Vector Control of Permanent Magnet Synchronous Motors Using Embedded Coder & Sensorless Control Simulation Using Sliding Mode Observer

A THESIS
SUBMITTED TO THE FACULTY OF THE GRADUATE SCHOOL
OF THE UNIVERSITY OF MINNESOTA
BY

Puneeth Kumar Srikanta Murthy

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
Master of Science in Electrical Engineering

Prof. Ned Mohan

July, 2016

Acknowledgements

First, I would like to thank my advisor Prof. Ned Mohan for giving me the opportunity to work on this project. His suggestions and guidance were valuable for completing this project.

I thank Prof. William Robbins and Prof. Adam Rothman for being the members of the examining committee and for their support.

I thank Siddharth Raju for his suggestions and help during the course of the project.

Finally, I would like to thank everyone in the research group for being there providing valuable support when needed.

Dedication

To my family, teachers and friends.

Abstract

Availability of cheap and better performing floating point digital signal processors has given a boost for the digital control market in the field of power and motor control.In addition to that availability of graphical programming tools for micro-controllers has driven research into the ease of use and effectiveness of DSP micro-controllers for power and motion control in the academic field as well as in the industry. In that direction a floating point DSP micro-controller was used to perform vector control of permanent magnet synchronous motor. *Matlab Simulink Toolbox:Embedded Coder*, an optimized code generation toolbox for embedded systems was used to develop the required code. Also, a study of space vector pulse width modulation techniques was carried out to operate AC machines at reduced harmonic distortion. The effectiveness and ease of programming of floating point DSP micro-controllers was shown. It is recommended to be used in academic and research centers for the ease of development, effective performance and reduction in time to market. Along with this a simulation model for sensorless control of permanent magnet synchronous motor is provided.

Contents

Acknowledgements **i**

Dedication **ii**

Abstract **iii**

List of Tables **vi**

List of Figures **vii**

1 Introduction **1**

1.1 Background 1

1.2 Objective of Study 2

1.3 Dissertation Organization 3

2 Code Generation Using Embedded Coder **4**

2.1 Introduction to Embedded Coder 4

2.2 Embedded Coder TI C2000 Blocks 6

3 Modeling And Control of Permanent Magnet Synchronous Motor **10**

3.1 Introduction to Permanent Magnet Motors 10

3.2 Mathematical Model of PMSM 11

3.2.1 Rotating Reference Frame 12

3.3 Vector Control 13

3.3.1 Current(Torque) Controllers 14

3.3.2 Speed Controller 15

3.4 Matlab Model Simulation and Results 16

4 Vector Control using Embedded Coder 19

4.1 Embedded Coder Model 19

4.1.1 Simulink Model 20

4.2 Hardware Setup 28

4.3 Hardware Results 29

5 Sensorless Vector Control Using Sliding Mode Observer 31

5.1 Sliding Mode Control 31

5.2 Sliding Mode Observer for PMSM 32

5.3 Position and Speed 34

5.4 Simulation in Matlab And Results 34

6 Conclusion and Future Work 40

References 42

Appendix A. Getting Started With Embedded Coder 44

Appendix B. Space Vector PWM Techniques for Reduced Harmonic Distortion in AC Machines 49

Appendix C. Hardware Setup 55

List of Tables

3.1 Simulation Motor Parameters 17
4.1 Simulation Motor Parameters 29
B.1 Switching Sequences 51

List of Figures

2.1 Block Diagram of Code Generation 5
2.2 Embedded Coder Support Package for TI C2000 7
2.3 Simulation Model 8
2.4 Code Generation Model 8
2.5 Highlighted Part For Code Comparison 9
3.1 Cross-sectional View of Permanent Magnet Motor 11
3.2 d-axis Equivalent Circuit 12
3.3 q-axis Equivalent Circuit 13
3.4 Block Diagram of Current Controller 15
3.5 Block Diagram of Speed Controller 16
3.6 PMSM Simulation Model 17
3.7 Simulation Results 18
4.1 Embedded Coder Model for Vector Control 20
4.2 eQEP General 22
4.3 eQEP Position Counter 22
4.4 eQEP Speed Calculation 22
4.5 Speed and Position Calculation 23
4.6 ADC Scaling 24
4.7 ADC COnfiguration - 1 25
4.8 ADC COnfiguration - 2 25
4.9 Counter Compare Operation 26
4.10 ePWM General 27
4.11 ePWM Channel A 27
4.12 ePWM Event Trigger 27

4.13 Anti-Windup PI Controller 27
4.14 Block Diagram of the Hardware Setup 28
4.15 Line to Line Voltage 29
4.16 Line Current 30
5.1 Sensorless Vector Control Model 35
5.2 Position and Speed Estimation 36
5.3 Estimated and Measured Speed 36
5.4 Theta Before And After COmpensation 37
5.5 Measured and Estimated Currents 38
5.6 Measured Back EMF and Back EMF from Observer 38
5.7 D and Q Axis Currents from Estimated Position 39
A.1 Code Generation Model 45
A.2 Step 1 - 1 45
A.3 Step 1 - 2 46
A.4 Step 2 47
A.5 Step 3 47
A.6 Step 4 48
B.1 Duty Ratio Waveforms 50
B.2 Sector Diagram 50
B.3 Harmonic Distortion vs Frequency 54
C.1 Hardware Setup 55
C.2 PMAC Motor 56
C.3 Control Hardware 56
C.4 Control Card And Extension Board 57

Chapter 1

Introduction

1.1 Background

As the need for efficiency and conservation of power in industry increased application of power electronic circuits for operation of electrical ac machines increased. With the introduction of vector control for ac machines, they have further contributed to the efficient operation by generating maximum torque for every ampere of current drawn from the source.

With the development of digital electronics and introduction of microprocessors, FPGA's, DSP's etc, digital control of ac drives has taken prominence. For digital control the processors provides the user with the flexibility to perform complex algorithms, parameter modification, memory and reduces time to market unlike analog control.Though they have a lot of advantages programming them becomes an additional complex task in an already multi-disciplinary field of power electronics, machines, digital control, sensors and other peripheral circuit designs. Hence, model based design comes into picture for which *Matlab : Embedded Coder Toolbox* has been proposed to be used in university research and laboratories.

Though induction motors are the work horse of the industry, permanent magnet synchronous motors have started playing significant role not only in the field of automation. But they have started replacing induction motors or being the primary in the places with direct drive applications such as electric drives, vacuum cleaners, washing machines etc. The PMSM motors not only have high power density and efficiency

but also are able to provide high torque at low speeds, fast dynamic response, precise positioning, high torque to inertia ration and reliability. hence, they are not only able to operate in automation industries but also in stand alone equipment's.

Three phase inverters are used to provide three phase alternating current waveform to ac machines. The three phase ac waveforms are generated by providing pulse width modulated signals which are either sine wave modulated or generated using space vector algorithm. [1] Proposes three space vector pulse width modulation techniques to operate ac machines at reduced harmonic distortion. The various space vector pwm techniques are studied and simulated.

Field oriented control of permanent magnet synchronous motor was achieved by obtaining the rotor position information quadrature encoder placed on the motor. The algorithm to implement FOC was first simulated using *Simulink* and the same model was used along with embedded coder functions to be deployed to hardware.

1.2 Objective of Study

The research is expected to contribute to the usability of embedded coder at university level for undergraduate, graduate and doctoral students. The control hardware containing the floating point processor along with a three phase inverter board was used to perform vector control of a non-salient pole permanent magnet synchronous motor.

The main objectives of the thesis include:

- Identify a suitable floating point digital signal processor capable of carrying out a complex power and motion control algorithm
- Provide a platform for and test model based code generation using *Matlab : Embedded Coder Toolbox*.
- Study various space vector pulse width modulation techniques for operation of ac machines to achieve less harmonic distortion.
- Design vector control algorithm for a permanent magnet motor using *Matlab : Embedded Coder Toolbox* and implement it on hardware.

- Provide a suitable simulation model for sensorless vector control which can be implemented on hardware in the future

1.3 Dissertation Organization

- Chapter 1 provides introduction to the thesis.
- Chapter 2 presents an introduction to code generation using *Embedded Coder* with the procedure to setup the custom blocks provided by the manufacturer.
- In Chapter 3 briefly explains the mathematical model of the permanent magnet synchronous motor and provides control design principle.
- Chapter 4 provides the vector control model for the PMSM in embedded coder. The hardware results are also presented
- Chapter 5 provides information on sliding mode observer based sensorless vector control
- Chapter 6 concludes the thesis document and provides suggestions for future work.

Chapter 2

Code Generation Using Embedded Coder

In this chapter an overview of code generation using *Embedded Coder toolbox* is provided. Later the files generated through code generation is explained. Through a simple example of open loop dc motor control the code generated is compared with the *Simulink* model used to generate the code. Going through this chapter helps one to understand the code generation process and it's advantages.

2.1 Introduction to Embedded Coder

Mathworks Inc., along with various micro-controller manufacturers has developed a *Simulink* toolbox in order to generate processor specific embedded code. Matlab provides processor specific input output block set's which along with the blocks from *Simulink* can be used to generate code.

Simulink allows testing the built model through regular simulation before it could be deployed on to the hardware. The model can be built and deployed to hardware through processor specific software like Code Composer Studio, Cross Core Embedded Studio, Arduino IDE etc. The model can also be deployed directly from Simulink on to the hardware. Both the procedures require that the processor specific integrated development tool be installed.

If a user builds a model in *Simulink* for testing. Once, testing is done, the same

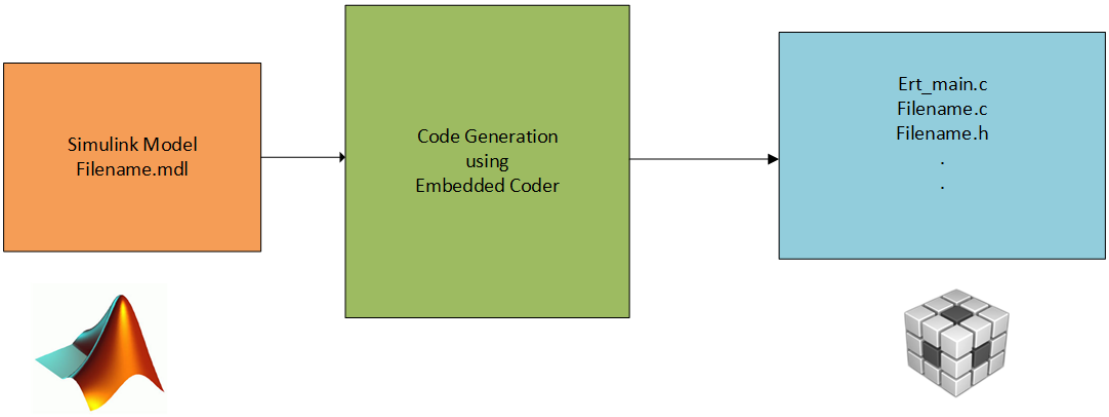

Figure 2.1: Block Diagram of Code Generation

model can be used to generate code by placing the required processor specific blocks and making necessary changes in the configuration menu. Such an example is shown in figures 2.3 and 2.4 . Figure 2.3 contains a model used to test the algorithm being built. Figure 2.4 shows the same algorithm with processor specific blocks at the place where *Simulink* scope was present.*Embedded Coder* provides seamless integration capabilities over multiple processor's not only from the same manufacturer but from different manufacturers as well. This can be achieved by just replacing the processor specific block set's. Hence, the designer does not have to worry about rewriting the whole code again to ensure compatibility. In the figures shown TI-DSP C2833x block set's were used and configured, if the user tends to use a different processor then he just needs to switch those blocks and make the necessary changes in the configuration window, rather than rewriting or rebuilding the whole model again. *Embedded Coder* not only makes the job of a embedded software engineer easy, but also that of an hardware engineer, with a freedom to change the processor at any stage of the project. All these points make *Embedded Coder* a viable tool to be used for research and development in academic field as well as in industries. Processor specific integrated development tool needs to be selected in the configuration menu. Once that is done *Simulink* generates a specific set of files and folders. The description of the same is given below.

- A main file to initialize the system and check for errors.
- A c file to initialize the board peripherals along with interrupts.

- A c file containing the algorithm designed in the *Simulink* model.
- A c file containing the constants and variables used in the *Simulink* model.
- A c file containing the data types of the various constants and variables used in the model.
- A c file to initialize the hardware as per the requirement of the program. This c file is supported by additional c files and header files to initialize the hardware.
- Additional header files are generated to support the code generation are generated or pulled in from the Texas Instruments Control Suite.

A detailed description of the files generated through embedded coder could be found in *Mathworks* documentation [2].

2.2 Embedded Coder TI C2000 Blocks

Mathworks provides additional toolbox along for Texas Instruments microcontrollers. These blocks can be used in *Simulink* like the regular blocks. These blocks represent the functionality of the specified modules. Hence, a knowledge of the configurations of the processor under use is essential.Sophisticated algorithms can be built using available *Simulink* blocks and the TI Embedded Coder blocks can be used to generate the codes. The TI C2000 Embedded Coder tool set provided by Mathworks is shown in figure 2.2 below, As we can see from the figure above, the support package consists of processor specific modules, optimization tools for fixed point operations, RTDX instrumentation for real time operation, Target Communication and scheduling blocks. The processor specific modules includes all the peripheral blocks available in that particular module. The figure 2.2 shows the modules available for C2833x family processors. A brief introduction of the modules commonly used for power and motion control is provided below. The other modules are listed.

- ADC(Analod to Digital Converter) : This block is used to configure the ADC module to obtain digitized signals from the analog inputs. The output from this module is an unsigned 16 bit integer.

