Sensorless PMSM Control for an H-axis Washing Machine Drive

Designer Reference Manual

MC56F8013/23/25
Digital Signal Controller

DRM110
Rev. 0
02/2010

freescale.com

Sensorless PMSM Control for an H-axis Washing Machine Drive

Designer Reference Manual

by: Peter Balazovic
Freescale Czech Systems Laboratories
Roznov pod Radhostem, Czech Republic

To provide the most up-to-date information, the revision of our documents on the World Wide Web will be the most current. Your printed copy may be an earlier revision. To verify you have the latest information available, refer to:

http://www.freescale.com

The following revision history table summarizes changes contained in this document. For your convenience, the page number designators have been linked to the appropriate location.

Revision History

Date	Revision Level	Description	Page Number(s)
February 2010	0	Initial release	N/A

Chapter 6
Software Design

Chapter 1
Introduction

Recent world-wide interest towards environmental friendliness, water consumption, and energy saving particularly impinge on the home appliance areas. Employing variable speed motor drives with PMSM yield an opportunity to increase overall energy efficiency. Power density and intelligent control consequently outperforms conventional uncontrolled drives. In PMSM achieving the variable speed drive requires optimal motor speed determination and position by using a shaft position sensor to successfully perform the PM motor field oriented control (FOC). Therefore, the aim is not to use this mechanical sensor to measure the position directly, but employ some indirect technique to estimate the rotor position.

The variable speed motor drives are used in modern belt-driven washing machines, where the driving motor works at lower speeds and higher torque levels during a tumble-wash cycle and high speed during a spin-dry cycle. The proposed position estimation techniques are well suited for such applications enabling better washing performance. An electronically controlled three phase interior PM motor provides a unique feature set with higher efficiency and power density. A variable speed operation with optimum performance of the interior PM motor can only be achieved when its excitation is precisely synchronized with the instantaneous rotor position.

To be able to use FOC, the position of the rotor flux has to be known prior to any control action executed. Therefore, a detection algorithm based on the injection of the pulsating HF signal in synchronous frame is used to estimate the rotor position of the rotor at start-up. This technique avoids generating unwanted rotor movements, common when conventional alignment processes are used. The enhanced back-EMF observer detects the voltages induced by the stator windings PM flux. These signals are used to calculate the rotor position and speed needed for control. Because the observed variables are not available at a low angular speed, an open loop starting procedure is implemented. To test the performance both in steady-state and transient operations, the resulting control structure has been fully analyzed through experiments.

The application software was implemented on a 16-bit fixed point DSC56F8023 digital signal controller. Sensorless drive performance over a wide range of operating conditions within a washing machine is demonstrated. It is also illustrated that the 16-bit digital signal controller that combines both MCU and DSP capabilities can realize such a demanding control technique.

Chapter 2
Washing Machine

The vertical axis (V-axis) washing machine dominates the non-European market today as residential clothes washers. There is also a large portion of them in the commercial sector. In V-axis washers, clothes move around a central agitator and must be fully immersed in water to be washed properly. The direct drive V-axis washing machine used for development of the proposed sensorless algorithm is depicted in Figure 2-1.

In horizontal axis (H-axis) washing machines, the wash drum rotates in alternate directions around a horizontal axis instead of being fully immersed. The use of H-axis washing machines is expected to reduce energy consumption significantly by reducing the water used and its heating. Currently, this type of washing machine dominates the European market. Nowadays, a belted motor drive is conventionally used in either type of washing machine. An ongoing effort is to replace such an approach using a direct drive arrangement. This arrangement provides a significant improvement in reliability and reparability. Here, the drive belt is eliminated and the direct drive washing machine's cabinet consumes less space than its belted counterpart.

The motor drive in the washing machine handles a wide range of speed. Torque and speed characteristics of a washing machine drive is shown on Figure 2-1.

