CHALMERS

Vector Control of an Induction Motor based on a DSP

Master of Science Thesis

QIAN CHENG
LEI YUAN

Department of Energy and Environment
Division of Electric Power Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
Göteborg, Sweden 2011

Vector Control of an Induction Motor based on a DSP

QIAN CHENG
LEI YUAN

Department of Energy and Environment
Division of Electric Power Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
Göteborg, Sweden 2011

Vector Control of an Induction Motor based on a DSP

QIAN CHENG
LEI YUAN

Department of Energy and Environment
Division of Electric Power Engineering
Chalmers University of Technology
SE–412 96 Göteborg
Sweden
Telephone +46 (0)31–772 1000

Chalmers Bibliotek, Reproservice
Göteborg, Sweden 2011

Vector Control of an Induction Motor based on a DSP

QIAN CHENG
LEI YUAN
Department of Energy and Environment
Division of Electric Power Engineering
Chalmers University of Technology

Abstract

In this thesis project, a vector control system for an induction motor is implemented on an evaluation board. By comparing the pros and cons of eight candidates of evaluation boards, the TMS320F28335 DSP Experimenter Kit is selected as the digital controller of the vector control system. Necessary peripheral and interface circuits are built for the signal measurement, the three-phase inverter control and the system protection. These circuits work appropriately except that the conditioning circuit for analog-to-digital conversion contains too much noise. At the stage of the control algorithm design, the designed vector control system is simulated in Matlab/Simulink with both S-function and Simulink blocks. The simulation results meet the design specifications well. When the control system is verified by simulations, the DSP evaluation board is programmed and the system is tested. The test results show that the current regulator and the speed regulator are able to regulate the stator currents and the machine speed fast and precisely. With the initial values of the motor parameters there is a 12.5% overshoot in the current step response. By adjusting the stator resistance and the inductance this overshoot could be removed and only minor difference remains between the simulated and measured current step responses.

Index Terms: Induction Motor, Inverter, 3-phase PWM, DSP, Vector Control

Acknowledgements

The deepest thanks to our supervisor, Dr. Stefan Lundberg, for the valuable guidance and suggestion he has given us in this project. We also want to thank Prof. Torbjörn Thiringer for introducing this master thesis topics. We would also like to thank Division of Electric Power Engineering for the technical support.

Qian Cheng
Lei Yuan
Göteborg, Sweden, 2011

Contents

Chapter 1

Introduction

1.1 Background

AC motors(Alternating Current motors) have a simple structure and has higher efficiency than the DC motors(Direct Current motors) when operated at high speed [2]. AC motors also provide more durable service with a lower cost compared to DC motors [24]. Besides, AC motors can easily be supplied directly from the grid and therefore they are widely used in industry. However, when it is directly connected to the grid, traditional AC motor gets the better performance only at the speed near the synchronous speed due to its working principle [2]. In some low cost applications where only DC is available, e.g. in electrical vehicles, the usages of AC motors are limited.

For a variable-speed application or if the power source is DC, a variable-speed driver is needed, which is usually realized with a three-phase inverter(DC-to-AC converter). With feedback or estimation of the field orientation, AC motors can be modeled as a separately magnetized DC motor through a series of coordinate transformations based on the orientation. For the transformed AC motor model, DC motor control methods can be applied, which can obtain good transient performance and eliminate oscillation in the produced torque [37]. This method of controlling the transformed AC motor is called vector control.

Mostly, an embedded microcontroller governs the overall operation of AC motor controller. Among various microcontrollers, Digital Signal Processor(DSP) can perform complicated computation, such as coordinate transformations, field estimation algorithms and controller algorithms. Before implementing the vector control system on the microcontroller, the control algorithm could be verified with the help of the simulation softwares, such as Matlab/Simulink.

1.2 Purpose and Objectives

The task is to select an evaluation board according to the specifications and to design and implement the vector control system on the board for an induction machine. Also, documenting the implementation and commenting the program codes are parts of the task, which might make it easier to use or rebuild the vector control system. The electrical interface of the evaluation board should be modified to meet the standard of the dSPACE DS1103 system, because the systems must be exchangeable according to the specifications. The final result of this project should be a vector control system of an induction machine based on an evaluation board. The vector control system will contain three main parts, transformations, controllers and the filed estimator. Eventually, the mature control system might play a role in the experiment of master students or Ph.D. students at the *Division of Electric Power Engineering* at *Chalmers*.

1.3 Scope and Methods

This project covers several academic areas, such as computer architecture, programming, debugging, analog and digital circuit design and construction, induction machine modeling and vector control theory with the representation of three-phase quantities in the complex plane.

For the selection of an appropriate evaluation board, various data sheets from different semiconductor companies, webpages and academic articles are studied. The relevant information for the fulfillment of the specifications are collected and presented. Based on this the most suitable board is selected. The electrical interfaces, analog and digital, are designed according to the application circuits in the selected components' data sheets. The vector control system is first designed on-paper as a continuous time system and it is verified with the simulation using Matlab/Simulink. Before the control system is implemented on the evaluation board the interfaces of the board are tested, such as the three-phase PWM, Analog-to-Digital conversion and so on. To verify the implementation of the coordinate transformations and the field estimation an open-loop volts per hertz(V/f) controller is first implemented on the evaluation board. After debugging and checking the implementation the vector current regulator is added and finally the speed regulator.

Chapter 2

Induction Machine Vector Control System Description

In this chapter a brief description of induction machine control system is given, in Fig. 2.1 for a system overview. The induction machine vector control system includes hardware devices, such as induction machine and inverter, and technical theory, for instance vector control strategy and Pulse Width Modulation (PWM). In the latter chapters more details about the different parts will be given.

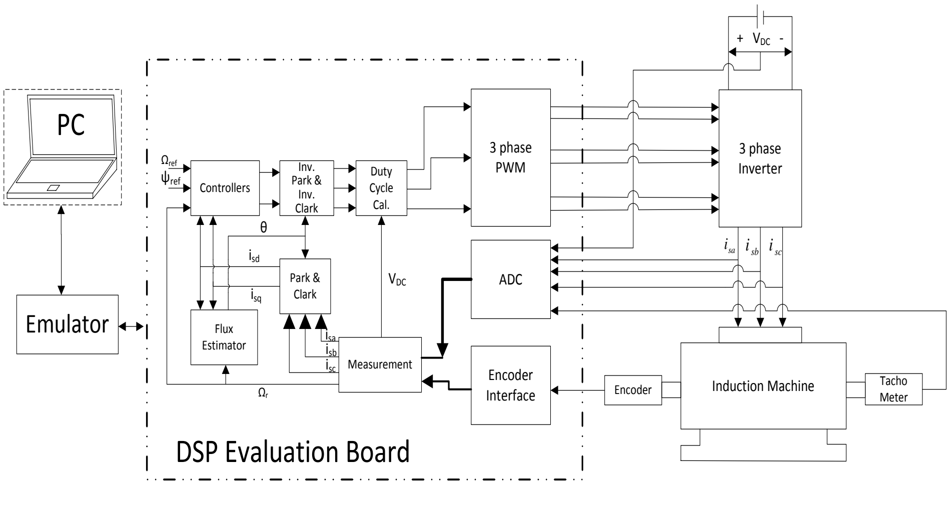
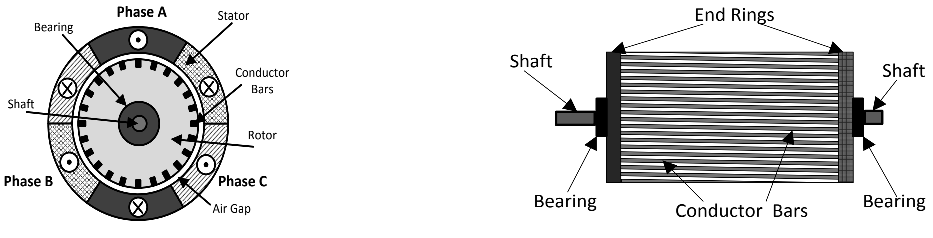


Fig. 2.1 Overview of induction machine vector control system.



(a)

(b)

Fig. 2.2 Structure of an induction machine

2.1 Induction Machine

Induction machine can also be called asynchronous machine, which is the object to be controlled in this project. The induction machine is composed of a stator and a rotor as shown in Fig. 2.2(a). The stator is fixed while the rotor rotates inside with a small air gap between them. The rotors of almost 90% induction machine are squirrel cage rotors [9], which contain conductor bars and end rings as shown in Fig. 2.2(b). The rotor cage is a closed conductor as the conductor bars are all short-circuited by the end rings. The rotor is mounted on the shaft with two bearings [9]. Normally one of the two shaft ends is used for driving the load while the other one for mounting the shaft position or the speed measurement devices. The stator is made up of several pole pairs. Each pole pair has three windings placed symmetrically in space as illustrated in Fig. 2.2(a). Based on the arrangement of the three windings, the flux linkages($\vec{\Psi}_A$, $\vec{\Psi}_B$ and $\vec{\Psi}_C$)generated by the windings have 120°displacement in space. As shown in Fig. 2.3(a),$\vec{i}_A$, $\vec{i}_B$ and $\vec{i}_C$ are the unit vectors which stand for the orientations of the fluxes generated by each winding. The magnitudes of flux linkages(Ψ_A, Ψ_B and Ψ_C) could be expressed as

$$\Psi(t) = \int (u_s(t) - Ri_s(t))dt,$$

where $u_s(t)$ is the applied voltage to each winding, R is the resistance of each winding and $i_s(t)$ is the current through each winding. If a three-phase voltage is applied on the stator windings, i.e.

$$\begin{aligned} u_{sa}(t) &= U_{pk} sin(360ft + 90°) \\ u_{sb}(t) &= U_{pk} sin(360ft + 210°) \\ u_{sc}(t) &= U_{pk} sin(360ft + 330°), \end{aligned}$$

where U_{pk} and f are the peak value and the frequency of the applied voltage respectively, this will result in a sinusoidal flux linkage in each phase that lags the phase voltage by 90°if the winding resistance is neglected, that is

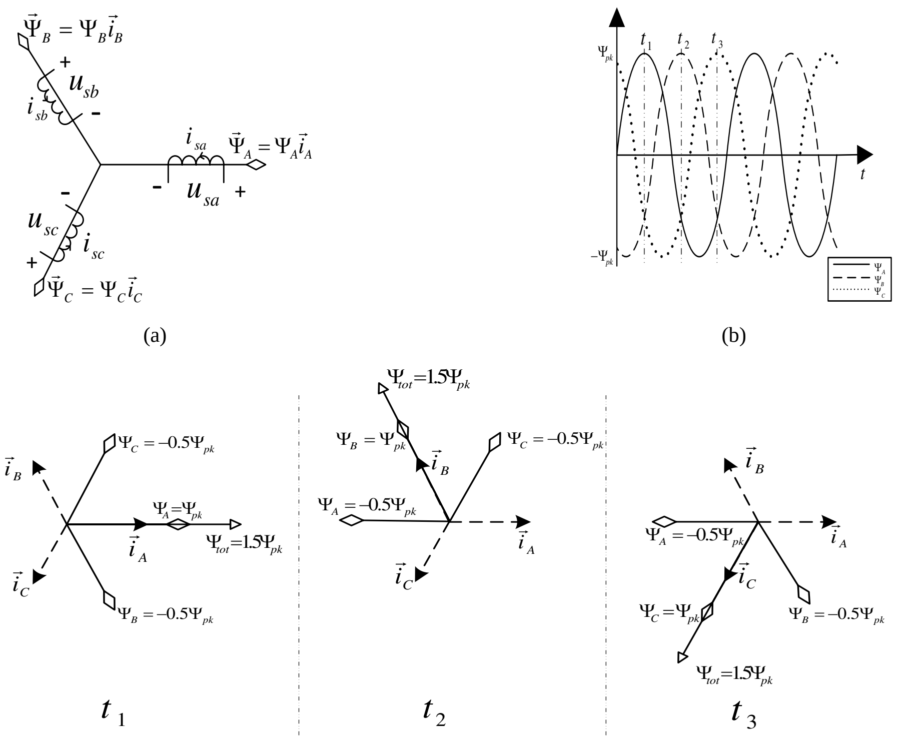
$$\begin{aligned} \Psi_A(t) &= \Psi_{pk} sin(360ft) \\ \Psi_B(t) &= \Psi_{pk} sin(360ft + 120°) \\ \Psi_C(t) &= \Psi_{pk} sin(360ft + 240°). \end{aligned}$$

The waveforms of the three flux linkages are plot in Fig. 2.3(b). In each period, the linkage generated by each winding reaches its maximum value once. At the moment of t_1, Ψ_A gets to its peak value of Ψ_{pk} while both Ψ_B and Ψ_C are minus half peak values. As shown in the left subfigure of Fig. 2.3(c), the resultant flux has the same orientation with $\vec{\Psi}_A$ and a magnitude of $1.5\Psi_{pk}$. At the time of t_2 when Ψ_B reaches the peak, the resultant flux still has a magnitude of $1.5\Psi_{pk}$ but coincides with $\vec{\Psi}_B$ as shown in the middle subfigure of Fig. 2.3(c). Similarly, as shown in the right subfigure of Fig. 2.3(c), the resultant flux's magnitude doesn't change but coincides with $\vec{\Psi}_C$ at t_3 when Ψ_C reaches the peak. From the above analysis of three moments, it could be noticed that the resultant flux rotates with a constant magnitude. The magnitude of this rotating resultant flux is 1.5 times of the peak value of each wingding's flux linkage. The rotating speed N_s (in *rpm*) of the resultant flux is called synchronous speed, calculated as

$$N_s = 60\frac{f}{N_p},$$

where f is three phase voltage frequency and N_p is number of pole pairs.

The rotating resultant stator flux linkage cuts the rotor bars, which will induce a voltage in the rotor bars. Since the rotor bars are short-circuited, this inducted voltage will drive a current in the rotor bars. By Lenz's Law, *An induced current is always in such a direction as to oppose the motion or change causing it*, the induced current will generate a torque trying to make the rotor follow the rotating flux generated by the stator windings. However, if the rotor speed is exactly equal to the synchronous speed, there will not be an induced current and no toque. Other external factors, such as friction torque and load torque, will make the rotor slow down. So the induction machine rotor runs at a slightly slower speed than the synchronous speed in order to get enough induced current and driving torque. There always exists a gap between machine speed and synchronous speed, which is referred to as slip. This is why induction machine gets its another name of asynchronous machine, to emphasize the difference between the machine speed and the flux rotating speed.



(a)

(b)

(c)

Fig. 2.3 RotatingFlux

2.2 Three-phase Inverter and Pulse Width Modulation(PWM)

The three-phase inverter supplies the induction machine with three-phase alternating voltage. From the schematic of three-phase inverter in Fig. 2.4, it is seen that the three-phase inverter is fed by a DC voltage as the energy input and has three legs, each consisting of two IGBTs and two diodes. It is worth noting that in Fig. 2.4, each IGBT in three-phase inverter is replaced with an ideal switch. Define the DC-link voltage fed into the three-phase inverter as V_{dc}. Take phase A for example. Based on Fig. 2.4, if switch T1 is turned on, the A-phase voltage relative to ground U_{AN} will be equal to $+\frac{V_{dc}}{2}$; while if switch T4 is turned on, U_{AN} will equal $-\frac{V_{dc}}{2}$.

Although in this perspective it seems that the three-phase inverter only can generate two possible voltage values for phase voltage, $+\frac{V_{dc}}{2}$ or $-\frac{V_{dc}}{2}$, Pulse Width Modulation(PWM) makes it possible for the three-phase inverter to convert the DC-link voltage into three-phase alternating voltage by controlling the on-times and off-times of the IGBTs. By adjusting the proportion of on-time in one PWM switching period for the IGBTs on the same bridge leg, any value between $+\frac{V_{dc}}{2}$ and $-\frac{V_{dc}}{2}$ can be obtained as the phase voltage. The ratio is described as duty cycle [33]. Define the on-time of an IGBT as T_{on}, the off-time of the IGBT as T_{off} and one PWM switching period as T_{PWM}. Duty cycle(D) can be formulated as:

$$D = \frac{T_{on}}{T_{on} + T_{off}} = \frac{T_{on}}{T_{PWM}}, \tag{2.1}$$

which is usually expressed in percent.

The basic PWM generation principle is to compare waveforms of a triangular wave(carrier wave) and a sine wave(reference wave) shown in Fig. 2.5(a). When the magnitude of the sine wave is larger than that of the triangular wave, the PWM signal will output a high voltage level to switch on the corresponding IGBT. It is the case of the segment between point A and point B presented in Fig. 2.5(a) and Fig. 2.5(b). On the contrary, when the magnitude of the sine wave is smaller than that of the triangular wave, the PWM signal will produce a low voltage level to shut down that IGBT. It is the case for the segment between point C and point D displayed in Fig. 2.5(a) and Fig. 2.5(b). Therefore, the corresponding time moments of the intersections of the two waves serve as the turning points to switch on or off the IGBT in the inverter. From Fig. 2.5(b), it is observed that the generated PWM signal is composed of a series of pulses with fixed

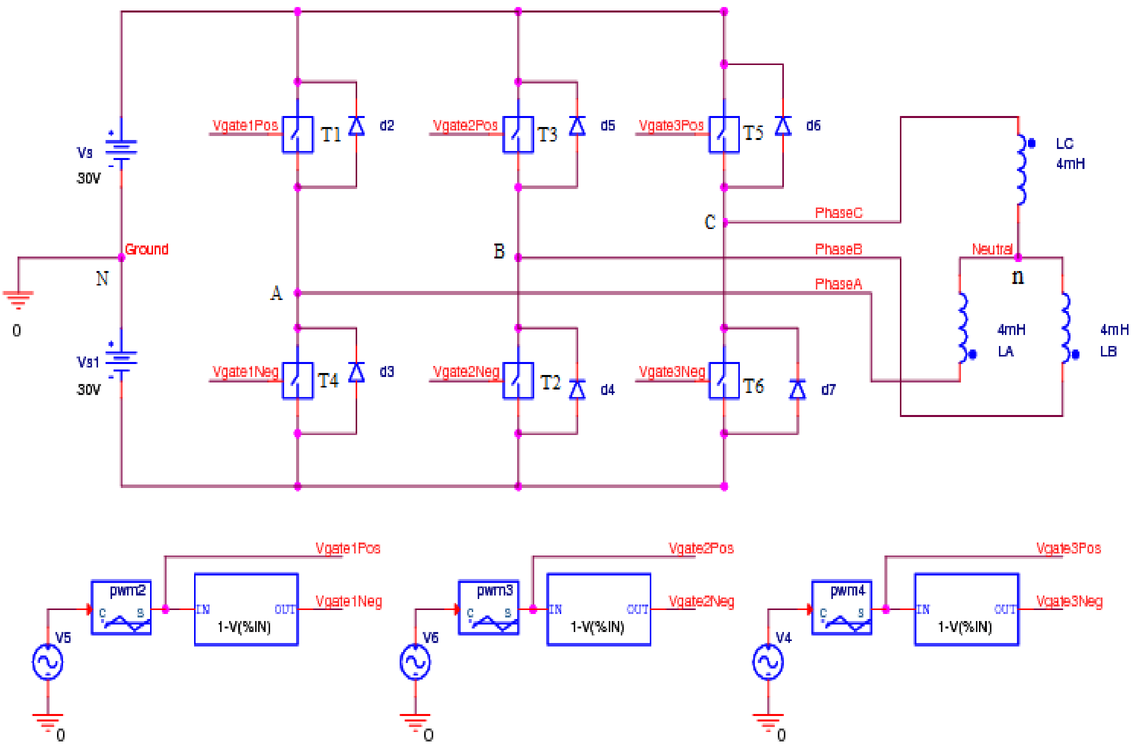


Fig. 2.4 Three Phase PWM Inverter Schematic.

magnitude but variable duty cycle.

To better understand the operation of the three-phase inverter, a simulation of the three-phase inverter in Fig. 2.4 by PSpice is performed. In the simulation, the reference voltages for the three phases are assumed to be constant in a short period as the frequency of PWM carrier wave is much higher than that of the modulation wave, $V_{A,ref} = V_5 = 27V = -V_4 = -V_{C,ref}$, $V_{B,ref} = V_6 = 0$ and the DC-link voltage is $\pm 30V$. Besides, the loads of the inverter are assumed to be purely inductive and connected in a Y-configuration.

In Fig. 2.6(a) the three-phase reference voltages and the carrier wave are shown. Still take phase A for example, the voltage waveform of phase A with respect to ground, V_{AN}, is displayed in Fig. 2.6(c) and the voltage between the neutral point and ground, V_{nN}, is presented in Fig. 2.6(b), thereby the phase voltage for phase A, V_{An}, can be obtained easily by subtracting V_{nN} from V_{AN} shown in Fig. 2.6(c) as well. The similar cases for phase B and phase C are shown in Fig. 2.6(e) and Fig. 2.6(d) respectively. The voltage between the neutral point and ground, V_{nN}, can be obtained by voltage division, as:

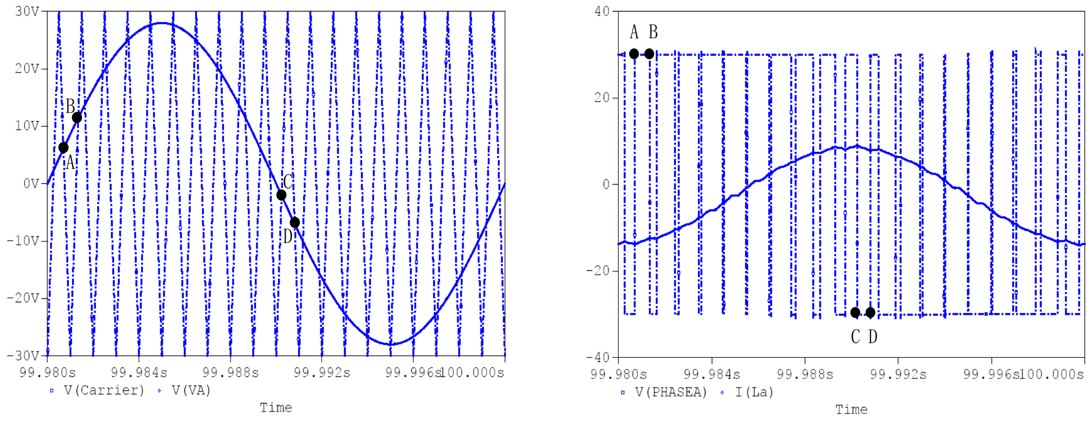
$$V_{nN} = \frac{V_{AN} + V_{BN} + V_{CN}}{3}.$$

From Fig. 2.6(b), Fig. 2.6(c), Fig. 2.6(e) and Fig. 2.6(d) it can be noticed that at $96ms$ all the phase voltages relative to ground, V_{AN}, V_{BN} and V_{CN}, are $30V$ and the voltage between the neutral point and ground, V_{nN}, is also equal to $30V$. As a fact, the average phase voltages for one PWM switching period are always equal to the values of the reference waves, which justifies the PWM method. The average A-phase voltage, $u_{a,AVG}$, can be calculated as:

$$u_{a,AVG} = \frac{1}{T}\int_0^T V_{An}dt = \frac{1}{T}\int_0^T V_{AN} - V_{nN}dt = \underbrace{\frac{1}{T}\int_0^T V_{AN}dt}_{=V_{A,ref}} - \underbrace{\frac{1}{T}\int_0^T V_{nN}dt}_{=0} = V_{A,ref}. \tag{2.2}$$

The same results can be obtained for the other two phases, i.e. $u_{b,AVG} = V_{B,ref}$ and $u_{c,AVG} = V_{C,ref}$.

As the voltage applied to the phase winding is a pulse wave instead of a sinusoidal wave, there will be a ripple in the phase current, which can be found in Fig. 2.6(b). To investigate the relation between the current ripple and the carrier wave, the current of phase B, i_B, is also plotted in Fig. 2.6(e). Comparing



(a)

(b)

Fig. 2.5 Pulse Width Modulation(PWM) generation principle. In (a) the dash line is the carrier wave and the solid line is the reference wave for phase A voltage; In (b) the dash line is the generated PWM signal and the solid line is A-phase current in the PSpice simulation.

Fig. 2.6(a) and Fig. 2.6(e), it can be noticed that, at every moment that the carrier wave reaches its peak value the phase current, i_B, is equal to its average value, in this case $0A$. Due to the principle of the PWM the voltage reference is constant over one switching period . Due to this fact, the controller can not control the current in the switching period, it can only control the average value of the current. This means that the control system sets the average voltage for one switching period to control the average value of the current. To measure the average value of the current for the feedback to the control system it is important that the measurement is synchronized with the current wave peaks. The current samples must be taken on the peaks of the carrier wave in order to set the average value of the current.

2.3 Vector Control System

As a control strategy used in variable frequency drives, vector control provides a feasible solution to torque/speed control of AC machines by controlling the phase currents into the machine even if it gives rise to a considerable computation burden for the processor where the control algorithms are implemented. The most noticeable merit of vector control is to get rid of machine speed dependency on power grid frequency and make it possible to reach the desired machine speed within safety and power limits.

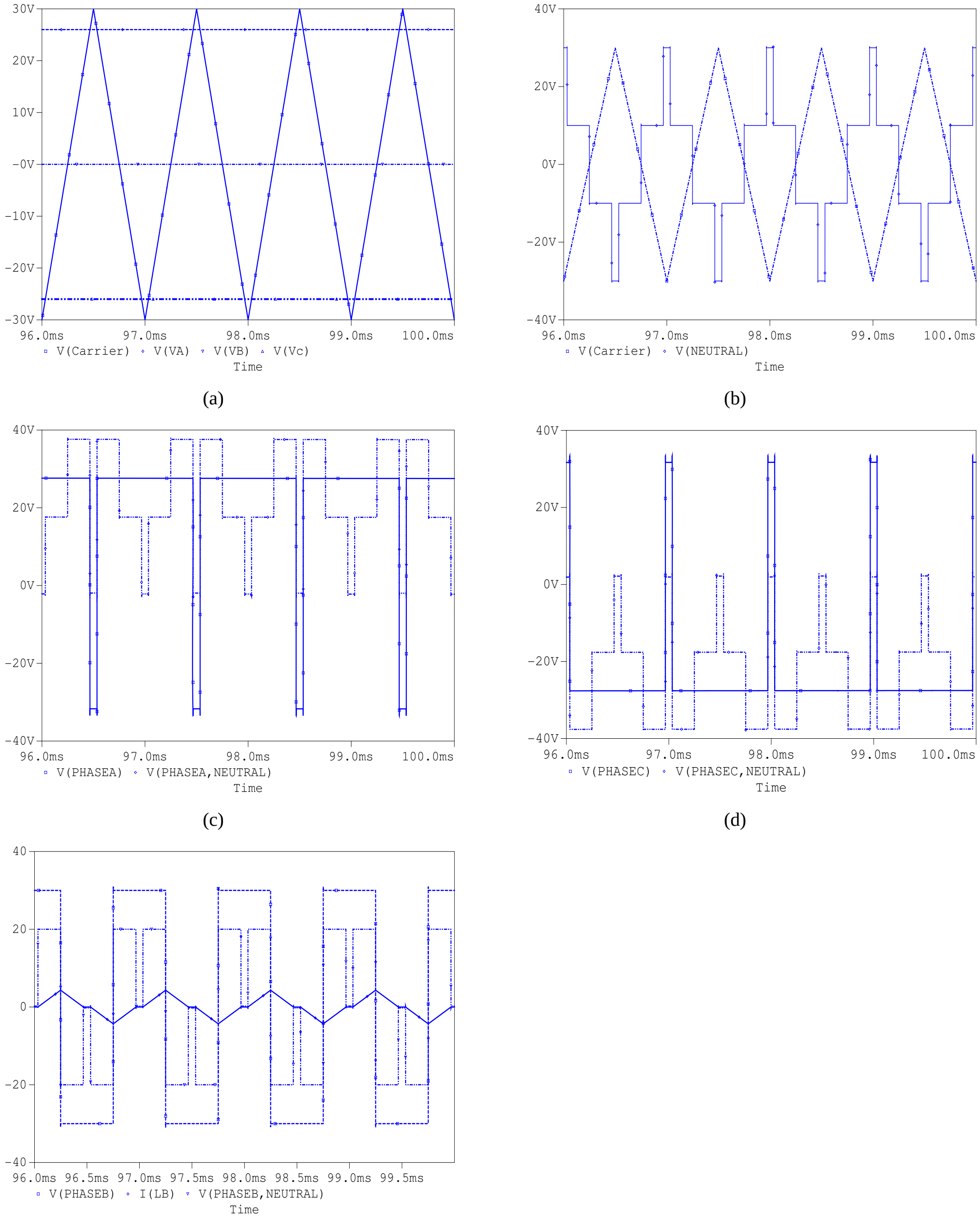
Usually in a vector control system, the phase currents of the machine and the DC-link voltage of the inverter and sometimes the rotor shaft position and/or the machine speed are taken as the control system inputs, while the phase voltages to the machine are picked as the outputs. Two important transformations are involved in the vector control system to transform AC machine into a separately magnetized DC machine, namely Clarke Transformation and Park Transformation. With these transformations the three-phase sinusoidal quantities can be transformed into DC quantities in steady state.

2.3.1 Clarke and Park Transformations

If the three-phase stator coils are arranged symmetrically in space, i.e. with a 120°displacement as shown in Fig. 2.7(a), and the sum of the phase voltages is zero($u_a(t) + u_b(t) + u_c(t) = 0$, neglecting the case of zero sequences), the three-phase stator voltages can be expressed by an equivalent space voltage vector $\vec{u}_s^s$, i.e. with two-phase quadrature quantities as

$$\begin{aligned}
\vec{u}_s^s(t) &= K(u_{sa}(t) + u_{sb}(t)e^{j\frac{2\pi}{3}} + u_{sc}(t)e^{j\frac{4\pi}{3}}) \\
&= K(u_{sa}(t) - \frac{1}{2}u_{sb}(t) - \frac{1}{2}u_{sc}(t) + j\frac{\sqrt{3}}{2}(u_{sb}(t) - u_{sc}(t))) \\
&= u_{s\alpha} + ju_{s\beta}, \qquad (2.3)
\end{aligned}$$

where K is set to be $\frac{2}{3}$ in order to make the two-phase quantities to have the same amplitude as the three-phase quantities, i.e.$|\vec{u}_s^s| = max(u_{sa}) = max(u_{sb}) = max(u_{sc})$. It can be seen from Fig. 2.7(a) that



(a)

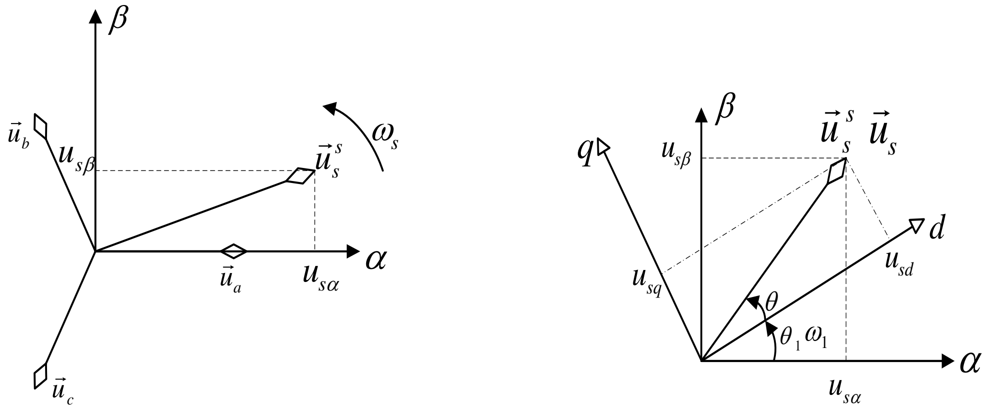
(b)

(c)

(d)

(e)

Fig. 2.6 The simulation results of three-phase inverter by PSpice. In (a) the solid line is the carrier wave and the dash lines stand for reference voltages of three phases respectively; in (b) the solid line is V_{nN} and the dash dot line is the carrier wave; in (c) the solid line is V_{AN} and the dash dot line is V_{An}; in (d) the solid line is V_{CN} and the dash dot line is V_{Cn}; in (e) the solid line is B-phase current i_B, the dash line is V_{BN} and the dash dot line is V_{Bn}.



(a)

(b)

Fig. 2.7 Clarke and Park Transformations.

$\vec{u}_s^s$ can be treated as a vector rotating in the $\alpha\beta$ coordinate system. The α-axis of $\alpha\beta$ coordinate system is aligned with the direction of the A-phase space vector. As the $\alpha\beta$ coordinate system is stationary with respect to the coils and the coils are fixed to the stator, the $\alpha\beta$ coordinate system is stationary as well.