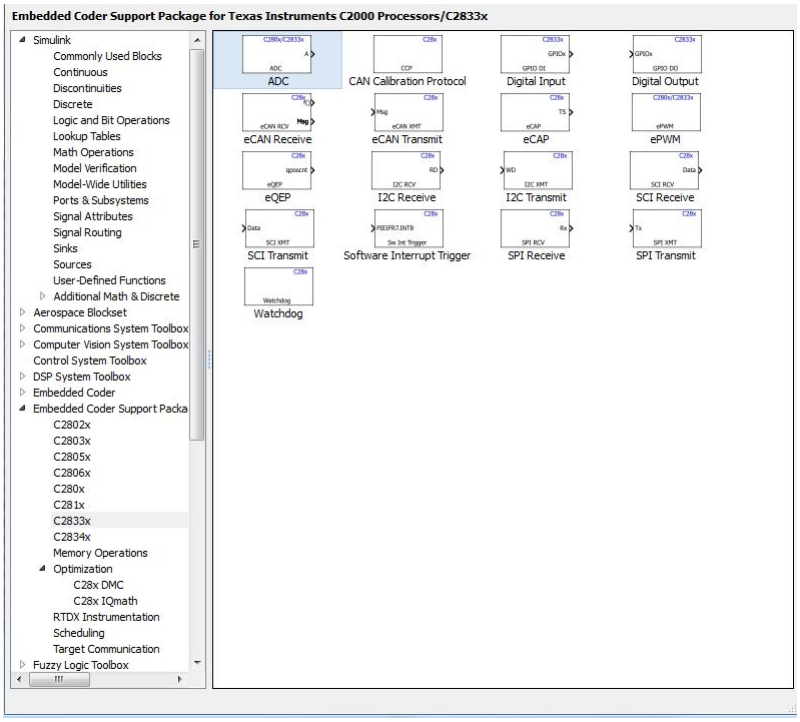

Figure 2.2: Embedded Coder Support Package for TI C2000

- eQEP (enhanced Quadrature Encoder Pulse) : This block is used to capture encoder signal from the motor and can be configured to provide position and speed information.
- ePWM(enhanced Pulse Width Modulation) : This module is used to configured to generate pulse width modulation signals at the required switching frequency. It takes in unsigned 16 bit unsigned integer as counter compare value.
- Software Interrupt Trigger
- Digital Input and Digital Output
- Analog Input and Output
- SPI Receive and Transmit
- eCAN Receive and Transmit

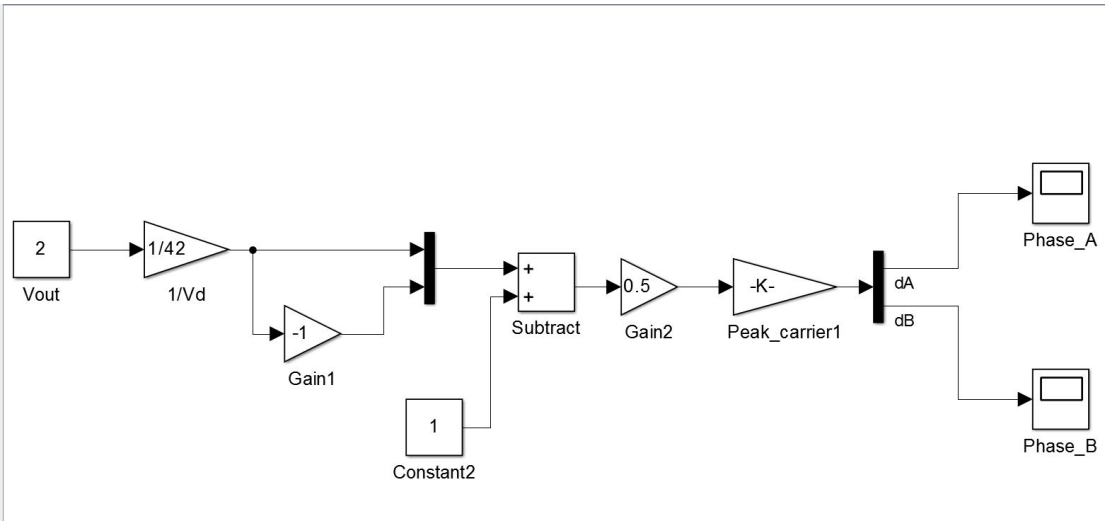

Figure 2.3: Simulation Model

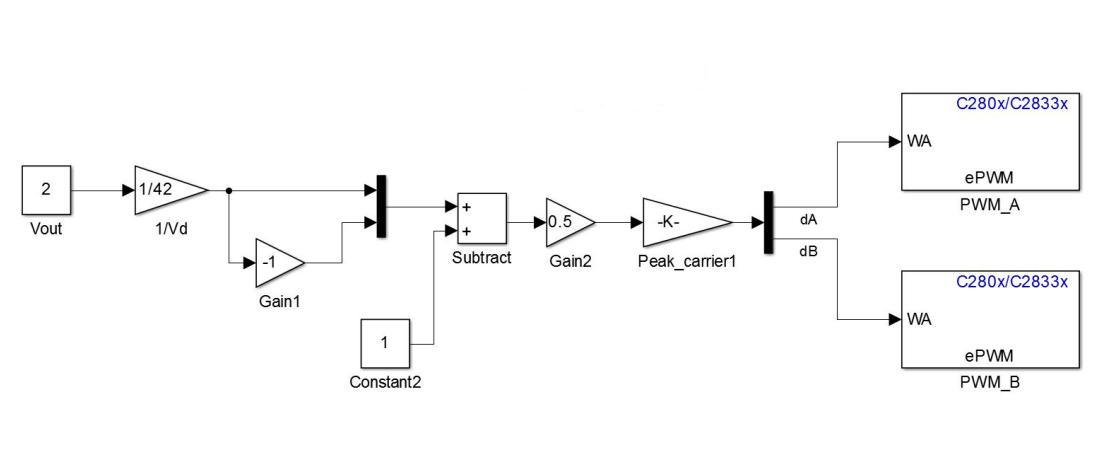

Figure 2.4: Code Generation Model

- I2C Receive and Transmit
- eCAP
- Watchdog

The user need not have to worry about the data type being loaded into the module in Simulink. The code generator ensures that a value with the right data type is written

on to the register.For example let us take a look at the example below. The figure 2.5 shows a highlighted part whose code is given below it.The ePWM module takes in 16 bit unsigned integer values. The user can ignore this fact while building the program, as it can be seen from the generated program. The embedded coder based on configuration takes care of these things.The embedded coder convert whatever values it is given into 16 bit unsigned integer values. The value on the right hand side is converted into 16 bit unsigned integer and then assigned to the counter compare value in the PWM module. The generated code also contains comments which makes it easier to trace the code back to the *Simulink* model.

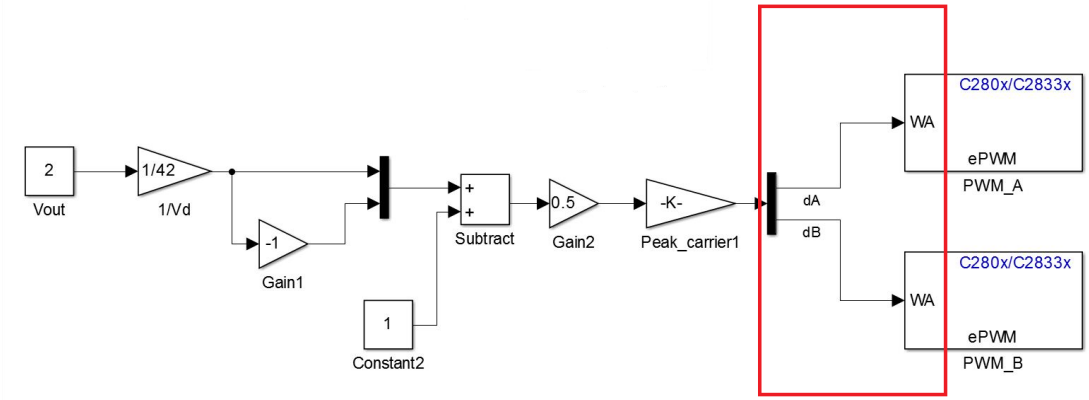

Figure 2.5: Highlighted Part For Code Comparison

```
/* S-Function (c280xpwm): '<Root>/PWMLA' */
/*-- Update CMPA value for ePWM4 --*/
{
  EPwm4Regs.CMPA.half.CMPA = (uint16_T)(rtb_Peak_carrier1[0]);
}
/* S-Function (c280xpwm): '<Root>/PWMLB' */
/*-- Update CMPA value for ePWM6 --*/
{
  EPwm6Regs.CMPA.half.CMPA = (uint16_T)(rtb_Peak_carrier1[1]);
}
```

Chapter 3

Modeling And Control of Permanent Magnet Synchronous Motor

In this chapter the a brief introduction about permanent magnet synchronous motors and it's different types is given.Further, the mathematical equations of the motor is presented in two phase rotating reference frame.Vector control of motor model presented is discussed. The motor modeled in matlab is presented along with vector control of it.

3.1 Introduction to Permanent Magnet Motors

Similar to induction motors the permanent magnet motors have a stator made up of sinusoidally distributed windings. But, their rotor is made up of a permanent magnet. Hence, the need to induce currents in the rotor to produce flux is eliminated. Therefore the independent rotor flux due to the permanent magnet is capable of following the flux produced by the stator. This implies that the rotor flux is able to rotate at synchronous speed and hence the name.

In *Chapter 2* of *Advanced Electric Drives: Analysis, Control and Modeling Using Simulink* [3] Prof. Ned Mohan elegantly shows us the need to analyze sinusoidal alternating current(ac) machines in two phase rotating frame. Furthermore the two phase

rotating frame provides us the advantage of decoupled control. Similar to direct current motors the torque and flux components of the motor can be controlled independently in ac machines as well.

The permanent magnet motors can be classified as sinusoidal and trapezoidal motors based on the kind of current excitation provided to them. The discussion henceforth will be on sinusoidal permanent magnet synchronous motors.

Sinusoidal PM motors can be further classified based on their saliency.

- **Non-salient pole** motors or surface mount permanent magnet motors have a uniform air gap and hence provide the same reluctance along any direction.
- **Salient pole** motors or interior permanent magnet motors the equivalent air gap is not uniform hence the reluctance along is different along different directions of the motor.

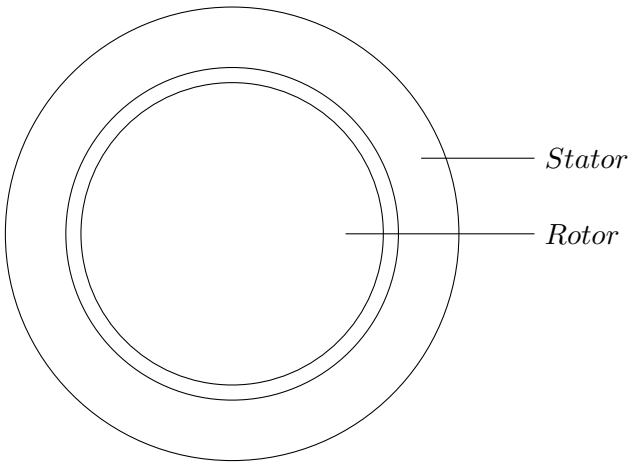

Figure 3.1: Cross-sectional View of Permanent Magnet Motor

3.2 Mathematical Model of PMSM

In this section a mathematical description of the motor model is provided with equations. In the following analysis magnetic saturation, eddy current and hysteresis loss

are neglected. Since the rotor is made up of a permanent magnet which has a high resistivity hence the induced currents in the rotor can be assumed to be zero.

In order to understand and design control for permanent magnet synchronous motors, it is required to have a knowledge of the dynamic model of the PMSM. Mathematical equations which describe the performance of PMSM motor in both steady state and transient operations are provided in state space form.

3.2.1 Rotating Reference Frame

The following equations provide the stator voltage, stator flux linkage and electromagnetic torque equations are given in DQ frame.

$$v_{sd} = R_s i_{sd} + \frac{d}{dt}\lambda_{sd} - \omega_m \lambda sq \tag{3.1}$$

$$v_{sq} = R_s i_{sq} + \frac{d}{dt}\lambda_{sq} + \omega_m \lambda sd \tag{3.2}$$

$$\lambda_{sd} = L_s i_{sd} + \lambda_{fd} \tag{3.3}$$

$$\lambda_{sq} = L_s i_{sq} \tag{3.4}$$

$$T_{em} = \frac{p}{2}(\lambda_{sd} i_{sq} - \lambda_{sq} i_{sd}) \tag{3.5}$$

The figures 3.2 and 3.3 provide an equivalent circuit for the motor in both D and Q axis frame.

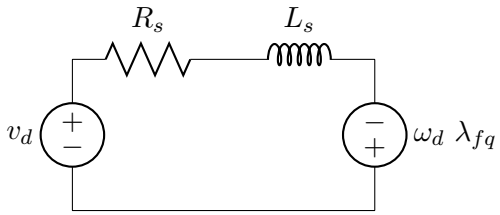

Figure 3.2: d-axis Equivalent Circuit

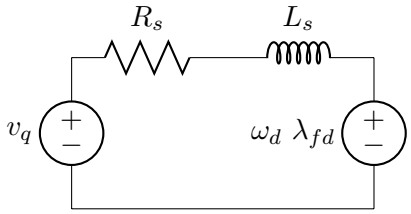

Figure 3.3: q-axis Equivalent Circuit

3.3 Vector Control

In order to obtain high performance operation in alternating current machines vector control strategies is used. Rotor oriented vector control scheme is used for sinusoidal permanent magnet motor because of it's practicality and relative simplicity. The direct axis corresponds to the flux competent whereas the quadrature axis corresponds to the torque compenent.Through decoupled control of both direct axis and quadrature axis current components the magnitude and position of the phase current can be controlled.

The method of indirect vector control is used, where terminal quantities such as voltage and currents are used to measure the field quantities. As a physical explanation, in order to have a maximum torque per amp the the stator current is made 90 degrees ahead of the rotor flux. This implies that the d-axis component of the current should be made zero, since the rotor flux is aligned along the d-axis.Looking at it mathematically, we can write down the input and output power equations. Then we could obtain the value of d-axis current for maximum efficiency. The steps are shown in the following equations.

$$P_i = \frac{3}{2}(v_d i_d + v_q i_q) \tag{3.6}$$

$$P_o = T_{em} = \frac{p}{2}\lambda_{fd} i_{sq} \tag{3.7}$$

$$\eta = \frac{P_o}{P_i} \tag{3.8}$$

$$\frac{d\eta}{di_{sd}} = 0 \tag{3.9}$$

$$i_{sd} = 0 \tag{3.10}$$

Making i_{sd} zero in the equations 3.3 and 3.5 take the form

$$\lambda_{sd} = \lambda_{fd} \tag{3.11}$$

$$T_{em} = \frac{p}{2}\lambda_{sd} i_{sq} \tag{3.12}$$

Using the value of λ_{sd} from equation 3.11 in equation 3.12, we have

$$T_{em} = \frac{p}{2}\lambda_{fd} i_{sq} \tag{3.13}$$

The controller design as explained in *Chapter 8* of *Electric Drives and Machines* [4] is started from the innermost loop. A cascade control structure is used, with the speed controller being the outer loop and the current controller the inner loop.