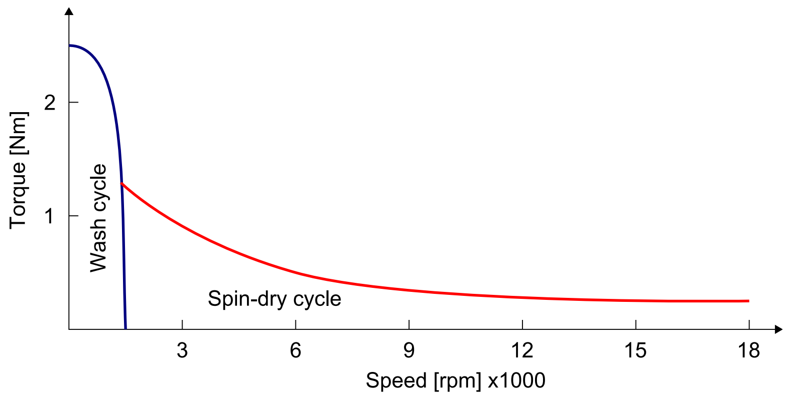


Figure 2-1. Torque and speed characteristic of an electric drive for a washing machine

Figure 2-1 shows the drive operation in one direction, but the actual drive implementation has to be able to perform speed reversal with the same torque and speed characteristic.

The actual time diagram of the tumble wash operation is shown in Figure 2-2. Here, the washtub operates in slow speed. This figure shows a speed profile diagram where initially the washtub starts accelerating to a low speed level, then there is a certain time interval of a steady spin, this is followed by the deceleration back to zero speed, and then the washtub stops. This operation sequence is marked as positive in Figure 2-2. After completion of this positive operation sequence the negative sequence is followed employing the same speed profile, but targeting the negative speed level. These operations alternate during the tumble-wash cycle. The wash program chosen determines the overall time period of the wash cycle. After completion of the wash cycle, the washing machine starts operation in the spin-dry cycle as shown in Figure 2-2. The washing machine's washtub accelerates to a pre-defined high speed level for a spin-dry cycle where it remains for a short time. Next, the washtub is commanded to decelerate either to a stop or to proceed to a low speed level. This cycle might be repeated several times, depending on the wash program chosen.

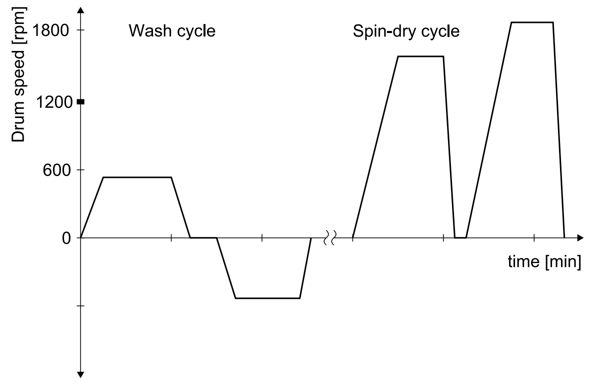


Figure 2-2. Washing machine operating cycles

Employing variable speed motor drives yields an opportunity to have more sophisticated washing programs. This enables enhancing the appliance performance and increases overall energy efficiency. An improved tumble-wash cycle can be achieved using various speed profiles with different speed ramps and speed levels. The speed levels for the spin-dry cycle can also be set arbitrarily.

Thus washing machines equipped with variable speed drive and intelligent control consequently outperform conventional and two speed washers.

2.1 Variable Speed Motor Drives

Variable speed drives (VSD) allow motors to operate continuously over a full range of speed. There are different motor categories potentially applicable to drive washing machines such as an AC induction machine (ACIM), PM motor, or a switched reluctance motor (SRM). The VSD overall efficiency is a by product of the electronic drive efficiency and the motor efficiency.

2.1.1 ACIM Speed Drive Control

In general, to achieve a variable speed operation of the ACIM, the variable frequency and variable voltage need to be supplied to the ACIM. This mode of operation is referred to as constant volt per Hz control. The ACIM speed drive efficiency can be further improved by using FOC. This requires accurate velocity information sensed by a speed or position sensor attached to the rotor. However, an additional extra sensor, connector, and wirings increase the cost of the motor drives.

2.1.2 Permanent Magnet Motor

Permanent magnet (PM) motors make use of permanent magnets to establish the flux instead of creating it from the stator winding. Replacing the electromagnetic excitation by permanent magnets has several advantages. The most obvious one is the absence of excitation losses in PM motors.

The need to use a shaft position sensor to successfully perform the control of the PM motor makes for lowering the robustness and reliability of the overall system. Therefore, the aim is not to use this mechanical sensor to measure the position directly but instead to employ some indirect techniques to estimate the rotor position. These estimation techniques differ greatly in approach for estimating the position or the type of motor they can be applied to.