In vector control theory, the three-phase to two-phase coordinate transformation is named as Clarke Transformation. The reverse process that turns three-phase back to two-phase is called Inverse Clarke Transformation [8]. Clarke and Inverse Clarke Transformations can be expressed in the form of matrices, as:

$$\begin{bmatrix} u_{s\alpha} \\ u_{s\beta} \end{bmatrix} = \begin{bmatrix} \frac{2}{3} & -\frac{1}{3} & -\frac{1}{3} \\ 0 & \frac{1}{\sqrt{3}} & -\frac{1}{\sqrt{3}} \end{bmatrix} \begin{bmatrix} u_{sa} \\ u_{sb} \\ u_{sc} \end{bmatrix} \tag{2.4}$$

$$\begin{bmatrix} u_{sa} \\ u_{sb} \\ u_{sc} \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ -\frac{1}{2} & \frac{\sqrt{3}}{2} \\ -\frac{1}{2} & -\frac{\sqrt{3}}{2} \end{bmatrix} \begin{bmatrix} u_{s\alpha} \\ u_{s\beta} \end{bmatrix} \tag{2.5}$$

respectively. The Clarke Transformation can also be applied to other three-phase quantities, such as the flux, the rotor currents and so on, in order to get the corresponding space vectors.

However the α-components and the β-components of these rotating vectors are still AC sinusoidal quantities. A new coordinate system is defined, called the dq coordinate system, which has the same origin as the $\alpha\beta$ coordinate system but it rotates with the same speed as the flux. The d-axis of the dq coordinate system coincides with the space vector of the flux. The transformation of a space vector between the two coordinates can be achieved by multiplying the vector by $e^{j\theta_1}$ or $e^{-j\theta_1}$, where θ_1 is the angle between the two coordinate system. Take the stator voltage as an example. As shown in Fig. 2.7(b), the space vector of the stator voltage is referred to as $\vec{u}_s^s$ in the $\alpha\beta$-system and as $\vec{u}_s$ in the dq-system. From Fig. 2.7(b) it can be seen that $\vec{u}_s^s$ and $\vec{u}_s$ are transformed as

$$\vec{u}_s^s = |\vec{u}_s^s|e^{j(\theta+\theta_1)} = |\vec{u}_s^s|e^{j\theta}e^{j\theta_1} = \vec{u}_s e^{j\theta_1} \tag{2.6}$$

$$\vec{u}_s = \vec{u}_s^s e^{-j\theta_1}. \tag{2.7}$$

The transformation from the $\alpha\beta$ coordinate system to the dq coordinate system is called Park Transformation while its backward process is known as Inverse Park Transformation [8]. If $\vec{u}_s^s$ and $\vec{u}_s$ are written in the form as

$$\begin{aligned} \vec{u}_s^s &= u_{s\alpha} + ju_{s\beta} \\ \vec{u}_s &= u_{sd} + ju_{sq}, \end{aligned}$$

and $e^{j\theta_1}$ and $e^{-j\theta_1}$ are expanded using Euler's formula as

$$\begin{aligned} e^{j\theta_1} &= cos\theta_1 + jsin\theta_1 \\ e^{-j\theta_1} &= cos\theta_1 - jsin\theta_1, \end{aligned}$$

Eq. 2.6 and Eq. 2.7 can be expressed in matrices forms as

$$\begin{bmatrix} u_{sd} \\ u_{sq} \end{bmatrix} = \begin{bmatrix} \cos\theta_1 & \sin\theta_1 \\ -\sin\theta_1 & \cos\theta_1 \end{bmatrix} \begin{bmatrix} u_{s\alpha} \\ u_{s\beta} \end{bmatrix} \tag{2.8}$$

$$\begin{bmatrix} u_{s\alpha} \\ u_{s\beta} \end{bmatrix} = \begin{bmatrix} \cos\theta_1 & -\sin\theta_1 \\ \sin\theta_1 & \cos\theta_1 \end{bmatrix} \begin{bmatrix} u_{sd} \\ u_{sq} \end{bmatrix}. \tag{2.9}$$

2.3.2 Digital Implementation of a Vector Control System

In digital implementation of vector control, the microprocessor must be equipped with analog-to-digital converter in order to measure the phase currents and the DC-link voltage. If the speed of the machine is measured with a tachometer, this signal also needs to be measured with the AD-conversion. If an incremental encoder is used to measure the position of the rotor, the speed can be calculated from the position. Considering the aliasing from the current ripple, the microprocessor should be able to trigger the sampling of currents on the peak of the carrier wave as mentioned in Sec. 2.2, which also means that the sampling could take place once or twice during one PWM switching period. Due to this fact, all the calculation steps of the vector control algorithm must be finished in one or a half of PWM switching period.

The basic procedure of a vector control algorithm is that the measured three-phase currents are converted into a complex space vector by Clarke Transformation. The space vector is then transformed from the $\alpha\beta$-system to the dq-system by a Park Transformation where usually the flux angle is used as the transformation angle. In this way a flux orientated system is obtained. For the DC-currents obtained after the transformations, usually a standard PI-regulator together with some feed-forward terms are used. The output of the current regulator is the stator voltage space vector, expressed in the dq-system. The stator voltage space vector is transformed back to three-phase by Inverse Park and Inverse Clarke Transformations and the obtained phase voltages are used as reference for the PWM in Fig. 2.4. In Fig. 2.8 the PWM is shown as the block Duty Cycle Output. The vector control algorithm consists of a wide variety of mathematical calculations, such as trigonometric functions $\sin(\theta_1)$, $\cos(\theta_1)$, multiplication, division, addition and subtraction. To meet all the practical system demands(performing AD-conversion, all the calculations and updating the PWM within one or a half of PWM switching period) the processor used for vector control system needs to have a high execution speed [36].

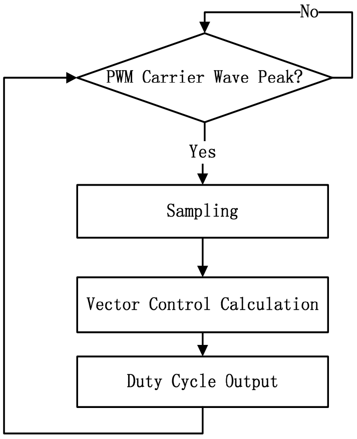


Fig. 2.8 Digital implementation of a vector control system.

Chapter 3

Evaluation Board Selection

3.1 Evaluation Board Specifications

Based on the knowledge introduced in the previous chapter and the demands put by the division, the evaluation board for the vector control system should have the following characteristics:

1. Compulsory:

   - **4 ADC Input Channels** Three inputs for 3-phase current measurement respectively, one for DC link voltage. More channels are preferable, for example, one channel for sampling a speed signal from a tachometer, if used.
   - **3-phase PWM** Three pairs of PWM outputs that can be synchronized with each other, with a switching frequency of 10kHz. The PWM should have sufficient resolution. Additionally, it should be possible to synchronize the AD conversion with the carrier wave of the PWM.
   - **Interface of incremental encoder** Specific encoder module or external event capture.
   - **On-board programming** Emulator is a superior solution for programming, debug and download.
   - **Cost** High performance-price ratio is desirable.

2. Optional:

   - **CPU computation capability** As almost all the quantities in vector control are decimals, a microprocessor with a floating-point unit is highly recommended. The differences between fixed-point and floating-point DSP will be shortly described in Sec. 3.3.
   - **Math function library** Some DSP manufacture may provide highly-optimized math function, such as $sin()$ and $tan()$, in libraries, which can improve code execution efficiency.
   - **Available program example** A set of mature example codes can contribute to the design.
   - **Communication interface** It is an advantage to have USB and CAN interfaces on the board for other applications.
   - **DAC output** It can be used for debugging and to send signals to other devices.
   - **Distributor** A distributor in Göteborg is preferred.

3.2 Evaluation Board Candidates

Following the specifications above, the search task is focused on the large semiconductor companies. These companies usually list different types of chips and evaluation boards on their webpages. Nevertheless it is of low efficiency to look through every chip's data sheet. Fortunately, these chips and boards are grouped by different application fields, such as industrial, video&imaging, wireless and so on. For the industrial fields, the semiconductor companies will provide some application notes for induction motor control, where some microprocessors are recommended. In the first step of the selection process, all these recommended microprocessors are listed together. In the next step it is checked whether the microprocessors in the list have a corresponding evaluation board or not. Some of the evaluation boards are very expensive, as they have a small induction motor and/or a converter on the board, which is redundant for this project. Finally eight evaluation boards are picked as candidates and they are listed in Table 3.3.

3.3 Boards Comparison

Making an immediate decision on which evaluation board that is the most suitable for this project is not easy. So these eight evaluation boards are compared based on the following key specifications: CPU type, PWM carrier wave frequency and resolution, encoder interface, ADC as well as on-board programming. Once one board meets one of these key specifications, it will be marked with a ★.

As mentioned before, massive computations need to be performed in one vector control period and due to this fact the microprocessor's calculation speed plays an important role in the system performance. In a fixed-point processor, the decimal is represented by an integer. The mathematic operations of decimals are achieved by decimal point shift in the software algorithm, which will increase the code execution time. Besides, attention needs to be paid on overflow problem all the time. On the other hand, floating-point DSP implements more number operation in internal hardware unit instead of executing a segment of codes for decimal computation. Since the floating-point processor has a high dynamic range for numeric calculations [38], the overflow problem needs rarely to be considered. A computer lab provided by Texas Instruments can roughly reflect the code execution speed on different types of DSP. Table 3.1 shows the results from this lab of a simple math operation $k = i * i$ with different number formats. The middle column indicates the type of processor used. The DSP used in the lab is TI's F28335, which can be configured to run in either fixed-point mode or floating-point mode. The right column shows the number of CPU clock cycles that are used for the $i * i$ calculation.

Table 3.1: Speed test of fixed-point and floating-point DSP

Code in C	Type of DSP	No. of CPU clock cycles

`unsigned i=1;`
`unsigned k=0;`
`k=i*i;`
	Fixed-point	3

`unsigned i=1;`
`unsigned k=0;`
`k=i*i;`
	Floating-point	3

`float i=1.0;`
`float k=0.0;`
`k=i*i;`
	Fixed-point	113

`float i=1.0;`
`float k=0.0;`
`k=i*i;`
	Floating-point	5

From Table 3.1, it could be found that it takes only several CPU clock cycles for both fixed-point and floating-point DSP to execute the multiplication of two integers. There is no difference in computation speed for integer multiplications. However, when it comes to multiplication of two decimals, the fixed-point DSP has to cost more than 100 CPU clock cycles to finish the calculations while the floating-point DSP only costs 5 cycles. So it could be concluded that the floating-point DSP has advantages over the fixed one from the point of view of computation speed, especially for this vector control system that involves massive decimal calculations. So No.3 and No.4 in Table 3.1 are predominant as they are equipped with the floating-point unit. Infineon's XC866 8-bit microcontroller is also a possible choice because of its enclosed 16-bit vector computer, which could handle the Clarke & Park transformations in hardware. In other words, this vector computer could relieve some burden of trigonometric computations such as $sin()$ and $cos()$ for the main CPU. No.6 ADZS-BF506F EZ-KIT Lite is a fixed-point DSP, however, its CPU frequency is as high as $400MHz$, which implies a high computation speed. In this step, No.3, No.4, No.5 and No.6 are marked with a ★.

The second compulsory demand for the selection of evaluation board is the carrier wave frequency and its resolution. Usually, the carrier wave is generated by a counter which increments periodically and is reset at the end of every PWM period [26]. If the least significant output voltage of the inverter is $0.2V$ and the

DC link voltage is $800V$, the PWM resolution is required as

$$\frac{800V}{2^{PWM\,resolution}} \leq 0.2V \Rightarrow PWM\,resolution \geq 12\,bits,$$

with PWM resolution meaning the resolution of the PWM counter, i.e. the number of the bits used for the counter. It is can be found from Table 3.3 only No.7 can't meet the PWM resolution requirement. For a given PWM resolution, a higher bus frequency gives a higher PWM carrier wave frequency. In the project specifications the carrier wave frequency is specified to be $10kHz$. This, together with the resolution, gives the required updating frequency of the PWM counter, counter frequency, as

$$counter\,Freq. = carrier\,wave\,Freq. \times 2^{resolution} \times 2 = 10kHz \times 2^{13} = 81.92MHz,$$

where $\times 2$ stands for rising and falling ramps of each carrier wave, the carrier wave is a triangular wave. For the 8 listed evaluation boards except No.1 and No.2, the PWM counter is connected directly to the system clock, which means that the highest PWM counter frequency is equal to the system clock frequency. The evaluation boards with a system frequency lower than 81.92MHz are probably eliminated due to this(They are No.5, No.7, No.8, see Table 3.3). But Freescale's 56F8037EVM has the ability to let the PWM counter run with a frequency of 3 times the system frequency, which is only 32MHz. In this step, No.1, No.3, No.4 and No.6 are marked.

For the vector control system in this project, the speed of the rotor should be known in order to estimate the flux. In this project, the rotor speed is obtained either from a tachometer or by taking the time derivative of the rotor position from the incremental encoder. The working principles of the incremental encoder can be illustrated with Fig. 3.1.

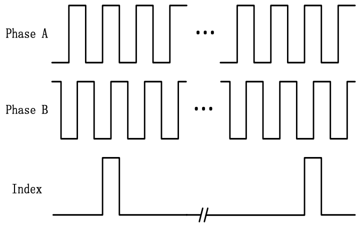


Fig. 3.1 Working principles of incremental encoder

The incremental encoder outputs three pulse sequence signals when it is rotating: Phase A, Phase B and Index. There is a quarter period displacement between Phase A and Phase B. The rotating direction can be fixed by the order of rising edges from the two phases. To determine the absolute position, the encoder will generate one index pulse per turn. In theory, as long as the microprocessor has external interrupts, the pulses from the encoder can be counted in the Interrupt Service Routines (ISR). The drawback of this method is that the frequent interrupts will disturb the main routine, which may affect the system performance. Consequently, special encoder modules have been integrated onto some microprocessors. These modules are able to treat the continual pulses without halting the main routine. The processors equipped with encoder module are No.1, No.2, No.3, No.4 and No.8 in Table 3.3, which are marked in this step.

As mentioned in Sec. 2.3.2, the Analog-to-Digital conversion needs to be finished as soon as possible after the PWM carrier wave peak. In this project, the PWM switching period is $100\mu s$ and the Analog-to-Digital conversion time is expected to be less than one tenth of the PWM switching period, that is $10\mu s$. Most of the candidate boards' Analog-to-Digital conversion times are far less than 10μs except for No.4 in Table 3.3, which has no internal ADC. In order to sample 4 analog inputs at the same time instant, four sample&hold circuits are needed. Only the 56F8367EVM can provide 4 internal sample&hold circuits. This problem can be solved by adding external Sample&Hold chips as peripheral circuit and assigning a digital I/O pin to output a pulse for synchronizing the external Sample&Hold circuits with the carrier wave. Based on the number of Sample&Hold circuits, only No.2 is marked in this step.

The last compulsory requirement to be checked is the ability of on-board programming. All candidate boards except No.7 have on-board programming ability, achieved by Joint Test Action Group(JTAG) technology. However the No.2 56F8367EVM's JTAG needs to communicate with the computer through a parallel port, which might not be found on modern computers. The JTAG of the other six evaluation boards can be connected to the computer via USB. Usually, to achieve on-board programming an emulator is used. The emulator serves for program downloading and on-line debugging through JTAG. Among these six evaluation boards, No.1, No.3 and No.8 have emulator included while No.4, No.5 and No.6 require the users to buy an additional emulator. The prices of emulators are high, usually about €100. In this step, No.1, No.3 and No.8 are marked.

Table 3.2: Results of the primary selection of evaluation board

No.	Evaluation Board	Score	No.	Evaluation Board	Score
1.	56F8037EVM	★★★	2.	56F8367EVM	★★
3.	TMS320F28335 Experimenter Kit	★★★★	4.	TMS320C28346 Control Card	★★★
5.	Easy Kit XC878	★	6.	ADZS-BF506F EZ-KIT Lite	★★
7.	ST7MC Control Board		8.	Explorer 16 Starter Kit	★★

After five steps of primary selection, seen in Table 3.2, No.3 TMS320F28335 Experimenter Kit ranks first among the eight evaluation candidates. TMS320F28335(F28335 for short) is a 32-bit floating-point DSP. The emulator XDS100 has been integrated into the experimenter kit, which allows on-board programming and debugging. F28335 can provide up to two groups of 16-bit three-phase PWM outputs, both of which are able to trigger the starting of ADC. F28335 is also equipped with two 32-bit quadrature encoder interfaces. It also has various serial port peripherals, such as CAN, UART, McBSP, SPI and I2C. As a member of TI's C2000 microcontroller family, users of TMS320F28335 can share rich resource of example codes, application notes, math algorithms and other users' experience, all of which can be got from TI's forum on the Internet. Besides, a full-sized Integrated Development Environment(IDE) CCS v4.0 will be given for free with the evaluation board. There are two types of TMS320F28335 evaluation boards. One type has a complete set of peripherals while the other one leave a blank docking station providing more flexibility and they cost $179 and $99 respectively. What's more, TI has a distributor in Göteborg. Based on the considerations above and opinions from other people, TMS320F28335 Experimenter Kit with a blank docking station is chosen to be the evaluation board used in this project.

Table 3.3: Evaluation board candidates

No.	Evaluation Borad Manufacture	CPU Type	System Frequency	3-phase PWM No. of Channels Max. Resolution @Max. Counter Freq.	ADC No. of Channels Max. Resolution @Max. Sampling Freq. Or Min. Conversion Time	ADC&PWM Synchronized	DAC	Event Capture External Interrupt	Encoder Module	CAN	USB	JTAG in which way?	Emulator included	Math Library	Price
1	56F8037EVM Freescale	16-bit Fixed-point	32MHz	6-channel 15-bit @96MHz	2*8 Channels 12-bit @2.67MSPS	Yes	12-bit 2 Channels	Yes	Quadrature Counter*2	Yes	No	Yes by USB	Yes	Yes	$199.1
		http://www.freescale.com/webapp/sps/site/prod_summary.jsp?code=MC56F8037EVM&fsrch=1													
2	56F8367EVM Freescale	16-bit Fixed-point	60MHz	6-channel*2 15-bit @60MHz	4*4 Channels 12-bit @5MSPS	Yes	No	Yes	Quadrature Decoder*2	Yes	No	Yes by Parallel port	Yes	Yes	$299.1
		http://www.freescale.com/webapp/sps/site/prod_summary.jsp?code=MC56F8367EVME&fsrch=1													
3	TMS320F28335 Experimenter Kit Texas Instruments	32-bit Floating-point	150MHz	6-channel*2 16-bit @150MHz	2*8 Channels 12-bit @80ns	Yes	No	Yes	Quadrature Decoder*2	Yes	No	Yes by USB	Yes	Yes	$99
		http://focus.ti.com/docs/toolsw/folders/print/tmdsdock28335.html													
4	TMS320C28346 ControlCARD Texas Instruments	32-bit Floating-point	300MHz	6-channel*3 16-bit @300MHz	No	Yes with external ADC	No	Yes	Quadrature Decoder*3	Yes	No	Yes by USB	No	Yes	$125
		http://focus.ti.com/docs/prod/folders/print/tms320c28346.html													
5	Easy Kit XC878 Infineon	8-bit MCU+ 16-bit vector computer	24MHz	6-channel 16-bit @24MHz	8 Channels 10-bit @200ns	Yes	No	Yes	No	Yes	No	Yes by USB	No	No	€99
		http://www.infineon.com/dgdl/Microcontroller-XC87x-Data-Sheet-V15-infineon.pdf?folderId=db3a304412b407950112b40c497b0af6&fileId=db3a30432239cccd01231293d56908b1													
6	ADZS-BF506F EZ-KIT Lite Analog Devices	16-bit*2 Fixed-point	400MHz	6-channel*2 16-bit @400MHz	2*6 Channels 12-bit @2MSPS	Yes	No	Yes	No	Yes	No	Yes by USB	No	No	$199
		http://www.analog.com/en/embedded-processing-dsp/blackfin/BF506-HARDWARE/processors/product.html													
7	ST7MC control board STMicroelectronics	8-bit MCU	8MHz	6-channel 8-bit @8MHz	1*16 Channels 10-bit @3.5us	N/A	No	Yes	No	No	No	No	No	No	$96.36
		http://www.st.com/internet/mcu/product/117672.jsp													
8	Explorer 16 Starter Kit Microchip	16-bit Fixed-point	40MHz	6-channel 16-bit @40MHz	2*16 Channels 12-bit @0.5MSPS	Yes	No	Yes	Quadrature Decoder	Yes	No	Yes by USB	Yes	N/A	$299
		http://www.microchip.com/stellent/idcplg?IdcService=SS_GET_PAGE&nodeId=1406&dDocName=en027853													

Chapter 4

Signal Conditioning

In the previous chapter, the F28335 floating-point DSP has been selected as the evaluation board of this project. As mentioned in Sec. 1.2, one of the objectives of this project is to make the final vector control system exchangeable with the dSPACE system. Hence, some peripheral circuits have to be added to achieved this objective and in addition some steps are also taken in order to enhance the system's performance and to protect the DSP. For example, a buffer circuit will be built to protect the DSP's digital input from over-voltages. In the first part of this chapter, the relative signal and interface specifications of dSPACE are listed and according to these specifications, the signal conditioning circuits needed for DSP are described in detail.

4.1 dSPACE Signal and Interface Specifications

The dSPACE system used in the lab is the DS1103. According to the manual [10] of the DS1103 system, the specifications are listed below.

1. Signal specifications:

   - **Analog Inputs.** Bipolar inputs with $\pm 10V$ input span.
   - **Digital Input.** Transistor-Transistor Logic(TTL). The voltage range of the TTL signal is from $0V$ to $5V$. For TTL input, $\leqslant 0.8V$ is defined as 'low', $\geqslant 2.0V$ as 'high'.
   - **PWM and Digital Outputs.** TTL. For TTL output, $\leqslant 0.4V$ is defined as 'low', $\geqslant 2.4V$ as 'high'.
   - **Incremental Encoder Interface.** Either TIA-422 or TTL

2. Interface specifications:

   - **Analog Inputs.** Female Bayonet Neil-Concelman(BNC) connector.
   - **Digital Inputs and Outputs.** In this project, the digital inputs and outputs are mainly used for inverter's fault detection and relay control. The digital I/O interface of DS1103 is a male 50-pin D-Subminiature(D-Sub).
   - **PWM Outputs.** 37-pin female D-Sub.
   - **Incremental Encoder Interface.** 15-pin female D-Sub.

4.2 Analog Signal

The Analog-to-Digital Converter(ADC) samples the analog signal and converts it into the bit form which can be processed by the DSP. In the project, the ADC is used to measure three phase currents, the DC voltage and maybe the speed if tachometer is used. The precision of these measurements is very important for the performance of the whole vector control system.

4.2.1 About ADC Input Voltage

As mentioned before, the analog inputs for the dSPACE system are in the range of $\pm 10V$ while the analog inputs for the DSP are in the range from $0V$ to $3V$. Due to this fact,the input signals need to be scaled down and biased with a conditioning circuit. For every input channel, three operational amplifiers(Op-Amp) are used to build one conditioning circuit. The selected operational amplifier is Texas Instruments' TLC074A, which has the advantages of low noise, high slew rate and high common mode rejection ratio(CMRR) [14]. What's more, four Op-Amps are integrated into one package, which can save space and reduce the soldering complexity. The conditioning circuit for one channel is shown in Fig. 4.1.

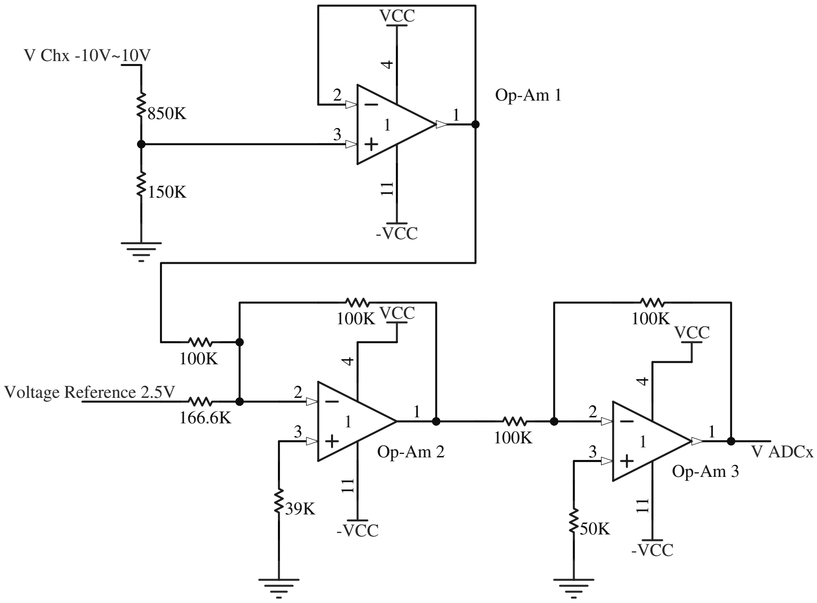


Fig. 4.1 Conditioning circuit for ADC.

In Fig. 4.1, following the signal direction from the input, the first Op-Amp is connected as a voltage follower. It provides a high input impedance for the purpose of eliminating the loading effect of the source signal. This is also the reason why a high-value voltage-dividing resistance is chosen on the input. The two resistances of $850k\Omega$ and $150k\Omega$ scale the input voltage of $\pm 10V$ to $\pm 1.5V$. The second Op-Amp is a summing amplifier, which shifts the input voltage by $1.5V$, but the signal polarity is inverted. The output of this Op-Amp is $-3V \sim 0V$. The third Op-Amp will invert the signal back to $0V \sim 3V$. In the circuit design, the resistance is set to be in $k\Omega$-range, because $M\Omega$-range resistor will cause excessive thermal noise [32] and Ω-range resistor will consume more power. Practical operational amplifiers draw a small current from each of their inputs due to bias requirements and leakage [32]. This bias current will cause voltage drop across the resistors connected to the input. This can be seen in Fig. 4.2(a) where the non-inverting pin of the amplifier is connected to the ground. According to the virtual short-circuit property of operational amplifiers, the electric potential of the inverting input pin is also $0V$. In ideal case, the output voltage of the Op-Amp should also be $0V$. However as bias current exists,

$$V_{out} = V_- + R_f I_b = 0V + R_f I_b = R_f I_b \neq 0.$$

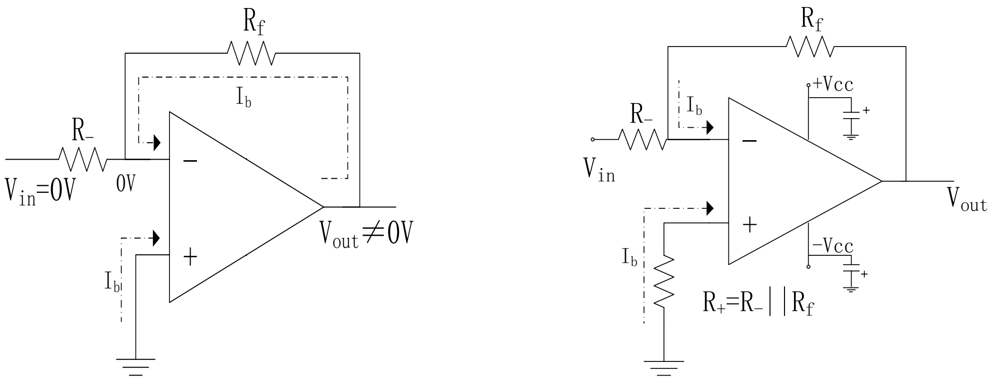
The solution to compensate for this voltage drop is to let the resistance seen from each input be matched, as shown in Fig. 4.2(b),

$$V_- = V_+ = -R_+ I_b \tag{4.1}$$

$$I_b = \frac{V_{in} - V_-}{R_-} + \frac{V_{out} - V_-}{R_f}. \tag{4.2}$$

If $V_{in} = 0V$, it is required that $V_{out} = 0V$ as well. Substitute $V_{in} = V_{out} = 0V$ and Eq. 4.1 into Eq. 4.2, it can be found that the compensating resistor should be selected as:

$$R_+ = \frac{R_f R_-}{R_f + R_-} = R_f \parallel R_-.$$



(a)

(b)

Fig. 4.2 Bias current effect and compensation

The compensating resistor connected to the non-inverting input is used for both the second and the third Op-Amps. The output of an operational amplifier cannot swing negative when operated from a single positive supply [6]. Two of the three amplifiers in the conditioning circuit will output negative voltages and due to this fact both a negative and a positive voltage supplies are needed. Due to wire inductance, the power may not be passed to the amplifier from the voltage source immediately, which could cause waveform distortion when the input signal changes fast. For each amplifier package, two tantalum capacitors are connected between $+V_{cc}$ and the ground as well as $-V_{cc}$ and the ground, which can supply energy to the amplifier temporarily, as shown in Fig. 4.2(b). These two capacitors can also reduce the noises coming from the voltage source.

4.2.2 ADC Calibration

In ideal conditions, the input-output relation of the conditioning circuit should be:

$$V_{ADCx} = \frac{3}{20} \times V_{Chx} + 1.5V$$
$$V_{Chx} = \frac{20}{3} \times V_{ADCx} - 10.0V$$

However, mainly due to the inaccurate resistance value, there will be a small gain error in the conditioning circuit. The metal film resistor used in this project has a tolerance of $\pm 1\%$, from the nominal value. Even though there is a gain error, there is still a linear relation between V_{ADCx} and V_{Chx}. This relation can be described as:

$$\begin{aligned} V_{ADCx} &= A_x \times V_{Chx} + B_x \\ V_{Chx} &= A\,'_x \times V_{ADCx} + B\,'_x. \end{aligned}$$

To find the actual coefficients A_x, B_x, $A\,'_x$ and $B\,'_x$ of the conditioning circuits, a set of voltages are applied to the input, and the corresponding output voltage is measured. In Table 4.1, the measured input-output relation for Channel 1 is shown.

Table 4.1: Measured input and output voltages for Channel 1

$V_{Ch1}(V)$	-10.05	-9.59	-9.04	-8.52	-8.01	-7.56	-7.04	-6.49	-6.06	-5.5	-5.02
$V_{ADC1}(V)$	0.004	0.072	0.153	0.231	0.307	0.373	0.45	0.532	0.595	0.678	0.75
$V_{Ch1}(V)$	-4.47	-4.05	-3.499	-2.995	-2.507	-2.014	-1.498	-1.08	-0.505	0	0.49
$V_{ADC1}(V)$	0.83	0.892	0.975	1.051	1.123	1.196	1.273	1.335	1.42	1.496	1.569
$V_{Ch1}(V)$	1.008	1.513	2.024	2.488	3.023	3.508	3.985	4.51	5.0	5.52	6.05
$V_{ADC1}(V)$	1.646	1.721	1.797	1.867	1.946	2.019	2.090	2.169	2.241	2.318	2.397
$V_{Ch1}(V)$	6.44	7.02	7.47	8.03	8.51	9.06	9.51	10.07			
$V_{ADC1}(V)$	2.455	2.542	2.609	2.693	2.764	2.846	2.913	2.996			

In Matlab the command $P = polyfit(V_{Chx}, V_{ADCx}, 1)$ can be applied to get $P_x = [A_x, B_x]$ and command $P_{x,inv} = ployfit(V_{ADCx}, V_{Chx}, 1)$ results in $P_{inv} = [A'_x, B'_x]$. Five conditioning circuits have been built for five channels of analog inputs. The calibration results for these five channels are shown in Table 4.2. Comparing the calibration results with the ideal values of A, B, A' and B', it could be concluded that the five conditioning circuits behave correctly as designed, but with some coefficient errors being less than 1%.

Table 4.2: Calibration results of five ADC conditioning circuits

A_1	0.1488	B_1	1.4969	A'_1	6.7213	B'_1	-10.0609
A_2	0.1500	B_2	1.4976	A'_2	6.6660	B'_2	-9.9828
A_3	0.1485	B_3	1.4927	A'_3	6.7356	B'_3	-10.0541
A_4	0.1502	B_4	1.4976	A'_4	6.6595	B'_4	-9.9735
A_5	0.1481	B_5	1.4877	A'_5	6.7514	B'_5	-10.0443

The Analog-to-Digital Converter in the DSP has a 12-bit resolution and a conversion range of $0V \sim 3V$. From this the input voltage to the ADC, V_{ADCx}, can be calculated from the register value, Reg_x, as

$$V_{ADCx} = Reg_x \times \frac{3}{4096}. \tag{4.3}$$

The nominal input and output relations of the current sensors, the DC-link voltage sensor and the tachometer are [21]:

$$V_{cx} = 2.5 + 0.0625 i_{sx} \tag{4.4a}$$

$$V_{DC,s} = 0.0965 V_{DC} \tag{4.4b}$$

$$V_{tacho} = \frac{1}{600} \Omega_r, \tag{4.4c}$$

where i_{sx} is the stator phase current in *A*, V_{cx} is the output of the current sensor in *V* and x could be a, b or c indicating which phase that is measured.V_{DC} is the DC-link voltage and $V_{DC,s}$ is the output of the DC-link voltage sensor. Ω_r is the rotating speed of the rotor in *rpm* and V_{tacho} is the output of the tachometer in *V*.