A block diagram of the vector control architecture is provided in figure 123. The transformation equation used in the equation is provided.

The transformation of current from abc from to dq frame is given below.

$$\begin{bmatrix} i_{sd}(t) \\ i_{sq}(t) \end{bmatrix} = \sqrt[2]{\frac{2}{3}} \begin{bmatrix} cos(\theta_{da}) & cos(\theta_{da} - \frac{2\pi}{3}) & cos(\theta da - \frac{4\pi}{3}) \\ -sin(\theta_{da}) & -sin(\theta_{da} - \frac{2\pi}{3}) & -sin(\theta_{da} - \frac{4\pi}{3}) \end{bmatrix} \begin{bmatrix} i_a(t) \\ i_b(t) \\ i_c(t) \end{bmatrix}$$

The transformation of current from dq from to abc frame is given below.

$$\begin{bmatrix} i_a(t) \\ i_b(t) \\ i_c(t) \end{bmatrix} = \sqrt[2]{\frac{2}{3}} \begin{bmatrix} cos(\theta_{da}) & -sin(theta_{da}) \\ cos(\theta_{da} + \frac{4\pi}{3}) & -sin(\theta_{da} + \frac{4\pi}{3}) \\ cos(\theta_{da} + \frac{2\pi}{3}) & -sin(\theta_{da} + \frac{2\pi}{3}) \end{bmatrix} \begin{bmatrix} i_{sd} \\ i_{sq} \end{bmatrix}$$

3.3.1 Current(Torque) Controllers

Equations 3.1 and 3.2 can be used to come up with the transfer function for the current controllers. Rewriting equation 3.1 below,

$$v_{sd} = R_s i_{sd} + \frac{d}{dt}\lambda_{sd} - \omega_m \lambda sq$$

We can see that only the first two components are the d-axis components which contribute to the d-axis stator voltage, the last term might be considered as a disturbance or used as a compensation term while designing controllers. Hence the transfer function for the d-axis stator voltage is given by,

$$\frac{I_{sd}(s)}{V_{sd}(s)} = \frac{1}{R_s + sLs} \tag{3.14}$$

Similarly, the transfer function equation for q-axis component is given by,

$$\frac{I_{sq}(s)}{V_{sq}(s)} = \frac{1}{R_s + sLs} \tag{3.15}$$

As we can see the transfer function for both the d-axis and q-axis controllers is the same. Hence, the value of gain constants in the PI controller(shown in figure)is the same for both the axes.

The current controller has the highest bandwidth which is an order of magnitude less than the switching frequency of the power inverter. The phase margin is chosen to be 60 degrees.

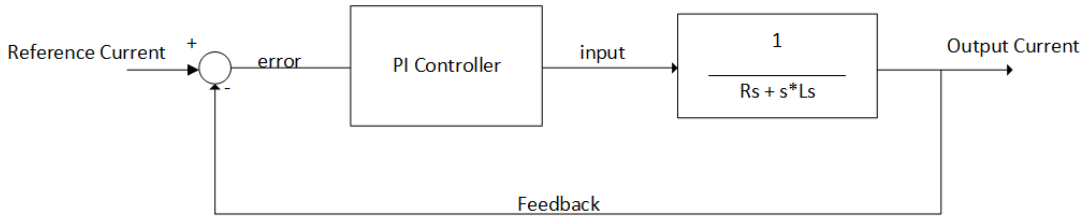

Figure 3.4: Block Diagram of Current Controller

3.3.2 Speed Controller

In order to design speed loop controller which generates the reference value q-axis stator current, it is assumed that the current loop is working ideally and is considered as unity(figure).

The rotor speed, ω_{mech} is given by

$$\omega_{mech} = \int \frac{T_{em} - T_L}{J_{eq}} dt \tag{3.16}$$

From 3.13, we have

$$T_{em} = k i_{sq} \tag{3.17}$$

Where k is given by,

$$k = \frac{p}{2}\lambda_{fd} \tag{3.18}$$

The transfer function for the speed controller is given by,

$$\frac{\omega_{mech}(s)}{I_s q(s)} = \frac{k}{sJ_{eq}} \tag{3.19}$$

The bandwidth for the speed controller is one order of magnitude less than that of the current controller, since the inner most loop should have the highest bandwidth in cascade control mode.

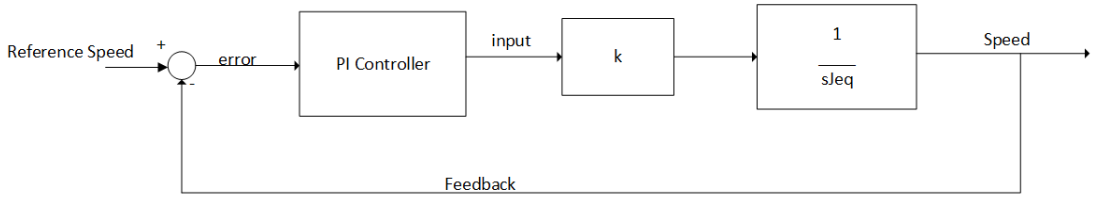

Figure 3.5: Block Diagram of Speed Controller

3.4 Matlab Model Simulation and Results

The dynamical equations presented in Section 3.2 were used to develop a motor model in Simulink. The concept of Vector Control provided in Section 3.3 was used to come with gain constant values for the PI controllers for both speed and current controllers. Computer simulations were performed to verify the performance of the controllers. The machine parameters and the gain constants of the controllers are shown in table 3.1.

The *Simulink* model is shown in the figure 3.6 below.

The simulation results are shown in the figure 3.7

Motor Parameter	Value
Rated Frequency, f	60 Hz
Stator Resistanse, R_s	0.416 Ohm
Stator Inductance, L_s	1.365e-3 mH
No. of Pole Pairs, P	4
Inertia, J	3.4e-3 $Kgm-2$
Rated Speed, N_s	6000 RPM
Back EMF Constant, k_e	0.0957 Vs

Table 3.1: Simulation Motor Parameters

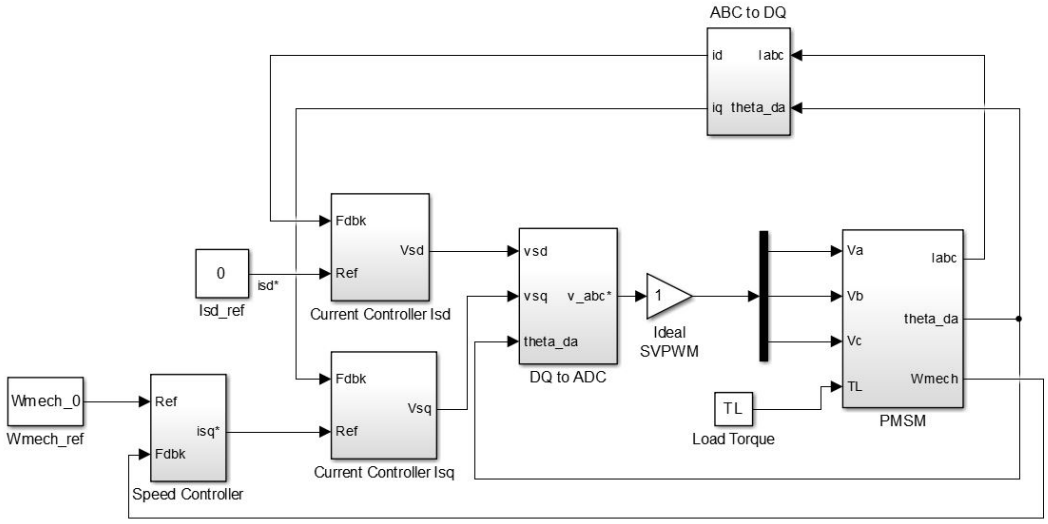

Figure 3.6: PMSM Simulation Model

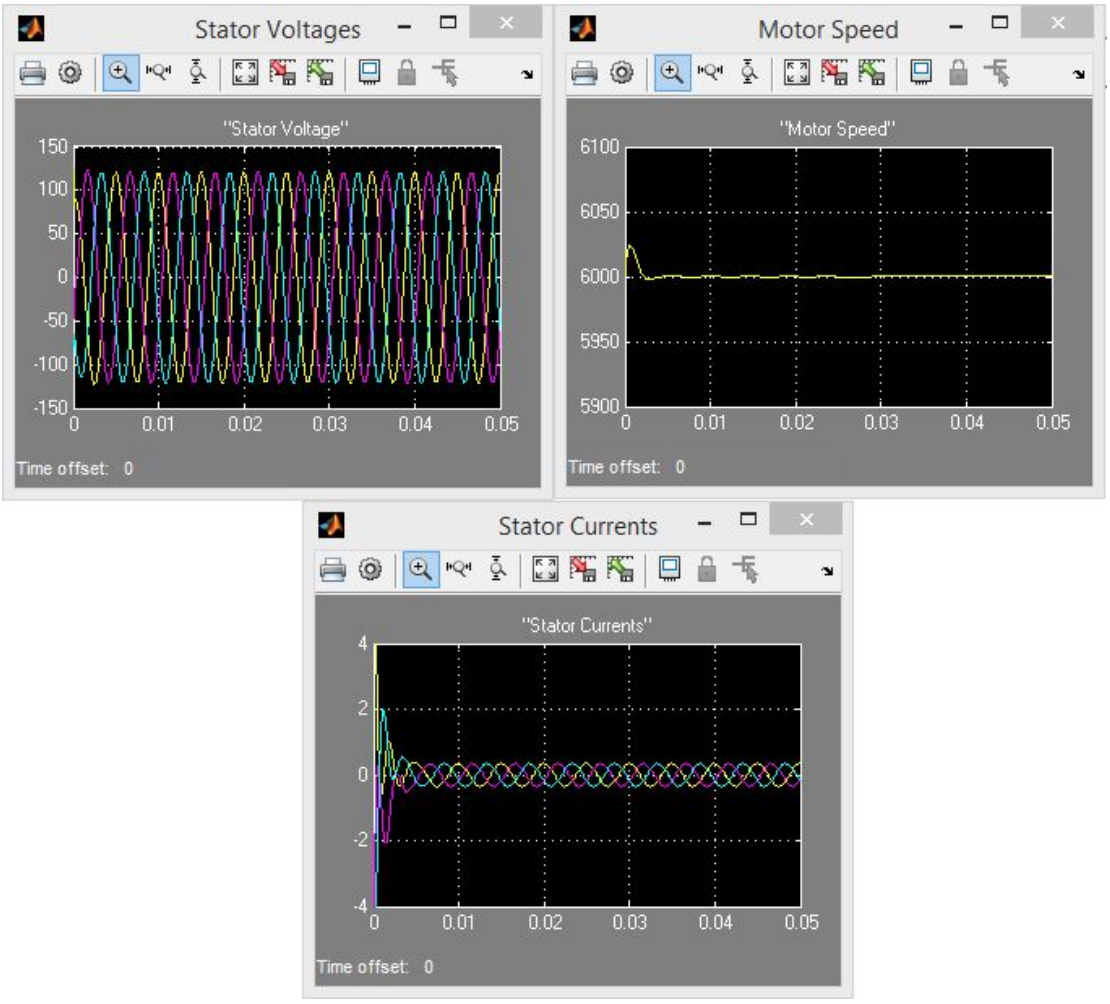

Figure 3.7: Simulation Results

Chapter 4

Vector Control using Embedded Coder

In this chapter,a detailed description to develop the algorithm to perform vector control of a permanent magnet synchronous motor using embedded coder is provided. A description of hardware used is provided. Finally, the hardware results are presented.

4.1 Embedded Coder Model

As mentioned in chapter 2. One of the advantages of Code Generation using Embedded Coder is the need to eliminate development of a new model different from the simulation model. The model used for simulation only can be used to develop the model for embedded coder with a few additional blocks for hardware description blocks, scaling and other calculations.In this section we further look at the description of the blocks used to develop the model.

Texas Instruments C2000 support package for Embedded Coder toolbox was used to develop the model in *Simulink*. The DSP used was F28335 [5], corresponding C2833x hardware modules from the support package was used to develop the code for the module. These blocks were configured as per the requirement for operation after thoroughly going through the data sheets.

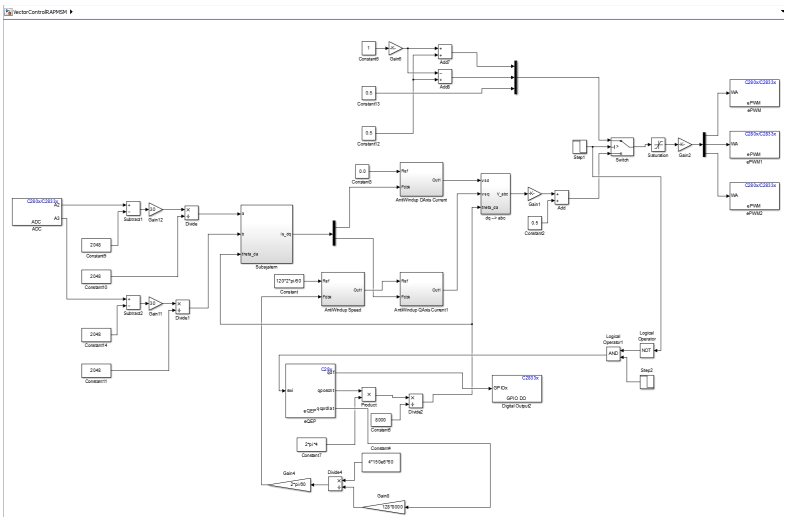

Figure 4.1: Embedded Coder Model for Vector Control

4.1.1 Simulink Model

The *Simulink* model for the vector control of PMAC motor is shown in the figure 4.1 . The model is similar to the one developed to perform simulation studies, as shown in Chapter 3. The motor model is replaced by hardware modules either acting as inputs or outputs accordingly.

The output module includes the enhanced Pulse Width Modulation module. This module provides the PWM signals for the inverter in order to generate the required voltage. The input modules include the Analog to Digital Converter and enhanced Quadrature encoder Peripheral. The ADC module provides the current information. The eQEP module provides the position information needed to perform transformation and speed information. These three modules act as a substitute for the motor model provided in the simulation.

