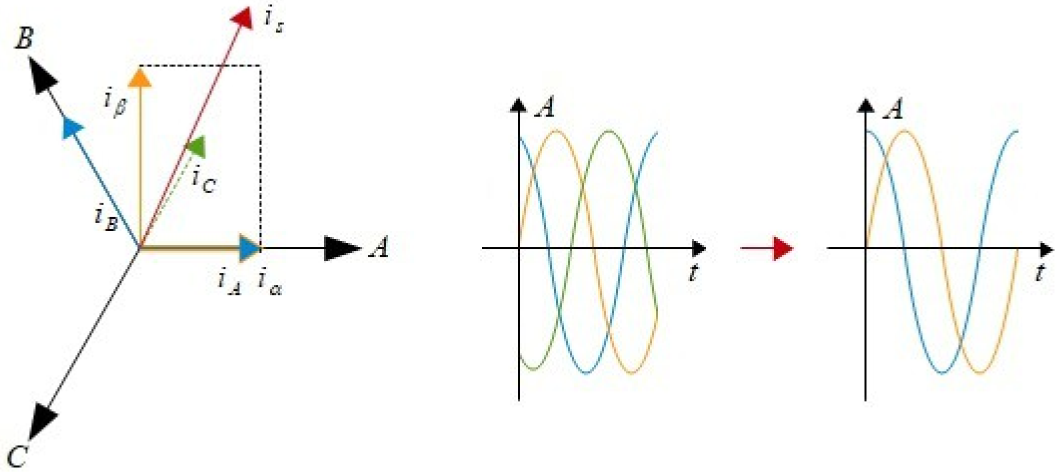


Figure 16: Forward Clarke transformation

```
void vClarke(T_IaIbData* ptIaIbData, T_IAIBData* ptIAIBData)
{
     ptIAIBData->IA = ptIaIbData->Ia;
     ptIAIBData->IB = (ptIaIbData->Ia + 2 * ptIaIbData->Ib) / SQRT3;
}
```

Figure 17: Clarke transformation implementation

Park transformation translates the two-axis orthogonal system from Clarke's transformation into a rotating orthogonal system. Park transformation is given by the following equation [18]:

$$\begin{vmatrix} i_d \\ i_q \end{vmatrix} = \begin{vmatrix} \cos(\Theta) & \sin(\Theta) \\ -\sin(\Theta) & \cos(\Theta) \end{vmatrix} . \begin{vmatrix} i_\alpha \\ i_\beta \end{vmatrix} \qquad (4.7)$$