Although there are several types of PM motors, they are broadly divided into brushless DC and AC synchronous motors. According to the placements of the PM in the rotor PM motors it can be further divided into several subgroups, each representing unique electrical characteristics. These include a variation in the stator core saturation, stator resistance, or inductance variation and so on, and everything dependent on the rotor position.

Chapter 3
Three-phase Permanent Magnet Synchronous Motors

Because the development of magnet materials, permanent magnet motors have become an attractive solution compared to DC or induction motors for various drive applications. This permanent magnet (PM) machine type offers important advantages. The moment of inertia can be kept minimal. The rotor construction is more robust than DC motors. An alternative construction with buried (interior) magnets are possible. Their efficiency is relatively high due to small rotor losses. For a short time, the motor currents can be applied over the rated values. Due to the permanent magnetization, the motor can be operated without a reactive current component which causes optimal inverter use.

There are two principal classes of permanent magnet AC machines. The first type, the permanent magnet synchronous motor (PMSM) is sinusoidally excited. The second type, the trapezoidal excited machine is the brushless DC (BLDC) motor. The construction differences are that while the stator windings of the trapezoidal PM machines are concentrated into a narrow-phase pole, the windings of a sinusoidal machine are typically distributed over multiple slots to approximate a sinusoidal distribution. These differences in construction are reflected in their corresponding motion characteristics. This implies the consequence that the first type of PMSM provides sinusoidal back-electromotive force (back-EMF) generation, and the second type provides trapezoidal back-EMF.

The PM synchronous motor is a rotating electric machine with a classic three-phase stator like that of an induction motor. The rotor has surface-mounted permanent magnets (see Figure 3-1).

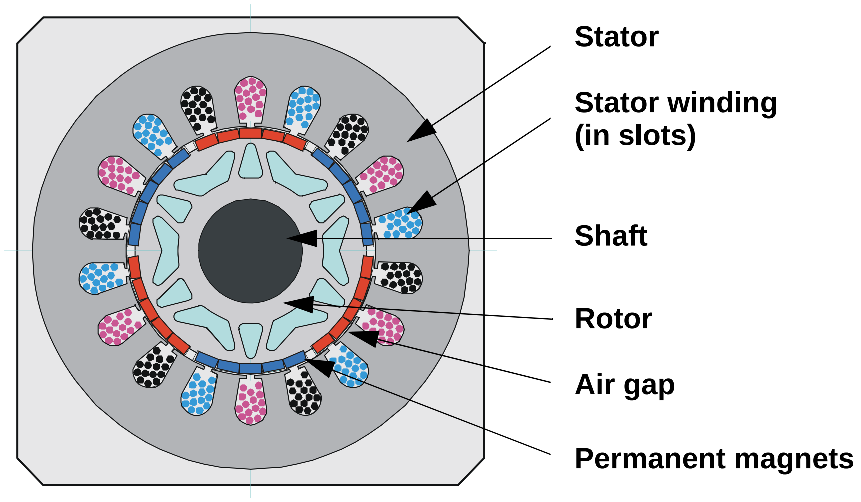


Figure 3-1. PM synchronous motor — cross section

The PM synchronous motor is equivalent to an induction motor. The air gap magnetic field is produced by a permanent magnet, so that the rotor magnetic field is constant. The PM synchronous motors offer a number of advantages when used in modern motion control systems. The use of a permanent magnet to generate a substantial air gap magnetic flux makes it possible to design highly efficient PM motors.

The torque ripple associated with the sinusoidal PM (PMSM) machine is generally less than that developed in trapezoidal (BLDC) machines. This provides the reasons that sinusoidal motors are becoming popular in many different motor control applications such as washing machines, electrical power steering, and electromechanical braking. This work targets the PMSM with only interior permanent magnets.