The following subsection provides information on the scaling factors used along with these blocks as well as the way to configure these blocks to obtain the required information.

Speed Calculation And Position Information

In motor drive applications obtaining position and speed information is absolutely necessary to perform precise control. This information is generally obtained through encoder, which converts the speed and position information into electrical signals. The quadrature encoder on the motor model provides three signals A, B and index. The A and B signal's are lag or lead each other by 90 degrees depending on the direction of rotation of the motor. The index signal is used to determine the home position of the motor.

The encoder provided on the motor being used for this experiment has 2000 lines per revolution. This indicates that there are 8000 discrete position information being provided per revolution. Hence, the resolution of the quadrature encoder is 360.0/8000 = 0.045 degrees per position information.

These three signals are connected to the TI-F28335 chip which has eQEP module [6]. The eQEP module has two capture unit's, eQEP1 and eQEP2. eQEP1 is used to capture the quadrature encoder signals.

The eQEP module has different count modes viz, quadrature count mode, direction count mode, up-count mode and down-count mode.Since a quadrature encoder is being used, the quadrature count mode is used. The positive direction of rotation was assumed to be the counter clockwise direction of rotation. These settings are made under general tab.

Under the position counter tab, the *Output Position Counter* is selected. The maximum position counter value is set to $2^{16} - 1 = 65535$. The position counter is reset on index value. Also, the position counter is set to an initial value of zero when the motor is released from the lock position.

Under Speed calculation tab, the eQEP capture is enabled. Upon unit time out event the timer and period are captured. The timer period latched value which is enabled here is used to calculate the required speed, since we are performing low speed operation. In order to obtain the unit timer period value the system clock is divided by the sample time used for the module.

The equation used to perform low speed calculation can be given by,

$$Speed = \frac{No.of PositionEvents}{TotalClockCycles} \times ClockPeriod \times \frac{OneRevolution}{No.of Positions} \times \frac{60Seconds}{1minute} \quad (4.1)$$

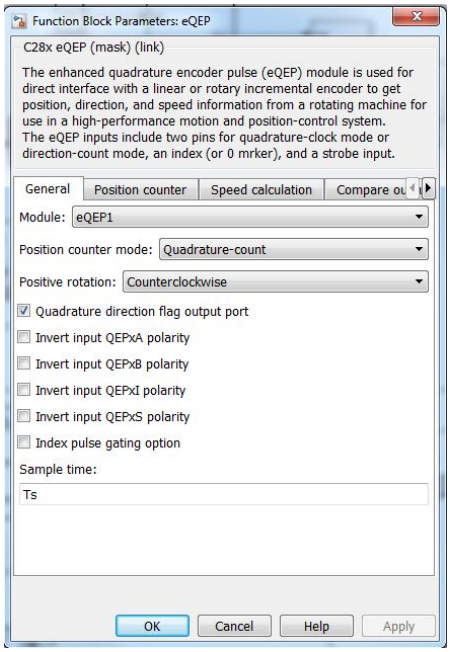

Figure 4.2: eQEP General

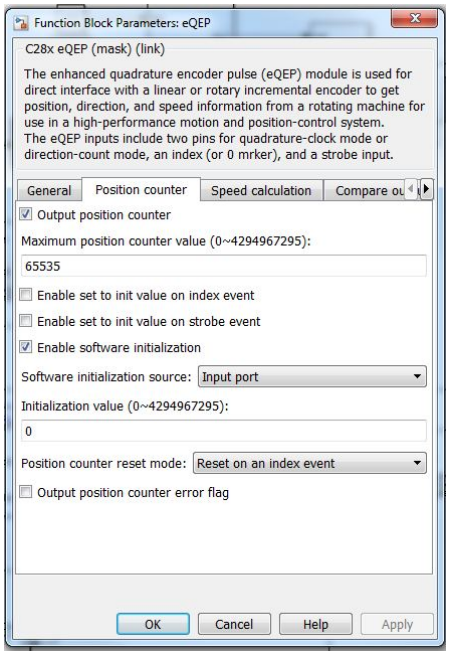

Figure 4.3: eQEP Position Counter

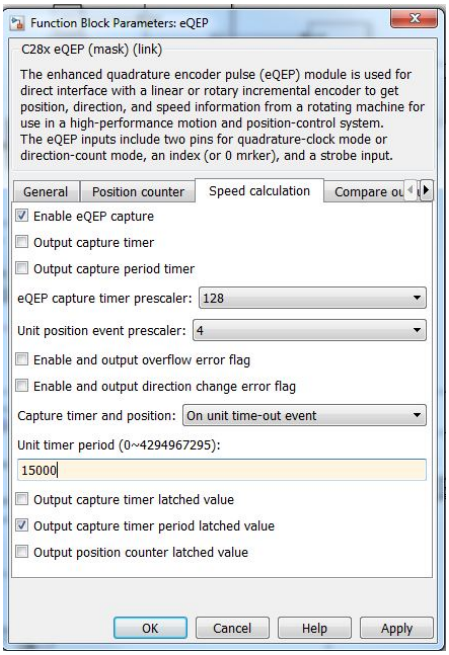

Figure 4.4: eQEP Speed Calculation

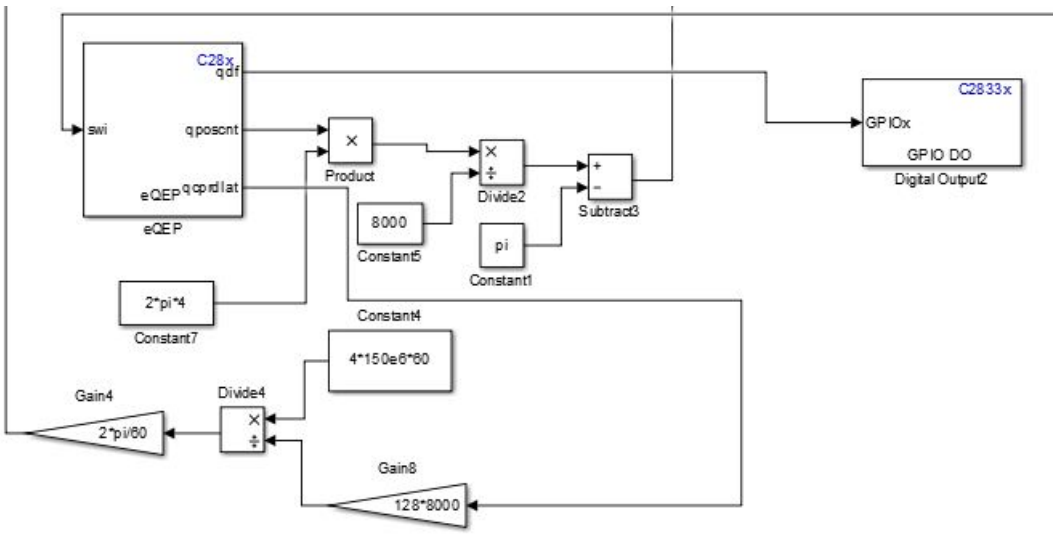

Figure 4.5: Speed and Position Calculation

The rotor position from the position counter value is obtained as follows,

$$\theta_{da} = \frac{PositionCounterValue}{8000} \times 2\pi \tag{4.2}$$

The part of the simulink model which perform's the above operation is shown in the figure 4.5

Analog to Digital Module

The analog to digital converter module obtains the analog value at the it's terminals and provides the digital value at it's result registers. In order to provide the digital value, the ADC module passes the analog signal through multiplexers, sample and hold circuit and the conversion core.

The ADC module on TI-F28335([7]) can take in values between 0V and 3V. In order to capture negative values as well an offset of 1.5V is used on the control board. The ADC input ports on the control board can take in a voltage value between plus or minus 15V. The input circuitry scales it appropriately to give the ADC input channels vales between 0-3V centered around 1.5V.

Since, the ADC module inputs are scaled and the current sensor output on the inverter is also scaled in order to obtain the right values certain mathematical operations

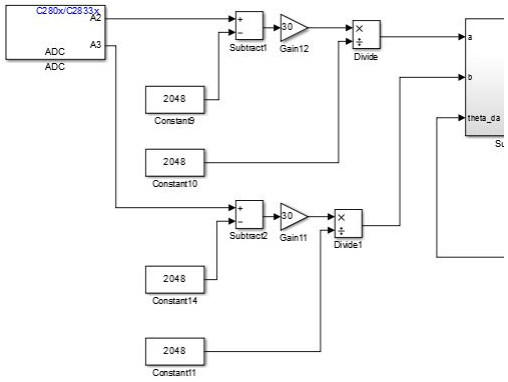

Figure 4.6: ADC Scaling

need to be performed on the signals coming out from the ADC module. On the inverter board for every one amp flowing through it a voltage of 0.5 volts is measured across the current sensor output. Taking all this into consideration, the scaling equation is given by,

$$ActualValue = [ADCModuleOutput - 2048] \times \frac{30}{2048} \tag{4.3}$$

The part of the simulink module executing this code is shown in the figure 4.6,

The ADC module is used to obtain the current information of the motor.Out of the 16 ADC channels available 2 ADC channels were used to obtain the current information. The currents for Phase A and Phase B are obtained using Channels A2 and A3 sequentially. Post interrupt at the end of each conversion is enabled.

The configuration of ADC blocks is shown in the figures 4.7 and 4.8.

PWM Module

Once the PI controllers give the DQ voltage values, the DQ to ABC transformation block provides the 3-hase voltages that need to be provided in order to run the motor at the required speed. In order for the inverter to generate these voltages appropriate switching singals need to be provided. These switching signals are obtained by generating appropriating modulating signals. These signals are then compared with a repeating sequence. Generally this repeating sequence for three phase inverter is an

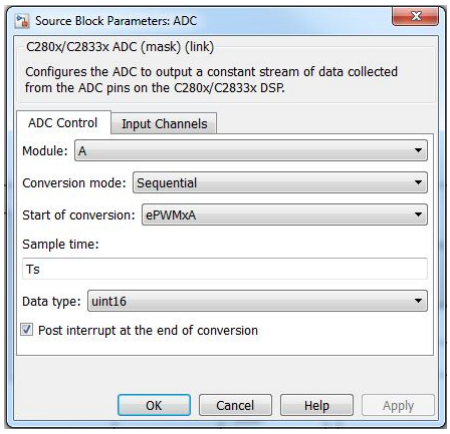

Figure 4.7: ADC COnfiguration - 1

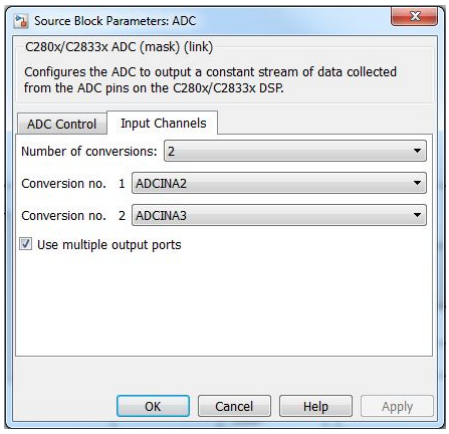

Figure 4.8: ADC COnfiguration - 2

up-dow counter. Whenever the modulating signal is greater than the carrier signal the output is a one, otherwise the output is zero. The frequency of the carrier signal represents the switching frequency of the inverter switches, whereas the frequency of the modulating signal represents the fundamental frequency of the voltage that needs to be produced at the inverter output

A single PWM module in TI-F28335([8]) represents two PWM outputs A and B. In our hardware we are using only channel A for the PWM outputs. The processor is capable of generating either up count, down count or up-down count carrier signals.

For the required switching frequency appropriate timer base period value should be set. The value can be calculated using the following formula,

$$TBPRD = \frac{1}{2} \times \frac{f_{clk}}{f_{pwm} \times CLKDIV \times HSPCLKDIV} \tag{4.4}$$

The ePWM module can be used to initiate the ADC conversion as well. This can be configured under the event trigger tab. The TBPRD value in equation 4.4 corresponds to $MaxPRD$. The counter compare A register is provided with the values of the modulating signal. The modulating signal is scaled appropriately with the TBPRD value. The counter compare value can be used to either switch on or off the PWM signal as shown in figure 4.9.

The configuration for the ePWM modules for one of the phases is shown in the

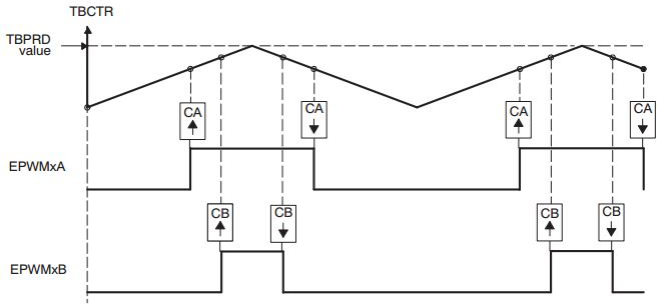

Figure 4.9: Counter Compare Operation

figures 4.10,4.11 and 4.12.

Anti Windup Integrator's and SVPWM Block

The vector control model with regular PI blocks entered into saturation mode quickly. The motor ran at the maximum voltage specified by the maximum upper and lower saturation limits of the saturation block. In order to over come this issues, anti windup compensation was used. In anti windup PI block, integrator is enabled or disabled based on the output of the saturation block present after the integrator.The anti windup PI block implemented in Simulink is shown in figure 4.13.

The integrator works on the following logic explained below. Let the output of the PI block be called u_{out}. The upper and lower saturation limits are respectively u_{+sat} and u_{-sat}.

- If, $u_{out} = u_{+sat}$ $AND > 0.0$ $Disable\ Integrator$
- $Elseif$, $u_{out} = u_{-sat}$ $AND < 0.0$ $Disable\ Integrator$
- $Else$, $Enable\ Integrator$

The space vector PWM signal is generated using a common mode signal. The common mode signal used is nothing but one half of the maximum and the minimum values among the three phase signal. The common mode signal here is a triangular waveform. A more detailed explanation for the space vector pwm technique used here is provided in Appendix B.