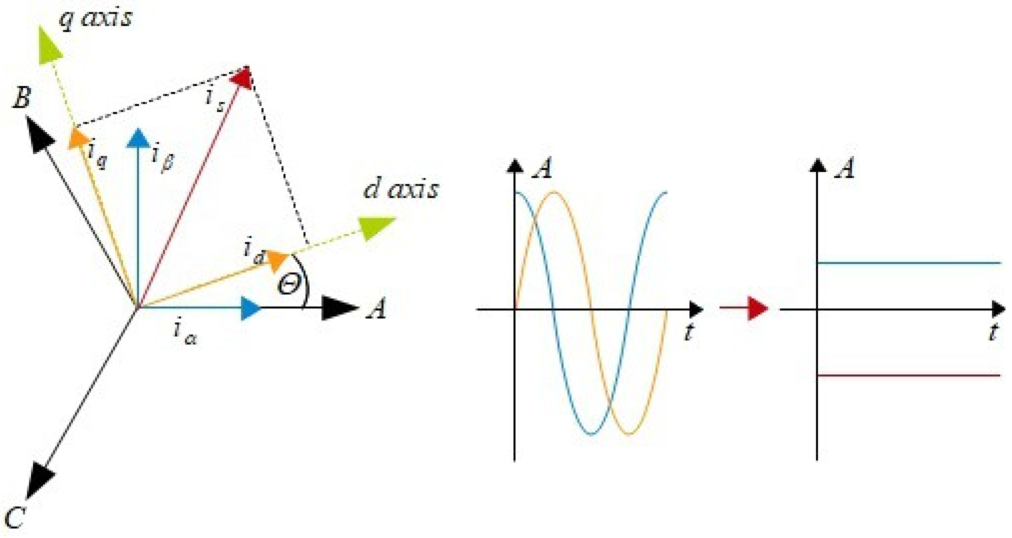


Figure 18: Forward Park transformation

The effect of those transformation is that the values along the direct and quadrature axis are independently controllable. The q-axis current value is the torque producing current and the d-axis current is the magnetization current.

4.4.2 Inverse Park and Clarke transformations

The inverse Park transformation, as the name suggests, is a translation from the rotating orthogonal system back to stationary orthogonal system. It is given by equation [18]:

$$\begin{vmatrix} V_\alpha \\ V_\beta \end{vmatrix} = \begin{vmatrix} \cos(\Theta) & -\sin(\Theta) \\ \sin(\Theta) & \cos(\Theta) \end{vmatrix} . \begin{vmatrix} V_d \\ V_q \end{vmatrix} \tag{4.8}$$

Inverse Clarke transformation transforms the voltage vectors in the stationary orthogonal two axis system back to three phase quantities [18].

$$\begin{vmatrix} V_A \\ V_B \\ V_C \end{vmatrix} = \begin{vmatrix} V_\alpha \\ \frac{-1}{2} V_\alpha + \frac{\sqrt{3}}{2} V_\beta \\ \frac{-1}{2} V_\alpha - \frac{\sqrt{3}}{2} V_\beta \end{vmatrix} \tag{4.9}$$

4.5 Space vector modulation implementation

Pulse width modulation within the microcontroller is implemented with counter and a comparison value. To implement SVM, three timers are configured to run synchronously in up-down counting mode, meaning that timer starts to count up from zero, with a specified rate, up to a predetermined value called timer period. When this period value is reached the timer starts to count down towards zero. When zero value is reached the process starts all over.

Implementation of space vector modulation is based on three timers running synchronously. With each timer a comparison value is established that is compared with the timers counter value. Output of the MCU GPIO pin is changed every time timers counter value is equal to the comparison value. Namely when timer is counting up and the counter value becomes equal to the comparison value the GPIO pin is set to a logic high level. When timer is counting down and counter value becomes equal to the comparison value the GPIO pin is set to a logic low level. This procedure is inverted for the low-side transistors in the inverter. The compare levels are obtained using inverse Clarke transformation described previously.