3.1 Mathematical Model of a PM Synchronous Motor

A clear and comprehensive description of the synchronous machines dynamic behavior is a fundamental requirement for their application in speed or torque controlled drive systems. The set of motor differential equations (Equation 3-1) are well known books documented in numerous publications by reputed authors. The stator voltage equations can be expressed as follows:

$$\begin{bmatrix} u_a \\ u_b \\ u_c \end{bmatrix} = R_S \cdot \begin{bmatrix} i_a \\ i_b \\ i_c \end{bmatrix} + \frac{d}{dt}\begin{bmatrix} \Psi_a \\ \Psi_b \\ \Psi_c \end{bmatrix} \qquad \textbf{Eqn. 3-1}$$

Equation 3-2 is the flux-linkage of each of the stator windings. and the equation for magnetic flux at the rotor are:

$$\begin{bmatrix} \psi_a \\ \psi_b \\ \psi_c \end{bmatrix} = \begin{bmatrix} L_{aa} & L_{ab} & L_{ca} \\ L_{ab} & L_{bb} & L_{bc} \\ L_{ac} & L_{bc} & L_{cc} \end{bmatrix} \cdot \begin{bmatrix} i_a \\ i_b \\ i_c \end{bmatrix} + \psi_{PM} \cdot \begin{bmatrix} \cos(\theta_e) \\ \cos\left(\theta_e - \frac{2}{3}\pi\right) \\ \cos\left(\theta_e + \frac{2}{3}\pi\right) \end{bmatrix} \qquad \textbf{Eqn. 3-2}$$

The PM synchronous machine is usually a symmetrical machine. All phase resistances, phase and mutual inductances, and flux-linkages can be thought as equal or as a function of the rotor electrical position θ_r with a 120° displacement. In the case of an interior permanent magnet machine, the saliency is presented in the motor and can be found as self and mutual inductances with variance in rotor position.

After applying substitutions to the stator phase equations (Equation 3-1), they can be transformed into the following matrix form:

$$\begin{bmatrix} u_a \\ u_b \\ u_c \end{bmatrix} = R_S \cdot \begin{bmatrix} i_a \\ i_b \\ i_c \end{bmatrix} + \begin{bmatrix} L_{aa} & L_{ab} & L_{ca} \\ L_{ab} & L_{bb} & L_{bc} \\ L_{ac} & L_{bc} & L_{cc} \end{bmatrix} \cdot \frac{d}{dt}\begin{bmatrix} i_a \\ i_b \\ i_c \end{bmatrix} + \frac{\partial}{\partial t}\begin{bmatrix} L_{aa} & L_{ab} & L_{ca} \\ L_{ab} & L_{bb} & L_{bc} \\ L_{ac} & L_{bc} & L_{cc} \end{bmatrix} \cdot \frac{d\theta_r}{dt} \cdot \begin{bmatrix} i_a \\ i_b \\ i_c \end{bmatrix} + \begin{bmatrix} u_{backEMFa} \\ u_{backEMFb} \\ u_{backEMFc} \end{bmatrix} \qquad \textbf{Eqn. 3-3}$$

Where the $\mathbf{u_{backEMF}}$ vector column matrix means:

$$\begin{bmatrix} u_{backEMFa} \\ u_{backEMFb} \\ u_{backEMFc} \end{bmatrix} = \omega_r \cdot \psi_{PM} \cdot \frac{d}{dt}\begin{bmatrix} \cos(\theta_r) \\ \cos\left(\theta_r - \frac{2}{3}\pi\right) \\ \cos\left(\theta_r + \frac{2}{3}\pi\right) \end{bmatrix} \qquad \textbf{Eqn. 3-4}$$

The number of equations can be reduced if an appropriate transformation is introduced. They can be transformed either into a stationary reference frame (a two phase system is fixed to the stator and is called $\alpha\beta$) or a rotating reference frame (a two phase system is fixed to the rotor and is called *dq*).

Analogous to standard DC machines, and AC machines develop maximal torque when the armature current is perpendicular to the flux linkage. Thus, if only the fundamental harmonic is considered, the torque T_e developed by an AC machine is given by:

$$T_e = \frac{pp}{2} \cdot \left(\frac{i_a \cdot u_{backEMFa}}{\omega_m} + \frac{i_b \cdot u_{backEMFb}}{\omega_m} + \frac{i_c \cdot u_{backEMFc}}{\omega_m} \right) = \frac{i_a \cdot u_{backEMFa}}{\omega_r} + \frac{i_b \cdot u_{backEMFb}}{\omega_r} + \frac{i_c \cdot u_{backEMFc}}{\omega_r} \qquad \textbf{Eqn. 3-5}$$

Where:

- ωm is the mechanical shaft speed of the rotor
- ωr is the electrical speed of the rotor in electrical radians per second
- pp is the number of poles for the motor.