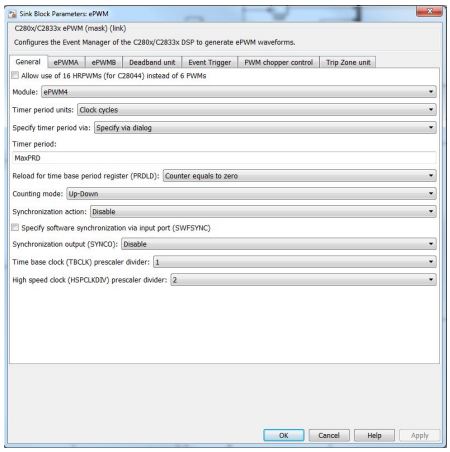

Figure 4.10: ePWM General

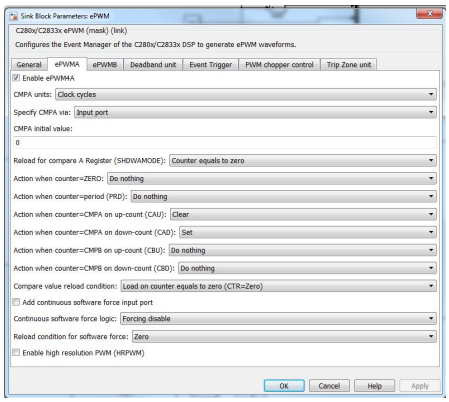

Figure 4.11: ePWM Channel A

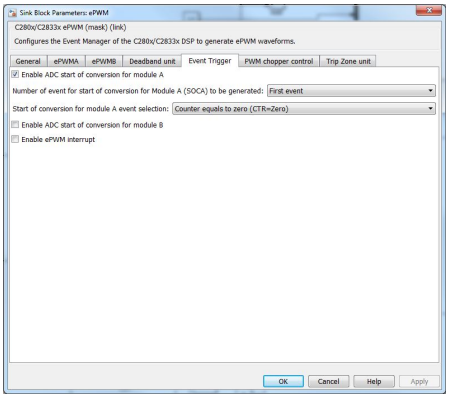

Figure 4.12: ePWM Event Trigger

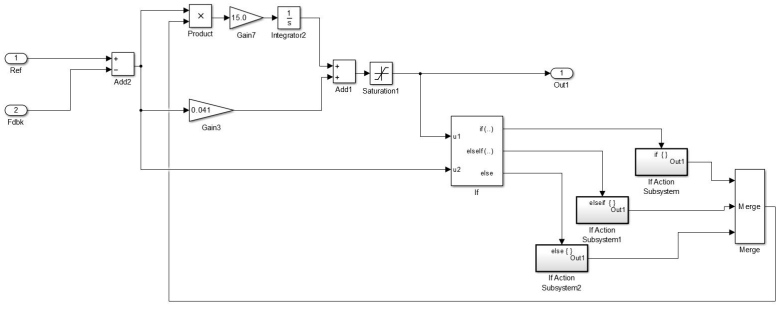

Figure 4.13: Anti-Windup PI Controller

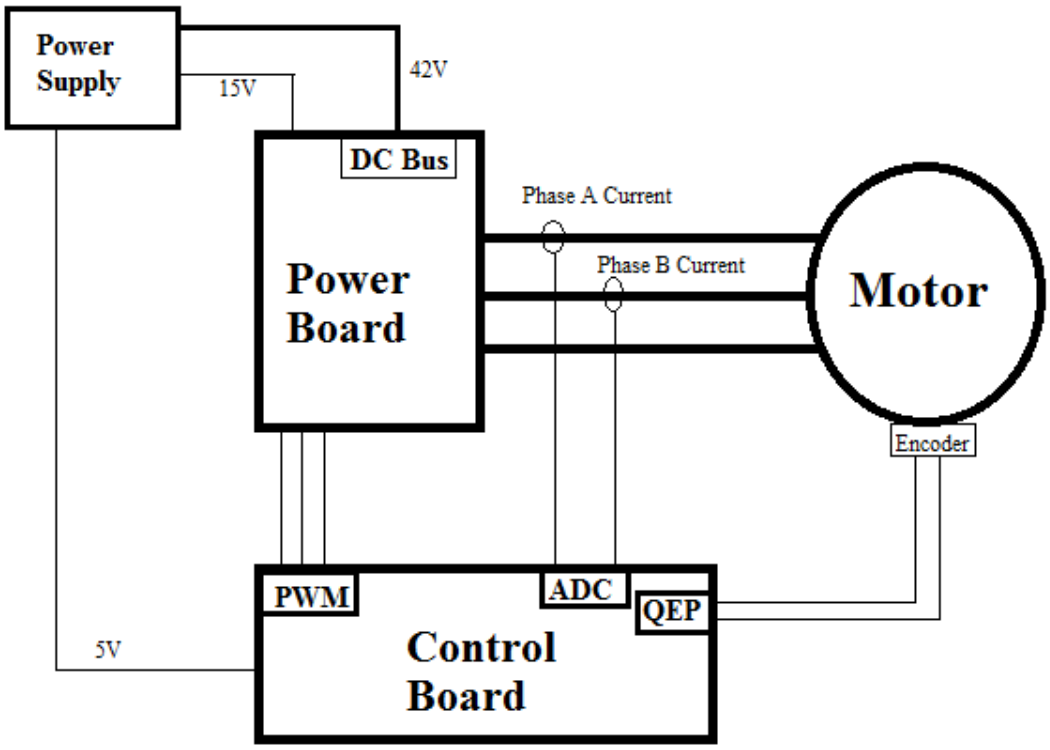

Figure 4.14: Block Diagram of the Hardware Setup

4.2 Hardware Setup

The block diagram of the hardware setup is shown in figure 4.14.

The control board provides the PWM signals to the inverter board. It also takes in current information through it's ADC channels and position and speed information through it's quadrature input modules. The three phase inverter provides the output voltage to be applied to the motor. The inverter has current sensor for two phases, these current sensors are connected to the ADC channels on the extension board. The quadrature encoder signals from the motor are connected to the quadrature input module of the processor.

Motor Parameter	Value
Rated Frequency, f	60 Hz
Stator Resistanse, R_s	0.9 Ohm
Stator Inductance, L_s	6e-3 mH
No. of Pole, P	8
Inertia, J	7.8e-5 $Kgm-2$
Back EMF Constant, k_e	0.2647 Vs

Table 4.1: Simulation Motor Parameters

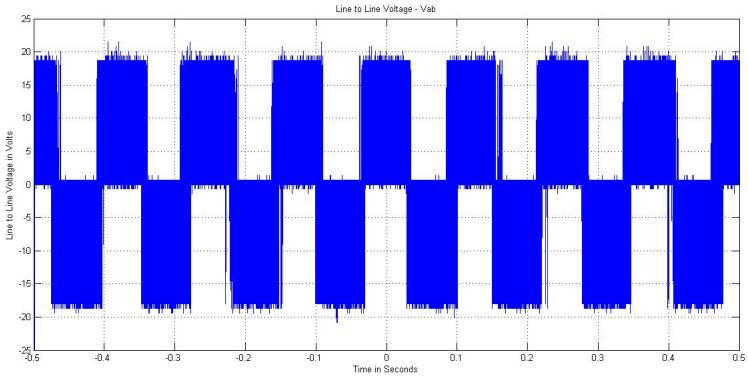

Figure 4.15: Line to Line Voltage

4.3 Hardware Results

The hardware results which include the line to line voltage, line to neutral voltage, phase current and speed is provided in the figures .The extension board for the Texas Instruments control card was developed by Siddharth Raju and Sai Priya.The results are shown in figure 4.15 and 4.16. Table 4.1 provides the motor parameters.

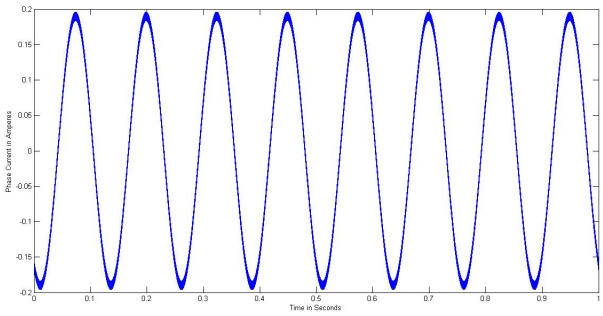

Figure 4.16: Line Current

Chapter 5

Sensorless Vector Control Using Sliding Mode Observer

A brief introduction to sliding mode control is presented. The concept of sliding mode control is used to control the motor model developed in Chapter 3. The simulation model and the simulation results are presented.

5.1 Sliding Mode Control

Let us consider a non-linear system with an m-dimensional input vector, u and n-dimensional state vector ,x given by

$$\dot{x} = f(x,t) + B(x,t)u(t) \tag{5.1}$$

The symbol t denotes the time. Let us consider *h(x,t)* an n-dimensional vector represent the disturbances in the system. Therefore equation becomes,

$$\dot{x} = f(x,t) + B(x,t)u(t) + h(x,t) \tag{5.2}$$

If,

$$h(x,t) \in span[B(x,t)] \tag{5.3}$$

then there exists a control input u such that the system is invariant to the disturbance signal *h(x,t)*. This implies that sliding mode control is independent of the disturbances. This is one of the main advantages of sliding mode control.

The control for such a system can be chosen such that,

$$u = \begin{cases} u_i^+, & \text{if } \sigma_i(x) > 0 \\ u_i^-, & \text{if } \sigma_i(x) < 0 \end{cases} \tag{5.4}$$

Where $\sigma_i(x)$ is an m-dimensional vector. By using his type of discontinuous control the sliding mode eventually occurs at the intersection of m surfaces of $\sigma_i(x)$. There might be an order reduction in the motion equations, i.e, the order m might be less than the order of the original system. The trajectory resulting from the above equations reaches the sliding surface where $\sigma_i(x) = 0$.

5.2 Sliding Mode Observer for PMSM

Sensorless control of a PMSM is used to describe the control of a PMSM machine by not measuring the main control variables which are speed and position. The terminal quantities such as phase currents and voltages are measured. The terminal quantities are used to estimate the primary control variables.

Stator currents are used as the sliding surface. Once the sliding mode happens, regardless of disturbances the estimation error becomes zero. For this method, the motor model in stationary reference frame is used instead of the rotational reference frame. The stationary reference frame does not need the motor rotor position information in order to be transformed from three phase to two phase frame.

The PMSM model in the stationary reference frame is given below,

$$L_s \frac{d}{dt} i_\alpha = -R_s * i_\alpha - e_\alpha + v_\alpha \tag{5.5}$$

$$L_s \frac{d}{dt} i_\beta = -R_s * i_\beta - e_\beta + v_\beta \tag{5.6}$$

Where,

$$e_\alpha = -\lambda_f \omega_e sin\theta \tag{5.7}$$

$$e_\beta = \lambda_f \omega_e cos\theta \tag{5.8}$$

i_α, i_β represent the stator phase currents

v_α, v_β represent the stator phase voltages

e_α, e_β represent the back EMF

R_s is the stator phase resistance

L_s is the stator phase inductance

λ_f is the rotor flux linkage of the PMSM

ω_e is the electrical angular velocity θ is the rotor position

The stator currents were chosen as the sliding surface as proposed in [9], it is given by,

$$\sigma(x) = \hat{i_s} - i_s = 0 \tag{5.9}$$

Where, $\hat{i_s}$ is the estimated value and i_s is the measured value.

For the PMSM motor model in equation 5.5 and 5.6, the sliding mode observer can be written as,

$$L_s \frac{d}{dt} \hat{i_\alpha} = -R_s * \hat{i_\alpha} + v_\alpha - k * sgn(\hat{i_\alpha} - i_\alpha) \tag{5.10}$$

$$L_s \frac{d}{dt} \hat{i_\beta} = -R_s * \hat{i_\beta} + v_\beta - k * sgn(\hat{i_\beta} - i_\beta) \tag{5.11}$$

Where k is a constant observer gain. By subtracting the above equation from equation 5.5 and 5.6, we have the mismatch dynamics given by

$$L_s \frac{d}{dt} \bar{i_\alpha} = -R_s * \bar{i_\alpha} + e_\alpha - k * sgn(\bar{i_\alpha}) \tag{5.12}$$

$$L_s \frac{d}{dt} \bar{i_\beta} = -R_s * \bar{i_\beta} + e_\beta - k * sgn(\bar{i_\beta}) \tag{5.13}$$

Where $\bar{i_\alpha} = \hat{i_\alpha} - i_\alpha$ and $\bar{i_\beta} = \hat{i_\beta} - i_\beta$ denote the observation errors. Here, the back emf components are the disturbance parameters. But from previous section we know that they are bounded, hence they can be suppressed by the discontinuous inputs. The value of the constant observer gain k is critical to achieve sliding mode. By choosing,

$$k > max(|e_\alpha|, |e_\beta|) \tag{5.14}$$

sliding mode can be achieved and the stability of the system ensured. This can be proved using Lyapunov equations[10].

During the sliding mode, $\bar{i_\alpha} = 0$ and $\bar{i_\beta} = 0$. Therefore setting the differentials of these terms to zero, from equations 5.12 and 5.13 we have,

$$e_\alpha = k * sgn(\bar{i_\alpha}) \tag{5.15}$$

$$e_\beta = k * sgn(\bar{i_\beta}) \tag{5.16}$$

The obtained back emf parameters should be passed through a low pass filter in order to filter out the high frequency components. The cutoff frequency should be designed appropriately.

5.3 Position and Speed

Once the disturbance parameter, back emf is obtained. It can be used to obtain the required speed and position information. The position signal is estimated as,

$$\hat{\theta} = -arctan\frac{e_\alpha}{e_\beta} \tag{5.17}$$

Since, there is high frequency components involved and filters are used. The θ value should be compensated for shift in phase. A phase locked loop(pll) can be used to estimate the speed. The back emf equations from equation 5.15 and 5.16 are converted from their stationary reference frame to the rotational reference frame(dq) using the position information from equation 5.17.Under steady state conditions, the d axis component of ht eback emf must be zero, hence the estimated speed can be given by the following equation,

$$\omega_(est) = \frac{E_q - sgn(E_d) * E_d}{\lambda_f} \tag{5.18}$$

An alternative approach would be to use the magnitude of the back emf in the stationary reference frame and divide it by λ_f. This would provide the speed in steady state. The equation is given by,

$$\omega_(est) = \frac{\sqrt[2]{e_\alpha^2 + e_\beta^2}}{\lambda_f} \tag{5.19}$$

5.4 Simulation in Matlab And Results

The vector control model with rotor flux lagging the phase a axis by 90 degrees is used to perform the simulation. The motor is started in open loop by giving external position command until it reaches the rated speed. The position command is a ramp signal starting from -π/2. The simulation model, the observer model and the corresponding

results are provided below. The SMO observer gain, k was chosen to have a value of 1000, smaller value would lead to inaccurate estimation of current values. Larger value would lead to longer settling time in case change in load parameters. The chosen back emf value is larger than the magnitude of the back emf value. The motor was started with all the initial conditions set to zero.