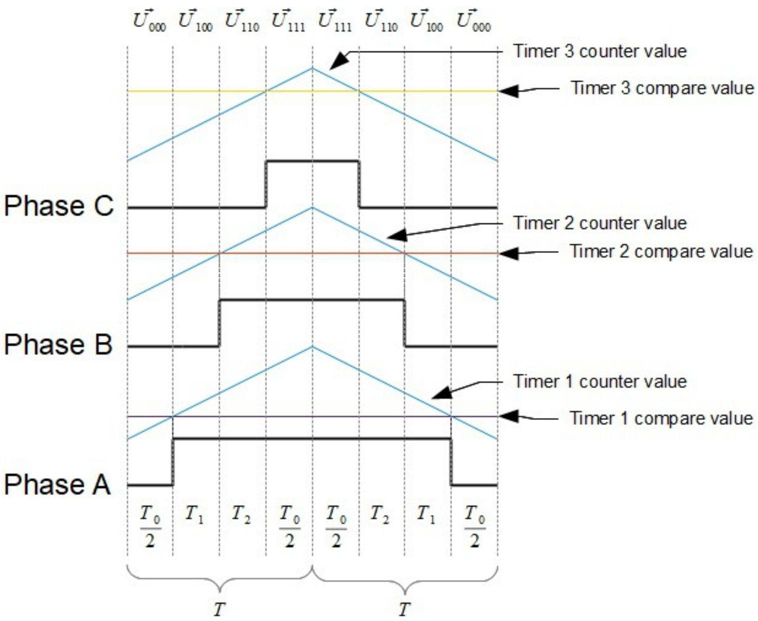

$\vec{U}_{000}$ $\vec{U}_{100}$ $\vec{U}_{110}$ $\vec{U}_{111}$ $\vec{U}_{111}$ $\vec{U}_{110}$ $\vec{U}_{100}$ $\vec{U}_{000}$
Timer 3 counter value
Timer 3 compare value
Phase C
Timer 2 counter value
Timer 2 compare value
Phase B
Timer 1 counter value
Timer 1 compare value
Phase A
$\frac{T_0}{2}$ T_1 T_2 $\frac{T_0}{2}$ $\frac{T_0}{2}$ T_2 T_1 $\frac{T_0}{2}$
T
T

Figure 19: Space Vector Modulation implementation

In the figure T is the switching period. Modulation period consists of two consecutive switching periods.

One method for ordering the time periods calculated by equations (2.2), (2.3) and (2.4) within the switching period is to use alternate-reverse strategy.

Namely, the time zero vector T_0 is divided into two equal time intervals, one used for vector $\vec{U}_{000}$ and the other for vector $\vec{U}_{111}$ (**see** Table 1, **page** 14).

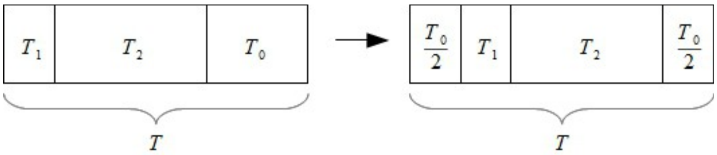


Figure 20: SVM time intervals

To efficiently implement the composition of modulation period the order of the time intervals for the second switching period within the modulation period is reversed. This has advantage of also reducing the number of times transistor is opened and closed within the modulation period, therefore reducing switching losses.

4.6 PID controllers

The system composes has three process variables (motor velocity, direct and quadrature axis currents) that need to keep their values close to the required value regardless of disturbances. One of the method to achieve this is a negative feedback based algorithm called PID. PID is an abbreviation and it is a function composed of three separate terms: proportional action, integral action and derivative action [19].

$$u(t)=K_p e(t)+K_i \int e(\tau)d\tau+K_d \frac{d}{dt}e(t) \tag{4.10}$$

Implementation created is based on equation (4.10) but has modifications regarding the integral term. Namely the result of integration is limited. This is there to protect from the integral part from growing to a large value when a big step change is applied to the input and thus creating a large overshoot and increases settling time.

```
void vPID(T_PID* ptPid)
{
    float fError = ptPid->f_Input - ptPid->f_Feedback;

    float fP = fError * ptPid->f_Kp;
    float fD;

    if(fabsf(ptPid->f_Integrator + fError*(ptPid->f_Ki))
    < ptPid->f_IntegratorLimit)
    {
        ptPid->f_Integrator += fError*(ptPid->f_Ki);
    }
    fD = (fError - ptPid->f_PrevError) * ptPid->f_Kd;

    ptPid->f_Output = fP + ptPid->f_Integrator + fD;

    ptPid->f_PrevError = fError;
}
```

Figure 21: PID regulator implementation

In order for the system to operate we must tune the three coefficients of the three PID regulators within the system. In this system, due to the fact that the motor model isn't created, PID control loop tunign can be performed as an iterative process taking into account the effects of independently tuning P, I and D constants. The effects of separately tuning the constants is given in the table below.