Any mismatch between the back-EMF waveform and the corresponding phase currents result in a torque ripple. The torque ripple is minimized by maintaining sinusoidal phase motor currents. Generally, it is assumed that stator windings are approximated as sinusoidally distributed windings. The majority of the PM synchronous motors are designed so that the stator windings produce a relatively fine approximation of a sinusoidal distributed air gap magneto-motive force.

The relation between mechanical and electrical quantities are described by the mechanical equation as follows:

$$J_m \cdot \frac{d}{dt}\omega_r = T_e - \mathrm{sign}(\omega_r) \cdot T_L - B_m \cdot \omega_r - T_f \qquad \textbf{Eqn. 3-6}$$

Where:

- J_m is a total mechanical inertia $[kg \cdot m^2]$
- B_m viscous friction coefficient $[N \cdot m \cdot s]$
- T_f is a Coulomb friction torque $[N \cdot m]$
- T_l is the mechanical load on the shaft $[N \cdot m]$

3.1.1 Vector Transformations

This section gives an overview to the theory of the most commonly used reference frames and provides equations that allow for easy conversion amongst them. Using these techniques, it is possible to transform the phase variable machine description to another reference frame. The choice of the reference frame can considerably simplify the complexity of the mathematical model of the PM synchronous motor.

Three-phase PM machines can be conventionally modeled using a phase variable notation as can bee seen in Equation 3-3. However, for a three phase star-connected machine, the phase quantities are not independent variables. Due to this redundancy, it is possible to transform the three phase system to an equivalent two-phase representation. Hence, the transformation from three-phase to two-phase quantities is as follows:

$$\begin{bmatrix} f_\alpha \\ f_\beta \end{bmatrix} = \frac{2}{3} \cdot \begin{bmatrix} 1 & \cos\left(\frac{2}{3}\pi\right) & \cos\left(\frac{4}{3}\pi\right) \\ 0 & \sin\left(\frac{2}{3}\pi\right) & \sin\left(\frac{2}{3}\pi\right) \end{bmatrix} \cdot \begin{bmatrix} f_a \\ f_b \\ f_c \end{bmatrix} \qquad \textbf{Eqn. 3-7}$$

This transformation in Equation 3-7 is valid for the current, voltage, and flux linkages as well. Introduced transformation can be written in the inverse form as follows:

$$\begin{bmatrix} f_a \\ f_b \\ f_c \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ \cos\left(\frac{2}{3}\pi\right) & \sin\left(\frac{2}{3}\pi\right) \\ \cos\left(\frac{4}{3}\pi\right) & \sin\left(\frac{4}{3}\pi\right) \end{bmatrix} \cdot \begin{bmatrix} f_\alpha \\ f_\beta \end{bmatrix} \qquad \textbf{Eqn. 3-8}$$

Transformation Equation 3-7 and Equation 3-8 are commonly known as the forward Clarke and the inverse Clarke transformation, respectively. Defined vector quantities are shown in Figure 3-2.

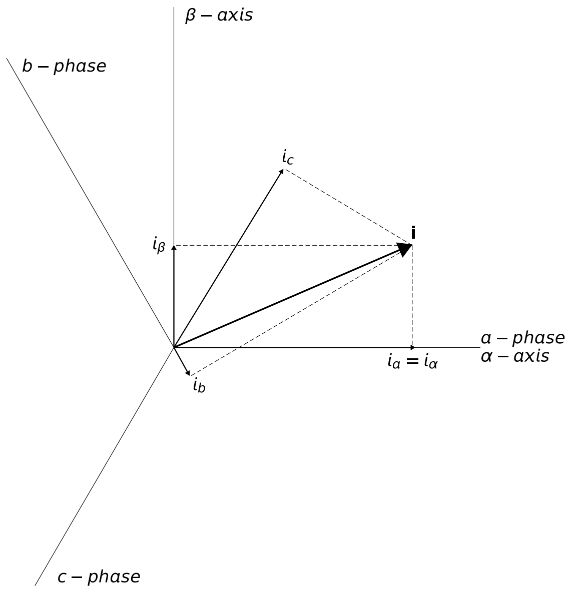


Figure 3-2. Stationary reference frame in relation with phase quantities

The real $\alpha - \text{axis}$ of the defined stationary coordinate system is chosen to coincide with the $a - \text{axis}$. The $\beta - \text{axis}$ lies in quadrature with the $\alpha - \text{axis}$. This reference frame with α, β quantities is known as a stationary reference frame that is fixed to the stator.