If Channel 1-5 of the analog inputs are used to measure the stator phase currents(i_{sa}, i_{sb} and i_{sc}), the DC link voltage V_{DC} and the machine speed Ω_r respectively, then these five quantities to be measured can be expressed as

$$i_{sa} = \frac{1}{0.0625} \times ((Reg_1 \times \frac{3}{4096} \times A'_1 + B'_1) - 2.5) \tag{4.5a}$$

$$i_{sb} = \frac{1}{0.0625} \times ((Reg_2 \times \frac{3}{4096} \times A'_2 + B'_2) - 2.5) \tag{4.5b}$$

$$i_{sc} = \frac{1}{0.0625} \times ((Reg_3 \times \frac{3}{4096} \times A'_3 + B'_3) - 2.5) \tag{4.5c}$$

$$V_{DC} = \frac{1}{0.0965} \times (Reg_4 \times \frac{3}{4096} \times A'_4 + B'_4) \tag{4.5d}$$

$$\Omega_r = 600 \times (Reg_5 \times \frac{3}{4096} \times A'_5 + B'_5) \tag{4.5e}$$

4.3 Digital Signal

The digital signals used in this project are three input pulse signals from the encoder, six output PWM signals for the inverter and five normal input and output signals. The F28335 DSP employs Low-Voltage TTL(LVTTL), which defines that $V_{CC} = 3.3V; V_{OH} \geqslant 2.4V, V_{OL} \leqslant 0.4V; V_{IH} \geqslant 2V, V_{IL} \leqslant 0.8V$. The digital signals can not be connected to the DSP directly until they have been conditioned since they are in the range from 0V to 5V and not in the range from 0V to 3.3V as those produced by the DSP.

4.3.1 Interface for Encoder

The speed of the motor could be obtained by taking the derivative of the motor position got from an incremental encoder mounted on the motor shaft. The electric signal generated by the incremental encoder is of TIA-422-B standard [7]. By this standard, the TIA-422 receiver receives two differential voltage inputs, inputA and inputB. If inputA is $200mV$ higher than inputB the output is high level and if inputB is $200mV$ higher than inputA the output is low level. As mentioned before, the DSP F28335 applies a LVTTL standard that is different from the TIA-422-B standard. There are some different types of 422 receiver ICs available to convert TIA-422 logic to TTL. When selecting an appropriate chip, several issues should be concerned, such as the supply voltage, output voltage, number of receivers per package and the signaling rate. There is an 1000-line encoder available for the project and its tolerable maximum rotational speed is $10000rpm$. From these parameters, the maximum data sequence rate from one encoder phase can be calculated as

$$\begin{aligned} Data\,Rate(bit/s) &= Number\,of\,Lines \times 2 \times \frac{Maximum\,Rotational\,Speed}{60} \\ &= 1000 \times 2 \times \frac{10000}{60} = 0.33(Mbps). \end{aligned} \tag{4.6}$$

The supply voltage of the receiver chip should be near $5V$ and it would be an advantage if there are at least 3 receivers per package. The chip's signaling rate must be larger than $0.33M$ and the output of the receiver should be compatible with LVTTL. Texas Instruments' AM26LS32A meets all the requirements above except the output voltage, which is TTL. A voltage divider should be applied to avoid destroying the inputs of the DSP. After tests, the high-level output of this chip is measured to be $4.4V$. The input voltage range of F25335 DSP is $0 \sim 3.3V$, so the fractional ratio of the voltage divider is set to be $\frac{2}{3}$. The circuit schematic is shown in Fig. 4.3. The signals from the encoder are fed into the interface board through a type of 15-pin D-Sub connector DB-15 with the same pin configuration as the dSPACE system.

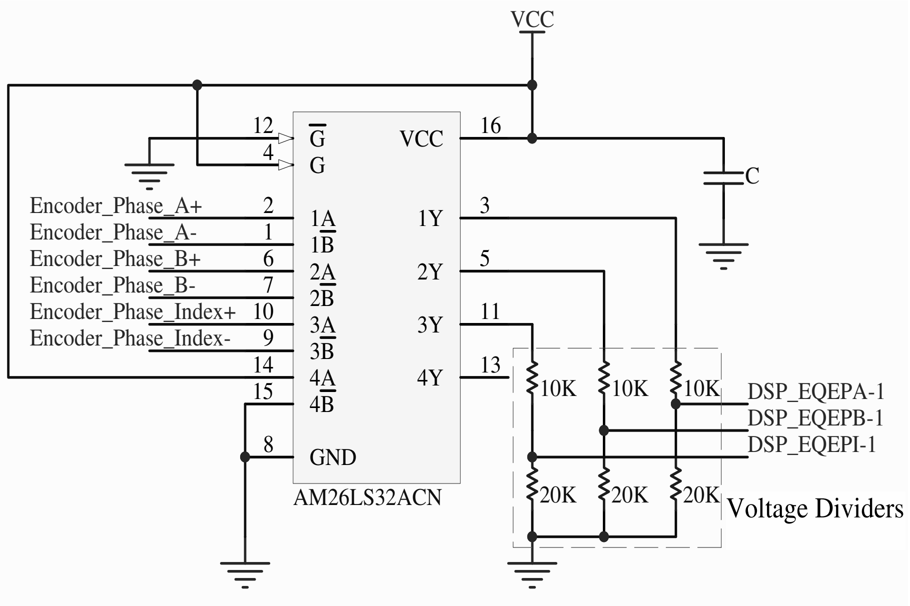


Fig. 4.3 Circuit schematic for encoder interface

4.3.2 Interfaces for PWM and Digital I/O

From the technical specifications it can be noticed that the LVTTL is compatible with TTL [35]. However, for digital inputs, a buffer chip can provide an isolation between the DSP and the outside to protect the DSP. For digital outputs, a line driver chip can improve the output drive capacity. Texas Instruments' SN74LS241 is chosen as both input buffer and output driver. For the PWM outputs, the chip's signaling rate should be considered. Compared with PWM cycle period of 0.1ms, SN74LS241's signaling time of around 10ns can be neglected. For the digital inputs, a voltage divider is still necessary between the input buffer and the DSP, in the same way as the encoder interface but with a different voltage divider fraction of $\frac{3}{5}$. The designing procedure is similar to that of the encoder interface and therefore it will not be repeated. What needs to be mentioned is that though the digital I/O interface of DS1103 is a male 50-pin D-Sub, female BNC connectors are still used as digital I/O interfaces, as the digital interfaces of the existing inverter are also BNC connectors.

4.4 Voltage Source and Reference

4.4.1 +5V Voltage Source

There are plug-in DC power supplies available in the electronic market, which can provide $5V$ directly. However as these, nowadays, usually are switching-mode supplies, there exists a residual ripple on the output, which is about from $30mV$ to $120mV$ [5]. For the digital circuit, this voltage ripple can be accepted. But for analog circuits, the ripple should be as small as possible in order to guarantee the measurement accuracy. The switching frequency of the used power supply is $130kHz$ [5], so the ripple frequency might be $260kHz$. According to the data sheet [11] and the typical power supply ripple rejection(PSRR) curve [12], the linear regulator 7805 has a ripple rejection ratio of $40dB$ at $260kHz$. By the definition of PSRR that

$$PSRR = 20log\frac{Ripple_{input}}{Ripple_{output}},$$

when $PSRR = 40dB$,

$$\frac{Ripple_{input}}{Ripple_{output}} = 10^{\frac{40}{20}} = 100 \Rightarrow Ripple_{output} = \frac{Ripple_{input}}{100},$$

which means that the voltage ripple generated by switching-mode supply will be reduced to $0.3 \sim 1.2mV$ by the linear regulator 7805.

However the 7805 functions properly only when the voltage drop across it is larger than $2V$. Therefore a $9V$ plug-in DC power supply is chosen to work together with the 7805 linear regulator. A typical 7805 application circuit is shown in Fig. 4.4.

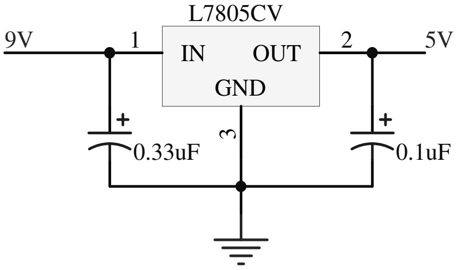


Fig. 4.4 Typical 7805 application circuit

A heat sink is required for cooling the 7805 circuit. The heat sink specification can be calculated from a thermal resistance model as

$$T_j = T_a + (R_{\theta jc} + R_{\theta cs} + R_{\theta sa}) \times P, \tag{4.7}$$

where T_j is the junction temperature, which according to the 7805 data sheet [11] should be less than 150°C. T_a is the ambient temperature, which can be estimated as room temperature 20°C. $R_{\theta jc}$ is the thermal resistance from junction to the casing of the 7805 package, which is 5°C/W [11]. $R_{\theta cs}$ is the thermal resistance from the casing to the heat sink. If the heat sink is directly attached onto the 7805, it is approximately 0.4°C/W [21]. $R_{\theta sa}$ is the thermal resistance from the heat sink to the ambient. The smaller it is, the better cooling capacity the sink heat has. P denotes the power dissipated in the 7805, which is the product of the current through and the voltage drop across the 7805. The voltage drop is fixed to be $4V$. The current is measured under the condition that the DSP and the interface board run at a full load. The experimental value of the current is $0.486A$, but in order to keep a safe margin, the current is doubled to be $1A$. This gives a power loss of 4W in the 7805 circuit. Substitute T_j, T_a, $R_{\theta jc}$,$R_{\theta cs}$ and P into Eq. 4.7, it can be found that $R_{\theta sa}$ should not be larger than 27°C/W. A heat sink of 12°C/W is selected in the project.

4.4.2 -5V Voltage Source

The $-5V$ voltage is generated by Texas Instruments' MC34063 voltage regulator. The MC34063 is a fixed-frequency switching-mode regulator, which requires minimal external components for building the DC-to-DC converter topologies boost, buck and inverting [19]. Fig. 4.5 shows the schematic of MC34063's application circuit that outputs a negative voltage.

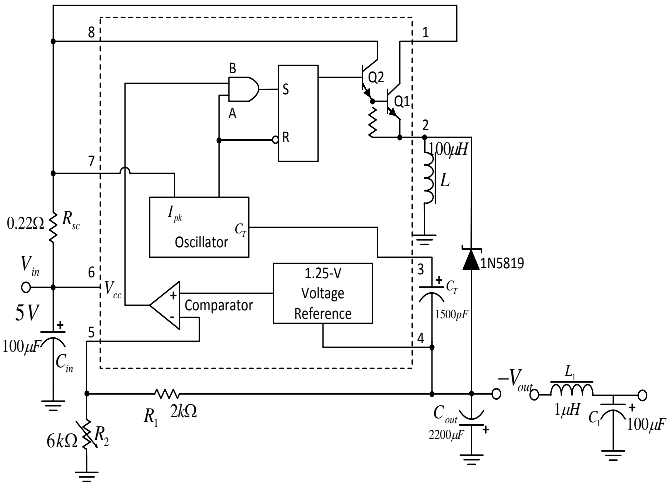


Fig. 4.5 Circuit schematic of MC34063 inverting regulator

The inverting DC-to-DC topology can be simplified with the diagram in Fig. 4.6, where the NPN Darlington transistor(Q1Q2) and the Schottky diode 1N5819 in Fig. 4.5 are replaced with the transistor(Q) and diode(D) respectively.

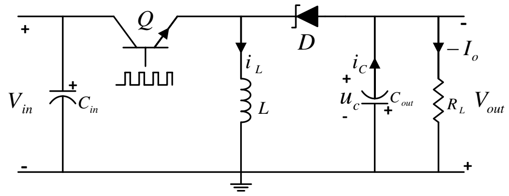


Fig. 4.6 Diagram of inverting-topology DC-to-DC regulator

When the transistor Q in Fig. 4.6 is turned on, V_{in} is directly applied on the inductor, leading to an increasing current in the inductor as can be noticed in the i_L subfigure of Fig. 4.7. Meanwhile, C_{out} draws current I_{out} from the load as the voltage across it is negative, which makes the output voltage increase. After the transistor is turned off, the inductor has to draw current i_L through the diode D as the current in the inductor cannot stop immediately. Though at this moment current I_{out} is still drawn from the load, i_L is larger than I_{out}, which means that the net current is drawn from C_{out} and the output voltage decreases. The transistor control circuit consists of a comparator, an oscillator, an AND gate, a NOT gate and a RS flip-flop. Fig. 4.7 shows the waveforms of the MC34063 when it is working in the inverting topology and in steady state. The comparator compares the values of $u_c\frac{R_2}{R_1+R_2}$ and $u_c + 1.25$, which results in that the

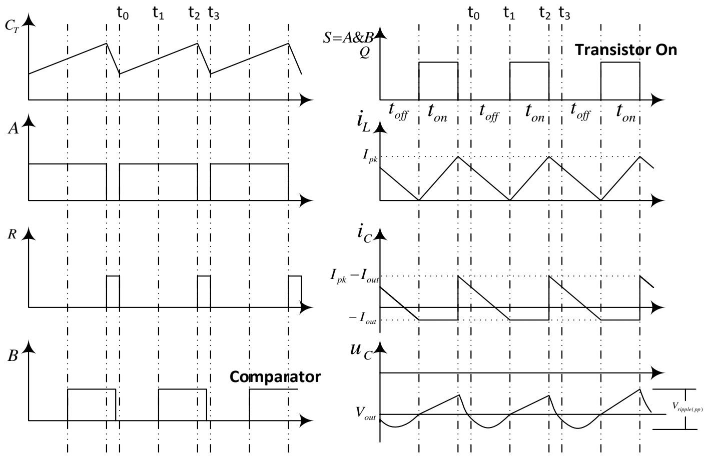


Fig. 4.7 Waveforms of the inverting DC-to-DC regulator

comparator outputs ”1” if $u_c > -1.25(1+\frac{R_2}{R_1})$ and outputs ”0” if $u_c < -1.25(1+\frac{R_2}{R_1})$. As u_c is actually the output voltage, which is almost equal to the desired output voltage $-V_{out}$ but with some ripples due to the switching operations. So the desired output voltage $-V_{out}$ has a relation with R_1 and R_2 as

$$-V_{out} = -1.25(1+\frac{R_2}{R_1}).$$

The oscillator charges and discharges the external timing capacitor C_T. When the voltage across the C_T touches the upper threshold or the lower threshold, the output of the oscillator will be toggled. The switching frequency f is only decided by the value of C_T. The waveform of the oscillator output is the same as the A input of the AND gate, as they are connected together as shown in Fig. 4.5. The B input of the AND gate are connected with the output of the comparator. The output of the oscillator is also connected to the R(Reset) input of the RS flip-flop through a NOT gate. The S(Set) input of the flip-flop is connected to the output of the AND gate. Besides, the output of the flip-flop is connected to the the base of the Darlington transistor(Q1Q2). In this way of connection, the transistor is turned on only when both the comparator and the oscillator output ”1”. As shown in Fig. 4.7, at t_0 moment, the oscillator output ”1” but u_c is lower than $-V_{out}$, so the transistor is still off. Between t_0 and t_1, u_c first decreases and then increase as the current i_c from the the C_{out} is changing from being positive to being negative. At t_1, u_c crosses the desired output voltage $-V_{out}$, transistor is turned on. u_c keeps increasing until the oscillator is toggled to ”0” that the transistor is forced to be turned off at t_2. The inductor starts to draw current from the C_{out} and u_c deceases. From t_3, the procedure above will be repeated. The R_{sc} is used to limit the current. When the voltage drop across R_{sc} is larger than $330mV$ [16], the oscillator will provide an additional current path to the timing capacitor, in order to accelerate the capacitor charging and shorten the on-time of the transistor, which will reduce the energy stored in the inductor.

The value of the MC34063’s peripheral components is selected based on calculation table in the reference [19, p.10]. Firstly, the designing specifications and some components’ parameters are listed below.

- Switching Frequency: $f = 15kHz$
- Input voltage: $V_{in} = 5V$
- Output voltage:$-V_{out} = -5V$
- Output current:$-I_{out} = -0.5A$
- Maximum voltage ripple:$V_{ripple(pp)} = 100mV$
- 1N5819 forward voltage drop:$V_F = 0.5V$ [15]
- NPN Darlington transistor saturation voltage:$V_{sat} = 1V$ [19]

The calculation is done in the following way, where I_{pk} is the peak value of inductor current and the L_{min} is the minimum value the inductance should be.

$$\begin{aligned}
t_{on}+t_{off} &= \frac{1}{f} = 6.6\times10^{-5}s \\
\frac{t_{on}}{t_{off}} &= \frac{V_{out}+V_F}{V_{in}-V_{sat}} = 1.3875 \\
t_{on} &= 3.8356\times10^{-5}s \\
t_{off} &= 2.76\times10^{-5}s \\
C_T &= 4\times10^{-5}t_{on} = 1.543nF \\
I_{pk} &= 2I_{out}(\frac{t_{on}}{t_{off}}+1) = 2.3875A \\
L(min) &= (\frac{V_{in}-V_{sat}}{I_{pk}})t_{on} = 6.42\times10^{-5}H \\
R_{sc} &= \frac{0.3}{I_{pk}} = 0.13\Omega \\
C_{out} &= \frac{9I_{out}t_{on}}{V_{ripple(pp)}} = 1.7\times10^{-3}F
\end{aligned}$$

For some components, it might be hard to find the type with the exact value expected. So according to the product list of `www.elfa.se`, the actual components’s value used in this circuit are

$C_{in} = 100\mu F$	$C_{out} = 2200uF$
$R_{sc} = 0.22\Omega$	$R_1 = 2k\Omega$
$L = 100\mu H$	$R_2 = 6k\Omega$(Potentiometer)
$C_T = 1500pF$	

In Ref. [16], it is mentioned that the minimum value of R_{sc} should be 0.2Ω, so 0.22Ω resistor is chosen though the calculated R_{sc} is 0.13Ω. Besides, in order to make the output voltage adjustable, R_2 is replaced with a potentiometer with $10k\Omega$ range.

4.4.3 2.5V Voltage Reference

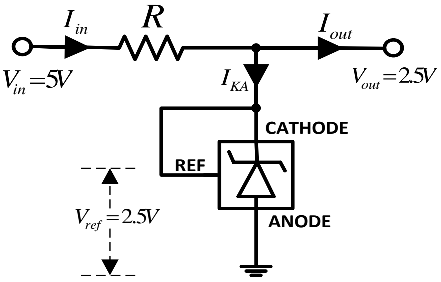


Fig. 4.8 TL431 typical application circuit

The $2.5V$ voltage reference used in ADC conditioning circuit is obtained by TI's TL431. The TL431 is a shunt voltage reference, which works similarly to Zener diodes. A typical application circuit for the TL431 is shown in Fig. 4.8. The output of this circuit is connected to the five ADC conditioning circuits shown in Fig.4.1 and from the figure the output current I_{out} could be calculated as

$$I_{out} = 5 \times \frac{2.5V - 0V}{166.6k\Omega} = 0.075mA.$$

The input of the circuit is connected to the $5V$ supply, giving an input current of

$$\begin{aligned} I_{in} &= \frac{V_{in} - V_{out}}{R} = \frac{2.5V}{R} \\ &= I_{KA} + I_{out}, \end{aligned}$$

where the I_{KA} is the cathode current. Then cathode current can be calculated as

$$I_{KA} = \frac{2.5V}{R} - I_{out} = \frac{2.5V}{R} - 0.075mA.$$

The recommended operating conditions of I_{KA} is from $1mA$ to $100mA$ [25], which gives

$$\begin{aligned} 1mA < \frac{2.5V}{R} - 0.075mA < 100mA \\ \Rightarrow 24.98\Omega < R < 2.33k\Omega. \end{aligned}$$

In this project, R is selected to be $1k\Omega$.

Chapter 5

Vector Control Design

One of the core concepts of vector control is to replace the three-phase sinusoidal quantities with two DC quantities so that the control strategies effectively used on DC machine could now be applied to induction machines. To achieve this aim, a series of coordinate transformations, flux estimation, decoupling and regulating will be done. The procedure of vector control design is introduced in detail in this chapter. Firstly, an overview of vector control design is exhibited as

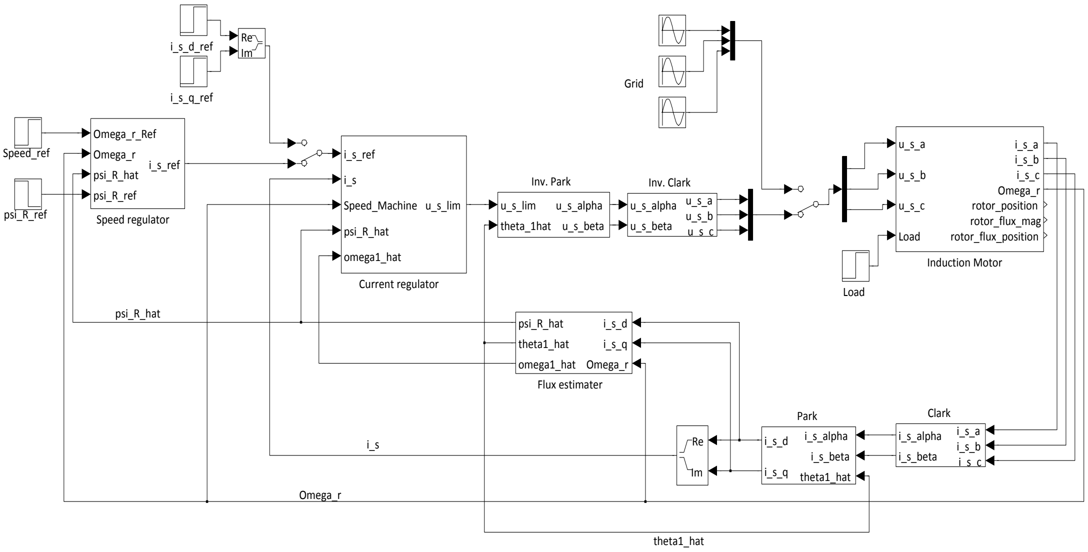


Fig. 5.1 Overview of the vector control block diagram.

5.1 Dynamic Models of Induction Motor

5.1.1 T Model

The dynamic model of the induction machine is described by using space vectors. There are many types of dynamic models for the induction machine. Among them, the T model defined in the stationary $\alpha - \beta$ system is chosen to be used to model the induction machine in the simulation. This kind of induction motor model could be expressed with Eq. 5.1, Eq. 5.2 and Eq. 5.3.

The stator and rotor dynamic equations:

$$\vec{u}_s^s = R_s \vec{i}_s^s + \frac{d\vec{\Psi}_s^s}{dt} \tag{5.1a}$$

$$\vec{u}_r^s = 0 = R_r \vec{i}_r^s + \frac{d\vec{\Psi}_r^s}{dt} - j\omega_r \vec{\Psi}_r^s \tag{5.1b}$$

The stator and rotor flux linkage equations:

$$\vec{\Psi}_s^s = L_s\vec{i}_s^s + L_m\vec{i}_r^s = L_{sl}\vec{i}_s^s + L_m\vec{i}_m^s \tag{5.2a}$$

$$\vec{\Psi}_r^s = L_r\vec{i}_r^s + L_m\vec{i}_s^s = L_{rl}\vec{i}_r^s + L_m\vec{i}_m^s \tag{5.2b}$$

The produced electromagnetic torque equation:

$$T_e = \frac{3n_p}{2}Im\{\vec{\Psi}_s^{s*}\vec{i}_s^s\} = \frac{3n_p}{2}(\Psi_{s\alpha}i_{s\beta} - \Psi_{s\beta}i_{s\alpha}) \tag{5.3}$$

Where:
Superscript of X^s indicates stationary coordinate is taken as reference. Subscript of X_s or X_r means the variable belongs to stator or rotor and subscript X_α and X_β are used to make clear distinction between quantity of α direction and of β direction. The quantities used in the induction motor model are listed below.

$\vec{u}_s^s = u_{s\alpha} + ju_{s\beta}$: stator voltage
$\vec{u}_r^s = u_{r\alpha} + ju_{r\beta}$: rotor voltage
$\vec{i}_s^s = i_{s\alpha} + ji_{s\beta}$: stator current
$\vec{i}_r^s = i_{r\alpha} + ji_{r\beta}$: rotor current
$\vec{i}_m^s = \vec{i}_s^s + \vec{i}_r^s$: magnetizing current
$\vec{\Psi}_s^s = \Psi_{s\alpha} + j\Psi_{s\beta}$: stator flux linkage
$\vec{\Psi}_r^s = \Psi_{r\alpha} + j\Psi_{r\beta}$: rotor flux linkage
ω_r: electrical rotor angular speed
R_s: stator phase resistance

R_r: rotor phase resistance
L_s: stator phase inductance
L_r: rotor phase inductance
L_m: mutual inductance
$L_{sl} = L_s - L_m$: stator leakage inductance
$L_{rl} = L_r - L_m$: stator leakage inductance
n_p: number of pole pair
T_e: electromagnetic torque

The dynamic circuit described by Eq. 5.1 and Eq. 5.2 is known as the T-form circuit and it is shown in Fig. 5.2.

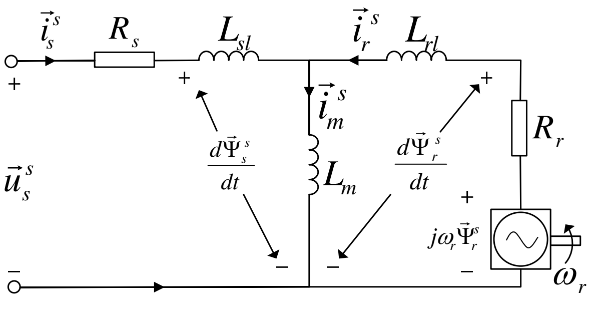


Fig. 5.2 Equivalent T circuit of the induction motor dynamic model.

Substitute Eq. 5.2 into Eq. 5.1, the dynamic relation between voltage and current could be obtained as:

$$\begin{aligned}\vec{u}_s^s &= R_s\vec{i}_s^s + L_s\frac{\mathrm{d}\vec{i}_s^s}{\mathrm{dt}} + L_m\frac{\mathrm{d}\vec{i}_r^s}{\mathrm{dt}}\\ 0 &= R_r\vec{i}_r^s + L_r\frac{\mathrm{d}\vec{i}_r^s}{\mathrm{dt}} + L_m\frac{\mathrm{d}\vec{i}_s^s}{\mathrm{dt}}\end{aligned}$$

from which the time derivatives of $i_{s\alpha}$, $i_{s\beta}$, $i_{r\alpha}$ and $i_{r\beta}$ could be solved as Eq. 5.4.

$$\frac{\mathrm{d}i_{s\alpha}}{\mathrm{d}t} = \frac{-L_ru_{s\alpha} + L_rR_si_{s\alpha} - L_mR_ri_{r\alpha} - L_m\omega_rL_ri_{r\beta} - L_m\omega_rL_mi_{s\beta}}{L_m^2 - L_rL_s} \tag{5.4a}$$

$$\frac{\mathrm{d}i_{s\beta}}{\mathrm{d}t} = \frac{-L_ru_{s\beta} + L_rR_si_{s\beta} - L_mR_ri_{r\beta} + L_m\omega_rL_ri_{r\alpha} + L_m\omega_rL_mi_{s\alpha}}{L_m^2 - L_rL_s} \tag{5.4b}$$

$$\frac{\mathrm{d}i_{r\alpha}}{\mathrm{d}t} = \frac{L_mu_{s\alpha} + L_sR_ri_{r\alpha} - L_mR_si_{s\alpha} + L_s\omega_rL_ri_{r\beta} + L_s\omega_rL_mi_{s\beta}}{L_m^2 - L_rL_s} \tag{5.4c}$$

$$\frac{\mathrm{d}i_{r\beta}}{\mathrm{d}t} = \frac{L_mu_{s\beta} + L_sR_ri_{r\beta} - L_mR_si_{s\beta} - L_s\omega_rL_ri_{r\alpha} - L_s\omega_rL_mi_{s\alpha}}{L_m^2 - L_rL_s} \tag{5.4d}$$

Eq. 5.4 is used for building the model of the induction motor in Matlab/Simulink.

5.1.2 Inverse Γ Model

The three inductor currents $\vec{i}_m^s$, $\vec{i}_s^s$ and $\vec{i}_r^s$ are not linearly independent, as $\vec{i}_m^s = \vec{i}_s^s + \vec{i}_r^s$. One leakage inductance - not two- is sufficient for modeling the behavior of the induction motor from the stator to the shaft. Due to the fact that the T-model is over-parameterized, it is not good for dynamic analysis or controller design [39]. The desired equivalent circuit of the induction motor dynamic model is shown in Fig. 5.3, where the rotor leakage inductance is removed.

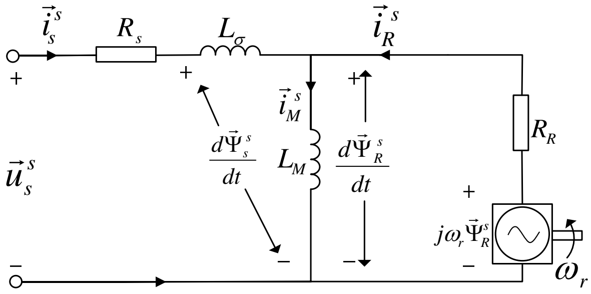


Fig. 5.3 Equivalent Inv. Γ circuit of the induction motor dynamic model.

To remove the rotor leakage inductance from the T-model, a transformation constant b is introduced and two new variables are defined as:

$$\vec{\Psi}_R^s = b\vec{\Psi}_r^s \tag{5.5}$$

$$\vec{i}_R^s = \frac{\vec{i}_r^s}{b}. \tag{5.6}$$

Combine Eq. 5.2b, Eq. 5.5 and Eq. 5.6, $\vec{\Psi}_R^s$ could be expressed as:

$$\vec{\Psi}_R^s = b\vec{\Psi}_r^s = b(L_r b\vec{i}_R^s + L_m\vec{i}_s^s) = b^2 L_r\vec{i}_R^s + bL_m\vec{i}_s^s. \tag{5.7}$$

From Fig. 5.3, it can be found that $\vec{\Psi}_R^s = L_M(\vec{i}_s^s + \vec{i}_R^s)$, which also requires that $\vec{i}_R^s$ and $\vec{i}_s^s$ in Eq. 5.7 have the same coefficients. So

$$b^2 L_r = bL_m \Rightarrow b = \frac{L_m}{L_r} \tag{5.8}$$

$$\vec{\Psi}_R^s = \frac{L_m^2}{L_r}(\vec{i}_s^s + \vec{i}_R^s) = L_M(\vec{i}_s^s + \vec{i}_R^s) \Rightarrow L_M = \frac{L_m^2}{L_r} \tag{5.9}$$

Substitute the newly-defined b, $\vec{i}_R^s$ and L_M into Eq. 5.2a, then L_σ can be derived as

$$\begin{aligned}\vec{\Psi}_s^s &= L_s\vec{i}_s^s + L_m\vec{i}_r^s = L_s\vec{i}_s^s + bL_m\vec{i}_R^s = L_s\vec{i}_s^s + \frac{L_m^2}{L_r}\vec{i}_R^s = L_s\vec{i}_s^s + L_M\vec{i}_R^s \\ &= (L_\sigma + L_M)\vec{i}_s^s + L_M\vec{i}_R^s \Rightarrow L_\sigma = L_s - L_M\end{aligned} \tag{5.10}$$

To obtain the expression of R_R, Eq. 5.6 and Eq. 5.5 are substituted into Eq. 5.1b, then

$$\begin{aligned}0 &= \vec{u}_r^s = bR_r\vec{i}_R^s + \frac{1}{b}\frac{\mathrm{d}\vec{\Psi}_R^s}{\mathrm{d}t} - j\frac{1}{b}\omega_r\vec{\Psi}_R^s = b^2R_r\vec{i}_R^s + \frac{\mathrm{d}\vec{\Psi}_R^s}{dt} - j\omega_r\vec{\Psi}_R^s \\ &= R_R\vec{i}_R^s + \frac{\mathrm{d}\vec{\Psi}_R^s}{\mathrm{d}t} - j\omega_r\vec{\Psi}_R^s \Rightarrow R_R = b^2R_r\end{aligned} \tag{5.11}$$

Based on the new variables and parameters defined for the inverse Γ model, the dynamic equations and flux linkage equations in T model are rearranged as:

$$\vec{u}_s^s = R_s\vec{i}_s^s + \frac{d\vec{\Psi}_s^s}{dt} \tag{5.12a}$$

$$0 = R_R\vec{i}_R^s + \frac{\mathrm{d}\vec{\Psi}_R^s}{\mathrm{d}t} - j\omega_r\vec{\Psi}_R^s \tag{5.12b}$$

$$\vec{\Psi}_s^s = L_s\vec{i}_s^s + L_M\vec{i}_R^s = (L_M + L_\sigma)\vec{i}_s^s + L_M\vec{i}_R^s \tag{5.12c}$$

$$\vec{\Psi}_R^s = L_M\vec{i}_s^s + L_M\vec{i}_R^s \tag{5.12d}$$

In the vector control system, the stator current $\vec{i}_s^s$ and the rotor flux $\vec{\Psi}_R^s$ are the variables to be controlled, to the contrary, the stator flux $\vec{\Psi}_s^s$ and the rotor current $\vec{i}_R^s$ are not expected to appear in the dynamic equations. Derived from Eq. 5.12c and Eq. 5.12d, $\vec{\Psi}_s^s$ and $\vec{i}_R^s$ can be re-expressed using $\vec{i}_s^s$ and $\vec{\Psi}_R^s$ as:

$$\vec{\Psi}_s^s = L_\sigma \vec{i}_s^s + \vec{\Psi}_R^s \tag{5.13a}$$

$$\vec{i}_R^s = \frac{\vec{\Psi}_R^s - L_M \vec{i}_s^s}{L_M}. \tag{5.13b}$$

Insert Eq. 5.13 into Eq. 5.12a and Eq. 5.12b, the dynamic equations with $\vec{i}_s^s$ and $\vec{\Psi}_R^s$ as state variables could be obtained as:

$$L_\sigma \frac{\mathrm{d}\vec{i}_s^s}{\mathrm{d}t} = \vec{u}_s^s - (R_s + R_R)\vec{i}_s^s - (j\omega_r - \frac{R_R}{L_M})\vec{\Psi}_R^s \tag{5.14a}$$

$$\frac{\mathrm{d}\vec{\Psi}_R^s}{\mathrm{d}t} = R_R \vec{i}_s^s - (\frac{R_R}{L_M} - j\omega_r)\vec{\Psi}_R^s \tag{5.14b}$$

So far, both the T model and the inverse Γ model of the induction motor are built in stationary $\alpha - \beta$ coordinates. In order to get DC quantities, it is necessary to transfer Eq. 5.14 from the stationary coordinates into the rotating coordinates, which is referred to as $d - q$ coordinate.