Table 3: Effects of independent P, I and D tuning [20]

Closed-loop response	Rise time	Overshoot	Settling time	Steady-State error	Stability
Increasing Kp	Decrease	Increase	Small increase	Decrease	Degrade
Increasing Ki	Small decrease	Increase	Increase	Large decrease	Degrade
Increasing Kd	Small decrease	Decrease	Decrease	Minor change	Improve

Tuning process was split into two: tuning of current regulators and tuning of speed controller. In the tuning of current regulators the quadrature axis current regulator was tuned first, as the output of this regulator produces torque on output and is a prerequisite for tuning the direct axis current. Namely when we have tuned the PID for quadrature axis regulator, direct axis current regulator can be tuned with a rotating motor by monitoring the current consumed by the motor and tuned so that the direct axis current value is close to 0. The constants that were found in the tuning process are visible in **Table** 4.

Table 4: PID constants found by manual tuning process

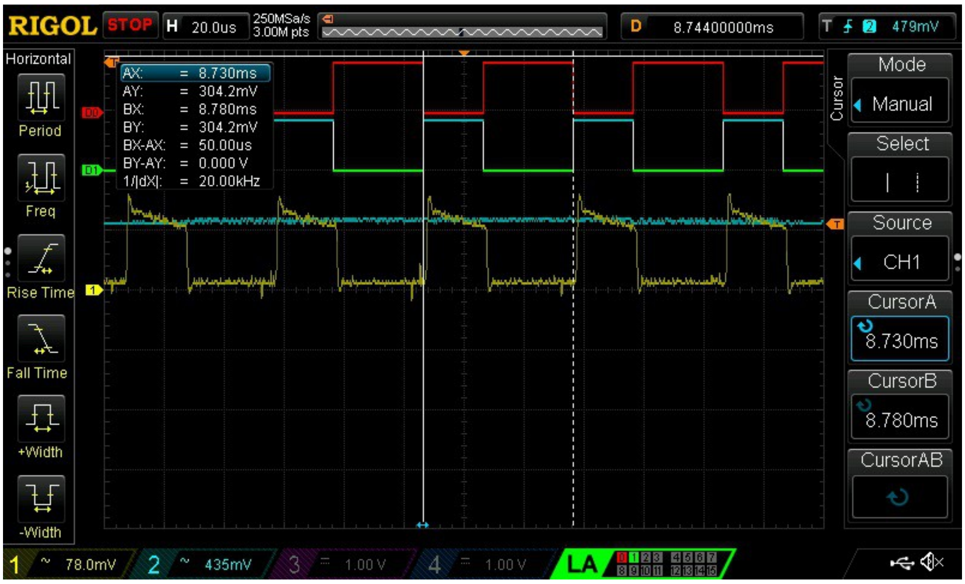
	Kp	Ki	Kd
Quadrature axis current PID	$4.0\cdot10^{-6}$	$10.0\cdot10^{-6}$	$10.0\cdot10^{-5}$
Direct axis current PID	$1\cdot10^{-4}$	$10.0\cdot10^{-6}$	0
Speed controller	$9.0\cdot10^{-2}$	$10.0\cdot10^{-6}$	1.5

5 System performance evaluation

In this paragraph measurement result from the implemented system are presented, analysed and compared to the original requirements.

5.1 Current measurement verification

To verify current measurement functionality an oscilloscope was used. One of the oscilloscope input channels was connected directly to the DRV8305 current sense amplifier output. Second oscilloscope input channel was connected to MCU DAC output. The DAC input from software was updated after every measurement with the value read from ADC.



H 20.0us
250MSa/s
3.00M pts
D 8.74400000ms
T 2 479mV

AX: = 8.730ms
AY: = 304.2mV
BX: = 8.780ms
BY: = 304.2mV
BX-AX: = 50.00us
BY-AY: = 0.000 V
1/|dX|: = 20.00kHz