The simulation model and the results are provided in the following pages.

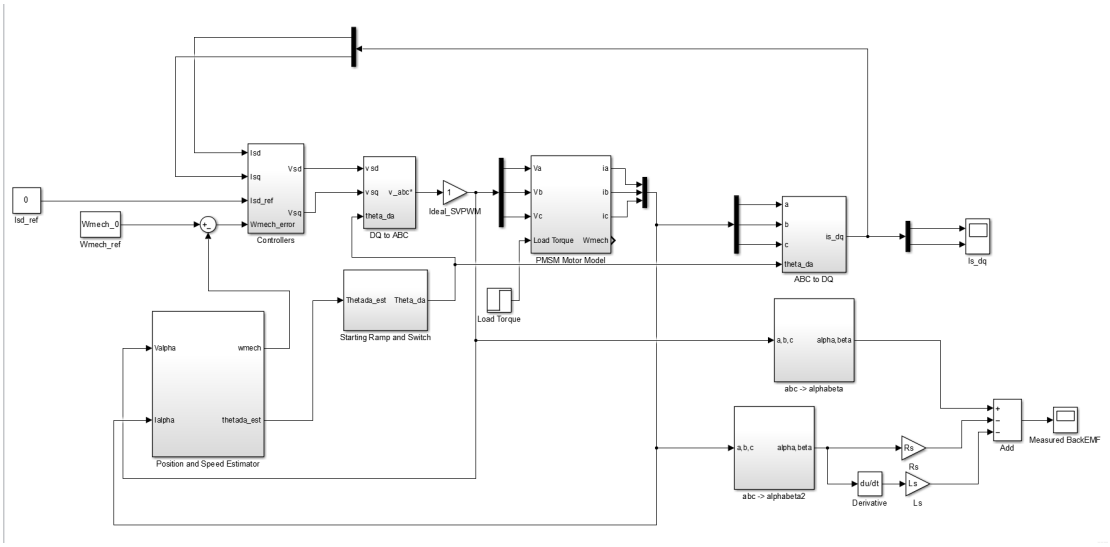

Figure 5.1: Sensorless Vector Control Model

The position and speed estimator model are provided below,

As can be seen from the various plots, the inrush current is high and hence during hardware implementation appropriate limits must be used. Though the D-axis current from the estimated position contains disturbance, over a long run it settles down to zero. As we can see the amount of time taken to reach zero is very less. The estimated stator currents and the back emf from the observer in the stationary reference frame follow the measured quantities very well, indicating the robustness of the model even in the presence of change in load torque.