The $d - q$ coordinate system has the same rotating speed as the rotor flux. Besides, the d-axis of $d - q$ coordinate is the direction in which the rotor flux is oriented. If the angle between the $d - q$ coordinate system and the $\alpha - \beta$ coordinate system is θ_1 and the rotating speed of the $d - q$ coordinate system is ω_1, then the dynamic equations of the induction motor in $d - q$ coordinates could be achieved as:

$$L_\sigma \frac{\mathrm{d}\vec{i}_s}{\mathrm{d}t} = \vec{u}_s - (R_s + R_R + j\omega_1 L_\sigma)\vec{i}_s - (j\omega_r - \frac{R_R}{L_M})\vec{\Psi}_R \tag{5.15a}$$

$$\frac{\mathrm{d}\vec{\Psi}_R}{\mathrm{d}t} = R_R \vec{i}_s - (\frac{R_R}{L_M} + j(\omega_r - \omega_1))\vec{\Psi}_R, \tag{5.15b}$$

by replacing $\vec{i}_s^s$ with $\vec{i}_s e^{j\theta_1}$ and $\vec{\Psi}_R^s$ with $\vec{\Psi}_R e^{j\theta_1}$ in Eq. 5.14. In this report, if it has no superscript, the vector $\vec{X}_y$ is defined in $d - q$ coordinates. Eq. 5.15a and Eq. 5.15b are used for current regulator design and rotor flux estimation respectively.

5.2 Current Regulator

The current regulator is composed of feed forward, active damping, PI regulator and anti-windup. The complete structure of current regulators is illustrated in Fig. 5.4.

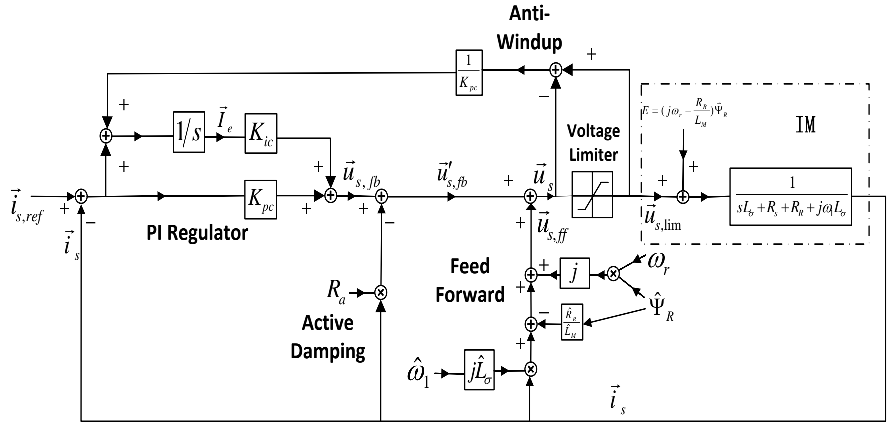


Fig. 5.4 Overview of current regulator

For current regulator, the input signals are:

- $\vec{i}_{s,ref}$, which is calculated based on the reference value of the torque and the rotor flux.

- $\vec{i}_s$, the stator current in $d-q$ coordinates, which is obtained from the Clark and Park transformation of the measured machine phase currents.
- ω_r, the electrical rotor angular speed, gotten by multiplying the number of pole pairs n_p with the mechanical rotor angular speed Ω_r measured by tachometer or other devices.
- $\hat{\omega}_1$ and $\hat{\Psi}_R$, the angular speed and the magnitude of the rotor flux $\vec{\Psi}_R$, which are calculated by flux estimator.

The output signal are:

- $\vec{u}_{s,lim}$, the output stator voltage saturated by voltage limiter. In Fig. 5.4, $\vec{u}_{lim}$ is directly applied on the machine. In reality, $\vec{u}_{lim}$ needs to be processed by Inv. Park and Inv. Clark transformations and the inverter, and then is applied on the induction machine as three-phase voltages.

5.2.1 Current Reference Determination

The torque expression of the T model in $\alpha-\beta$ coordinates has been given in Eq. 5.3 from which the torque expression of the Inv. Γ model in $d-q$ coordinates can be derived as:

$$\begin{aligned} T_e &= \frac{3n_p}{2} Im\{\vec{\Psi}_s^{s*}\vec{i}_s^s\} = \frac{3n_p}{2} Im\{(L_\sigma \vec{i}_s^s + \vec{\Psi}_R^s)^* \vec{i}_s^s\} \\ &= \frac{3n_p}{2} Im\{L_\sigma \vec{i}_s^{s*}\vec{i}_s^s + \vec{\Psi}_R^{s*}\vec{i}_s^s\} = \frac{3n_p}{2} Im\{\vec{\Psi}_R^{s*}\vec{i}_s^s\} \\ &= \frac{3n_p}{2} Im\{(\vec{\Psi}_R e^{j\theta_1})^* \vec{i}_s e^{j\theta_1}\} = \frac{3n_p}{2} Im\{\vec{\Psi}_R^* e^{-j\theta_1} \vec{i}_s e^{j\theta_1}\} \\ &= \frac{3n_p}{2} Im\{\vec{\Psi}_R^* \vec{i}_s\} = \frac{3n_p}{2} (\Psi_{Rd} i_{sq} - \Psi_{Rq} i_{sd}). \end{aligned} \tag{5.16}$$

The d-axis of the $d-q$ coordinate system is defined to have the same orientation as $\vec{\Psi}_R$, which means that Ψ_{Rq} is always 0 and the magnitude Ψ_R of $\vec{\Psi}_R$ equals to Ψ_{Rd}. The torque equation can be simplified as:

$$T_e = \frac{3n_p}{2} \Psi_R i_{sq}. \tag{5.17}$$

The reference value $i_{sq,ref}$ of i_{sq} can be derived from the desired torque $T_{e,ref}$ as:

$$i_{sq,ref} = \frac{2}{3n_p \hat{\Psi}_R} T_{e,ref}. \tag{5.18}$$

In steady state, the real part of Eq. 5.15b can be rewritten as

$$0 = R_R i_{sd} - \frac{R_R}{L_M} \Psi_{Rd} + (\omega_r - \omega_1) \Psi_{Rq}.$$

As $\Psi_{Rq} = 0$, the reference value of i_{sd} can be obtained as:

$$i_{sd,ref} = \frac{1}{L_M} \Psi_{Rd,ref} \tag{5.19}$$

The structure of current reference determination block can be found in Fig. 5.8 in Sec. 5.4.

5.2.2 Feed Forward

From the dynamic equation of the stator current $\vec{i}_s$ of the Inv. Γ model in $d-q$ coordinates,

$$L_\sigma \frac{\mathrm{d}\vec{i}_s}{\mathrm{d}t} = \vec{u}_s - (R_s + R_R + j\omega_1 L_\sigma)\vec{i}_s - (j\omega_r - \tfrac{R_R}{L_M})\vec{\Psi}_R, \tag{5.20}$$

it could be found that the term $(j\omega_r - \frac{R_R}{L_M})\vec{\Psi}_R$ needs to be removed in order to get a linear relation between $\vec{i}_s$ and $\vec{u}_s$. Besides, the existence of $j\omega_1 L_\sigma \vec{i}_s$ introduces the cross-coupling between d and q components. As cross-coupling and back-emf terms are not expected during current regulator design, a feed-forward signal $\vec{u}_{s,ff}$ is applied to eliminate the two terms and it is shown as:

$$\vec{u}_{s,ff} = \underbrace{j\omega_1 L_\sigma \vec{i}_s}_{Decoupling} + \underbrace{j\omega_r \hat{\vec{\Psi}}_R - \frac{R_R}{L_M}\hat{\vec{\Psi}}_R}_{EliminateBack-emf}. \tag{5.21}$$

Besides the feed-forward signal $\vec{u}_{s,ff}$, the actuating voltage also contains the feedback signal as

$$\vec{u}_s = \vec{u}'_{s,ff} + \vec{u}_{s,ff}, \tag{5.22}$$

which is also illustrated in Fig. 5.4. By inserting Eq. 5.21 and Eq. 5.22 into Eq. 5.20 and assuming perfect estimates, the feed-forward signal will eliminate the cross-coupling and the back-emf terms and Eq. 5.20 can be reduced to

$$L_\sigma \frac{\mathrm{d}\vec{i}_s}{\mathrm{d}t} = \vec{u}'_{s,fb} - (R_s + R_R)\vec{i}_s. \tag{5.23}$$

5.2.3 Active Damping

The transfer function $G(s)$ from $\vec{u}'_{s,fb}$ to i_s^k is $\frac{1}{sL_\sigma + R_s + R_R}$. A larger resistance R can provide better suppression of load disturbance [42]. However it is difficult to change the machine structure to increase the resistance, so a dummy damping resistance R_a is introduced in Eq. 5.23 as

$$\left.\begin{array}{l} L_\sigma \frac{\mathrm{d}\vec{i}_s}{\mathrm{d}t} = \vec{u}'_{s,fb} - (R_s + R_R)\vec{i}_s \\ \vec{u}'_{s,fb} = \vec{u}_{s,fb} - R_a\vec{i}_s \end{array}\right\} \Rightarrow L_\sigma \frac{\mathrm{d}\vec{i}_s}{\mathrm{d}t} = \vec{u}_{s,fb} - (R_s + R_R + R_a)\vec{i}_s,$$

where $\vec{u}_{s,fb}$ is the output of the PI regulator as marked in Fig. 5.4. The block diagram of active damping is shown in Fig. 5.5. Then the new transfer function $G'(s)$ from $\vec{u}_{s,fb}$ to $\vec{i}_s$ is $\frac{1}{sL_\sigma + R_s + R_R + R_a}$. Apparently, the machine resistance is equivalently enlarged by R_a from the side of the PI regulator. How to choose an appropriate value of R_a will be given in the next section.

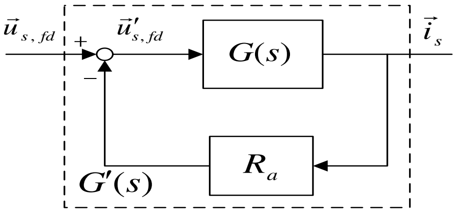


Fig. 5.5 Active damping

5.2.4 Internal Model Control

With feed forward and active damping mentioned above, the current control loop becomes a closed loop with unit feedback. Its structure is illustrated in Fig. 5.6.

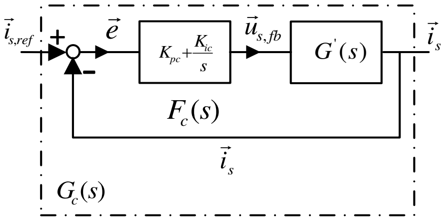


Fig. 5.6 Closed current control loop

The closed loop transfer function from reference current value to actual current value is $G_c(s) = \frac{F_c(s)G'(s)}{1+F_c(s)G'(s)}$. $F_c(s)$ is the transfer function of the PI controller, written as $F_c(s) = K_{pc} + \frac{K_{ic}}{s}$. If the desired closed loop transfer function $G_c(s) = \frac{\alpha_c}{s+\alpha_c}$, where α_c is the bandwidth of the closed loop system, the parameters of the PI controller can be calculated as:

$$\begin{aligned} K_{pc} &= \alpha_c \hat{L}_\sigma \\ K_{ic} &= \alpha_c(\hat{R}_s + \hat{R}_R + R_a), \end{aligned} \tag{5.24}$$

where a ˆ indicates a measured parameters or an estimated variable.

To make the process $G'(s) = \frac{1}{sL_\sigma + R_s + R_R + R_a}$ that the current PI regulator controls have the same bandwidth with the current control close-loop system $G_c(s) = \frac{\alpha_c}{s+\alpha_c}$, the active damping R_a is chosen as:

$$\frac{\hat{R}_s + \hat{R}_R + R_a}{\hat{L}_\sigma} = \alpha_c \Rightarrow R_a = \alpha_c \hat{L}_\sigma - \hat{R}_s - \hat{R}_R.$$

5.2.5 Voltage Limiter

Based on the working principles of the three-phase PWM and the inverter described in Sec. 2.2, it is known that the amplitude of the equivalent sinusoidal voltage applied to the machine stator is limited to half of the DC-link voltage. This fact can be modeled as a voltage limiter as shown in Fig. 5.4. What's more, in this project, the amplitude of modulation voltage is expected to be smaller than the half of the DC-link voltage because the inverter will output a constant voltage if the magnitude of the modulation voltage is larger than half DC-link voltage. This can be achieved by adding a voltage limiter in the current controller. For example, the DC link voltage is $60V$, then the peak value of the phase voltage is set to be within $\pm 28V$. As the scaling constant K of the Clark and the Inv. Clark transformations mentioned in Sec. 2.3.1 will make sure that $\|\vec{u}_s^s\|$ equals to the peak value of u_a, u_b or u_c and Park and Inv. Park transformations will never change the vector length. Due to this fact the voltage limiter in $d-q$ coordinate is set to $28V$.

5.2.6 Integrator Anti-Windup

In some occasions, due to the existence of the voltage limiter, the output voltage of the current regulator might not reach the value which is needed. Then the error between $\vec{i}_{s,ref}$ and $\vec{i}_s$ might let the integrator of the PI regulator integrate to a very large value. Even if the error is finally reduced, it would still take a long time for the integrator to go back to a normal value, which introduces a considerable delay [40]. This phenomenon is called integrator windup. In order to overcome the integrator windup problem, the difference between the voltage values before and after the voltage limiter is fed back to the integrator through the gain $\frac{1}{T_t}$, as shown in Fig. 5.4. T_t is the tracking-time constant, which determines how quickly the integrator is reset [43]. In this current regulator, T_t equals to K_{pc}.

5.2.7 Summary for Current Regulator

In order to facilitate the digital implementation of the current regulator, the relative equations are summed up in this section according to Fig. 5.4. As the DSP can not handle the complex quantities, these equations are split into real and imaginary parts. After flux estimator comes into steady state, $\hat{\Psi}_{Rq}$ is always equal to

0, then the terms containing $\hat{\Psi}_{Rq}$ are eliminated in these equations. Besides, $\hat{\Psi}_{Rd}$ is replaced by $\hat{\Psi}_R$.

$$u_{sd} = -\hat{\omega}_1 \hat{L}_\sigma i_{sq} - \frac{\hat{R}_R}{\hat{L}_M}\hat{\Psi}_R - R_a i_{sd} + K_{pc}(i_{sd,ref} - i_{sd}) + K_{ic} I_{ed} \tag{5.25a}$$

$$u_{sq} = \hat{\omega}_1 \hat{L}_\sigma i_{sd} + \omega_r \hat{\Psi}_R - R_a i_{sq} + K_{pc}(i_{sq,ref} - i_{sq}) + K_{ic} I_{eq} \tag{5.25b}$$

$$u_{sd,lim} = u_{sd}\frac{sat(\sqrt{(u_{sd})^2 + (u_{sq})^2}, 0, 28)}{\sqrt{(u_{sd})^2 + (u_{sq})^2}} \tag{5.25c}$$

$$u_{sq,lim} = u_{sq}\frac{sat(\sqrt{(u_{sd})^2 + (u_{sq})^2}, 0, 28)}{\sqrt{(u_{sd})^2 + (u_{sq})^2}} \tag{5.25d}$$

$$I_{ed} = \int_0^t \big((i_{sd,ref} - i_{sd}) + \frac{1}{K_{pc}}(u_{sd,lim} - u_{sd})\big)dt \tag{5.25e}$$

$$I_{eq} = \int_0^t \big((i_{sq,ref} - i_{sq}) + \frac{1}{K_{pc}}(u_{sq,lim} - u_{sq})\big)dt \tag{5.25f}$$

The $sat(x, min, max)$ in Eq. 5.25c and Eq. 5.25d is the saturation function. If x is some number between min and max the saturation function will return x otherwise the saturation function will return max if x is larger than max and return min if x is smaller than min.

5.3 Flux Estimator

There are two types of flux estimators, current model estimator and voltage model estimator. Compared with the voltage model, the current model estimator is more suitable for low speed application, in spite of sensitivity to parameter variations [23]. As a result, the flux estimator used in this project is the current model estimator implemented in Indirect Field Orientation(IFO), which indicates that the quantities used for the flux estimation are currents taken from the $d - q$ system. With $\vec{i}_s$ from the $d - q$ system and the electrical rotor angular speed ω_r as inputs, the flux determination equations for the estimator can be derived from Eq. 5.15b. Splitting Eq. 5.15b into real and imaginary part and assuming that $\vec{\Psi}_R = \Psi_{Rd} + j0$; since this is the wanted orientation of the $d - q$ system, the rotor flux equation can be expressed as:

$$\frac{\mathrm{d}\hat{\Psi}_R}{\mathrm{d}t} = R_R\hat{i}_{sd} - \frac{R_R}{L_M}\hat{\Psi}_R \tag{5.26a}$$

$$0 = R_R\hat{i}_{sq} - (\hat{\omega}_1 - \omega_r)\hat{\Psi}_R. \tag{5.26b}$$

By rewriting Eq. 5.26 into integral form, the desired outputs of flux estimator (amplitude, speed and position of $\vec{\Psi}_R$) can be calculated as:

$$\hat{\Psi}_R = \int_0^t (\hat{R}_R\hat{i}_{sd} - \frac{\hat{R}_R}{\hat{L}_M}\hat{\Psi}_R)dt \tag{5.27a}$$

$$\hat{\omega}_1 = \omega_r + \hat{\omega}_2 = \omega_r + \frac{\hat{R}_R\hat{i}_{sq}}{\hat{\Psi}_R} \tag{5.27b}$$

$$\hat{\theta}_1 = \int_0^t \hat{\omega}_1 dt, \tag{5.27c}$$

where $\omega_2 = \omega_1 - \omega_r$ is the difference speed between the synchronous speed and electrical rotor speed, the slip frequency. Based on Eq. 5.27, the structure of the flux estimator could be built as

5.4 Speed Regulator

The dynamic equation of the mechanical part of the induction motor can be expressed as

$$\frac{\mathrm{d}\Omega_r}{\mathrm{d}t} = -\frac{B}{J}\Omega_r + \frac{T_e - T_L}{J}, \tag{5.28}$$

where

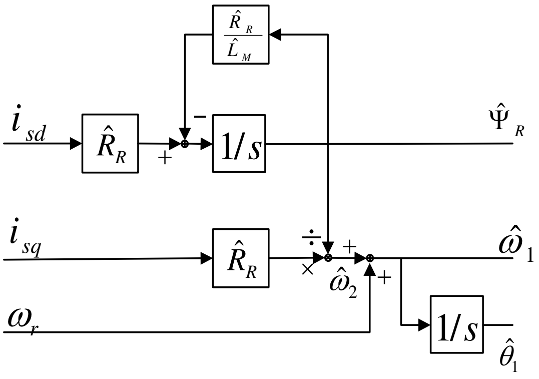


Fig. 5.7 Structure of the flux estimator.

Ω_r: the mechanical angular speed
T_e: electrodynamic torque(see Eq. 5.16)
T_L: load torque

B: friction coefficient
J: motor inertia

The transfer function from the applied torque to the rotor angular speed is $\frac{\Omega_r(s)}{T_e(s)} = \frac{1}{Js+B}$, which is very similar to the transfer function Eq. 5.23 used for the current regulator design. Due to this fact the design of the speed regulator is very similar to the current regulator design, that internal model control(IMC), active damping and anti-windup are also applied in the speed regulator design. If the desired closed-loop transfer function is $\frac{\Omega_r(s)}{\Omega_{r,ref}(s)} = \frac{\alpha_\omega}{s+\alpha_\omega}$ and an active damping B_a is introduced, the transfer function of the speed PI regulator becomes

$$\begin{aligned} F_\omega(s) &= K_{p\omega} + \frac{K_{i\omega}}{s} \\ K_{p\omega} &= \alpha_\omega \hat{J} \\ K_{i\omega} &= \alpha_\omega(\hat{B} + B_a) \\ B_a &= \alpha_\omega \hat{J} - \hat{B}, \end{aligned}$$

where the value of B_a is chosen so that the controlled plant for the speed PI regulator, the mechanical part of the machine along with the active damping, has the same bandwidth with the speed closed loop. As the q-current is limited within $\pm 5A$, the reference torque is de facto saturated as well in the speed regulator. To avoid large overshoots in the speed response, anti-windup is added and the tracking-time constant is selected to be $K_{p\omega}$.

To get a more compact structure of the vector control algorithm, the speed regulator and current reference determination that is described in Sec. 5.2.1 are packed into one block in both Matlab/Simulink simulations and in the DSP implementation. The structure of this combined block is shown in Fig. 5.8. As electrical dynamics are much faster than the mechanical, it can be assumed that the output signal of the speed regulator, the desired actuating torque, acts on the motor directly and immediately. In other words, seen from the speed control loop the inner current control loop is considered as a block with gain of unit one, whose input signal $\vec{i}_{s,ref}$ and output signal $\vec{i}_s$ are exactly the same. This can be realized by keeping the bandwidth of the speed regulator much smaller than the bandwidth if the current regulator. A factor of 10 to 100 is usually recommended [22], in this project, $\alpha_\omega = \frac{\alpha_c}{50}$.

To facilitate the digital implementation of the speed regulator, according to Fig. 5.8, relative equations are summed up as well.

$$i_{sd,ref} = \frac{1}{\hat{L}_M} \Psi_{R,ref} \tag{5.29a}$$

$$i'_{sq,ref} = \frac{2}{3n_p} \frac{1}{\hat{\Psi}_R} (K_{p\omega}(\Omega_{r,ref} - \Omega_r) + K_{i\omega} I_{e,\Omega} - B_a \Omega_r) \tag{5.29b}$$

$$i_{sq,ref} = sat(i'_{sq,ref}, -5, 5) \tag{5.29c}$$

$$I_{e,\Omega} = \int_0^t \big((\Omega_{r,ref} - \Omega_r) + \frac{1}{K_{p\Omega}} \frac{3n_p}{2} \hat{\Psi}_R (i_{sq,ref} - i^{k\prime}_{sq,ref})\big) dt. \tag{5.29d}$$

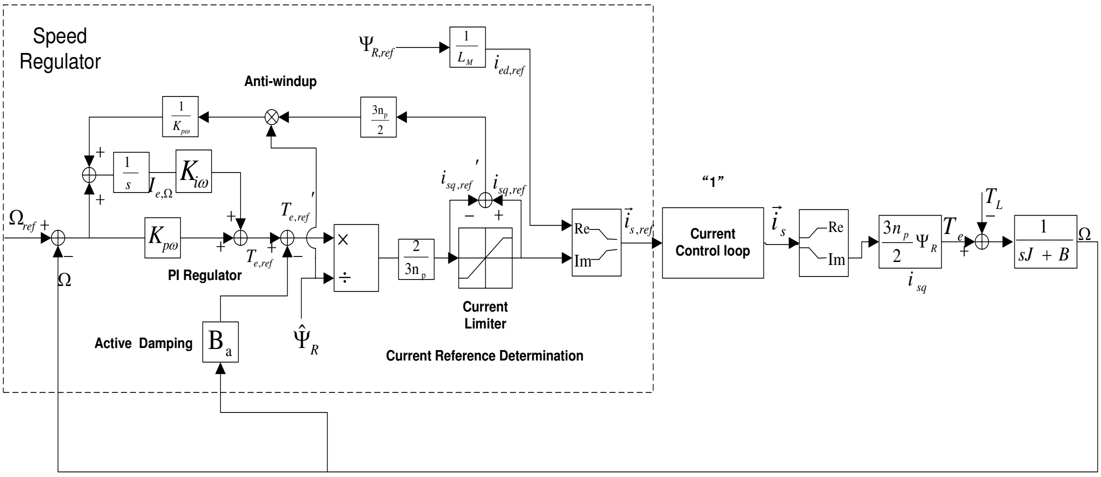


Fig. 5.8 Speed control loop with speed regulator.

5.5 Simulation

The designed vector control system is simulated in Matlab/Simulink. The top-layer Simulink block diagram is the same as Fig. 5.1. The structures of sub-blocks of the current regulator, flux estimator and speed regulator are almost the same as Fig.5.4, Fig.5.7 and Fig.5.8 correspondingly. However there is still some small differences. For example, the Simulink blocks *Integrator* and *Saturation* can not handle the complex number while the integral and the saturation of the complex number are needed in the system. For the integral in Simulink, the complex number is decomposed to the real part and the imaginary part, which are integrated separately as shown in Fig. 5.9.

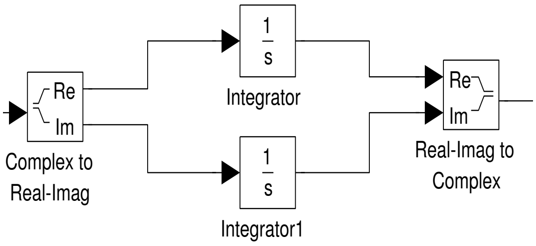


Fig. 5.9 Integral of the complex number in Simulink.

For the saturation in Simulink, the complex number is expressed in polar form, whose the radial part is saturated as shown in Fig. 5.10.

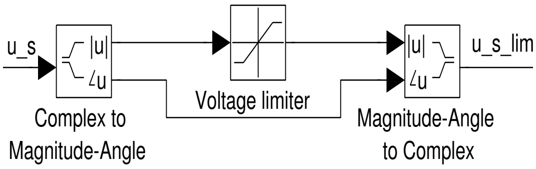


Fig. 5.10 Saturation of the complex number in Simulink.

The sub-block of the induction machine is built mainly based on the state space differential equations listed in Eq.5.4 and Eq.5.28, where $i_{s\alpha}$, $i_{s\beta}$, $i_{r\alpha}$, $i_{r\beta}$ and Ω_r are the state variables. The sub-blocks of the four coordinate transformations are constructed according to Eq.2.4, Eq.2.5, Eq.2.8 and Eq.2.9.

In the primary stage of simulation, the whole vector control system is built with Simulink blocks. The benefits of building the model with blocks is that it is easy to observe a variable by adding a scope wherever it is needed. Moreover, it is more flexible to modify the system structure. For instance, the two manual switches shown in Fig.5.1 allow rapid shifts between different simulation scenarios. Then, the sub-blocks of current regulator, speed regulator, flux estimator and induction machine are rebuilt using the S-function. The model implemented in the S-function can more or less help to verify the correctness of the model implemented in blocks, if the two models could get the same results. Besides, the code in the S-function could be reused for the DSP programming.

In Simulink, the simulation is configured to be fixed step with a step time of $0.0001s$, the same as the control period used in the DSP code. The simulation results of the two models implemented in blocks and in the S-function are the same, which will be shown in Chap. 7.

5.6 Discretization

All the design above is based on continuous time. In order to make the control algorithm executable in the DSP, discretization must be done to the equations of each block in the continuous system which are listed at the end of each section in this chapter. Since the control loop frequency is $10kHz$, which means that the control loop period is $0.1ms$, the sampling period h for the discretized system is selected as $0.1ms$. As mentioned in Sec. 2.3.2, at a certain time instant $t = kh$, in one cycle of the control loop sampling takes place firstly, then the sampled values at $t = kh$ are passed down into the function block chain in Fig. 2.8. Then each block takes its input signals and stored state variables into calculation, derives its output to feed into the next block and then get its state variable updated for the next iteration at time instant $t = (k + 1)h$.

In this project, Forward Euler approximation is used to transform the continuous system into discrete system. With $x(t)$ defined as a continuous variable, $x[kh]$ as its value at the current time instant and $x[(k + 1)h]$ for the next time instant, as illustrated in Fig. 5.11 the derivatives can be approximated with a forward difference as [40]

$$\frac{\mathrm{d}x(t)}{\mathrm{d}t} \approx \frac{x[(k+1)h] - x[kh]}{h}. \tag{5.30}$$

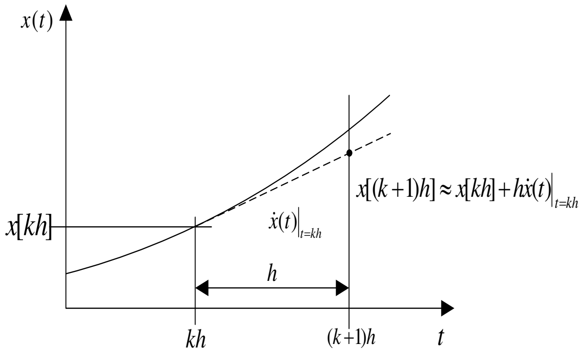


Fig. 5.11 Illustration of Forward Euler approximation.

The integral form of Forward Euler approximation is described in Fig 5.12, where the area below the curve of $x(t)$ corresponds to the integral of variable x, denoted as X. Over the period interval between kh and $(k + 1)h$, the area below the curve of $x(t)$ is approximated to the area of a rectangle with the height of $x[kh]$ and width of h. The integral of $x(t)$ can be calculated as, by using the Forward Euler approximation,

$$X[(k+1)h] \approx X[kh] + hx[kh]. \tag{5.31}$$

The way to apply Forward Euler approximation is that all the equations which describe the continuous system before are presented in integral form. For the blocks containing dynamic parts(integrals), Eq. 5.31 should be inserted into the original equations to replace the integrals and the argument t of all the continuous variables should be changed into kh to indicate a discrete system. For the blocks without dynamics included, the only change is that kh takes the place of t as the new argument of the discrete variables. In this way, the discrete forms for blocks with dynamic parts included are summed up below, where the argument kh of discrete variables is simplified as k and $(k + 1)h$ is replaced with $k + 1$.

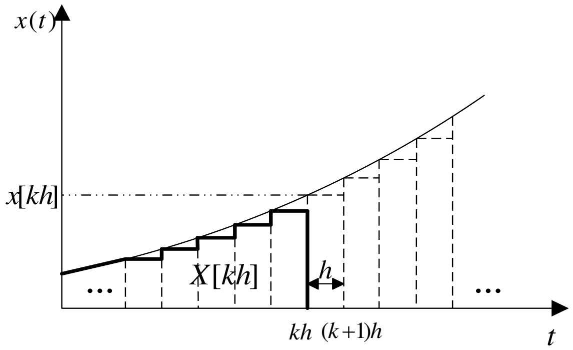


Fig. 5.12 Integral form of Forward Euler approximation.