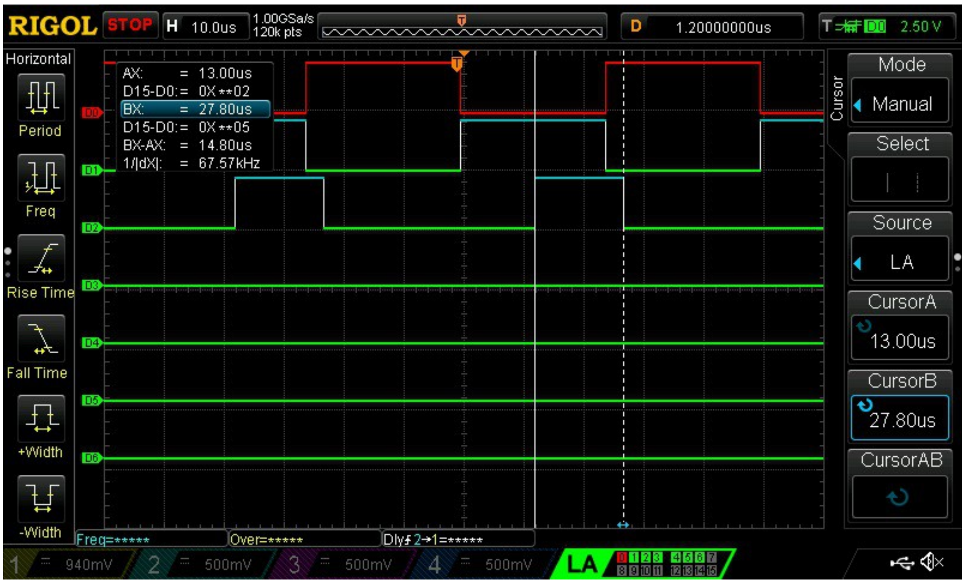
Mode: Manual
Select
Source: CH1
CursorA: 8.730ms
CursorB: 8.780ms
CursorAB

1 ~ 78.0mV
2 ~ 435mV
3 = 1.00 V
4 = 1.00 V

Figure 22: Current measurement

Digital signal D0 is connected to the high-side and digital signal D1 to the low-side transistor of one inverter leg. Oscilloscope channel 1 (yellow) is connected to the current sense amplifier output and channel 2 (blue) is connected to DAC output which is loaded with the measured value from ADC. It can be clearly seen that the current is measurable only when the lower transistor in the inverter leg is opened. Also we can see that the value measured by ADC corresponds to the CSA output value at the middle of low side transistor opening time, which was established as the sampling moment.

5.2 Algorithm execution time



```
AX:      = 13.00us
D15-D0:= 0X**02
BX:      = 27.80us
D15-D0:= 0X**05
BX-AX:   = 14.80us
1/|dX|:  = 67.57kHz
```

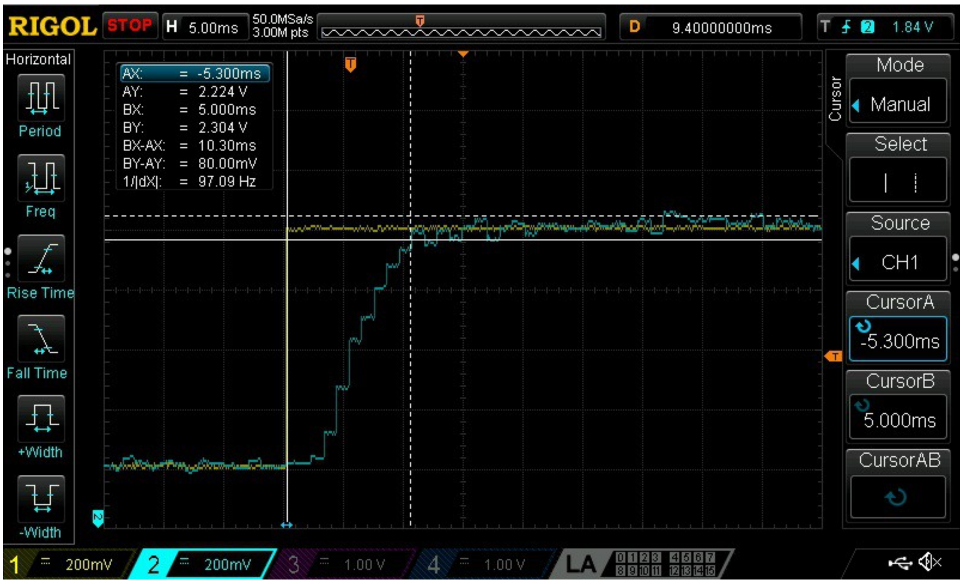
Figure 23: Algorithm execution time measurement