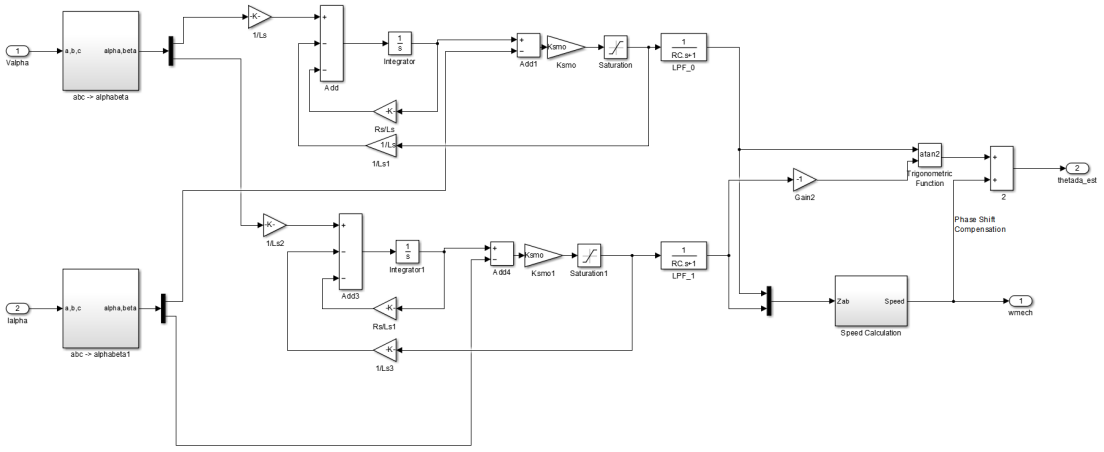

Figure 5.2: Position and Speed Estimation

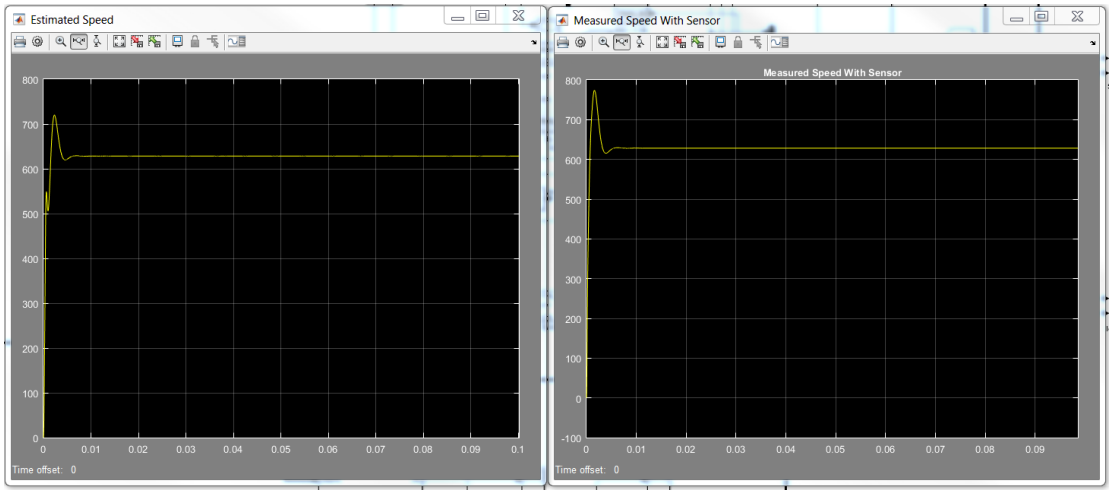

Figure 5.3: Estimated and Measured Speed

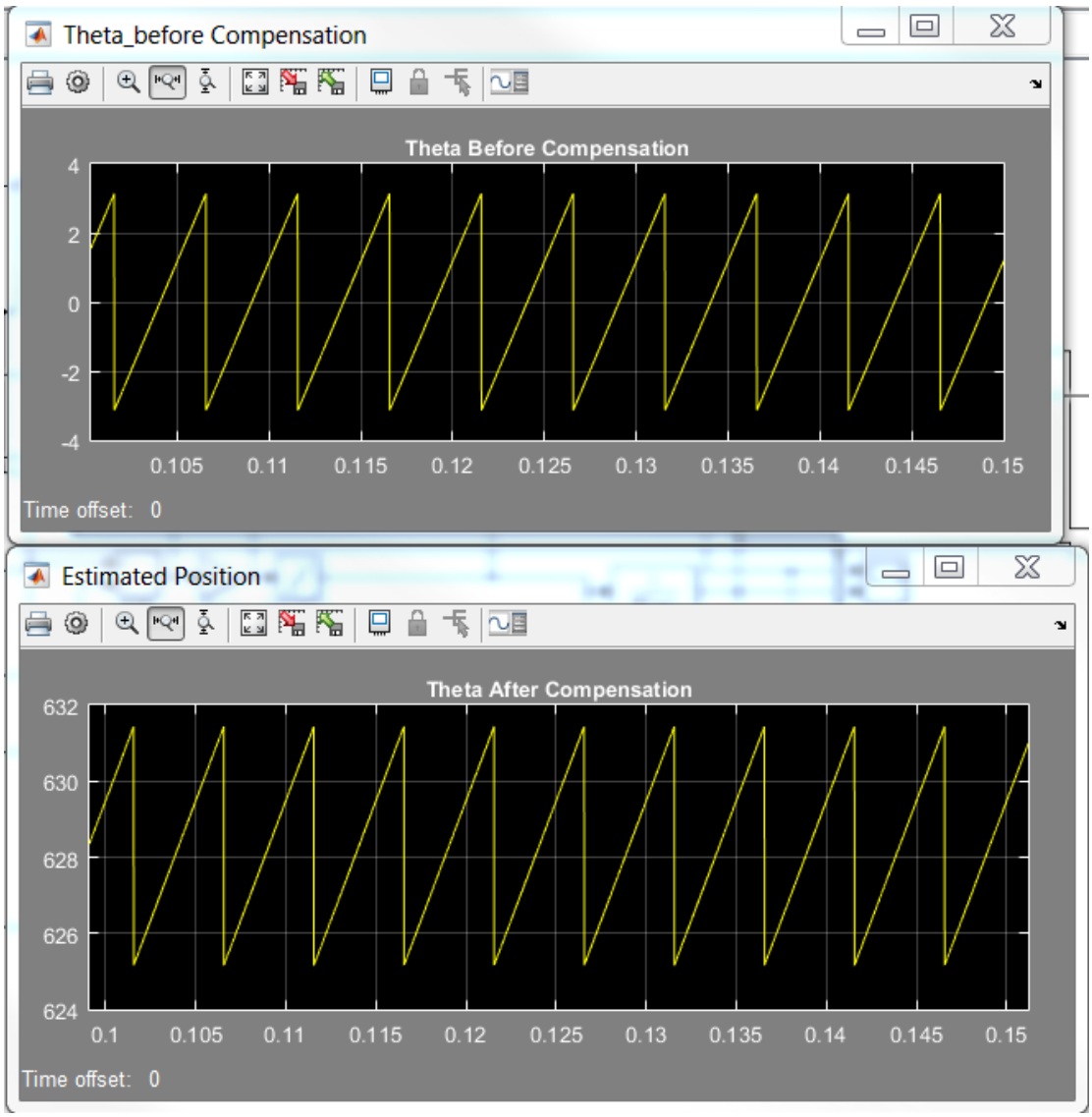

Figure 5.4: Theta Before And After COmpensation

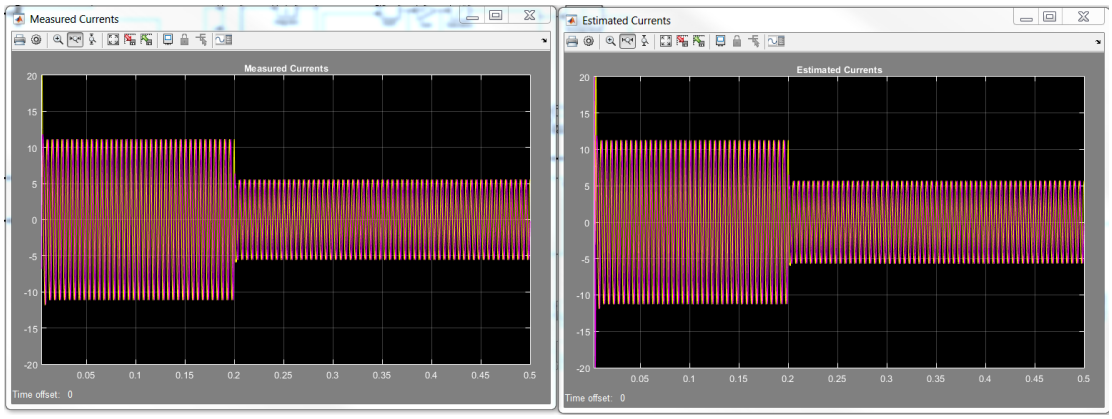

Figure 5.5: Measured and Estimated Currents

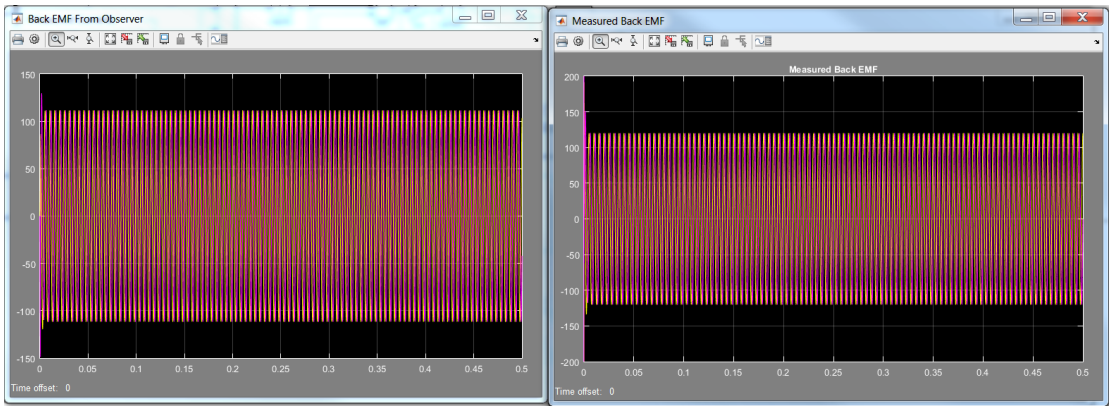

Figure 5.6: Measured Back EMF and Back EMF from Observer

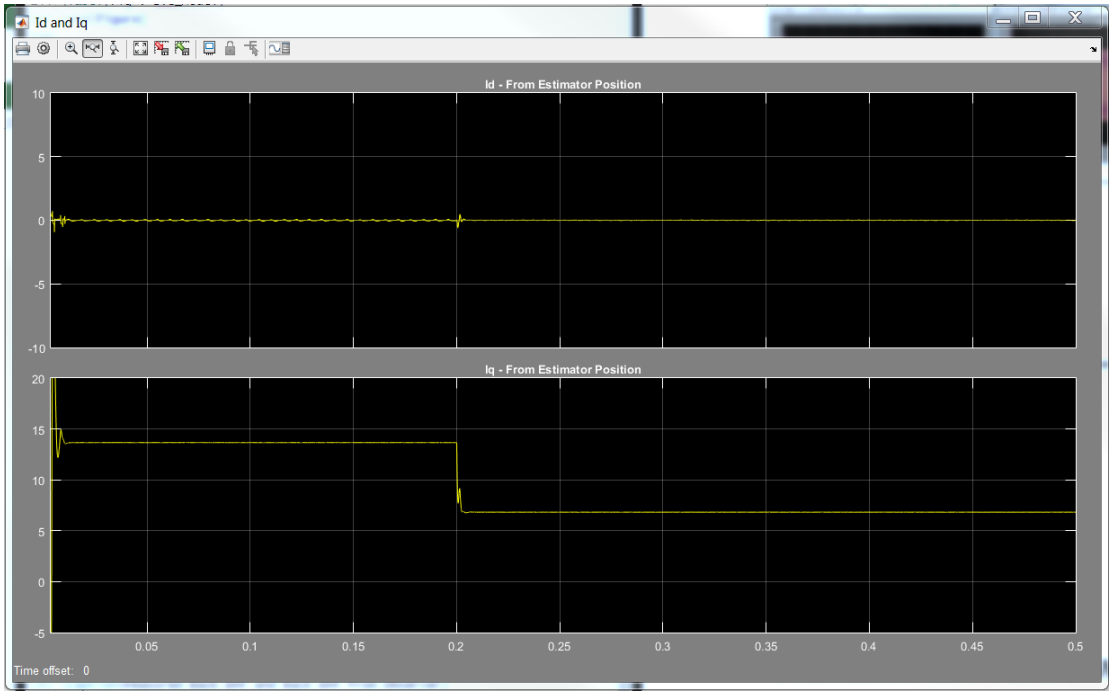

Figure 5.7: D and Q Axis Currents from Estimated Position

Chapter 6

Conclusion and Future Work

In this thesis work the general objectives mentioned in section 1.2 is achieved. Embedded coder successfully accelerates the design process for power and motion control applications. The hardware used TI-F28335, a floating point processor had the capabilities to perform complex mathematical operations without any issues. Vector control of PMAC motor was performed using the proposed hardware and software.A simulation model for sensorless control of PMSM using sliding mode observer was presented, which can be implemented in hardware. The hardware implementation requires more tuning of the filters and the observer gain.

There were some issues with the ADC's.Which contributed to the deviation in the current results, which were not as expected. A better compensation circuit for the ADC inputs was designed for the next version of the extension board by Siddharth Raju. This will enable better sensing capabilities and as well a better match in results to the one obtained in hardware.Also, there was an issue with the PI controller which was overcome by using anti windup compensation.

The present work was done without monitoring the signals real time on the computer. All the results that were obtained were through external measurements using equipment's such as oscilloscope, tachometer etc. By using a faster JTAG and software versions compatible for real time hardware in the loop operation data can be monitored directly on the computer.Also, the processor used here has the capability to control multiple systems(either motor's or power electronic circuit's) at the same time. Finally, the motor can be ran up to it's rated speed and operated with different space vector

pulse width modulation strategies studied in Chapter B in order to operate the machine at reduced harmonic distortion.

References

[1] Di Zhao, Hari, Gopalaratnam Narayanan, and Rajapandian Ayyanar. Space-Vector-based hybrid pulsewidth modulation techniques for reduced harmonic distortion and switching loss. *Power Electronics, IEEE Transactions on*, 25(3):760–774, March 2010.

[2] Mathworks. Files and folders created by build process - MATLAB & simulink.

[3] Ned Mohan. *Advanced Electric Drives: Analysis, Control and Modeling Using Simulink*. Mnpere.

[4] Ned Mohan. *Electric Machines and Drives*. Wiley, 1 edition, January 2012.

[5] Texas Instruments. TMS320F28335 — delfino f2833x.

[6] Texas Instruments. TMS320x2833x, 2823x enhanced quadrature encoder pulse eQEP - sprug05a.pdf.

[7] Texas Instruments. TMS320x2833x Analog-to-Digital converter (ADC - spru812a.pdf.

[8] Texas Instruments. TMS320x2833x, 2823x enhanced pulse width modulator (ePWM) reference guide (rev. a) - sprug04a.pdf.

[9] Vadim Utkin, Juergen Guldner, and Jingxin Shi. *Sliding Mode Control in Electro-Mechanical Systems*.

[10] Song Chi and Longya Xu. Position sensorless control of PMSM based on a novel sliding mode observer over wide speed range. In *Power Electronics and Motion*

Control Conference, 2006. IPEMC 2006. CES/IEEE 5th International, volume 3, pages 1–7. IEEE, August 2006.

[11] G. Narayanan and V. T. Ranganathan. Analytical evaluation of harmonic distortion in PWM AC drives using the notion of stator flux ripple. *Power Electronics, IEEE Transactions on*, 20(2):466–474, March 2005.

Appendix A

Getting Started With Embedded Coder

In this appendix a brief introduction to setup and run embedded coder on TI-F28335 processor is shown. The user can load the program on to the processor in two ways. One is through Matlab directly and the other way is by using the generated files in Code Composer Studio, the official Texas Instruments IDE for C2000 class processors. By going through this appendix the user will know how to code the processor directly from Matlab. The process of loading the program to the processor is provided in brief steps.

Let us take the example of open loop control of DC motor provided in Chapter 3. Once the model is ready as shown in Chapter 3 the user can use the steps mentioned below to successfully load the program on to the processor. For reference the open loop DC motor model is shown here again in figure A.1.

- Select code generation, click on browse to choose the $SystemTargetfile$. The default option will be grt.tlc, change it to ert.tlc as shown in figures A.2 and A.3.

- The default options once ert.tlc will be as shown in figure A.4. Select the Target Hardware as TI Delfino F2833x. Select the Toolchain for the appropriate Code Composer Studio available on your computer.

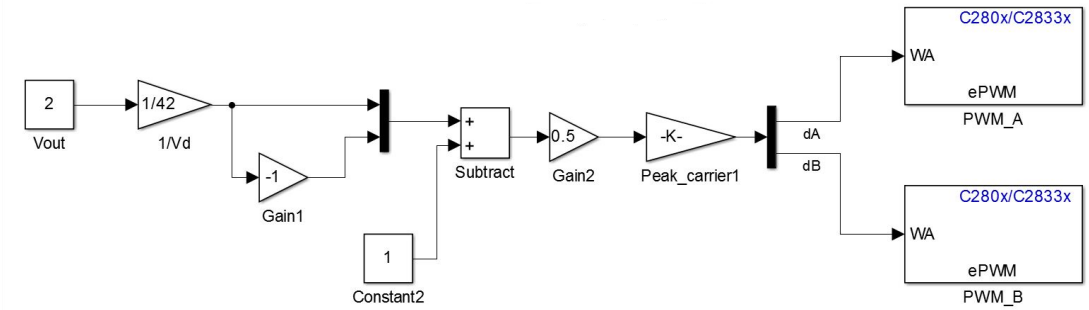

Figure A.1: Code Generation Model

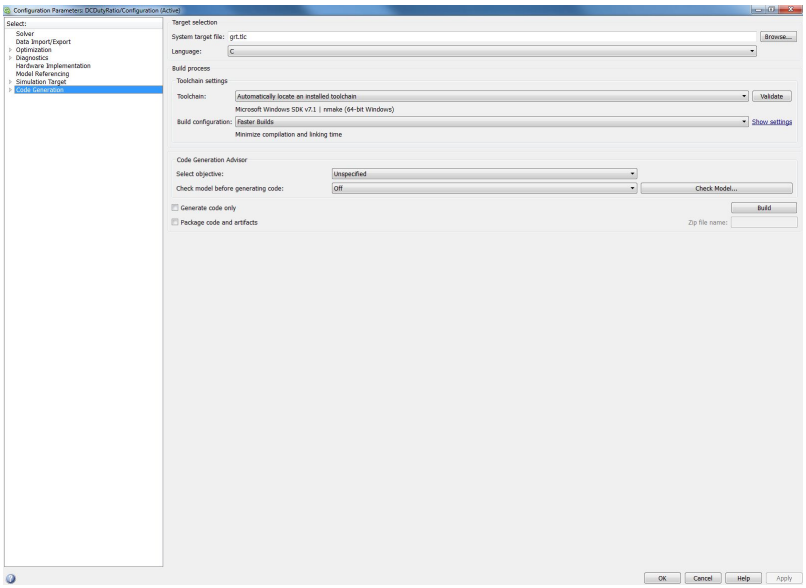

Figure A.2: Step 1 - 1

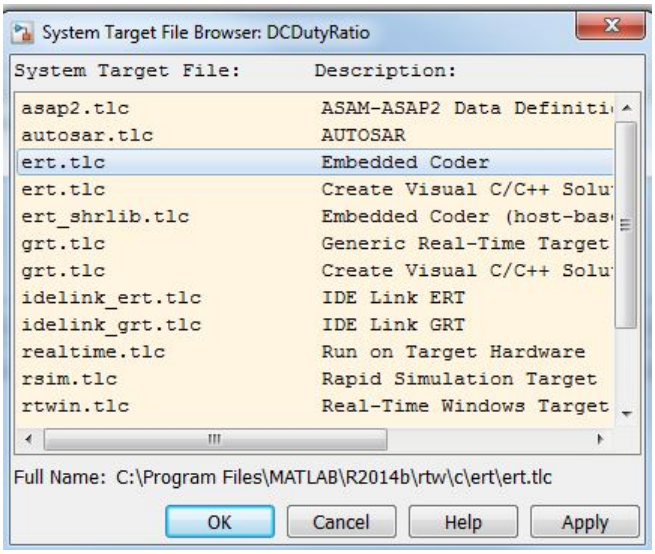

Figure A.3: Step 1 - 2

- Once that is done, the solver automatically sets the type to be fixed and the solver used will be discrete. The fundamental sample size will be auto and the user can set it to required value. (Note : The fundamental sample size should be a multiple of the sample size used in the Simulink blocks if not rate transition blocks need to be used). Also, under interface pane enable continuous states if continuous blocks in Simulink are used.(Figure A.5)

- Under Code Target, the build action needs to be changed from Build to Build, Load and run in order to run the program directly from Matlab. Select the device name as F28335. Click Apply.(Figure A.6)

- Press Ctrl+B or Click on load on to hardware. Once that is done the program is loaded on to the hardware and the required task is executed.

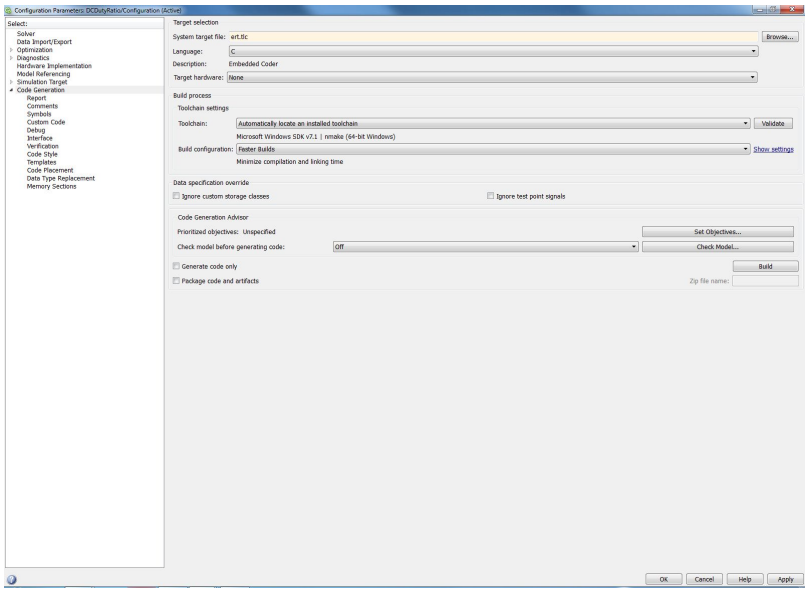

Figure A.4: Step 2

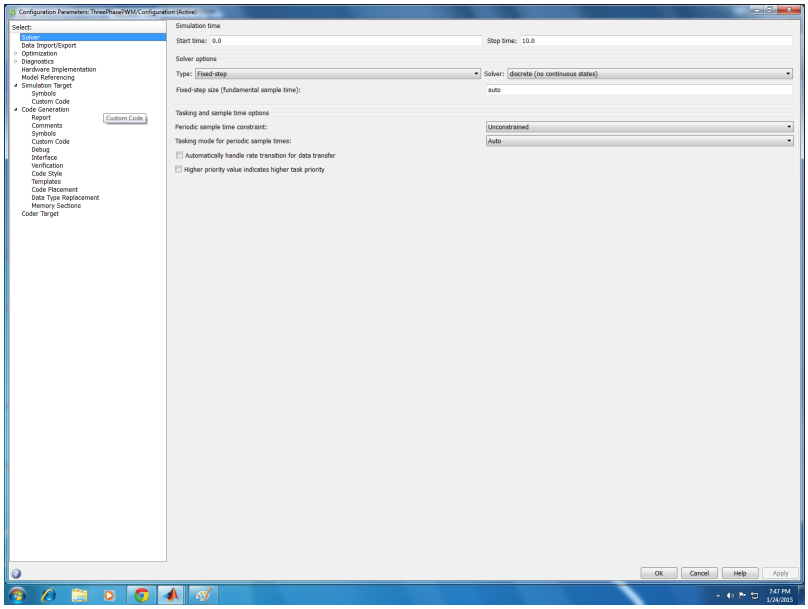

Figure A.5: Step 3

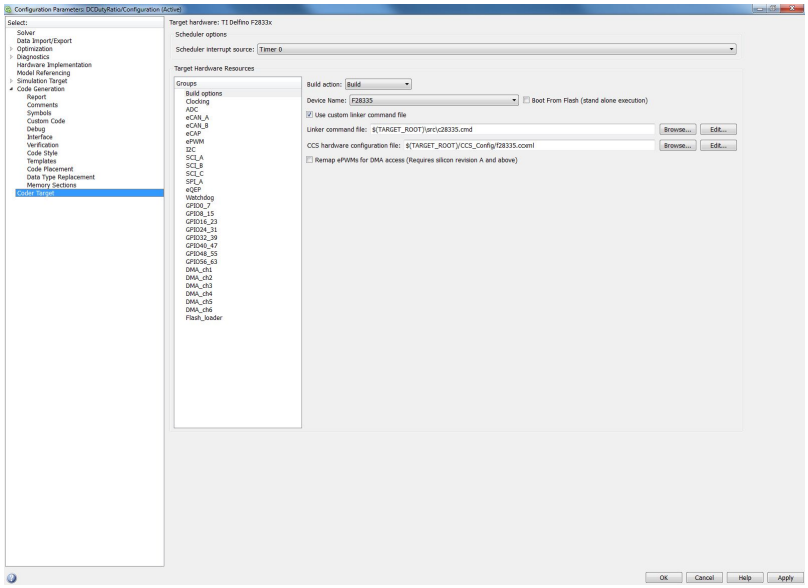

Figure A.6: Step 4

Appendix B

Space Vector PWM Techniques for Reduced Harmonic Distortion in AC Machines

There are various space vector pulse width modulation techniques studied and proposed to reduce the harmonic distortion in A.C. Machines. Among them, three space vector pwm techniques have been studied in [1], one of them is a new strategy proposed by the author to reduce harmonic distortion in A.C Machines.