Current Regulator. The inputs of the current regulator are: $\vec{i}_{s,ref}(k)$, $\vec{i}_s(k)$, $\omega_r(k)$, $\hat{\omega}_1(k)$ and $\hat{\Psi}_R(k)$. The outputs is $\vec{u}_{s,lim}(k)$.

$$u_{sd}(k) = -\hat{\omega}_1(k)\hat{L}_\sigma i_{sq}(k) - \frac{\hat{R}_R}{\hat{L}_M}\hat{\Psi}_R(k) - R_a i_{sd}(k) + K_{pc}(i_{sd,ref}(k) - i_{sd}(k)) + K_{ic}I_{ed}(k) \tag{5.32a}$$

$$u_{sq}(k) = \hat{\omega}_1\hat{L}_\sigma i_{sd}(k) + \omega_r(k)\hat{\Psi}_R(k) - R_a i_{sq}(k) + K_{pc}(i_{sq,ref}(k) - i_{sq}(k)) + K_{ic}I_{eq}(k) \tag{5.32b}$$

$$u_{sd,lim}(k) = u_{sd}(k)\frac{sat(\sqrt{(u_{sd}(k))^2 + (u_{sq}(k))^2}, 0, 28)}{\sqrt{(u_{sd}(k))^2 + (u_{sq}(k))^2}} \tag{5.32c}$$

$$u_{sq,lim}(k) = u_{sq}(k)\frac{sat(\sqrt{(u_{sd}(k))^2 + (u_{sq}(k))^2}, 0, 28)}{\sqrt{(u_{sd}(k))^2 + (u_{sq}(k))^2}} \tag{5.32d}$$

$$I_{ed}(k+1) = I_{ed}(k) + h\big((i_{sd,ref}(k) - i_{sd}(k)) + \frac{1}{K_{pc}}(u_{sd,lim}(k) - u_{sd}(k))\big) \tag{5.32e}$$

$$I_{eq}(k+1) = I_{eq}(k) + h\big((i_{sq,ref}(k) - i_{sq}(k)) + \frac{1}{K_{pc}}(u_{sq,lim}(k) - u_{sq}(k))\big) \tag{5.32f}$$

Speed Regulator. The inputs of the speed regulator are: $\Omega_{r,ref}(k)$, $\Psi_{R,ref}(k)$, $\Omega_r(k)$ and $\hat{\Psi}_R(k)$. The output is $\vec{i}_{s,lim}(k)$.

$$i_{sd,ref}(k) = \frac{1}{\hat{L}_M}\Psi_{R,ref}(k) \tag{5.33a}$$

$$i'_{sq,ref}(k) = \frac{2}{3n_p}\frac{1}{\hat{\Psi}_R(k)}(K_{p\omega}(\Omega_{r,ref}(k) - \Omega_r(k)) + K_{i\omega}I_{e,\Omega}(k) - B_a\Omega_r(k)) \tag{5.33b}$$

$$i_{sq,ref}(k) = sat(i'_{sq,ref}(k), -5, 5) \tag{5.33c}$$

$$I_{e,\Omega}(k+1) = I_{e,\Omega}(k) + h\big((\Omega_{r,ref}(k) - \Omega_r(k)) + \frac{1}{K_{p\Omega}}\frac{3n_p}{2}\hat{\Psi}_R(k)(i_{sq,ref}(k) - i'_{sq,ref}(k))\big) \tag{5.33d}$$

Flux Estimator. The inputs of the flux estimator are: $\vec{i}_s(k)$, $\omega_r(k)$. The outputs are: $\hat{\Psi}_R(k)$, $\hat{\omega}_1(k)$ and $\hat{\theta}_1(k)$.

$$\hat{\Psi}_R(k+1) = \hat{\Psi}_R(k) + h(\hat{R}_R\hat{i}_{sd}(k) - \frac{\hat{R}_R}{\hat{L}_M}\hat{\Psi}_R(k)) \tag{5.34a}$$

$$\hat{\omega}_1(k) = \omega_r(k) + \frac{\hat{R}_R\hat{i}_{sq}(k)}{\hat{\Psi}_R}(k) \tag{5.34b}$$

$$\hat{\theta}_1(k+1) = \hat{\theta}_1(k) + h\hat{\omega}_1(k) \tag{5.34c}$$

The discrete forms of other parts of vector control system, such as coordinate transformation, are almost the same as the continuous forms except for the argument of the variables, so they will not be repeated here.

Chapter 6

DSP Programming

Texas Instruments provides an integrated solution called *controlSUITE™* for industrial control purpose. *controlSUITE™* consists of necessary header files, source files, library, linker command files for developing an executable program. Besides it gives extensive example codes, which allows the new user to be able to quickly start. *Code Composer Studio™*(CCS) is the Integrated Development Environment(IDE) for TI's microcontrollers, which is an integrated user interface guiding users through each code developing step.

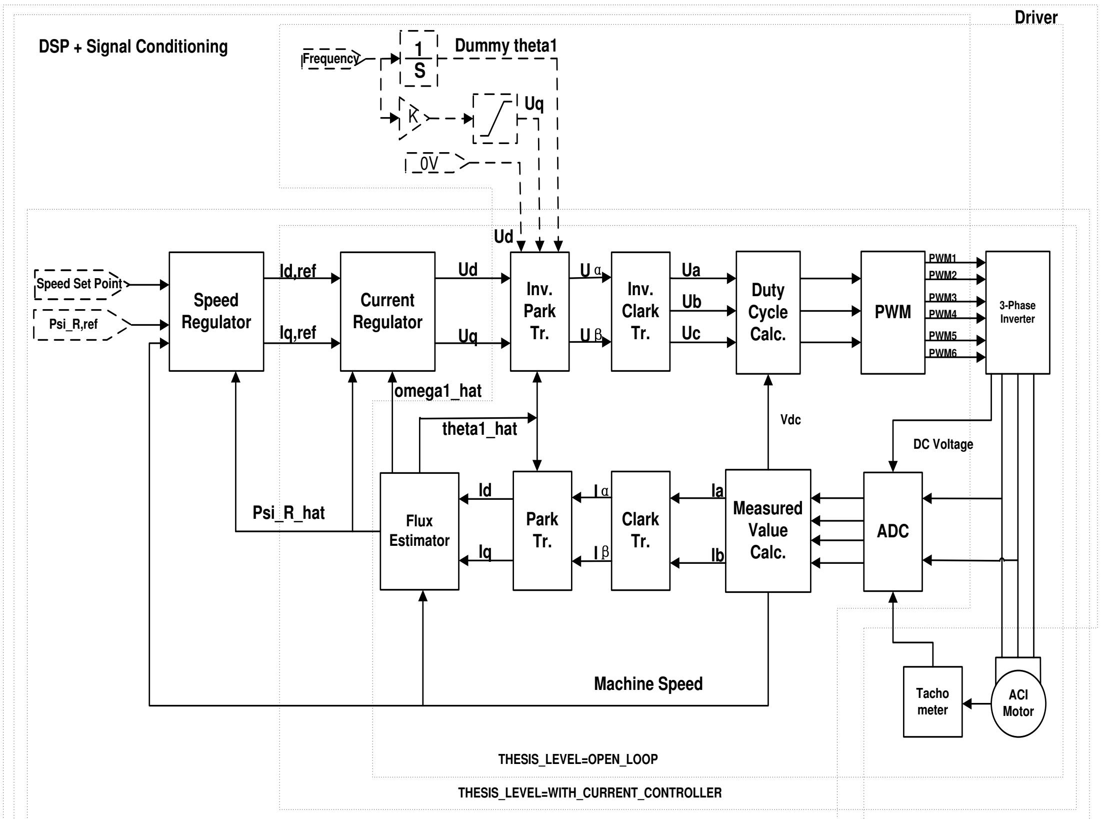


Fig. 6.1 Programming Diagram

The complete induction motor vector control system is basically designed according to the planned system blocks diagram shown in Fig. 2.1. However, considering some practical limits, some modifications to the original plan have been made during the implementation stage. These modifications can be noticed if the block diagram of the implemented system shown in Fig. 6.1 is compared with the block diagram in Fig. 2.1. Firstly, only two phase currents i_{sa} and i_{sb} are measured instead of all the three phases. It is sufficient to take the measurements of two phases for the Clark Transformation, if the three phase currents are well balanced with the condition of $i_{sa}(t) + i_{sb}(t) + i_{sc}(t) = 0$ holding. If the three phases are balanced one phase current can be calculated from the other two phase currents. The other change is that the incremental quadrature encoder is replaced by a tachometer whose output voltage is proportional to the machine speed, because the induction motor in the lab has no place to mount the encoder. A brief *V/f* control is also implemented at the primary stage of programming in order to test the functionality of each device. As the *V/f* control will not appear in the final result of this project, the blocks of the *V/f* control is drawn with dashed lines in Fig. 6.1.

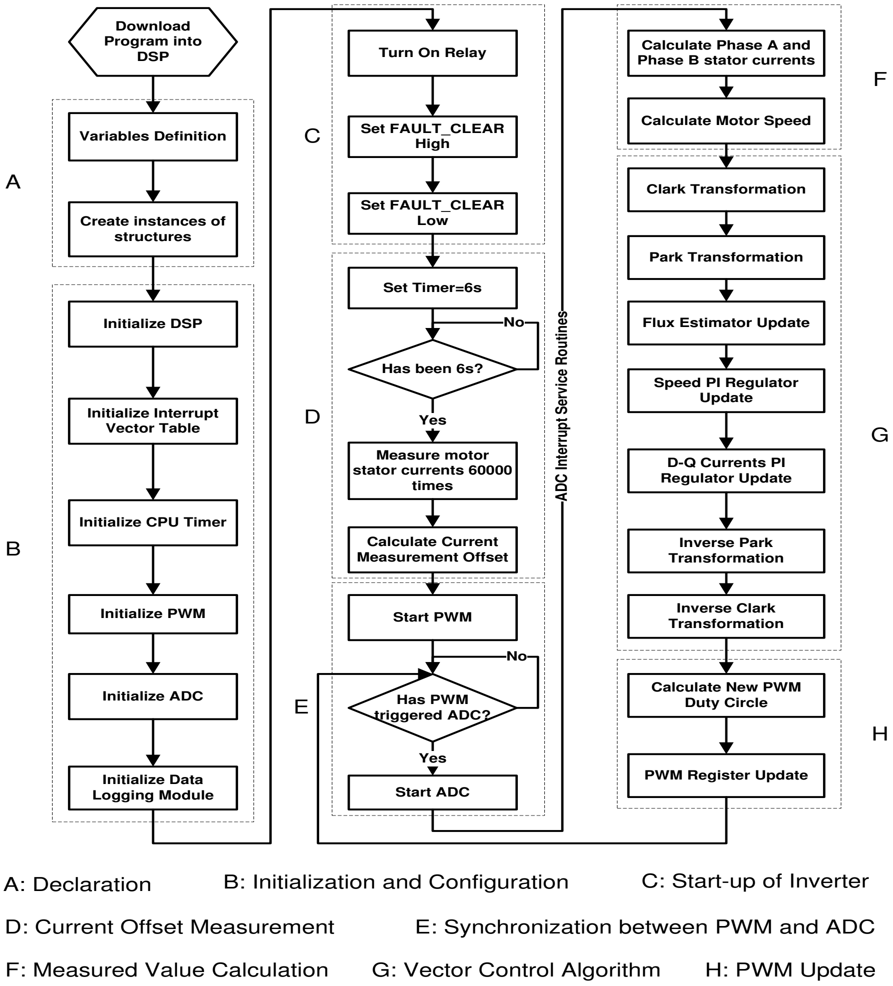

Download Program into DSP
A
Variables Definition
Create instances of structures
B
Initialize DSP
Initialize Interrupt Vector Table
Initialize CPU Timer
Initialize PWM
Initialize ADC
Initialize Data Logging Module
C
Turn On Relay
Set FAULT_CLEAR High
Set FAULT_CLEAR Low
D
Set Timer=6s
No
Has been 6s?
Yes
Measure motor stator currents 60000 times
Calculate Current Measurement Offset
E
Start PWM
No
Has PWM triggered ADC?
Yes
Start ADC
ADC Interrupt Service Routines
F
Calculate Phase A and Phase B stator currents
Calculate Motor Speed
G
Clark Transformation
Park Transformation
Flux Estimator Update
Speed PI Regulator Update
D-Q Currents PI Regulator Update
Inverse Park Transformation
Inverse Clark Transformation
H
Calculate New PWM Duty Circle
PWM Register Update
A: Declaration
B: Initialization and Configuration
C: Start-up of Inverter
D: Current Offset Measurement
E: Synchronization between PWM and ADC
F: Measured Value Calculation
G: Vector Control Algorithm
H: PWM Update

Fig. 6.2 The Flow Chart of the Program

In Fig. 6.2 a more detailed flow chart of the implemented system is shown compared to the basic flow chart shown in Fig. 2.8. From the flow chart in Fig. 6.2 it can be seen that the program starts with variable declarations(A), initialization and configuration of the used hardware in the DSP(B), start-up of the inverter and offset measurement(D). These parts are not shown in the basic flow chart in Fig. 2.8 since they are depending on the used hardware. Sequentially, the PWM is started and the program enters into the key control loop(E). Based on the control strategy of this project, the control loop is synchronized with the PWM carrier wave as discussed in Sec. 2.3.2. Every start of a new PWM period will trigger an ADC sequence. An ADC sequence contains measurements of phase currents, machine speed and DC link voltage. At the end of the ADC sequence(E), the program jumps into the interrupt service routines(ISR). In the ISR, firstly the measured current and speed values are calculated as Eq. 4.5, where the offset compensation is added(F). Following the measurement calculations, the vector control algorithm is executed(G). Finally the PWM duty cycles are updated(H) and the program waits for the next trigger from the PWM(E). In the following a brief introduction to the F28335 DSP and the used built in hardware modules will be given, followed by a more detailed description of the flow chart in Fig. 6.2.

6.1 Description of the Used Hardware in the F28335 DSP

In this section, first a brief introduction to the F28335 DSP is given, followed by a detailed description of the used modules and features of the F28335 DSP. Due to the fact that the PWM module plays an important role in this project, such as to output the three-phase signals to the inverter, to decide the control loop period and to synchronize the analog-to-digital conversions, the description of the PWM module will take up a large proportion of this section.

6.1.1 Overview of the F28335 DSP

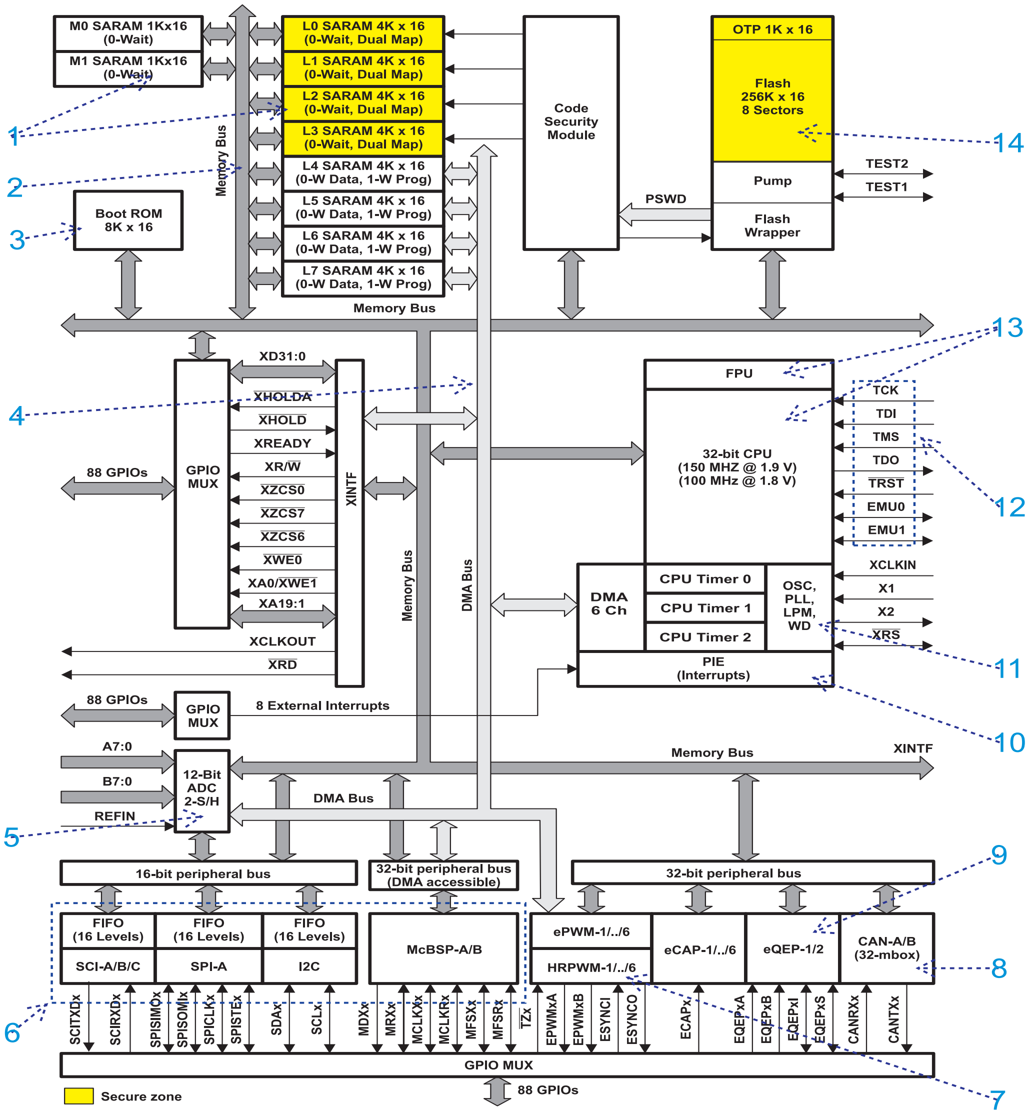


Fig. 6.3 Functional block diagram of F28335, quoted from Ref. [28, P.34,Figure3-1]

The computing unit(⑬ in Fig. 6.3) of the F28335 DSP consists of a 32-bit CPU and a single-precision 32-bit floating-point unit(FPU), which enables the floating-point computation to be performed in hardware. Besides, the CPU of the F28335 has a 8-stage pipeline structure, which makes the CPU be able to execute eight instructions simultaneously in one system clock period. The $150MHz$ system clock is provided by an on-chip oscillator and a phase-locked loop(PLL) circuit⑪. The oscillator generates a $50MHz$ clock signal, which is tripled to $150MHz$ by the PLL circuit. The F28335 applies the Harvard Bus Architecture, which means that there are independent logical memory spaces and separated memory buses for the program and the data as can be seen in Fig. 6.3. The memory bus② contains a program read bus, a data read bus and a data write bus. The physical memory of the F28335 comprises of a $34K \times 16$ single-access randomaccess memory(SARAM, ①), a $256K \times 16$ Flash⑭, a $8K \times 16$ read-only memory(ROM, ③), a $1K \times 16$ one-time programmable memory(OTP, above⑭) and the registers. SARAM, OTP and Flash memories are assigned and used according to the practical demands. The ROM has been pre-programmed by the DSP manufacturer. The program existing in the ROM provides a standard procedure for DSP booting as well as some optimized codes for mathematical functions. The registers control the behavior of the DSP and each peripheral module. For the F28335, reading from or writing to registers applies the *bit-field address structure*, which will be explained in Sec. 6.1.6. F28335 also has the feature of direct memory access(DMA). With the DMA bus④, the data can be passed from one part of the DSP to the other part without the interaction of the CPU [28], which increases the data transmission speed. As it is designed mainly for industrial applications, the F28335 has plenty of peripheral circuits. For example, the 16-channel, 12-bit ADC module⑤, the PWM module⑦ and the encoder module⑨ could be used for motor control purposes. Five kinds of communications could be achieved with by the controller area network(CAN) module⑧, the serial communication interface(SCI) module, the serial peripheral interface(SPI), the multichannel buffered serial port(McBSP) module and the inter-integrated circuit(I2C) module⑥. 96 interrupts are supported by F28335. These interrupts are governed by the peripheral interrupt expansion(PIE) block⑩, which could enable or disable some interrupts, decide the interrupts' priorities and inform the CPU of the occurrence of a new interrupt. The port marked with ⑫ in Fig. 6.3 is the joint test action group(JTAG) interface, which supports the real-time debugging. With the help of the JTAG, the user can watch and modify the contents of the memory and the registers without stopping the processor.

6.1.2 PWM Module

The TMS320F28335 has six independent enhanced PWM(ePWM) modules included. The so-called Enhanced PWM peripheral means that it can generate complex PWM waveform with the least CPU resources occupied [20]. Each of ePWM module has two output channel: ePWMxA and ePWMxB belonging to the ePWMx module. Each ePWM module contains seven submodules, which can realize different functions in the generation of PWM waveforms. They are time-base(TB) submodule, counter-compare(CC) submodule, action-qualifier(AQ) submodule, dead-band generator(DB) submodule, PWM-chopper(PC) submodule, trip-zone(TZ) submodule and event-trigger(ET) submodule. The complete structure of a single ePWM module with each submodule and the signal connections between its subsections included is shown in Fig. 6.4. Except for the PWM-chopper(PC) submodule and the trip-zone(TZ) submodule, the other submodules are all used in this project and they will be discussed individually later on.

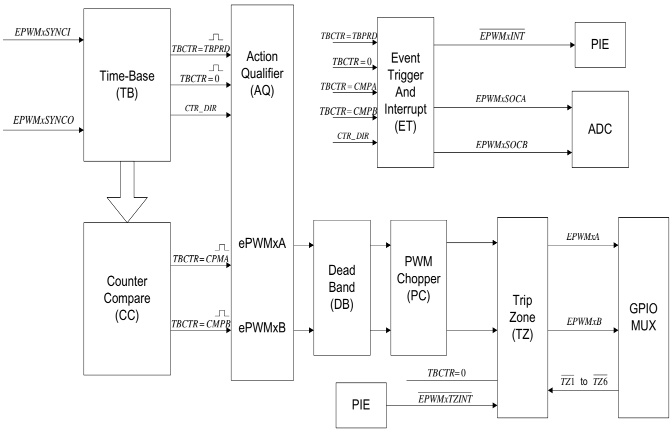


Fig. 6.4 Internal submodules of each ePWM module and signal connections.

Time-base submodule

As the most basic submodule in each ePWM module, the time-base submodule takes charge of the event timing for its own ePWM module. The structure block diagram of a time-base submodule containing main registers and key signal flows is shown in Fig. 6.5.

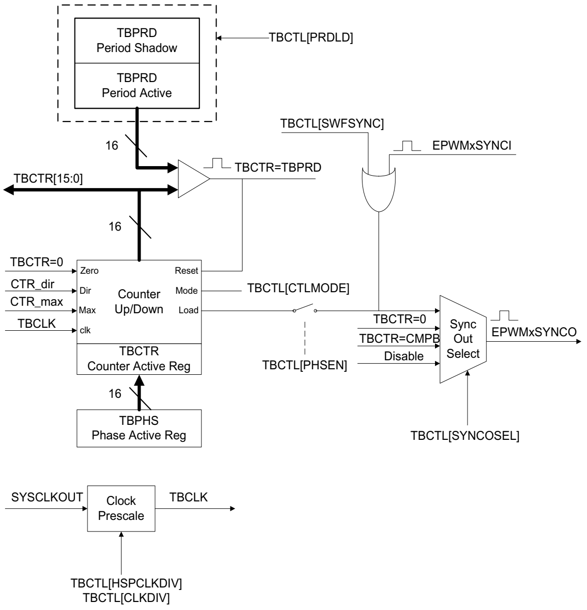


Fig. 6.5 Time-base submodule structure in each ePWM module.

One of the main tasks for the time-base submodule is to determine the PWM time-base clock relative to the system clock. The PWM time-base clock is used to regulate the timing of all the events in the PWM module. The period of the time-base clock T_{TBCLK} for the PWM module can be scaled to several times of the system clock period $T_{SYSCLKOUT}$ as:

$$T_{TBCLK} = T_{SYSCLKOUT} * CLKDIV * HSPCLKDIV, \tag{6.1}$$

where CLKDIV and HSPCLKDIV are bits in the time-base control register(TBCTL) serving for the time-base clock pre-scale.

The time-base submodule can also be used for specifying the period of the time-base counter(TBCTR) depending on its different operation modes. There are three types of operation modes for the time-base submodule, which can be selected in time-base control register (TBCTL), namely up-count mode, down-count mode and up-down-count mode. During a complete PWM period, for the former two modes the time-base counter(TBCTR) just keeps incrementing or decrementing all the time giving a saw tooth carrier wave. While for up-down mode the, the TBCTR increments in the first half the PWM period and then turns to decrement in the second part of the period giving a triangular carrier wave. The largest difference of up-down mode is that in one period the counter changes in a symmetrical pattern, where the corresponding moment to the PWM carrier peak time can be easily found. The peak time of the PWM carrier wave has to be known for signal sampling as explained in Sec. 2.2. So up-down-count-mode is selected in the PWM design of this project. To get the desired PWM frequency, the value in time-base period register(TBPRD) is supposed to be determined. For up-down-count mode, the relationship between time-based period (the value stored in TBPRD register)and PWM frequency (f_{PWM}) can be written as

$$T_{PWM} = 2 \times TBPRD \times T_{TBCLK} \tag{6.2a}$$

$$f_{PWM} = 1/T_{PWM}, \tag{6.2b}$$

where T_{PWM} stands for PWM period and T_{TBCLK} for time-base clock period that can be got from Eq. 6.1. Substitute Eq. 6.1 into Eq. 6.2, with the frequency of the system clock $f_{SYSCLKOUT} = \frac{1}{T_{SYSCLKOUT}}$ the value in time-base period register(TBPRD) can be determined as follows

$$TBPRD = \frac{1}{2} * \frac{f_{SYSCLKOUT}}{f_{PWM} * CLKDIV * HSPCLKDIV} \tag{6.3}$$

By Eq. 6.3, as long as the DSP system clock frequency and the desired PWM frequency is known, time-base period register(TBPRD) can be easily configured. According to the design specification of this project and the the DSP configuration, the specified PWM frequency(f_{PWM}) is $10kHz$ and the system clock frequency($f_{SYSCLKOUT}$) of the DSP F28335 is $150MHz$. For convenience, CLKDIV is selected as 1 and HSPCLKDIV is selected as 2. Based on Eq. 6.3, the value set in the time-base period register(TBPRD) can be calculated directly. It can be seen from Fig. 6.5 that the time-period register(TBPRD) has a shadow register that can provide preventive measures for corruption and spurious operation if time-base period shadow mode is applied [20].

$$\begin{aligned} TBPRD &= \frac{1}{2} * \frac{150M}{10k * 1 * 2} \\ &= 3750 \end{aligned} \tag{6.4}$$

Besides, synchronization between different ePWM modules can also be realized in the time-base submodules. The three-phase PWM is used to produce a three-phase alternating voltage, hence the synchronization between the three-phase PWM signal turns out to be very important. Each ePWM module has two signal for synchronization between different ePWM modules: a synchronization input EPWMxSYNCI and a synchronization output EPWMxSYNCO shown in Fig. 6.5. The time-base counter synchronization scheme for the F28335 is presented in Fig. 6.6. From Fig. 6.6, it is can be found that the ePWM modules are connected in a series with the synchronization output EPWMxSYNCO of the previous one fed into the synchronization input EPWMxSYNCI of the next one. Only the input synchronization for the master module (the first one) in the chain is taken from an external pin. For each ePWM module, once a pulse from the synchronization input is detected, the value in the time-base phase register(TBPHS) will be loaded into time-base counter(TBCTR), where time-base phase register(TBPHS) is used to store the time-base counter(TBCTR) phase value of the ePWM module with respect to the time-base of its synchronization input signal [20].

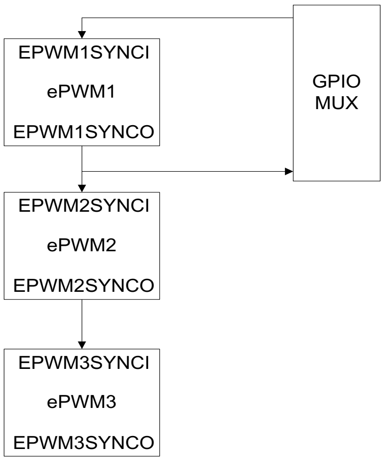


Figure 6.6: Time base counter synchronization scheme for F28335.

In this project, although the inverter output are three-phase voltages which are leading or lagging each other 120°, PWM signals have the same phase at any time moment. For example, the ePWM module ePWM1, ePWM2, ePWM3 are selected for the three-phase PWM generation. Hence, the time-base phase register(TBPHS) for the ePWM modules are assigned the same value 0. It means there is no phase shift between output signal ePWM1A, ePWM2A and ePWM3A. To synchronize different ePWM modules, the synchronization output select bit(SYNCOSEL) in the time base control register(TBCTL) is supposed to be configured based on the practical requirements [20, P. 98]. As shown in Fig. 6.6, ePWM1 is defined as the master phase to generate a synchronization output EPWM1SYNCO pulse each time its time-base counter(TBCTR) equals zero, while ePWM2 is defined as a slave phase whose synchronization input EPWM2SYNCI signal is enabled. Meanwhile, ePWM2's synchronization output EPWM2SYNCO signal is set equal to its synchronization input EPWM2SYNI signal to to drive it into ePWM3 unit. Except the master module ePWM1 the slave modules ePWM2 and ePWM3 have to load the time-base counter(TBCTR) with the time-base phase register(TBPHS) when a synchronization input EPWMxSYNCI pulse appears.

Counter-compare submodule

In the practical DSP implementation, the PWM waveform generation is achieved by the comparison between a counter(TBCTR) value(the carrier wave) and a set-point(the reference wave) which is store in a counter-compare register. Fig. 6.7 shows one way to generate a PWM waveform. In the PWM generation process the counter-compare(CC) submodule takes the part of event generation, while the action qualifier submodule assumes the task of taking actions, which will be discussed later. In counter-compare submodule, there are two counter-compare registers: counter-compare A register(CMPA) and counter-compare B register(CMPB) to store the values that is used to compare with the time-base counter(TBCTR) continuously. The time-base counter(TBCTR) is treated as the input while the generated event TBCTR=CMPA or TBCTR=CMPB is the expected output. The operation mode shown in Fig. 6.7 is exactly what is applied in this project, where TBCTR=CMPA is chosen as the output event of the counter-compare(CC) module. In Fig. 6.7, each intersection of TBCTR and CMPA value marked with letter A implies when the event TBCTR=CMPA occurs.

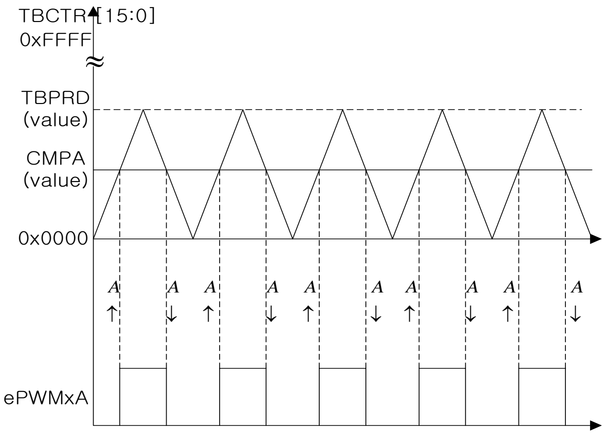


Figure 6.7: Counter-compare event, up-down-counter mode, dual edge symmetric waveform, with modulation on ePWMxA

Action qualifier submodule

After getting the generated events from the previous submodules, the action qualifier submodule is supposed to take action. There are two sources of the input events: time-base(TB) submodule and Counter-compare(CC) submodule. The former one produces event TBCTR=0 and TBCTR=TBPRD while the latter one generates event TBCTR=CMPA and TBCTR=CMPB. With time-base counter(TBCTR) state considered, incrementing or decrementing, the four events are expanded into 8 event combinations. When a specified event takes place, one of the 8 events, there are four possible actions to be triggered: to set high, to clear low, to toggle and to do nothing, which can determine the shape of the PWM waveform. The action qualifier output control register(AQCTLA) is used to define the actions that should be taken if specified events occur [20, P. 102]. In this project, to get the desired waveform same as Fig. 6.7, the action-qualifier submodule should be configured in this way that when TBCTR=CMPA and TBCTR is incrementing, ePWMxA is set high; while TBCTR=CMPA and TBCTR is decrementing, ePWMxA is cleared low. The up arrow or down arrow in Fig. 6.7 implies the direction of time-base counter(TBCTR) change.

Dead band submodule

In this project, what is required to fed into the upper and lower IGBTs in the same bridge leg of the inverter is a pair of mutual complimentary PWM signal. It can be implemented in the dead-band submodule to take ePWMxA as the signal source and then to produce the two mutually complementary PWM outputs: ePWMxA and ePWMxB, due to the unique structure of dead-band module shown in Fig. 6.8. The setup of the dead-band unit is based on six switches, S0 to S5. Different state combinations of the 6 switches bring about different modes for signal pairs. Although all the combinations can be achieved, not all of them are typical application modes. For practical safety consideration, a dead band should be inserted into the ideal PWM waveform to avoid that the two IGBTs on the same bridge leg of the inverter being turned on simultaneously. Therefore, operating mode *Active High Complementary*(AHC) is selected as the desired one for a pair of power switches in one phase of a 3-phase motor control system, which can be achieved by setting the states of the switches in Fig. 6.8, which can be configured in dead band control register(DBCTL) [20, p. 107]. In Fig. 6.8, a Rising Edges Delay block and a Falling Edges Delay block are used to insert a rising edge delay or a falling edge delay into the original PWM output. With the switch S4 and S5 set to 0, ePWMxA is chosen as the input source for both output A and B. By setting switch S2 to 0 and S1 to 1, a rising edge delay is inserted into the original ePWMxA signal; by setting switch S3 and S0 to 1, ePWMxA signal is reversed with a falling edge delay added, which is output as ePWMxB signal.