Algorithm execution time was measured by setting an general purpose input/output (GPIO) pin of the MCU to a logical high level when the PWM timer reached zero value and ADC sampling starts. At the end of the FOC algorithm calculations the same GPIO pin was set to a logical low level. By measuring the time during which the signal was on a logical high level algorithm execution time can be measured.

On the figure above the digital signals D0 and D1 (the upper two) are gate signals to one inverter leg. Signal D2 is connected to the GPIO mentioned above. As can be seen the execution interval is 14,80 microseconds. This is well below the requirement of half of modulation period, which is 25 microseconds. Considering this we can make a conclusion that the implementation of FOC meets its timing constraint.

5.3 Speed controller performance

In order to verify the speed controller performance a step response experiment was performed where to a standstill motor velocity reference of 500 RPM was given with no ramping. The reference and actual velocity values were sent to the DAC of the microcontroller and the DAC outputs were monitored with an oscilloscope. The output of the result can be seen on the following scope screen capture.



```
AX:      =  -5.300ms
AY:      =  2.224 V
BX:      =  5.000ms
BY:      =  2.304 V
BX-AX:   =  10.30ms
BY-AY:   =  80.00mV
1/|dX|:  =  97.09 Hz
```

Figure 24: System reaction to step response

In Figure 24 the yellow line is showing the input to the speed controller and the blue line is the measured rotor velocity. From the image it can be seen that the unloaded motor achieves the desired velocity in 10,30 milliseconds. Also it can be seen that the output velocity stays between the required steady-state error limits established (limits are visible as horizontal cursors).

Summary

As a result of this thesis a working embedded C application for field orientated control of BLDC motor was done. It contains the theoretical overview, implementation and performance analysis of the implemented system.

At first a problem statement was made, then a study of theoretical background was performed. In the end a working solution that complies with the requirements established. based on the theoretical knowledge gathered was created. In the course of the thesis several problems were solve, for example: current measurements in three phase inverters, calibration between rotor mechanical and flux angles and the implementation of space vector modulation.

The implemented solution provides a good starting point for further development in field orientated control. In this thesis the system used an absolute encoder for position feedback. This is not a viable solution in many cases due to the increased complexity and cost. There exists numerous different methods for obtaining rotor angle information without any physical angle sensors. This work could be used as a reference platform for comparing the angular information provided by sensorless algorithms with angular measurements from absolute encoder.

Kokkuvõte

Käesoleva töö tulemusena valmis C keeles kirjutatud programm mikrokontrollerile, mis võimaldab juhtida harjavabat alalisvoolumootorit, kasutades väljaorienteeritud juhtimist. Töö koosneb teoreetilise poole ülevaatest, kasutatud riistvarakomponentide kirjeldusest, süsteemi implementatsioonist ning süsteemi töötulemuste hindamisest.

Töö alguses püsitati lähteülesanne, mille põhjal koostati uurimus selle valdkonna teoreetiliste aluste kohta. Töö lõpuks valmis töötav lahendus, mis vastas töö käigus esitatud kriteeriumitele. Töö käigus lahendati erinevaid probleeme, näiteks faasivoolude mõõtmine 3-faasilises pingevaheldis, rootori mehaanilise asendi ja magnetvoo vahelise nurga kalibreerimine ja ruumvektormodulatsiooni implementeerimine.