The three space vector pwm techniques used are,

- Conventional Space Vector PWM (CSVPWM)
- Split Clamp Space Vector PWM(SCSVPWM)
- Advanced Split Clamp Space Vector PWM(ASCSVPWM)

The duty ratio waveforms for the three mentioned space vector pulse width modulation techniques are shown in figure B1.

The switching sequences are provided in table B1. There are six sectors, for SCSVPWM and ASCSVPWM each sector is further divided into two equal parts each 30 degree's. For a given revolving reference voltage vector of magnitude Vref and angle alpha as shown in figure B2, the active vector 1 is applied for T_1 seconds, active vector 2 for T_2

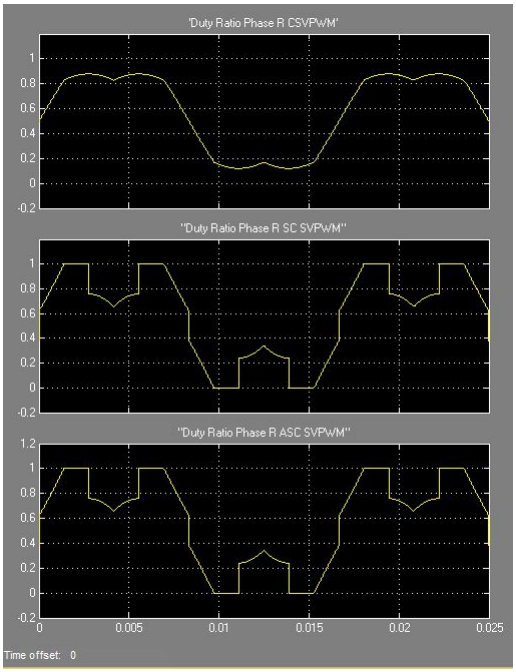

Figure B.1: Duty Ratio Waveforms

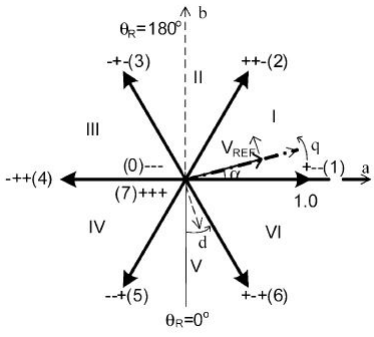

Figure B.2: Sector Diagram

Sector	Conventional	Split Clamp	Advanced Split Clamp
I	(0127,7210)	(012,210),(721,127)	(0121,1210),(7212,2127)
II	(7230,0327)	(723,327),(032,230)	(7232,2327),(0323,3230)
III	(0347,7430)	(034,430),(743,347)	(0343,3430),(7434,4347)
IV	(7450,0547)	(745,547),(054,450)	(7454,4547),(0545,5450)
V	(0567,7650)	(056,650),(765,567)	(0565,5650),(7656,6567)
VI	(7610,0167)	(761,167),(016,610)	(7616,6167),(0161,1610)

Table B.1: Switching Sequences

seconds and zero state vectors for T_z seconds. The expressions for T_1, T_2 and T_z are give below,

$$T_s = T_1 + T_2 + T_z \tag{B.1}$$

$$T_1 = \frac{V_{ref}}{V_{dc}} \frac{sin(60 - \alpha)}{sin(60)} T_s \tag{B.2}$$

$$T_2 = \frac{V_{ref}}{V_{dc}} \frac{sin(\alpha)}{sin(60)} T_s \tag{B.3}$$

$$T_z = T_s - T_1 - T_2 \tag{B.4}$$

Conventional Space Vector PWM

This is the most commonly and most widely used space vector pwm method. In this method, the zero state time is split equally between T_0 and T_7, the zero state vectors. T_1 is the amount of time for which active vector 1 is applied and T_2 is the duration for which active vector 2 is applied.

In order to obtain the waveform, the sector can be determined and then the corresponding duty ratio equations can be provided. The duty ratio equations for Sector 1 are provided below for all three phases,

$$d_R = \frac{0.5T_z + T_1 + T_2}{T_s}$$

$$d_Y = \frac{0.5T_z + T_2}{T_s}$$

$$d_B = \frac{0.5T_z}{T_s}$$

A common mode signal can also be used to generate the conventional space vector pwm. The common mode signal here is given by

$$m_{cm} = -0.5(m_{max} + m_{min})$$

This method is used to generate pwm signal in our model for vector control, since our model is running in the range where CSVPWM provides lower harmonic distortion compared to the other two methods. CSVMPW ensures a lower harmonic distortion in the frequency range 0-40.8Hz compared to the other the SVPWM methods mentioned.

Split Clamp Space Vector PWM

This method clamps one of the phases for a duration of 60 degrees. Also, it uses the same zero state for the entire duration of T_z unlike CSVPWM. Sector 1 is divided into two equal parts and in each part the zero state used is different. The duty ratio equations for sector 1 are provided below. The equations for first half of sector one are,

$$d_R = \frac{T_1 + T_z}{T_s}$$

$$d_Y = \frac{T_2}{T_s}$$

$$d_B = 0$$

The equations for second half of sector one are,

$$d_R = 1$$

$$d_Y = \frac{T_2 + T_z}{T_s}$$

$$d_B = \frac{T_z}{T_s}$$

The sampling time T_s here is one-third the switching frequency. For frequencies in the range 40.8-56.4Hz this method produces lower harmonic distortion.

Advanced Split Clamp Space Vector PWM

Even this method uses the same zero state for T_z seconds. But it divides the active vector time into two equal halves. This involves switching a phase twice, switching the second phase once and the third phase is clamped. At frequency in the range 56.4-60Hz this method produces the least harmonic distortion compared to the other two methods. The sampling time T_s here is one half times the switching frequency. The duty ratio equations for sector 1 are provided below.

The equations for first half of sector one are,

$$d_R = \frac{T_1 + T_z}{T_s}$$

$$d_Y = \frac{T_2}{T_s}$$

$$d_B = 0$$

The equations for second half of sector one are,

$$d_R = 1$$

$$d_Y = \frac{T_2 + T_z}{T_s}$$

$$d_B = \frac{T_z}{T_s}$$

For calculation and simulation purposes the equations and waveforms are similar to SCSVPWM.

Above figure from [1] provides a plot of harmonic distortion in the three space vector pwm techniques mentioned. The plot is for 50Hz it could b scaled for 60Hz, [1]. The

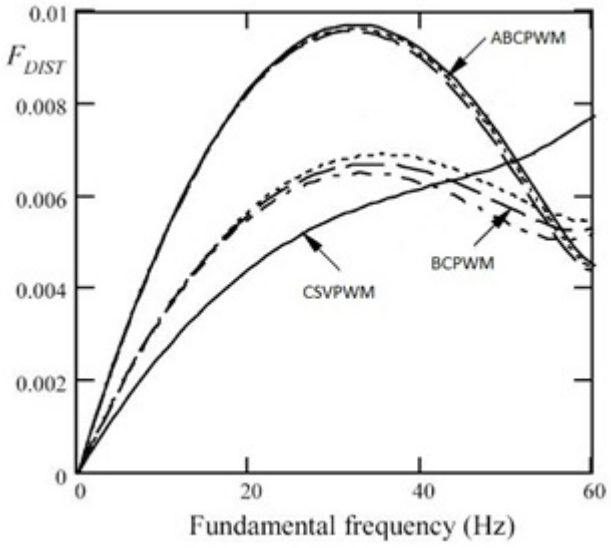

Figure B.3: Harmonic Distortion vs Frequency

result here has been obtained analytically by using the stator flux as explained in the paper [11].

Appendix C

Hardware Setup

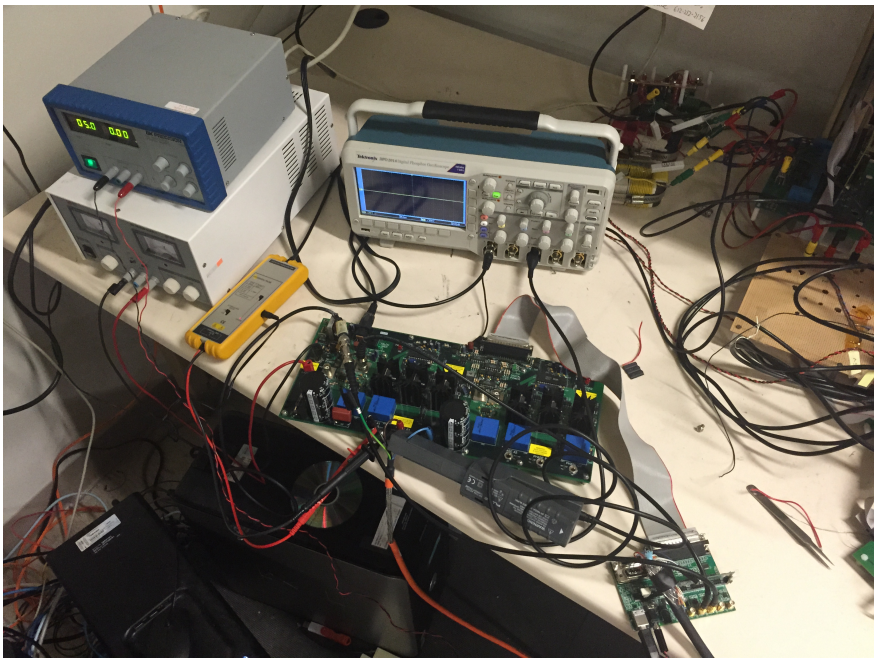

Figure C.1: Hardware Setup

Figure C.2: PMAC Motor

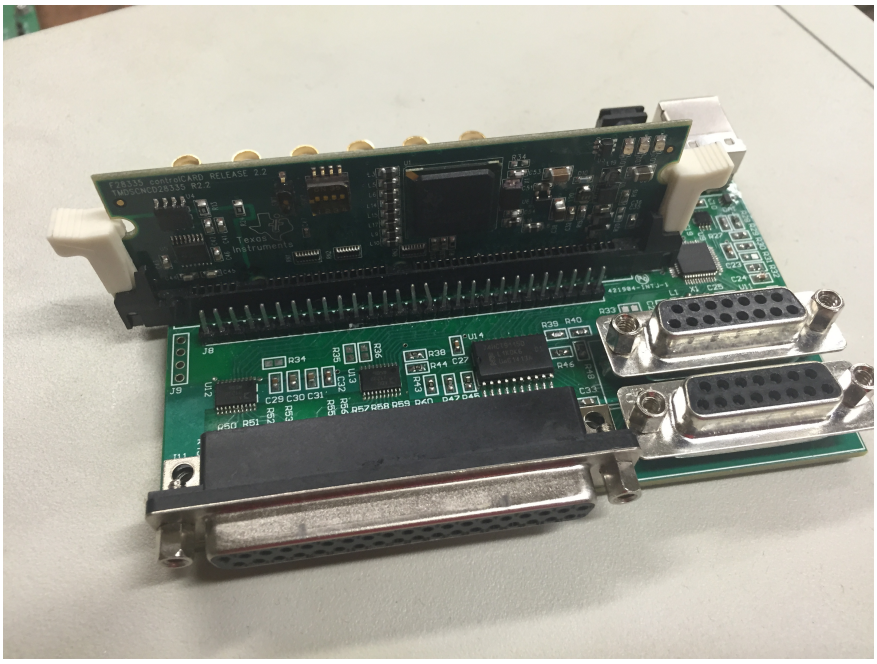

Figure C.3: Control Hardware

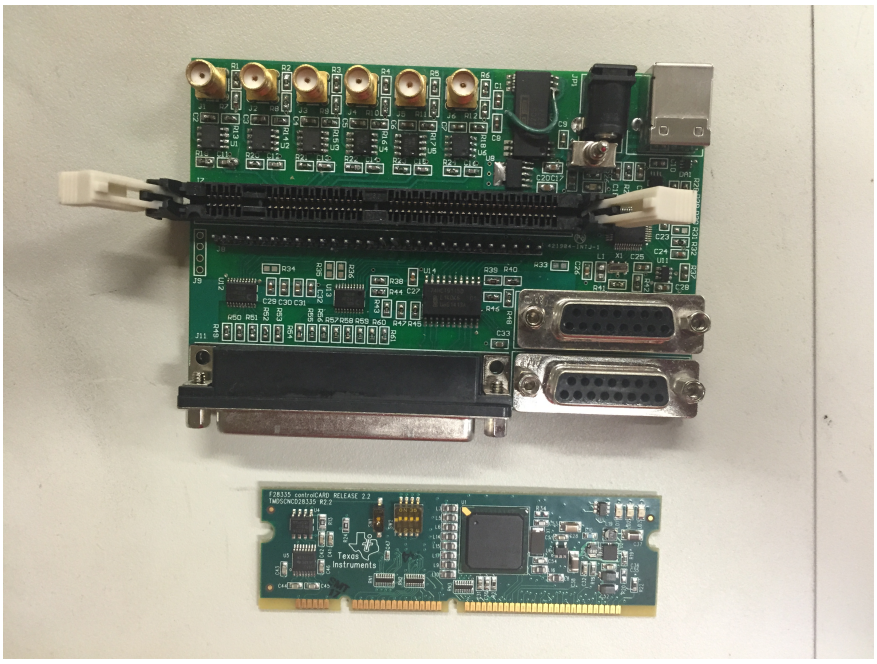

Figure C.4: Control Card And Extension Board