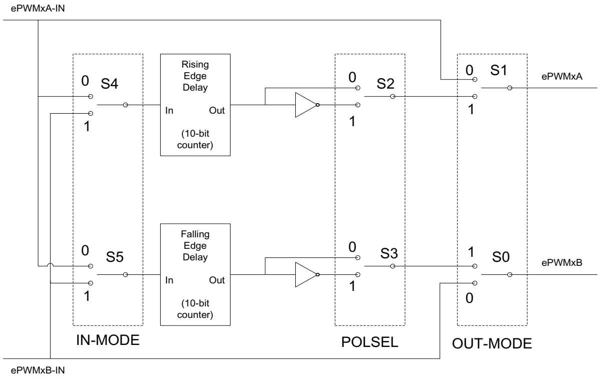


Fig. 6.8 Block diagram of ePWM dead band submodule.

The generated PWM signal in AHC mode looks in the way in Fig. 6.9. From Fig. 6.9, it is can be observed that there is an extremely short period of each PWM period, when both outputs of the ePWMx module are cleared, which avoids the case that the mutually complementary PWM signals are set high at the same time.

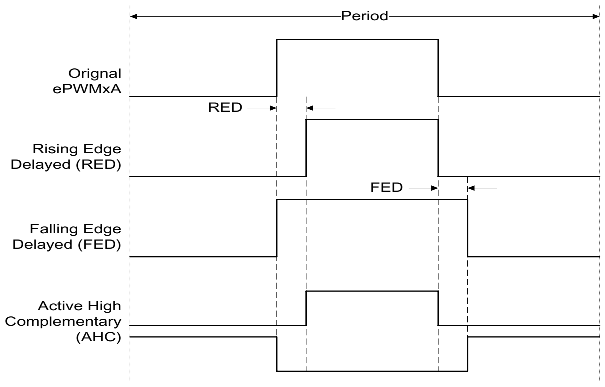


Fig. 6.9 Waveform in AHC mode with dead band inserted.

Event trigger submodule

During a sampling period, the Analog-to-Digital Conversion(ADC) is executed ahead, then the sampled data is used for calculate the PWM output. Therefore, the ADC and PWM are supposed to synchronize to function in the same pace. The event trigger module in ePWM is to issue interrupt request or ADC conversion after receiving event inputs. In order to avoid aliasing from current ripple, the carrier wave peaks are always chosen as the sampling time instants. Both TBCTR=0 and TBCTR=TBPRD can meet this requirement for sampling. And in this project it is unnecessary to sample twice at the both peaks in one period. Hence, TBCTR=0 is defined as the event that triggers ADC start of conversion A by EPWMxSOCA pulse shown in Fig. 6.4.

Duty cycle calculation

This is the key part of PWM generation. The value in counter-compare register corresponds to the duty cycle of the PWM period. So the counter-compare A register(CMPA) needs to be updated once in each PWM period to generate the PWM wave with varying duty cycle. Fig. 6.7 basically shows the PWM generation pattern used in this project. The roles that different submodules play in the PWM generation process are separately introduced before. From Fig. 6.7, the relation between the value set in CMPA register and PWM duty cycle(D) can be formulated as:

$$CMPA = TBPRD * (1 - D). \tag{6.5}$$

Recall some basic concepts mentioned in Sec. 2.2. At any moment, the specific output value of the invertor should be equal to the digital signal given out by the DSP as shown in Eq. 2.2. It would be better to start with the most extreme situations and take an example of phase A. As shown in Fig. 2.4, the output voltage for phase A is controlled by switch S1 and S4 and the PWM signals imposed on S1 and S4 are mutually complementary. Define V_{DC} as DC link voltage for the inverter. During a complete PWM period, if the switch S1 is turned on and meanwhile switch S4 is off for the whole PWM period, in the other word, the duty cycle of the PWM wave fed to switch S1 is 100%, the output voltage of phase A should be $+\frac{V_{DC}}{2}$; on the contrary, if the switch S1 stays off and meanwhile switch S4 is on for the whole cycle, alternatively, the duty cycle for switch S1 is 0%, the output voltage of phase A turns to be $-\frac{V_{DC}}{2}$. And if both of the switches are on for one half of the cycle period, the average voltage of phase A is supposed to be 0. According to the principle of PWM introduced in Sec. 2.2, as the carrier wave of the modulation is triangular wave, from the basic geometrical knowledge it is can be found that the relation between the output phase voltage from the inverter and the corresponding PWM duty cycle is linear. Now go back to a more general case. If the PWM duty cycle stands at any one point in the range from 0 to 1, the output voltage for phase A u_a can be gained as

$$u_a = (+\frac{V_{DC}}{2}) * D_a + (-\frac{V_{DC}}{2}) * (1 - D_a), \tag{6.6}$$

where D_a is donated as the duty cycle imposed on the upper IGBT of the bridge leg connected to phase A. From Eq. 6.6, the duty cycle can be solved as

$$D_a = \frac{u_a + \frac{V_{DC}}{2}}{V_{DC}}. \tag{6.7}$$

As mentioned in Sec. 6.1.2 the complementary PWM waveforms ePWMxB for the lower IGBT on the bridge leg can be generated by reversing ePWMxA in the dead band submodule. Hence, only one counter-compare register(CMPA) is required for the generation of one pair of PWM outputs on the same PWM bridge leg. Substitute Eq. 6.7 into Eq. 6.5, the value set in counter-compare A register (CMPA) can be easily calculated. The same way applies to the other two phases. As a result, the duty cycles for three phases can be derived as

$$CMPA_a = TBPRD_a(0.5 - \frac{u_a}{V_{DC}}) \tag{6.8a}$$

$$CMPA_b = TBPRD_b(0.5 - \frac{u_b}{V_{DC}}) \tag{6.8b}$$

$$CMPA_c = TBPRD_c(0.5 - \frac{u_c}{V_{DC}}), \tag{6.8c}$$

where the subscript implies the phase that the register values or variables belong to.

6.1.3 ADC Module

The block diagram of the F28335's ADC module is shown in Fig. 6.10. It has 16 analog input channels, which are connected to an analog multiplexer(MUX). A channel to be sampled can be selected by sending its corresponding 4-bit address to the MUX. The relation between the channels and the 4-bit addresses are listed in Table 6.1.

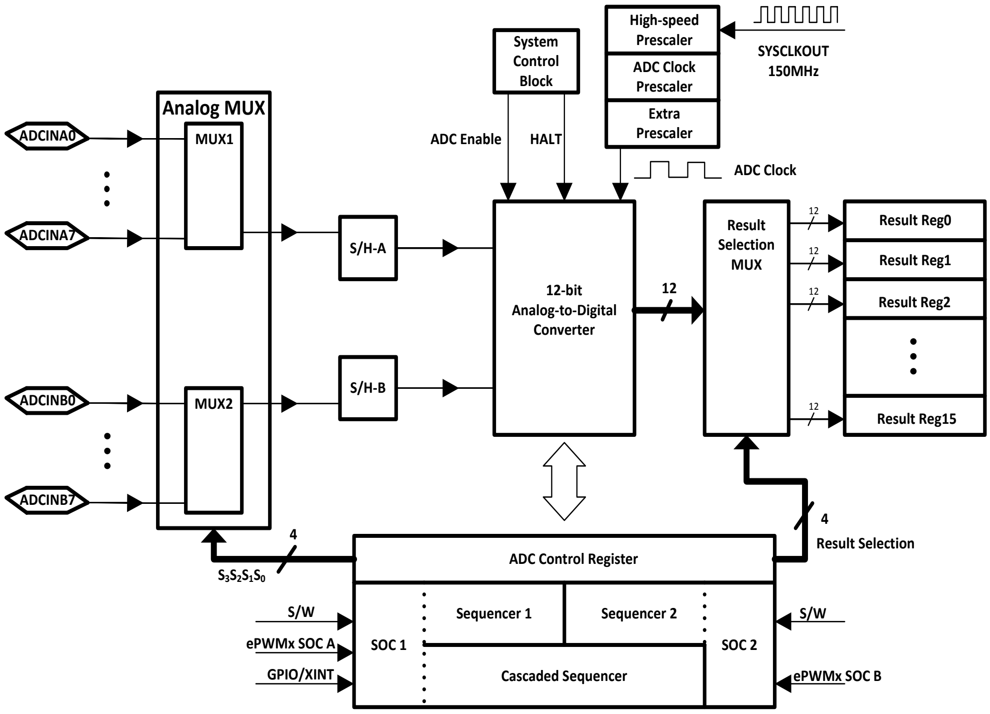


Fig. 6.10 Block diagram of ADC module

Table 6.1: Analog input channels and corresponding MUX addresses

Input channel	Address $S_3S_2S_1S_0$	Input channel	Address $S_3S_2S_1S_0$
ADCINA0	0000b	ADCINB0	1000b
ADCINA1	0001b	ADCINB1	1001b
ADCINA2	0010b	ADCINB2	1010b
ADCINA3	0011b	ADCINB3	1011b
ADCINA4	0100b	ADCINB4	1100b
ADCINA5	0101b	ADCINB5	1101b
ADCINA6	0110b	ADCINB6	1110b
ADCINA7	0111b	ADCINB7	1111b

The analog MUX consists of two 8-to-1 multiplexers, MUX1 and MUX2 in Fig. 6.10. The outputs of MUX1 and MUX2 are connected to two sample-and-hold(S/H) circuits, S/H-A and S/H-B, respectively. The two S/H circuits allow the possibility of sampling two analog signal simultaneously. After the S/H circuit has done the sampling, the analog-to-digital converter begins to transfer the analog signal held on the S/H circuit into a 12-bit binary number. The behavior of the ADC module is governed by the ADC control registers. As shown in Fig. 6.10, there are two sequencer blocks, Sequencer1 and Sequencer2, where the addresses of the channels to be sampled are placed in an appropriate order. For each sequencer, maximum eight channels could be selected. Besides, Sequencer1 and Sequencer2 can be merged into a cascaded sequencer, where maximum sixteen channels could be selected. In order to start an ADC sequence, a start-of-conversion(SOC) signal is needed. For Sequencer1, Sequencer2 or cascaded sequencer, the SOC signal could be given by the PWM module or the bit *S/W*, which can be set or reset in software. Sequencer1 and the cascaded sequencer could also be triggered by an external signal through a general-purpose input/output(GPIO) pin. The time that one conversion will take is decided by the ADC clock frequency. The ADC clock is obtained by prescaling the system clock. The total prescaling factor is equal to the product of the prescaling factors of the high-speed prescaler, the ADC clock prescaler and the extra prescaler. The detailed information about how to configure these three prescalers can be found in Ref. [29, P. 39,Table 18], Ref. [17, P. 38,Table 2 - 3] and Ref. [17, P. 34,Table 2 - 1]. The conversion results will be written into the result registers, through a result selection MUX. For each sequence, the result selection MUX will sent the first conversion result to Result Reg0, the second one to Result Reg1 and so on, until this sequence is finished.

Usually, when the analog-to-digital conversions are needed in two different moments during one control, the dual-sequence mode could be applied. However, in this project, all the analog signals are sampled at the same moment and the number of analog signals is less than eight. Due to this fact, one sequence of conversion per control loop is enough. However, cascaded-sequence mode is still applied, which can leave some flexibility to the channels and DSP pins assignment. If possible, it is desired that all the analog signals are sampled at the same moment exactly when the PWM carrier wave reaches its peak. But as there are only two S/H circuits, some compromises are made in the practical system implementation that the currents of phaseA and phaseB are sampled firstly and then the speed and the DC-link voltage are measured. The measurement of the phaseC current is given up. Anyway, the sampling mode of the ADC module has to be simultaneous. As to the ADC clock frequency, from the point of conversion speed, it is better to be as high as possible. However, too high ADC clock frequency might cause a nonlinearity to the conversion results [4]. So $12.5MHz$, the highest ADC clock frequency that the reference [4] recommends, is configured. Another issue worth considering is the width of the sampling window, which is number of clocks that the S/H circuit spends on sampling the signal. For those signals that change slow, a wider sampling window might bring the advantage of removing the noise by averaging the input signal. But in this project, the phase currents change fast, so wide sampling window will not help and might introduce some phase shifts. So the smallest possible sampling window, that is one ADC clock, is applied. Then the ADC module should work in *start/stop* mode, which means that the conversion is started by the SOC signal and stops when the sequence is finished, waiting for the next SOC signal. The SOC signal is triggered by the PWM module. At last, an interrupt request needs to be generated by ADC module once the conversion sequence is finished. The vector control algorithm will be executed in the interrupt service routines(ISR).

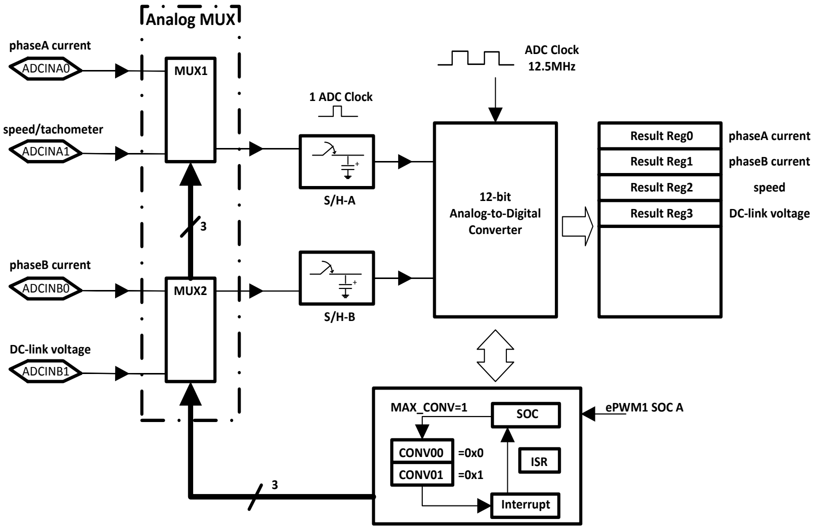


Fig. 6.11 ADC module configurations in this project

The simplified block diagram based on the ADC module configurations described above is shown in Fig. 6.11. How to configure the maximum number of conversions needs to be explained here. In simultaneous sampling mode, a pair of conversions counts for one. Besides, as the ADC module counts from 0 instead of 1, so the value of the *MAX_CONV* should be calculated as

$$\begin{aligned} MAX\_CONV &= \left\lceil \frac{No.\,of\,Conversions}{2} \right\rceil - 1 \\ &= \left\lceil \frac{4}{2} \right\rceil - 1 = 1. \end{aligned}$$

6.1.4 Encoder Module

The enhanced quadrature encoder pulse(eQEP) module of the F28335 is used to process the digital signal from the encoder. The eQEP module is able to run in two different operating modes. In quadrature-clock mode, the eQEP module receives two square wave signals from the encoder. These two square wave signals(A and B) have 90°phase shift with each other, which can be used to determine the rotation direction. In direction-count mode, one square wave signal and one direction signal are sent to the eQEP module. The counter in the module will increase and decrease depending on the direction. For both modes, an index pulse signal is used to determine the absolute position of the encoder. The operating mode of the eQEP module is decided by the type of the encoder. As the incremental encoder used in this project outputs the quadrature signals, the eQEP module is set to be working in quadrature-clock mode. A simplified block diagram of eQEP module is shown in Fig. 6.12, where only the functionalities used in the project are drawn, though the eQEP module is multi-functional [18].

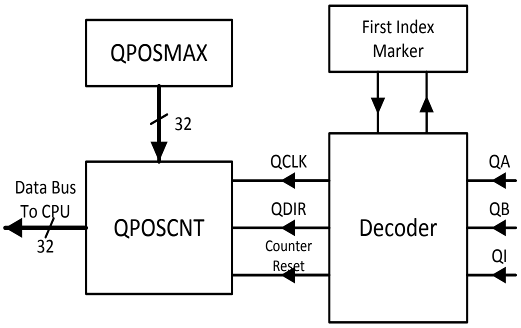


Fig. 6.12 Simplified block diagram of the eQEP module

Two quadrature waves are sent to the QA pin and the QB pin of the decoder block respectively. Every falling or rising edge of QA and QB will generate a clock signal(QCLK), which is passed to the position counter(QPOSCNT) from the decoder. The QPOSCNT will increase or decrease by 1 on each pulse of QCLK, depending on the direction signal(QDIR). If QDIR equals to 1/0, it means that the encoder rotates clockwise/anti-clockwise and the QPOSCNT will increase/decrease. The value of QDIR is determined by the relation of QA and QB. For clockwise rotation, QA leads to QB and vice versa. The detailed relation between QA, QB, QDIR and QPOSCNT are described in Table 6.2.

Table 6.2: Truth table of the decoder

Previous edge	Present edge	QDIR	QPOSCNT
QA↑	QB↑	1	increment
	QB↓	0	decrement
	QA↓	TOGGLE	increment or decrement
QA↓	QB↓	1	increment
	QB↑	0	decrement
	QA↑	TOGGLE	increment or decrement
QB↑	QA↑	0	decrement
	QA↓	1	increment
	QB↓	TOGGLE	increment or decrement
QB↓	QA↓	0	decrement
	QA↑	1	increment
	QB↑	TOGGLE	increment or decrement

An waveform example is give in Fig. 6.13 into order to provide an intuitive feeling about how eQEP works. The position counter(QPOSCNT) is reset by the index signal. When it meets the index signal for the first time, the eQEP module will remember the present edge and the rotating direction in the *first index marker* register. For example, if the first counter reset happens on the falling edge of QB during the clockwise direction(as shown in the left shadow area of Fig. 6.13), then all the later resets must be aligned with the falling edge of QB for the clockwise rotation and with the rising edge of QB for the anti-clockwise rotation [18](as shown in the right shadow area in Fig. 6.13). The position counter could also be reset by the overflow or the underflow of itself. For the underflow, the lower limiting value is 0. For the overflow,

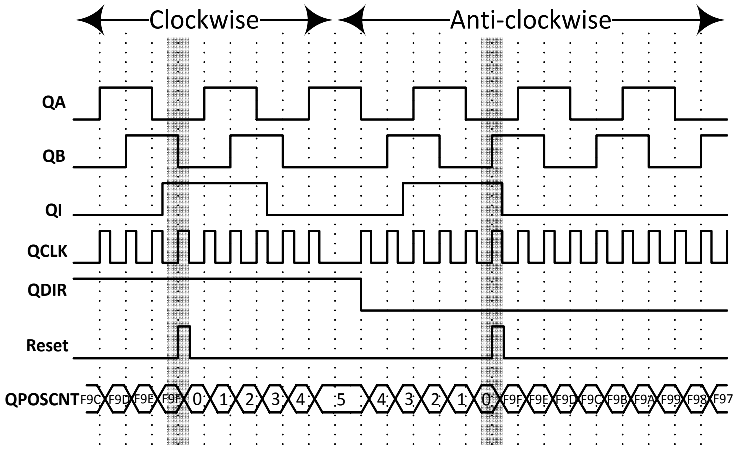

Clockwise
Anti-clockwise
QA
QB
QI
QCLK
QDIR
Reset
QPOSCNT
F9C F9D F9E F9F 0 1 2 3 4 5 4 3 2 1 0 F9F F9E F9D F9C F9B F9A F99 F98 F97

Fig. 6.13 An example waveforms of the eQEP

the upper limiting value is referred to as QPOSMAX, stored in the QPOSMAX Register. As the encoder in this project is an incremental quadrature encoder with 1000 lines, the QPOSMAX should be

$$1000 \times 4 - 1 = 3999 = 0xF9F.$$

No matter the counter is reset by the overflow, the underflow or the index signal, the counter is reset to 0 during the clockwise rotation and reset to QPOSMAX during the anti-clockwise rotation [18].

The period of the vector control loop in this project is only $0.1ms$, which means that the rotating speed of the encoder

$$n_{encoder} = \frac{1s}{0.1ms} \times 60 = 6 \times 10^5 rpm,$$

if the encoder rotates one revolution during one control loop period. Apparently it is of less possibility for the encoder to reach such high speed. Based on this fact, the algorithm that calculates the rotating speed of the encoder could be expressed using the following pseudo-code.

```
x(k-1)=x(k); //pass the old data
x(k)=QPOSCNT; //read the new position counter value
if(QDIR==1) //clockwise
{
  if(x(k)>=x(k-1))
  {
   n(k)=x(k)-x(k-1); //n(k) is the change of QPOSCNT
                     //in one control period
  }
  else
  {
   n(k)=x(k)+QPOSMAX-x(k-1);
  }
}
else(QDIR==0) //anti-clockwise
{
   if(x(k)<=x(k-1))
  {
   n(k)=x(k)-x(k-1); //n(k) is the change of QPOSCNT
  }                   //in one control period
  else
  {
   n(k)=x(k)-QPOSMAX-x(k-1);
  }
}
Speed(rpm)=n(k)/(QPOSMAX+1)*(60*1/T); //transfer result to rpm,
                                       //T is the control loop period.
```

6.1.5 DATALOG Module

The DATALOG module is not a hardware module of the F28335, instead, it is a section of highly-optimized code which will realize a virtual oscilloscope in the Code Composer Studio(CCS). The DATALOG software module stores 16-bit values of up to 4 observed variables in the data RAM, which could be illustrated in Fig. 6.14.

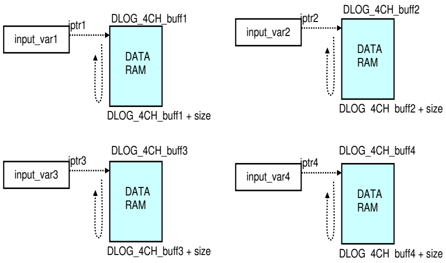


Fig. 6.14 Four memory sections of DATALOG module, quoted from [1, P. 4]

The values of the four data RAM sections are updated from the starting addresses to the ending addresses, with the four pointer(iptr1, iptr2, iptr3 and iptr4) moving from up to down. The data of these four data RAM sections are transmitted to CCS in realtime through the JTAG port and the emulator. After a set of appropriate configurations, the graphical waveforms of the observed variables are plotted in the CCS Graph Windows as Fig. 6.15 shows. The DATALOG module has brought a great convenience to the testing and the debugging of this project. The detailed information about the DATALOG module can be found in the controlSUITE™, if it has been installed in the computer. The file path is:

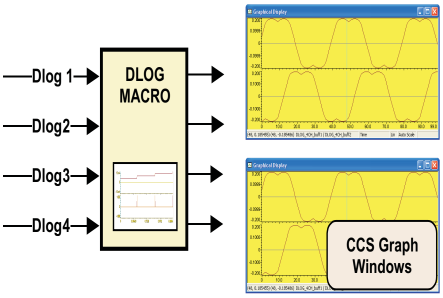


Fig. 6.15 DATALOG module and CCS Graph Windows, quoted from [27, P. 11,Figure 7]

\controlSUITE\libs\app_libs\motor_control\drivers\f2833x_v1.0\~Docs\data_log.pdf .

When using the DATALOG module, some issues needs to be considered:

- the first channel of the DATALOG module is the trigger channel, which means that only when the variable value of the first channel crosses the trigger point with rising edge, does the DATALOG module start to update. The range of trigger point is from -32768 to 32767.
- the maximum data buffer size for each channel is 32767.
- the parameter *prescaler* of the DATALOG module could be adjusted to observe the variables of the fast process or the slow process.
- 16-bit integer is the only data format that the DATALOG module uses, which means that the variables of other formats needs to be transferred to 16-bit integer first and then be written into the data RAM sections.

6.1.6 Bit-Field Address Structure

The F28335 DSP is using a 22-bit address for accessing the program memory and 32-bit address for reading or writting data memory. As ANSI C is the selected programming language, how to access the program

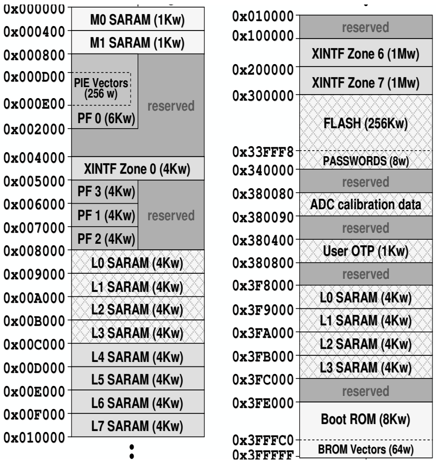


Fig. 6.16 Memory map of F28335, quoted from [3, P.2-12]

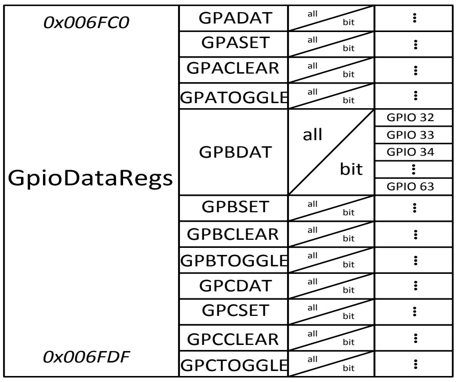


Fig. 6.17 Example of bit-field address structure

memory need rarely to be considered. But accessing the data memory is unavoidable because all the peripheral registers are arranged in the data memory.In ANSI C syntax, *structure*, *union* and *bit field* provides the possibility to access the peripheral registers rapidly and conveniently. Both *structure* and *union* are used as the data containers that contains the variables of the similar behaviors or functionalities. The biggest difference between *structure* and *union* is that members in an *union* share the same memory , while members in *structure* have their separate memory. The *bit filed* is the special member of the *structure* in ANSI C that allows the user to be able to operate some bits of an integer. The syntax for the *structure*, the *union* and the *bit-field* are well explained in Ref. [30].

Take *GpioDataRegs.GPBDAT.bit.GPIO34* as an example. It allows user to read/write value from/to No.34 general purpose input/output(GPIO) pin. As shown in Fig. 6.16, all kinds of memory are mapped into a same memory map, which means that each section of memory has its unique addresses. As mentioned before, the peripheral registers are also a part of the memory. All the registers of the peripheral modules are assigned the addresses from four different memory map areas, with the names of peripheral frame 0(PF0), peripheral frame 1(PF1), peripheral frame 2(PF2) and peripheral frame 3(PF3). The GPIO module belongs to the PF1 area. *GpioDataRegs* is an instance of a *structure*, which is assigned the address from 0x006FC0 to 0x006FDF. As seen in Fig. 6.17, this section of address covers all the 12 registers that operate the data of the GPIO module. The 12 members of *GpioDataRegs* corresponds the 12 registers respectively. In the example of *GpioDataRegs.GPBDAT.bit.GPIO34*, the *GPBDAT* is one of the 12 members of *GpioDataRegs*, which is assigned the same address as the registers that stores the states of the GPIO Port B. *GPBDAT* has two members. One is *all*, an unsigned 32-bit integer. The other one is *bit*. As *GPBDAT* is the instance of an *union*, its two members, *all* and *bit*, are assigned the same addresses. *bit* has 32 members: *GPIO32*, *GPIO33*. . . *GPIO61*, all of which are 1-bit numbers. Since both *all* and *bit* correspond to the data register of GPIO Port B, users can modify the value of GPIO Port B bit by bit like *GpioDataRegs.GPBDAT.bit.GPIO34=1* or treat it as a whole like *GpioDataRegs.GPBDAT.all=0xFFFF8888*, which increases the programming flexibility.

6.2 System Initialization and Configuration

6.2.1 Initialization

After being powered on and before executing the control loop routines, the DSP needs to be well initialized and set as shown in part B of the flow chart in Fig. 6.2. For initialization, *controlSUITE™* supports the users with initialization functions corresponding to each function unit. Users just need to call these functions at the beginning of the programme. The initialization functions only provide the most basic supports for running the system, for example to initialize the system clock or to enable some modules. The peripheral modules need to be further configured in order to be working in an expected way. The functioning units which have to be initialized in the project are list below.

- Code Start Branch
- PLL Circuit
- System Clock
- Peripheral Clock
- Interrupt Vector Table
- On-chip ADC calibration

The source files of these initialization functions could be found in *controlSUITE™* if following this file path:

```
\controlSUITE\device_support\f2833x\v132\DSP2833x_common\source
```

6.2.2 Configuration

In order to meet the requirements of different applications, each peripheral unit can run in different modes. Based on the specifications of this master thesis project, the peripheral units' configuration are set to the suitable operating modes as Table 6.3, Table 6.4 and Table 6.5. As the PWM configuration is critical to the system performance and safety, it has been explained in depth in Sec. 6.1.2.

Table 6.3: Analog-Digital converter configuration

ADC Clock	12.5MHz	Sampling Window	1 ADC Clock
Sampling Mode	Simultaneous Sampling	Sequence Mode	Cascaded
Operating Mode	Start/Stop	Trigger Source	PWM
No. of Conversions per Sequence	1	Interrupt	Sequence Finished

Codes of the ADC configuration

```
AdcRegs.ADCTRL1.bit.ACQ_PS = 0;//Width of sampling window
AdcRegs.ADCTRL1.bit.CPS = 0;
AdcRegs.ADCTRL3.bit.ADCCLKPS = 6;//ADC clock frequency
AdcRegs.ADCTRL1.bit.SEQ_CASC = 1;//Cascaded Mode
AdcRegs.ADCTRL2.bit.EPWM_SOCA_SEQ1=1;//ADC triggerred by PWM
AdcRegs.ADCTRL2.bit.INT_ENA_SEQ1 = 0x1;//Enable the interrupt
AdcRegs.ADCTRL3.bit.SMODE_SEL = 0x1;//Simutanous sampling
AdcRegs.ADCTRL2.bit.RST_SEQ1 = 0x1;//Reset the cascaded sequency
AdcRegs.ADCCHSELSEQ1.bit.CONV00 = 0;//Channel Select: ADC A0-> Phase A Current;
                                    // ADC B0-> Phase B Current
AdcRegs.ADCCHSELSEQ1.bit.CONV01 = 1;//Channel Select: ADC A1-> Speed;
                                    //ADC B1-> DC-link voltage
AdcRegs.ADCMAXCONV.bit.MAX_CONV1 = 1;//Maximum No. of Conversions
```

Table 6.4: PWM configuration

PWM frequency	10kHz	CTR Mode	Up-Down
Operating Mode	Shadow Mode	Load Mode	Load on CTR=0
Synchronization Mode	In Series	ADC Trigger Signal	CTR=0
Phase Shift	None	Dead Band Mode	Active High Complementary
Dead Band Source	ePWMxA	Action Mode	Toggle on CTR = CMPA

Codes of the PWM configuration

```
//Same part for EPWM1, EPWM2 and EPWM3:
EPwm1Regs.TBCTL.bit.CLKDIV=0; //Set CLKDIV bit as 1
                              //partly determine Time-base clock
EPwm1Regs.TBCTL.bit.HSPCLKDIV=1; //Set HSPCLKDIV as 2
                                 //partly determine Time-base clock
EPwm1Regs.TBCTL.bit.CTRMODE=2; //Set TBCTR operation mode as up-down-count mode
EPwm1Regs.TBPRD=3750; //Set TBCTR period as 3570 in up-down-count mode
EPwm1Regs.TBCTL.bit.PRDLD=0; //Load TBPRD from its shadow register
EPwm1Regs.CMPCTL.bit.LOADAMODE=0; //Load CPMA when TBCTR=0
EPwm1Regs.AQCTLA.bit.CAU=2; //When TBCTR is incrementing and TBCTR=CMPA
                            //set EPWM1A high
EPwm1Regs.AQCTLA.bit.CAD=1; //When TBCTR is decrementing and TBCTR=CMPA
                            //clear EPWM1A low
EPwm1Regs.DBCTL.bit.OUT_MODE=3; //Set both of switch S1 and S0 as 1
                                //in Dead-band submodule
EPwm1Regs.DBCTL.bit.POLSEL=2; //Set switch S3 as 1 and switch S2 as 0
                              //in Dead-band submodule
EPwm1Regs.DBCTL.bit.IN_MODE=0; //Set both of switch S4 and S5 as 0
                               //in Dead-band submodule
EPwm1Regs.ETSEL.bit.SOCAEN=1; //Enable EPWM1SOCA pulse
EPwm1Regs.ETSEL.bit.SOCASEL=1; //Generate an EPWM1SOCA pulse when TBCTR=0
//Different Part:
//For EPWM1:
EPwm1Regs.TBCTL.bit.PHSEN=0; //Synchronization event ignored
                             //and not load TBCTR with phase.
EPwm1Regs.TBCTL.bit.SYNCOSEL=1; //Generate a EPWM1SYNCO pulse when TBCTR=0
//For EPWM2:
EPwm2Regs.TBPHS.half.TBPHS=0; //Set TBCTR phase as 0 relative to EPWM2SYNI time-base
EPwm2Regs.TBCTL.bit.PHSEN=1; //Load TBCTR with TBPHS
                             //when synchronization event occurs
EPwm2Regs.TBCTL.bit.SYNCOSEL=0; //Set EPWM2SYNCO same as EPWM2SYNCI
//For EPWM3:
EPwm3Regs.TBPHS.half.TBPHS=0; //Set TBCTR phase as 0 relative to EPWM3SYNI time-base
EPwm3Regs.TBCTL.bit.PHSEN=1; //Load TBCTR with TBPHS
                             //when synchronization event occurs
```

Table 6.5: Quadrature encoder configuration

Interrupt	Disabled	Position CTR Input Mode	Quadrature-count
Maximum CTR Number	3999	Position CTR Reset Mode	Reset on Index Event

Codes of the quadrature encoder configuration

```
EQep1Regs.QDECCTL.bit.QSRC=0; //Quadrature-count mode
EQep1Regs.QDECCTL.bit.XCR=0; //2x resolution: Count the rising/falling edge
EQep1Regs.QEPCTL.bit.PCRM=0; //Position counter reset on an index event
EQep1Regs.QEPCTL.bit.QPEN =1; //eQEP position counter is enabled
EQep1Regs.QPOSMAX = 3999; //Maximum number of position counter
GpioCtrlRegs.GPAMUX2.bit.GPIO20=1; //GPIO20 as phase A pulse
GpioCtrlRegs.GPAMUX2.bit.GPIO21=1; //GPIO21 as phase B pulse
GpioCtrlRegs.GPAMUX2.bit.GPIO23=1; //GPIO23 as index
```

6.2.3 Converter Start-up

Two spare general purpose input/ouput(GPIO) pins on the DSP are assigned for the start-up of the inverter following. One is to turn on the inverter relay and the other one is to clear *FAULT_CLEAR* signal low. These two tasks will be done after system initilization and configuration and before the offset measurement as shown in the C part of Fig. 6.2.