Valminud süsteem pakub head lähtekohta edasiseks arendamiseks väljaorienteeritud juhtimise vallas. Antud töös on kasutatud süsteemi tagasisidestamiseks absoluutset asendiandurit, mis ei pruugi alati olla sobiv, sest lisakomponendina tõstab see süsteemi keerukust ja hinda. Alternatiivina eksisteerib mitmeid erinevaid meetodeid, kuidas mõõta harjavaba alalisvoolumootori rootori asendit ilma füüsiliste anduriteta. Seda tööd saaks kasutada referentsplatvormina, et võrrelda asendianduriteta juhtimisüsteemide poolt hinnatud rootori asendit asendianduri poolt mõõdetud asendiga.

References

[1] Kwang Hee Nam, AC Motor Control and Electric Vehicle Applications, Boca Raton: CRC Press, 2010

[2] Duane C. Hanselman, Brushless Permanent Magnet Motor Design, Lebanon: Magna Physics Publishing, second edition, 2003

[3] Valery Vodovozov, Dmitri Vinnikov, Electronic Systems of Motor Drive, Tallinn : Tallinn University of Technology, 2008

[4] Bahram Amin, Induction Motors: Analysis and Torque Control, Heidelberg: Springer-Verlag Berlin Heidelberg, 2001

[5] Peter Was, Sensorless Vector and Direct Torque Control, New York: Oxford University Press Inc. 1998

[6] Joseph P. John, S. Suresh Kumar, B.Jaya, Space Vector modulation based Field Oriented Control Scheme for Brushless DC motors, Nagercoil, International Conference on Emerging Trends in Electrical and Computer Technology , 2011

[7] LAUNCHXL-F28377S image. [WWW] http://43oh.com/wp-content/uploads/2015/06/TI_TMS320F28377s_launchpad_st.jpg (19.05.2017)

[8] TMS320F2837xS Delfino Microcontroller Techincal Reference Manual. [WWW] http://www.ti.com/lit/ug/spruhx5d/spruhx5d.pdf (19.05.2017)

[9] TMS320F2837xS block diagram. [WWW] http://www.ti.com/product/TMS320F28377S (19.05.2017)

[10] BOOSTXL-DRV8305 User guide. [WWW] www.ti.com/lit/ug/slvuai8/slvuai8.pdf (19.05.2017)

[11] BOOSTXL-DRV8305 image. [WWW] https://www.digikey.com/product-detail/en/Texas-Instruments/BOOSTXL-DRV8305EVM/296-43181-ND/5761583 (19.05.2017)

[12] AS5047D encoder datasheet [WWW] https://ams.com/eng/content/download/595083/1609657/file/AS5047D_DS000394_2-00.pdf (19.05.2017)

[13] AS5047D encoder image. [WWW] http://img.directindustry.com/images_di/projects/images-g/14-bit-on-axis-magnetic-rotary-position-sensor-11-bit-decimal-binary-incremental-pulse-count-35848-9922756.jpg (19.05.2017)

[14] Nanotec DF45L024048 image. [WWW] http://uk.farnell.com/productimages/standard/en_GB/2507571-40.jpg (19.05.2017)

[15] Nanotec DF45L024048 datasheet. [WWW] https://en.nanotec.com/fileadmin/files/Datenblaetter/BLDC/DF45/DF45L024048-A.pdf (19.05.2017)

[16] AN119: Calculating Settling Time for Switched Capacitor ADCs, Silabs [WWW]. https://www.silabs.com/documents/public/application -notes/AN119.pdf (19.05.2017)

[17] TMS320F2837xS Delfino Microcontrollers datasheet. [WWW] http://www.ti.com/lit/ds/symlink/tms320f28377s.pdf (19.05.2017)

[18] Mircea Popescu, Induction Motor Modelling For Vector Control Purposes, Espoo: Helsinki University of Technology, Laboratory of electromechanics, 2000

[19] K. Aström, T. Hägglund, PID Controllers: Theory, Design and Tuning, Durham: Instrument Society of America, 2nd edition, 1995

[20] Kiam Heong Ang, G. Chong, Yun Li, PID Control System Analysis, Design, and Technology: IEEE, 2005

Appendix 1 – Code

Full implementation of software is provided on CD.