6.2.4 Offset measurement

As a fact, both the current measurements and the speed measurement contain an small offset, that may be caused by the sensors or the ADC conditioning circuit. The offset of currents need to be measured when the machine stands still with all the peripheral devices switched on as shown in part D of Fig. 6.2. The averages of the two phase currents measured in this situation are subtracted from the ADC results in the code to compensate for the current offset. In this process, in order to get enough and accurate samples, firstly the DSP is requested to wait for 6sec in advance to let the system enter into its steady state; during the next $6sec$, 60000 samples can be collected with the control loop frequency of $0.1ms$, which are used to calculate the average. The speed offset compensation is derived by taking machine speed measurements with different speed settings at low speed and fitting the measurements in Matlab. The speed offset is estimated from the fitted curve.

6.3 Modular Programming

Considered the complexity of the whole vector control system implementation, it will only cause a mess and confusion to lump everything together. Consequently, in the code the whole system is divided into several modules and each module is packed into a head file. For each module, a specialized type of data structure is created to realize the operation of the variables in the corresponding modules.

Modular programming can lead to smoother system integration, reduced debug and troubleshooting time, a higher degree of visibility inside the software and quicker system reconfiguration. Besides, the module in one project might be reusable for other projects [27]. To be consistent with the control block diagram and the flow chart, the following modules are created and used in the program.

- 1.Clark Transformation
- 2.Park Transformation
- 3.Inv. Clark Transformation
- 4.Inv. Park Transformation
- 5.PWM
- 6.Encoder
- 7.Flux Estimator
- 8.Speed Regulator
- 9.Current Regulator
- 10.Data Logging

The tenth module *Data Logging* is an debugging tool provided by TI, which can display the variables of the code in real time when debugging. It can be used as a virtual probe and a virtual oscilloscope. Users can adjust the trigger value, size of displayed variables and pre-scalar based on practical needs.

To illustrate how the modules in the program are created and used, the module of *Clark Transformation* is taken as an example in the following.

6.3.1 Structure Definition

A module in the program is defined as a structure in C. The members of a module structure are mainly of three types: 1. interface variables, which are inputs and outputs of the module. 2. internal variables, which are intermediate values in the process of calculating the outputs, for example, the error integrator value of *Speed Regulator* module. 3.constant numbers, which reflect some properties of this module, for example, the voltage limit in *Current Regulator* module. The structure of *Clark Transformation* is defined as follows:

```
struct CLARK
{
 float Ialpha;
 float Ibeta;
 float Ia;
 float Ib;
};
```

Then a macro stands for the default values of the variables in the *Clark Transformation* module.

```
#define CLARK_DEFAULT \
{\
 0.0, \
 0.0, \
 0.0, \
 0.0  \
}
```

6.3.2 Module Operation Definition

The operations and calculations in the module are expressed as a macro with argument as following way.

```
#define CLARK_TRANS(p)\
(p.Ialpha)=(p.Ia);\
(p.Ibeta)=1/sqrt(3)*(p.Ia)+2/sqrt(3)*(p.Ib);
```

For the modules that contains the integral part, a rule is made that the outputs of these modules should be calculated first and then the integrators are updated, in order to avoid the disorder of the signals. The code of the speed regulator module is taken as an example.

```
#define SPEED_REGULATOR_UPDATE(p) \
{\
(p.speed_error)=((p.speed_ref)-(p.speed_mach))*0.10472;\
/*Calculate the error in rad/s*/\
(p.Te_ref)=\
K_P_Speed*(p.speed_error)+K_I_Speed*(p.speed_error_integral)-Ba*0.10472*(p.speed_mach);\
/*PI Regulator + Active Damping*/\
(p.i_s_q_ref_before)=(p.Te_ref)/(p.psi_R_hat)*2/(3*np);\
/*q current reference, before current limiter*/\
(p.i_s_q_ref)=_IQsat((p.i_s_q_ref_before),(Qcurrent_Limit),-(Qcurrent_Limit));\
/*q current reference, after current limiter*/\
(p.i_s_d_ref)=(p.psi_R_ref)/L_M;\
/*d current reference*/\
(p.speed_error_integral)+=\
h*((p.speed_error)+((p.i_s_q_ref)-(p.i_s_q_ref_before))*(p.psi_R_hat)/2*(3*np)/K_P_Speed);\
/*Update the error integrator*/\
}
```

6.3.3 Module Usages

At the beginning of the code, an instance of the module structure should be defined and initialized as:

```
struct CLARK Clark=CLARK_DEFAULT;
```

In the control loop of the code, firstly, the input values are passed to the module. Then the function-like macro is invoked to update the internal variables and calculate the outputs of the module. At last, the outputs are passed to the next module. In the case of the *Clark Transformation* module, the inputs are the measured phaseA and phaseB stator currents of the motor. The outputs are the corresponding Alpha and Beta currents, which are also the inputs of the *Park Transformation* module. The code below explains how to invoke the macro with argument.

```
Clark.Ia=current_A;
Clark.Ib=current_B;
CLARK_TRANS(Clark)
Park.Ialpha=Clark.Ialpha;
Park.Ibeta=Clark.Ibeta;
```

As *CLARK_TRANS(Clark)* is a macro, there is no need to add semicolon at the statement end.

6.4 Incremental Build Methodology

Even though modular programming is applied, sometimes it is difficult to debug if all the modules are lumped together at a time. Therefore the program structure is built level by level. Every time before building a new level, the old level has to be proved to work correctly.

As illustrated in Fig. 6.1, the program is spilt into three levels: open loop, with current controller and with speed controller.

In the first level. u_{sd} is set to be zero. u_{sq} is proportional to a setting frequency but is saturated within $\pm 28V$, given that the output of the DC voltage source in the lab is $60V$. The flux angle is got by integrating this setting frequency. Actually, the induction motor is working in direct-starting operating mode. If there is no error in this level, the following phenomena should be observed in steady state: 1. the non-loaded machine runs slightly less slowly than the synchronous speed; 2. I_d and I_q are constant; 3. $\hat{\omega}_1$ calculated from the flux estimator is supposed to be equal to $2\pi f_{setting}$.

In the second and the third levels, the current regulator and the speed regulator are added gradually, as indicated in Fig. 6.1. If these two level are built successfully, the dq currents and the machine speed should basically equal to their setting point. The testing results and the comparison with the simulation results will be discussed in detail in Chapter 7.

In order to switch among the three building levels conventionally, conditional compilation is applied, in the following way:

```
#define OPEN_LOOP 1
#define WITH_CURRENT_REGULATOR 2
#define WITH_SPEED_REGUALTOR 3
#define THESIS_LEVEL WITH_SPEED_REGULATOR
\\...
#if(THESIS_LEVEL==OPEN_LOOP)
\\...code1 ...
#endif
#if(THESIS_LEVEL==WITH_CURRENT_REGULATOR)
\\...code2 ...
#endif
#if(THESIS_LEVEL==WITH_SPEED_REGULATOR)
\\...code3 ...
#endif
```

Chapter 7

Testing and Evaluation

In this chapter, the testing results of each peripheral circuit are shown. For the tests with unsatisfactory results, methods to improve the performance are proposed. As mentioned in Chap. 6, the control system is programmed on three levels, namely the open loop system, the system with current regulator and the system with speed regulator, and its performance is evaluated for all three levels as well. The performance of the control system will be compared with both design specifications and simulation results.

7.1 ADC Performance

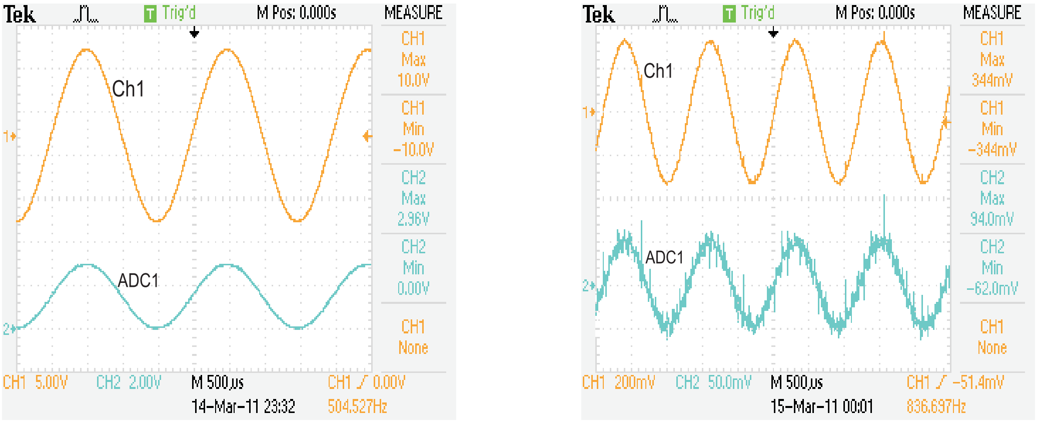
7.1.1 Full Range Test

In order to check whether the ADC conditioning circuit has met the design specification mentioned in Sec. 4.2.1, a sinusoidal signal with frequency of $500Hz$ and magnitude of $10V$ is applied on the No.1 input of the ADC conditioning circuit. The output signal of the ADC conditioning circuit is also measured. Their relation is plotted in Fig. 7.1(a).

Fig. 7.1(a) can well prove the dynamic characteristics of the circuit. At $500Hz$, there is no obvious waveform distortion or phase shift. Besides, the $-10V \sim +10V$ signal is scaled and shifted to $0V \sim 3V$. The Tektronix TDS2004 digital oscilloscope used for the testing, has only 3% DC gain accuracy at $2V/div$ [13]. Due to this fact the data read from oscilloscope can not be used to calibrate the ADC circuit, the method for calibration is explained in Sec. 4.2.2.

7.1.2 Small Signal Test

In Sec. 4.2.2, it is indicated that the relation between current sensor input and output is $V_{cx} = C + K \times i_{sx}$. In [21], it can be found that $K \approx 0.0625$ and $C \approx 2.5$. For safety reason, the DC link voltage is only 60V



(a)

(b)

Fig. 7.1 Test on ADC conditioning circuit(Oscilloscope View)

during the tests. Due to this fact the peak stator phase current in non-load and steady-state case is just about $2A$, while the rated peak current is $12.9A$. In this scenario with only $2A$, the current sensor coefficient K is too small, which means that the peak-to-peak value of the signal applied to the ADC conditioning circuit is only about $250mV$. To test the performance under the condition of small signals, a sinusoidal signal with peak values of $2.5 \pm 0.344V$ is input to the ADC conditioning circuit. The output signal from this test is not pleasant, as shown in Fig. 7.1(b), there are a lot of spikes on the output curve. The largest spike is about $50mV$, which is almost as large as the peak value of the output signal.

The testing signal mentioned above is provided by a signal generator. Moreover when the current sensor, conditioning circuit and the DSP are connected together, the situation gets worse. The current sensor itself contains much noise. If the current is calculated as Eq. 4.5, the noise will be enlarged to be intolerable because of the small sensor gain. The motor phase current calculated from the noisy voltage is plotted in Fig. 7.2 with the help of Data Logging function in Code Composer Studio(CCS). At about $10425ms$ of Fig. 7.2, the spike is almost as large as the peak-to-peak value of the current. From this figure, it can be concluded that the noise is so large that the ADC conditioning circuit cannot meet the requirements of vector control in the case of small signals.

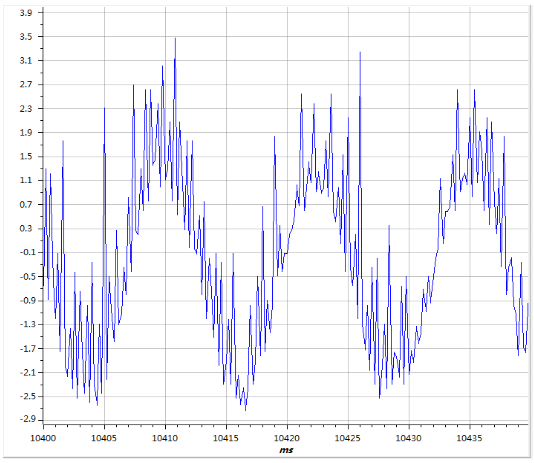


Fig. 7.2 Test on ADC conditioning circuit(CCS View)

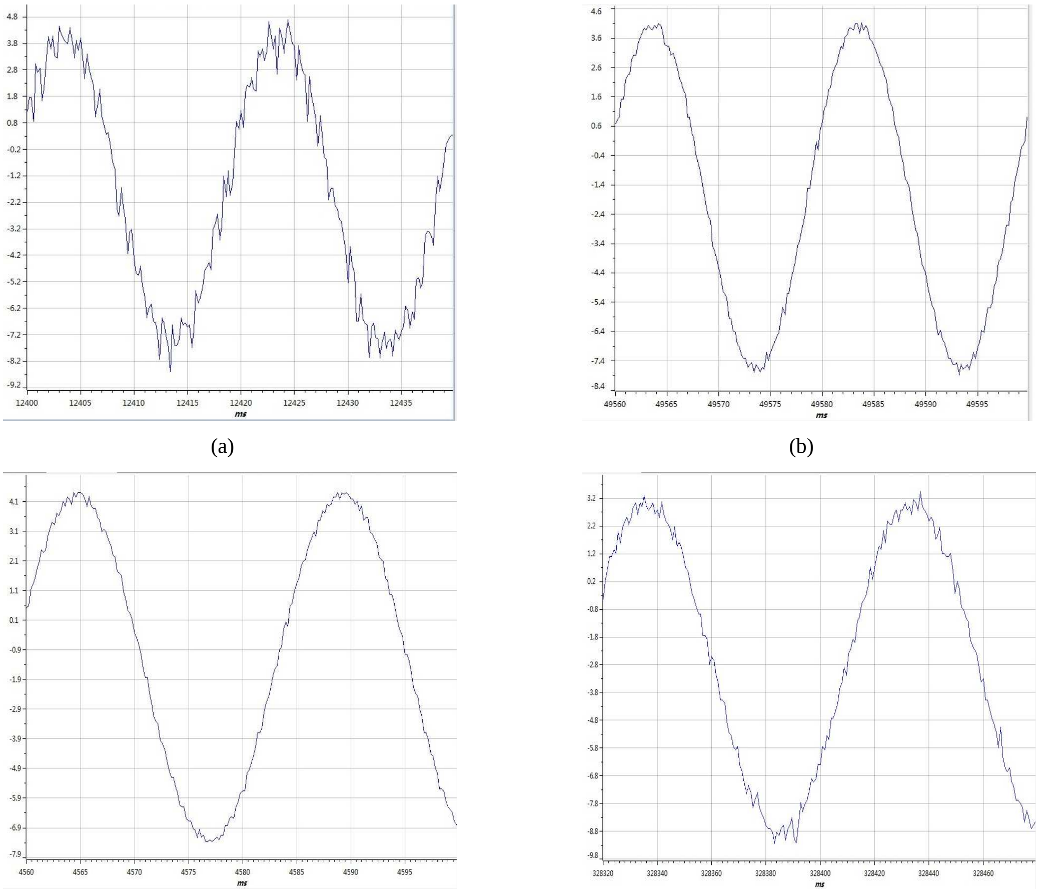
7.1.3 Possible Methods to Solve ADC Noise Problems

In this section, the possible sources of the ADC noise will be discussed and some potential solutions will be proposed. Some of the proposed solutions have been tested. To compare the different solutions under the same conditions, the testing is performed with a signal generator as input and the current is observed through the CCS. Fig. 7.3(a) shows the current wave form when no improving step is taken.

1. **Voltage Source**. In this project, $-5V$ voltage is produced by a switch-mode regulator. The on-off action of switch-mode regulator may cause the high-frequency voltage ripple on the output. Though there are much fewer types of linear negative-voltage regulators compared with positive-voltage ones, it is still highly recommended to use linear regulator to generate negative voltage. To verify this idea, the ADC conditioning circuit is powered by two 9V primary batteries. From Fig. 7.3(b) it can be reflected that a better voltage source does improve the circuit performance a lot.

2. **Layout & PCB**. There are some basic principles in electronic layout design, such as: keeping the ground of each electronic components close; keeping wire short; isolating high-frequency from low-frequency circuit and analog from digital circuit; avoiding sharp wire corner. As the whole circuit is built on experimental prototyping board in this project, it is unavoidable to break the principles above. Unlike prototyping board, PCB has advantages of copper-grounded, multi-layer and isolation. Maybe designing a PCB will cost some time, but it could be worth spending this time.

3. **Electronic Components**. In reality, no electronic component is ideal. For example, in this project, aluminium electrolytic capacitor is used to decouple the noise from the power bus. But compared with other types of capacitors, it has larger equivalent series inductance(L_{ESL}), which will reduce the



(a)

(b)

(c)

(d)

Fig. 7.3 Results of different solutions for current measurement improvement

decoupling capacity [31]. So a small ceramic capacitor is connected in parallel with each aluminium electrolytic capacitor to eliminate the effect of L_{ESL}.

4. **Reduced Number of Op-Amp**. It is guessed that the five groups of ADC conditioning circuits will affect each other. So four of the five groups are removed from the prototyping board and only one is left on the board. There are some improvements, which can be seen in Fig. 7.3(c). However, this solution is not operable, as the project needs at least four ADC channels.

5. **Independent Op-Amp**. There is an opinion on internet that single-channel Op-Amp gets better performance than dual-channel or quad-channel Op-Amp, see [34]. A similar ADC conditioning circuit was built using independent $\mu A741$ Op-Amp circuits. From Fig. 7.3(d), it can be found that independent Op-Amp helps reduce the noise even though the reason remains unknown.

6. **Only Using Voltage Divider**. Since both the current sensors used in the inverter and the speed sensor on the control panel in the lab give voltage signals between $0V$ and $3V$ and the DC-link voltage is only $60V$, it is possible to only use simple voltage dividers in this case. For the current speed signals the divider is with a gain of $\frac{1}{2}$ using two resisters of value $10k\Omega$. For the DC-link voltage signal, the gain of the divider is $\frac{1}{3}$ using two resisters of values $5k\Omega$ and $10k\Omega$.

To be safe and not encounter new problems, the simple voltage divider solution No.6 was selected to be used for the rest of the project. From the experimental results presented following, it can be seen that this method gives an acceptable noise level.

7.1.4 Sampling Issues

As the F28335 has only two sample-and-hold(S/H) circuits but only one analog-to-digital converter, the four signals to be sampled are arranged into two groups. The currents of phase A and phase B are in the first group, while the motor speed and DC link voltage are arranged into the second group. In the original plan, the currents of all the three phases should be measured. External S / H circuits needs to be built to make sure simultaneous sampling of the signals occurs. However, the original plan is given up based on the following reasons: 1. as mentioned in previous section, prototyping board is not a good choice for building accurate circuit; 2. the prototyping board has already full-arranged and has no redundant space left; 3. according to TI's motor control application notes [27], only sampling two phase will basically meet the requirements as long as the three-phase currents are well balanced.

To avoid aliasing from current ripple, the current should be sampled at the peak of the carrier wave. In this project, the peak of carrier wave signifies that the PWM counter equals 0 or 3750. But only the moment when the PWM counter equals 0 is used for sampling according to moderate accuracy specified for this project. An integer variable is inserted in the ADC interrupt service routines. Immediately after the conversion is finished, this integer will record the PWM counter value. The other integer variable is created to record the PWM counter value when the vector control calculation is completed. After the test, the previous variable for AD conversion time turns out to be 299 while the latter one for vector control calculation time appears to be 1639. These two moments are marked in Fig. 7.4. The entire AD conversion time only occupies about 4% of one control loop period and the total time of the vector control algorithm execution solely takes up about 22% of it as well.

Due to the fact that the DSP only has two sample-and-hold(S/H) circuits, only two signals can be measured simultaneously. Therefore, the four signals which need to be measured are divided into two groups. The signals in the first group are the currents of phase A and phase B, those in the other group are machine speed and DC-link voltage. The measurement procedure for each group goes as the order of ”sampling-converting-converting”. In other words, the two signal in the same group are sampled at the same time but converted sequentially. At the end of the first group conversion, the sampling of the second group will be started automatically. Considered that the entire AD conversion time is only 4% of one control loop period, it can be assumed that the sampling time of the current measurement in the first group is about 2% of it. Hence, it can be approximately considered that the whole current sampling process takes place at the peak time of the PWM carrier wave. It also can be pointed out that the difference time between the two groups is about 2% of the control loop period. Hence. Therefore, the four measured signals can be roughly thought to be sampled at the same time.

From Fig 7.4, it can be observed clearly that during one period the DSP spends more than three forth time on idling, which means that the potential of this processor has not been fully explored. If a high accuracy requirement for this vector control system is asked for, the plan for twice execution of control loop in one PWM period can be adopted. The DSP is definitely competent for it with sampling at both of the peaks of PWM carrier wave.

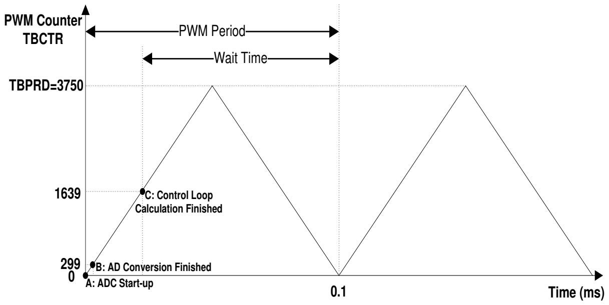
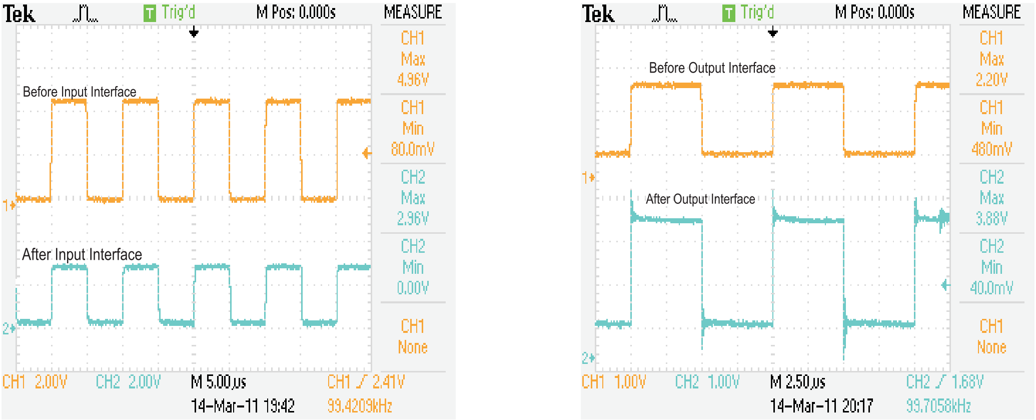


Fig. 7.4 AD conversion time and vector control algorithm execution time in one control loop period.

7.2 Digital Input and Output

As introduced in Sec. 4.3.2, the interface for digital input and output serves for three purposes: 1. Logic voltage level switch; 2. Input buffer protection; 3. Increase the output driving capacity. As the PWM signal frequency is $10kHz$, which is also the highest frequency of all the digital signals in the project, the testing signal is set to be $100kHz$ rectangular wave in order to leave a margin.



(a)

(b)

Fig. 7.5 Test of digital input and output interface

In Fig. 7.5(a), the upper wave is the input signal from the signal generator. Its high voltage is $4.96V$ and the low voltage is $0.08V$. After passing through the buffer circuit and the voltage divider, this signal is scaled down to be $2.96V$ for high voltage and $0V$ for low voltage, which agrees with LVTTL standard. Besides, the waveform still keeps the rectangular shape without distortion.

As seen in Fig. 7.5(b), the high and the low voltages of the testing output signal are $2.2V$ and $0.48V$ respectively. The processed output signal is $3.88V$ for high level and $0.04V$ for low level. It can be observed that there is spike with magnitude of about $0.4V$ on the processed output wave at the moment of step change. However, these spikes are tolerable in TTL signal standard.

7.3 Encoder Interface

As the induction motor in the lab has no place to mount the incremental quadrature encoder, the encoder is temporarily replaced by a speed sensor, which outputs voltage proportional to the speed. However, the interface and the code serving the encoder are still tested. To imitate the behavior or the encoder, two binary number 10011001b and 11001100b are shifted in cycle every $0.1ms$, where the letter b behind the numbers stands for binary number. The least significant bit of the two binary numbers are sent as digital outputs to the DSP's *EQEPA-1* and *EQEPB-1* pins respectively. After every 4000 shifts, a pulse signal will be sent to *EQEPI-1* pin. Pin *EQEPA-1*, *EQEPB-1* and *EQEPI-1* on the DSP are assigned for A, B and Index inputs from the encoder. In this scenario, if interface and code work correctly, the calculated speed should be $+150rpm$. If the two binary numbers are exchanged, the speed should be $-150rpm$. The testing results are exactly the same as expected.

7.4 Parameters Selection

In previous sections of this chapter, testing results of the peripheral circuits are exhibited. From this section, evaluation and analysis on the vector control system both in simulation and in experiment will be discussed. Firstly, all the parameters used in the simulation and and practical test in this project are listed below:

Original Motor Parameters

$R_s = 1.33\Omega$
$R_r = 1.24\Omega$
$L_{sl} = 0.008H$
$L_{rl} = 0.008H$

$L_m = 0.135H$
$J = 0.05kgm^2$
$B = 0.0007Nm/(rad/s)$
$n_p = 2$

Parameters defined in Sec. 5.1.2

$L_M = 0.1274H$
$L_\sigma = 0.0156H$
$R_R = 1.1051\Omega$

Relevant control system coefficients

$\Psi_{R,ref} = 0.08Wb$
$\alpha_c = 1000rad/s$
$\alpha_\omega = 20rad/s$
DC link voltage: 60V
Voltage limit: 28V
q current limit: 5A

$K_{pc} = 15.5524$
$K_{ic} = 15552$
$K_{p\omega} = 1$
$K_{i\omega} = 20$
$R_a = 13.1173\Omega$
$B_a = 0.993Nm/(rad/s)$

7.5 Open Loop Test with V/f Control

The open loop test is carried out by setting different synchronous frequency to the V/f controller and observing the corresponding motor speed. It should be remembered that all the tests are performed with a low DC-link voltage of $60V$ and therefore the inverter can only create a maximum phase voltage of $30V$. In this case the voltage limit is set to $28V$, thus the stator voltage is saturated at 5Hz and for higher frequencies the voltage magnitude is constant. The testing data are listed and compared with the data from simulations in Table 7.1.

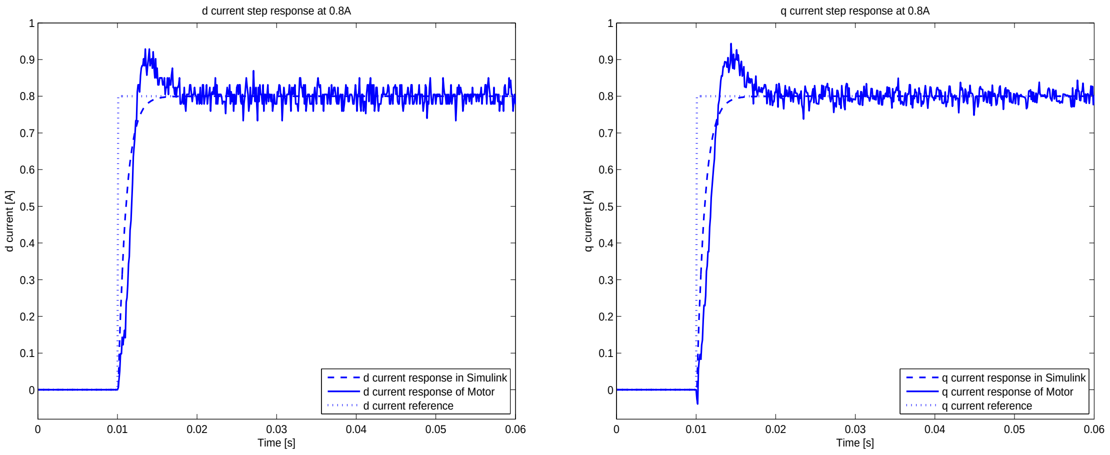
Table 7.1: Open loop test with the V/f control

Setting Frequency(Hz)	10	15	20	25	30	35	40
Synchronous Speed in Experiments(rpm)	300	450	600	750	900	1050	1200
Machine Speed (rpm)	297	444	588	729	866	996	1111
Slip	1%	1.33%	2%	2.8%	3.78%	5.14%	7.42%
Machine Speed in Simulations(rpm)	299	446	593	738	882	1024	1163
Difference between Experiments and Simulations	0.33%	0.44%	0.83%	1.2%	1.78%	2.67%	4.33%

From Table 7.1, it can be seen that the induction motor runs slightly slower than the synchronous speed. With the speed increasing, the slip will also increase. It is caused by two facts: 1. the stator voltage has been saturated which weakens the flux; 2. the friction torque becomes larger with increasing motor speed, which means that larger slip is needed to generate enough torque. Compared with the machine speed from simulations, the machine speed from the experiments has larger slip. What's more, the difference in machine speed is getting larger as the machine speed increases. This may be caused by the air drag force acting on the cooling fan when the machine runs. This force will exert an extra drag torque on the machine shaft, which is assumed to be proportional to the square of the machine speed as $T_{drag} \propto \Omega_r^2$. It is neglected in the simulations and will probably affect the machine speed in reality. The d and q currents at different frequencies are roughly compared as well. When the system enters its steady state, the average values for both d and q currents from experiments are almost the same as those from simulations in spite of noises.

7.6 Test with Current Regulator

One of the advantage of the online emulation is that the user is able to access the DSP's memory while the processor is still running. In order to capture the dynamic responses of d and q currents, two arrays are created. Once the step change in current reference is applied, in each control loop, the value of d or q current is stored in the array until the array is full. The size of the largest continuous data memory section is 2900 words, namely 2900 16-bit integers. It means that 2900 data points can be obtained for one variable response observation, 1450 data for dual variables, 725 data for quad variables et cetera. For signals with relatively slower response, such as motor speed, a scaler constant N can be added to record the variable's value every N control loop. In the code, most variables are represented by floating-point numbers. If these variables are converted directly to integer, the resolution will be greatly reduced. To guarantee enough resolution, before converting a variable, the variable is multiplied by 2048, 1024 or 512, depending on how large its maximum value is. Then when processing the recorded data in Matlab, it will be divided back to its true value.



(a)

(b)

Fig. 7.6 Step response of the d and the q current. Fig.(a) is the step response of the d current with a reference of 0.8A, and fig.(b) is the step response of the q current with a reference of 0.8A.

7.6.1 Simulation and Experimental Results

The procedure of obtaining the current step responses is that: 1. Switch to *Current Regulator* as the way introduced in Sec. 6.4; 2. Run the program and wait for about 9 seconds due to ADC calibration; 3. Change the reference of the d current from $0A$ to $0.8A$; 4. Change the reference of the q current from $0A$ to $0.8A$ after $0.5sec$. Fig. 7.6(a) and Fig. 7.6(b) show the d current and the q current step response respectively. The results from the computer simulation are also plotted in the same figure. To make a full comparison between simulation and experimental results, the time axis have been shifted.

The bandwidth α_c of the designed closed-loop current regulating system is $1000rad/s$, which means that the rise time (from 10% to 90%) of the current step response should be $ln(9)/\alpha_c s$ or $2.2ms$. From Fig. 7.6(a) and Fig. 7.6(b), it can be seen that the rise time of both i_d and i_q achieves the designing goal. Though there is noise on the current curve, it can still be noticed that the currents arrive at their reference values without a steady-state error. Unfortunately, two unexpected results are observed as well. One is that there is an overshoot in both d and q current response. Since the desired closed-loop system is a first-order system and anti-windup has been added, there should not be any overshoot as large as 12.5%. The other abnormal performance is that the current rises relatively slow at the beginning and then it has a higher rising rate until crossing the reference value for the first time. However, the step response of a first-order system is supposed to have the highest rising rate at the beginning and approach the setting point more and more slowly. In the next section, an investigation into these two unsatisfying performances will be discussed.

7.6.2 Analysis on Results

Firstly, the PI controller in the code should be proved to work in a correct way. To check this, i_d, u_d and the value of the error integrator are plotted both for the experiments and for the simulations in Fig. 7.7. Note that, the integrator output is the value which has been multiplied with K_{ic}. The same plots but with a current regulator bandwidth of $200rad/s$ are shown as well in Fig. 7.7. Observing step response of the low-bandwidth system has two benefits: 1. more clear waveform that shows how observed variables changes; 2.avoiding hitting the voltage limit means that linear analysis methods still work. In Fig. 7.7(d), at the moment when the reference step is applied, the u_d value outputted by the DSP has almost the same value as the value from the simulation. The small difference between the two u_d might be caused by the current measurement noise. Anyway, it could show that the P-part of the PI controller implemented in the DSP works appropriately. If the two solid curves in Fig. 7.7(b) and Fig. 7.7(f) are compared, it can be clearly seen that the peak value of the error integrator is reached exactly when the i_d crosses its reference value for the first time. It is reasonable, because when i_d is smaller/larger than the reference value, the error integrator will always increase/decrease. From these two curves, it can also be found that the more i_d differs from its reference value, the steeper the curve of the integrator is. These behaviors are in accordance with the characteristics of the integrators.

Both in Fig. 7.7(c) and in Fig. 7.7(d), in steady state, the u_d calculated from the DSP is larger than u_d in the simulations. There could be two possibilities which might lead to this problem. One is that the estimated value of the stator resistor is not accurate. In steady state and with ω_r and i_q being equal to 0, from Eq. 5.15 and Eq.5.26a, it can be deduced that $u_d \propto R_s$. So if R_s is increased to an appropriate value, u_d of the experiment and the simulation can be consistent with each other. The other possibility might be due to the voltage drop across the diodes and IGBT of the inverter, which implies that the actual voltage applied on the stator winding is about $1.5V$ less than expected. So in steady state, the u_d calculated by the DSP will contain a compensation for this voltage drop.

Some possible reasons are assumed to cause the overshoot. They are: inaccurate value of inductance and resistance, voltage drop, current measurement gain error and delay, noise as well as PWM dead band. Through simulations it is found that all these mentioned factors will cause more or less overshoot. However considering that the real situation is of little possibility to differ greatly from the ideal one, inaccurate value of inductance or the delay is thought to be the origin of the overshoot in the current step response. When the current sensor delay is 4 control periods($0.4ms$), the step response has about 12.5% overshoot. Then when the delay is larger than 10 control periods($1ms$), the system will become unstable. In [21], it is mentioned that the current sensor, $LEMHX10 - P/SP2$ transducer, has a bandwidth of $50kHz$, which is 5 times faster than the switching frequency and several hundreds times faster than the phase current frequency. So this possibility is basically excluded.

Slight variations of the inductance values will also cause an obvious overshoot. In simulation, the original parameters are used in the PI controller and flux estimator. Different combinations of L_s, L_r and L_m are tested as motor parameters in order to get a step response which can match the experimental results well. The step response of i_d, u_d and error integrator plotted in Fig. 7.8 employ such a set of parameters, that $L_s = 163mH$, $L_r = 123mH$ and $L_m = 120mH$. The curves of the simulation and the experimental step response match each other well except for the peak values.

7.6.3 Performance after Parameter Adjustment

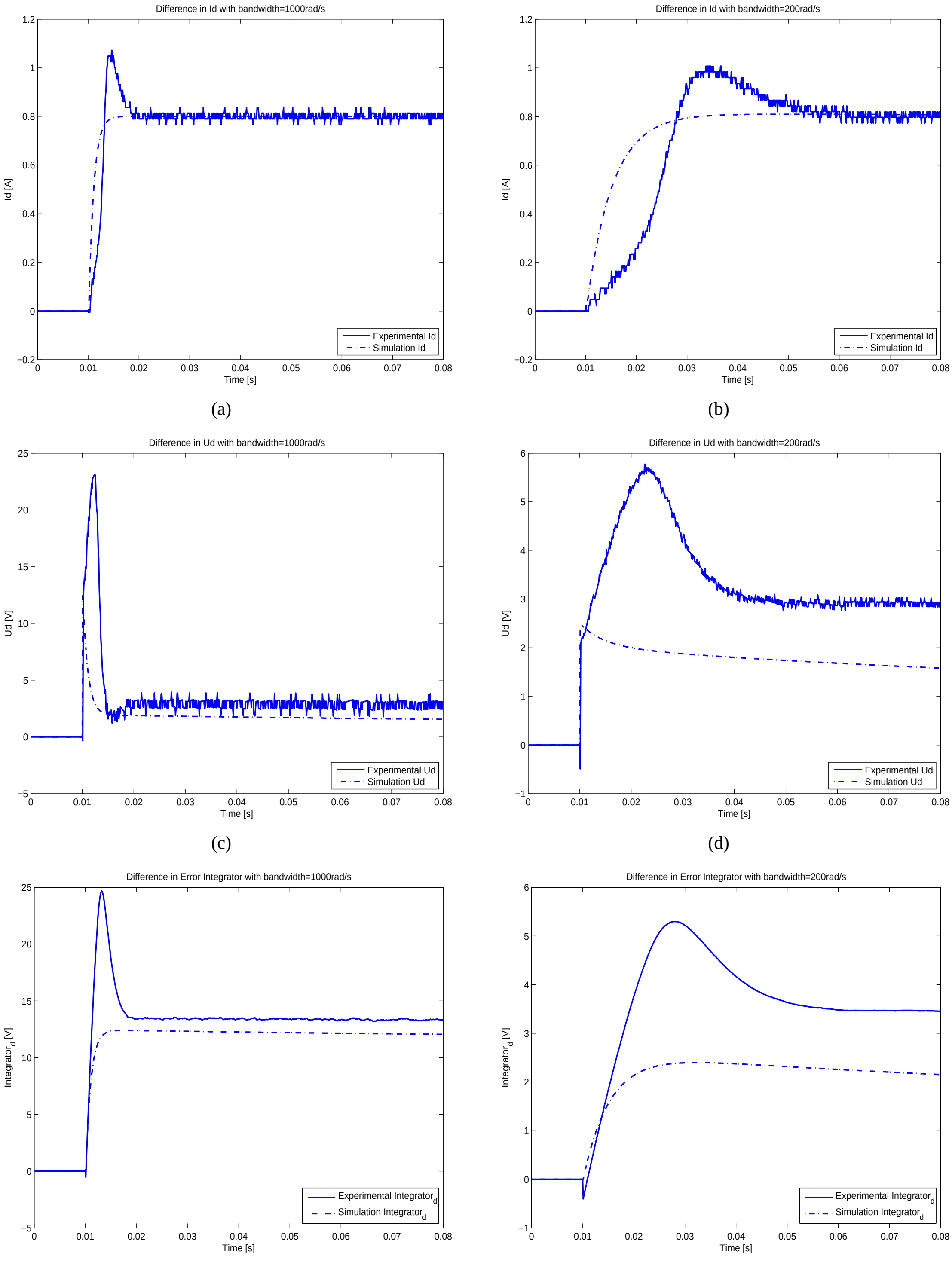
To verify the assumption of wrong motor parameter values, the current controller and the flux estimator are redesigned using the new parameters, which are $R_s = 2.7\Omega$, $L_s = 163mH$, $L_r = 123mH$, $L_m = 120mH$, $R_r = 1.24\Omega$. The stator resistance has been increased to compensate for the steady state error in the voltage. From Fig. 7.9, the step response of both d and q quantities are almost the same as the simulated ones. The biggest change compared to Fig. 7.7 is that the overshoots have disappeared. Finally, it should be emphasized that the variation of the inductance values is just a sufficient condition, but not necessarily the only condition to remove the overshoot. The exact reason(s) of the overshoot still remains unknown. The method mentioned above might not be scientific. However, it does solve the problem for this project in an engineering way.

7.7 Test with Speed Regulator

As the speed loop has a much slower response than the current loop, more data needs to be stored to study the step response of the speed loop. Due to the limit of memory used for data collection, a new variable to cut down the sampling frequency is defined in the code. With it the sampling time span for the motor speed is expanded and long enough for observing motor speed entering its steady state. In this case, 1000 memory units are reserved for speed data saving and the pre-scale factor for speed sampling is chosen as 150. With this setting the speed response information for 15 seconds can be traced and recorded in the DSP memory, using a PWM frequency of $10kHz$.

7.7.1 Simulation and Experimental Results

In the original design plan, the bandwidth for the speed control loop α_ω is chosen as $20rad/sec$. In order to eliminate the influence of nonlinearity, a much smaller bandwidth α_ω for the speed control loop and a smaller reference speed are applied to both simulation and experiment to avoid the q current hitting its limit. A clear comparison between simulation and experimental results are presented in Fig. 7.10(a), where the



(a)

(b)

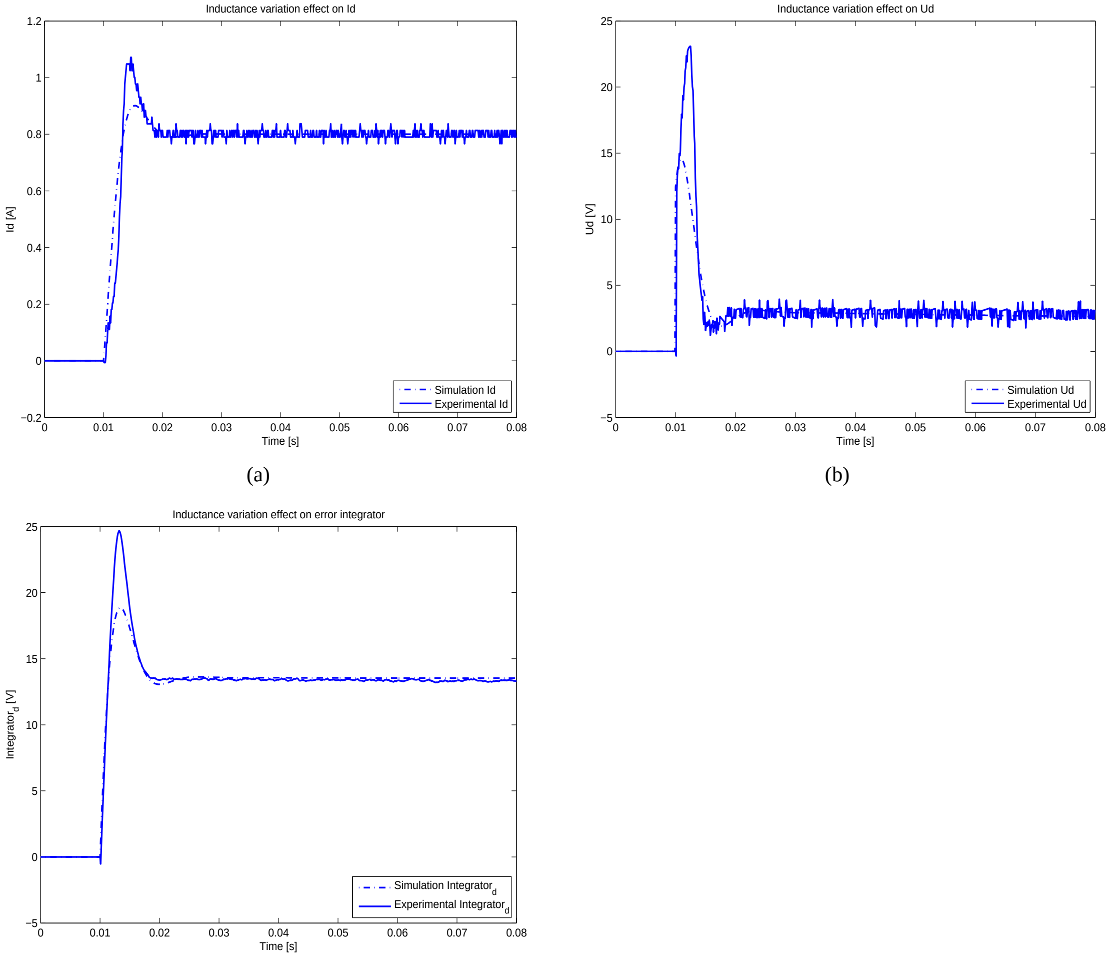
(c)

(d)

(e)

(f)

Fig. 7.7 Controller variables comparison between experimental and simulation results. Fig.(a), Fig.(c) and Fig.(e) are for a controller with a bandwidth of 1000rad/s; Fig.(b), Fig.(d) and Fig.(f) are for a controller with a bandwidth of 200rad/s. Fig.(a) and Fig.(b) are plots for the d current; Fig.(c) and Fig.(d) are plots for the d voltage; Fig.(e) and Fig.(f) are plots for the integral of the d current error. In all the figures, the solid line stands for the experimental curve and the dot-dash line stands for the simulation curve.

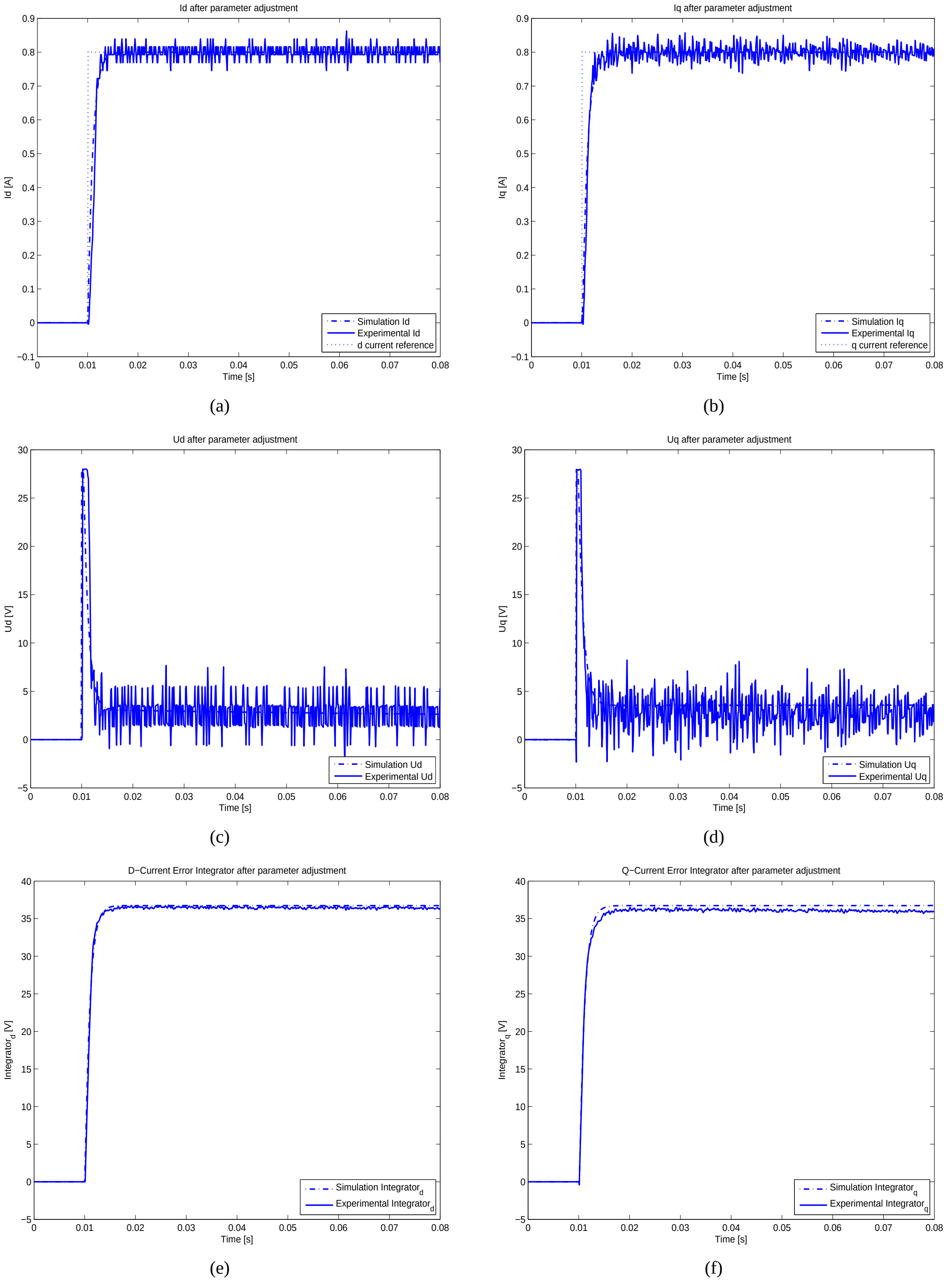


(a)

(b)

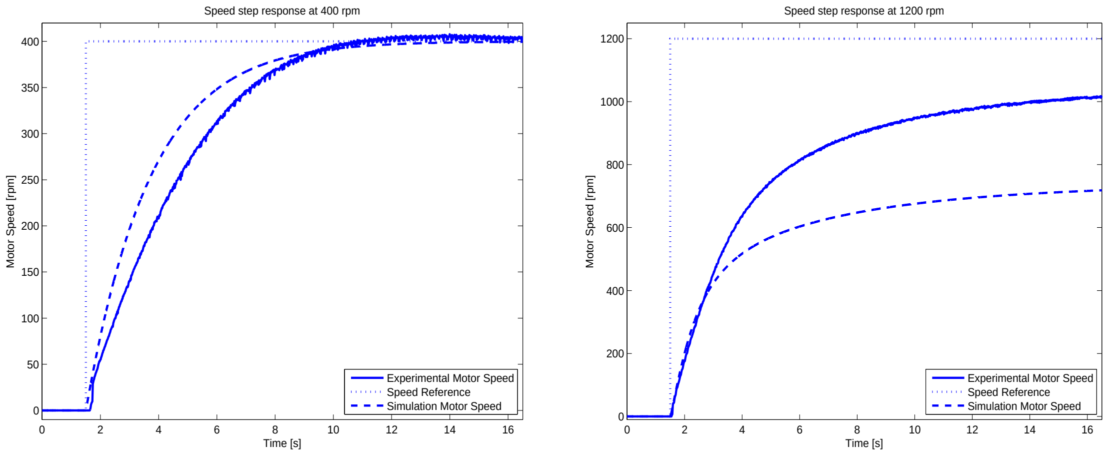
(c)

Fig. 7.8 Effect of variation in inductance values. Fig.(a) is for the d current, Fig.(b) is for the d voltage and Fig.(c) is for the integral of the d current error. In these figures, the solid line stands for the experimental curve and the dot-dash line stands for the simulation curve.



(a) (b) (c) (d) (e) (f)

Fig. 7.9 The current, the voltage and the current error integral for both d and q components after parameter adjustment. Fig.(a), Fig.(c) and Fig.(e) are for the d current, the d voltage and the integral of the d current error respectively; Fig.(b), Fig.(d) and Fig.(f) are for the q current, the q voltage and the integral of the q current error respectively. In all the figures, the solid line stands for the experimental curve and the dot-dash line stands for the simulation curve.



(a) (b)

Fig. 7.10 Speed step response at 400rpm and 1200rpm with bandwidth of 0.5rad/s. Fig.(a) is with the speed reference of 400rpm, and Fig.(b) is with the speed reference of 1200rpm. In these figures, the solid line stands for the experimental curve, the dash line stands for the simulation curve and the dot line stands for the speed reference.

bandwidth α_ω is $0.5rad/s$, the reference speed is $400rpm$ and the q current limit is $10A$. With the same bandwidth α_ω and q current limit, a high speed test is also presented to show a striking contrast with the low speed situation in Fig. 7.10(b), where reference speed is set to $1200rpm$. The causes of these differences will be explored explicitly in Sec. 7.7.2.

7.7.2 Analysis on Results

Given the speed control loop bandwidth α_ω of $0.5rad/s$, the rise time of the speed response is supposed to be 4.4 sec, which can be noticed in the simulation response in Fig. 7.10(a). By contrast, the practical response appears relatively slower than it should at the beginning. However, as it is approaching steady state it gets a larger acceleration than the simulation curve does, which causes an unnoticeable overshoot. Several causes could contribute to the difference between the simulation and experimental results. One of the potential reasons is that the reference torque generated by the speed controller is less than it should be, as a result of inaccurate speed sensing, incorrect motor parameters or inadequate coefficients of the speed PI controller. Since the gains of the speed PI controller are determined by the motor inertia J and friction coefficient B as well in IMC (Internal Model Control), it also can be affected by incorrect motor parameters. The other factor leading to this result could be inaccurate flux estimation, which could be induced by speed measurement ripple or incorrect motor parameters that are involved in the flux estimator design. Because the estimator designed in this control system is the current model, whose greatest drawback is parameter sensitivity, the variation of the parameters will affect the flux estimation significantly. The error between the estimated and the actual flux will lead to a wrong calculation of the reference of the q current, which will cause a series of chain reactions. With the assumption that current control loop has a perfect performance, $i_{sq} = i_{sq,ref}$, thus an inaccurate i_{sq} will be obtained if the estimated flux magnitude differs from the actual flux magnitude and thereby a wrong electromagnetic torque T_e will be produced, as can be seen from Fig 5.8. The error in estimated flux magnitude leads to that $T_{e,ref} \neq T_e$.

From Fig. 7.10(a), it also can be found that even if there exists some differences between simulation and experiment results, finally the reference speed of $400rpm$ can be achieved in the steady state for both situations. That will not always be the case. When the reference speed is increased to $1200rpm$ as in Fig. 7.10(b), it is impossible for the motor speed to catch up with the speed set-point for both simulation and experiment results. In the dynamic response part, the two curves can almost converge with each other, but it also can be seen clearly that the simulation still has a slightly faster response than the experiment. But the experimental curve reaches a higher level than the simulation curve, still far lower than speed set-point. The large difference in steady-state signifies one of the aspects which might affect the speed response is the difference between actual value and estimated value of the friction coefficient B. The other possibility is that the steady state difference is derived from difference between actual value and estimated value of the flux. When the machine is running at maximum speed, the back-emf is approximately equal to the voltage limit. Due to this fact, a lower flux magnitude will lead to that a higher speed can be achieved, if not considering the load.

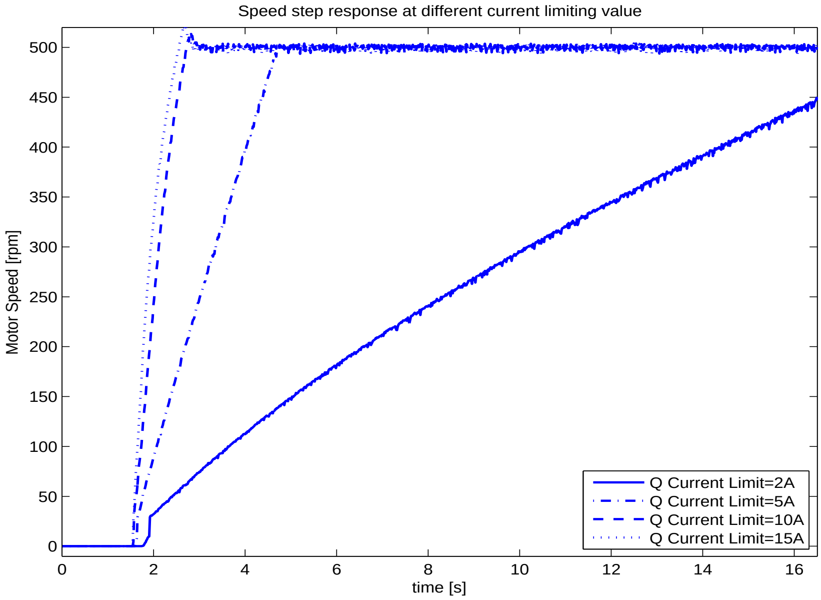


Fig. 7.11 Step responses with different current limits

Then turn back to focus on the nonlinearities for the speed controller, the saturation of the q current, and investigate the influence of different limit values of current saturation. Here, the bandwidth of the speed control loop α_ω is picked as $20 rad/s$ and the reference speed is set to $500 rpm$. The corresponding speed step responses with different current limits are listed in Fig. 7.11. There is an evident trend shown in Fig. 7.11 that the larger the q current limit is, the faster the speed step response is, which is natural. When the current limit value is really small, such as a current limit equal to $2A$, the system slips into deep saturation and a smaller q current brings about less torque produced and a smaller acceleration during the dynamic response phase. However, an unexpected difference between the responses with a high current limit and those with a low current limit is that a small overshoot can be observed in speed step response with a larger current limit, for example, current limit of $10A$ or $15A$. This overshoot is due to the fact that at these high current values the voltage is saturated for a short time. When the voltage is saturated, the actual and reference currents will not be the same and therefore the torque and the torque reference will differ for a short period. Hence, this overshoot in speed response can not be fixed by the implemented anti-windup in the speed regulator, because the speed regulator doesn't know the voltage has been saturated in the current regulator. But this problem can be solved by using the actual q current instead of the limited q current reference in the back calculation for the anti-windup for the speed controller.

Chapter 8

Conclusions

8.1 Results from Present Work

The purpose and objective of this project, mentioned in Sec. 1.2, are generally achieved. The selection of the evaluation board is proved to be a success. The F28335 DSP facilitates and accelerates the design work a lot. In this project, only a third of the processor power is utilized. For the interface board part, the digital signal interfaces, such as PWM and incremental encoder, work satisfactorily. However, the ADC conditioning circuit doesn't show pleasant performance as it adds a lot of noises to the measured signals. But this problem was solved eventually by using simple voltage dividers instead of the original ADC conditioning circuit. For the vector control design, no difficulty was met in the stage of simulation. After replacing the ADC conditioning circuit with voltage dividers, the control system was also implemented on the DSP. The vector control system on the DSP manages to regulate the dq currents and the speed fast and accurately.

When the comparison between the simulation results and the experimental results was made, some problems show up. One of them was an overshoot in both the d current and the q current during the current control loop test. Based on the assumption of parameter incorrectness, some parameter adjustment was tried to get a desirable performance, R_s was doubled and L_σ was tripled. With the new parameters used, the overshoot problem was basically overcome. But it still can't be said that the incorrectness of the parameters is exactly the reason for the overshoot, which is just one of the potential causes. The parameter adjustment changed the control coefficients in the PI regulators, K_i and K_p, and compensated for the differences between the simulations and the experiments. The other problem is that during the speed control loop test the speed response has a small overshoot and a longer rise time, compared with the results of the simulation. This is probably caused by some practical factors, such as the measured speed with noises and incorrect machine parameters, which will lead to wrong calculation of some critical variables in the vector control. Certainly, some small differences between experiments and simulations are tolerable, with these practical factors concerned.

8.2 Future Work

There remains a large potential to improve the performance of the present system. Since this thesis project is troubled by ADC noise a lot, the ADC conditioning circuit needs to be improved. More efforts should be paid on circuit layout. To avoid the disturbance of the noises in the analog signals, it would be a better choice to use digital signal instead of analog signal if possible. For example, the machine speed can be obtained by calculating the position signals from an incremental encoder (digital signal), not sensed directly by a tachometer (analog signal). Some other control strategies could also be considered for regulator design in order to make the system less sensitive to the noises or the parameter variations. Space vector PWM(SVPWM) might be introduced into this control system to replace the SPWM, as it can reduce the harmonic distortion and is more efficient in the use of supply voltage [41]. Finally, it is possible to control several motors simultaneously to make full use of the F28335 DSP's capacity, as the implementation of a single motor vector control system is far from reaching the capacity limit.

References

[1] "4-channel data logging uyility module," Digital Control System(DCS) Group, Texas Instruments, Tech. Rep.

[2] "Dc or ac drives? a guide for users of variable-speed drives (vsds)," ABB, Tech. Rep.

[3] "Digital signal controller tms320f28335 tutorial," Texas Instruments IncorporatedEuropean Customer Training Centre and University of Applied Sciences Zwickau, Tech. Rep.

[4] "F2833x analogue digital converter," Texas Instruments, Tech. Rep.

[5] "Models: Ac/dc-am9820 max 9 w," Nordic Power, Tech. Rep.

[6] "Use thelm158/lm258/lm358 dual,single supply op amp," National Semiconductor, Tech. Rep., August 1980.

[7] "Electrical characteristic of balanced voltage digital interface circuits, tia-422-b," Telecommunications Industry Association and Electrinic Industries Alliance, Tech. Rep., May 1994.

[8] "Clark & park transform on the tms320c2xx," Texas Instruments, Tech. Rep., 1997.

[9] "An887 ac induction motor fundamentals," Microchip, Tech. Rep., 2003.

[10] "Ds1103 ppc controller board hardware reference," dSPACE, Tech. Rep., March 2004.

[11] "Positive voltage regulators," STMicroelectronics, Tech. Rep., November 2004.

[12] "Understanding power supply ripple rejection in linear regulators," Texas Instruments, Tech. Rep., 2005.

[13] "Tds2000c series digital storage oscilloscopes user manual," Tektronix, Inc., Beaverton, OR 97077, USA, Tech. Rep., 2006.

[14] "Tlc070,tlc071,tlc072,tlc073,tlc074,tlc075,tlc07xa,family of wide-bandwidth high-output-drive single supply operational amplifier," Texas Instruments, Tech. Rep., September 2006.

[15] "1n5817 - 1n5819 schottky barrier rectifier," FairChild Semiconductor, Tech. Rep., August 2007.

[16] "Application of the mc34063 switching regulator," Texas Instruments, Tech. Rep., November 2007.

[17] "Tms320x2833x analog-to-digital converter (adc) module reference guide," Texas Instruments, Tech. Rep., October 2007.

[18] "Tms320x280x, 2801x, 2804x enhanced quadrature encoder pulse (eqep) module," Texas Instruments, Tech. Rep., December 2008.

[19] "1.5-a peak boost/buck/inverting switching regulator," Texas Instruments, Tech. Rep., December 2009.

[20] "Tms320x280x, 2801x, 2804x enhanced pulse width modulator (epwm) module," Texas Instruments, Tech. Rep., July 2009.

[21] "Four phase switch-mode inverter,construction and evaluation," Chalmers University of Technology, Tech. Rep., 2010.

[22] “Lecture slides of electric drives I,” Chalmers University of Technology, Department of Electric Power Enginering, Göteborg, Sweden, Tech. Rep., 2010.

[23] “Lecture slides of electric drives II,” Chalmers University of Technology, Department of Electric Power Enginering, Göteborg, Sweden, Tech. Rep., 2010.

[24] “Motor control 3 ac induction motor solutions by freescale,” Web page, September 2010. [Online]. Available: http://www.freescale.com/webapp/sps/site/application.jsp?code=APLINDMOT

[25] “Precision programmable reference,” Texas Instruments, Tech. Rep., June 2010.

[26] “Pulse-width modulation,” Web page, September 2010. [Online]. Available: http://en.wikipedia.org/wiki/Pulse-width_modulation

[27] “Tms320c2000 motor control primer, user guide,” Texas Instruments, Tech. Rep., September 2010.

[28] “Tms320f28335, tms320f28334, tms320f28332, tms320f28235, tms320f28234, tms320f28232, digital signal controllers (dscs) data manual,” Texas Instruments, Tech. Rep., March 2010.

[29] “Tms320x2833x, 2823x system control and interrupts reference guide,” Texas Instruments, Tech. Rep., March 2010.

[30] “C syntax,” Web page, February 2011. [Online]. Available: http://en.wikipedia.org/wiki/C_syntax#structures_and_unions

[31] “Capacitors and esr,” Web page, January 2011. [Online]. Available: http://wiki.xtronics.com/index.php/Capacitors_and_ESR

[32] “Operational amplifier applications,” Web page, January 2011. [Online]. Available: http://en.wikipedia.org/wiki/Operational_amplifier_applications

[33] “Pulse-width modulation,” Web page, June 2011. [Online]. Available: http://en.wikipedia.org/wiki/Pulse-width_modulation

[34] “Single or dual opamp package?” Web page, January 2011. [Online]. Available: http://www.diyaudio.com/forums/solid-state/24440-single-dual-opamp-package.html

[35] “Transistor - transistor logic,” Web page, May 2011. [Online]. Available: http://en.wikipedia.org/wiki/LVTTL

[36] “Vector control (motor),” Web page, June 2011. [Online]. Available: http://en.wikipedia.org/wiki/Vector_control_%28motor%29

[37] K. Aissa and K. D. Eddine, “Vector control using series iron loss model of induction, motors and power loss minimization,” *World Academy of Science, Engineering and Technology*, 2009.

[38] E. C. T. Center, “F2833x-numerical systems,” Texas Instruments, Tech. Rep.

[39] L. Harnefors, *Control of Variable-Speed Drives*. Department of electronics, Mälardalen University, Västerås, Sweden: Applied signal processing and control, 2002.

[40] B. W. Karl J.Åström, *Computer-Controlled Systems, theory and design*. Prentice Hall, 1997.

[41] A. Keyhani, “Pulse-width modulation(pwm) techniques,” Department of Electrical and Computer Engineering, the Ohio State University, Tech. Rep.

[42] S. Lundberg, “Control of dc-machine,” Chalmers University of Technology, Department of Electric Power Enginering, Göteborg, Sweden, Tech. Rep., 2010.

[43] K.-E. Årzen, *Real-Time Control System*. Lund Institute of Technology, The Royal Institute of Technology, 2008.