Meanwhile the vectors of the current, voltage and flux linkages rotate around these axes at a rate equal to the angular frequencies of the corresponding phase quantities. Besides this stationary reference frame attached to the stator, there can be formulated a general reference frame that rotates vector quantities through a known angle. This can be represented by following matrix formula as:

$$\begin{bmatrix} f_d \\ f_q \end{bmatrix} = \begin{bmatrix} \cos(\theta) & \sin(\theta) \\ -\sin(\theta) & \cos(\theta) \end{bmatrix} \cdot \begin{bmatrix} i_\alpha \\ i_\beta \end{bmatrix} \qquad \textbf{Eqn. 3-9}$$

The motor current vector consists of the direct axis component id and the component iq, that is called the quadrature axis component. The relationship of the direct and quadrature components of the current vector in the original stationary two-axis reference frame and the rotating reference frame is shown in Figure 3-3.

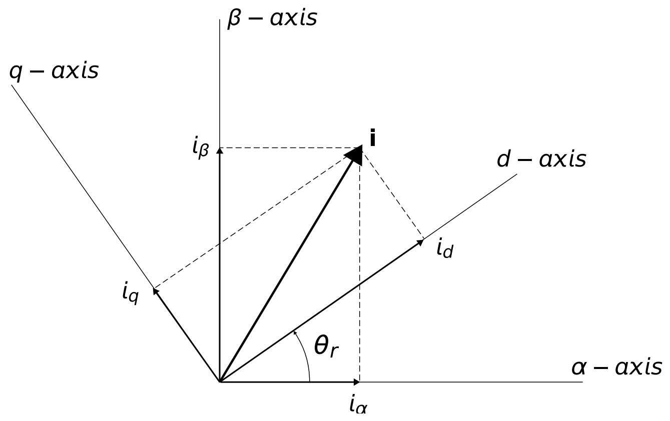


Figure 3-3. Rotating reference frame in relation with stationary reference frame

This transformation is known as the inverse Park transformation. The transformation from a rotating to a stationary reference frame is given as:

$$\begin{bmatrix} f_\alpha \\ f_\beta \end{bmatrix} = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix} \cdot \begin{bmatrix} f_d \\ f_q \end{bmatrix} \quad \textit{Eqn. 3-10}$$

This vector transformation from rotating reference frame to the stationary one is commonly known as the forward Park transformation. The elimination of position dependency from the PM motor electrical variables are the main advantage of an introduced vector rotation, see Equation 3-9 and Equation 3-10.

3.1.2 PM Motor in the Stationary Reference Frame

The PM motor equation in the stationary reference frame ($\alpha\beta$ coordinates) can be expressed in the scalar forms of the stator voltage and flux linkage defined as:

$$\begin{bmatrix} u_\alpha \\ u_\beta \end{bmatrix} = \begin{bmatrix} R_S & 0 \\ 0 & R_S \end{bmatrix} \cdot \begin{bmatrix} i_\alpha \\ i_\beta \end{bmatrix} + \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \cdot \frac{d}{dt} \begin{bmatrix} \Psi_\alpha \\ \Psi_\beta \end{bmatrix} \quad \textit{Eqn. 3-11}$$

This is an equivalent to Equation 3-12. There can be defined two useful terms by establishing direct axis inductance L_d, and quadrature axis inductance L_q. The mean inductance L_0 and differential inductance ΔL are as follows:

$$L_0 = \frac{L_d + L_q}{2}$$
$$\Delta L = \frac{L_d - L_q}{2} \quad \textit{Eqn. 3-12}$$

Where L_0 is the average inductance, ΔL is the zero-to-peak differential inductance is a direct measure of the spatial modulation of the inductance. In a surface mounted machine L_q and L_d are almost equal, so ΔL is very small. The buried magnet machine has a large difference between the d-axis and q-axis inductance due to the spatial modulation produced by the difference in the flux coupling between the stator and rotor.

The corresponding flux-linkage equations are:

$$\begin{bmatrix} \psi_\alpha \\ \psi_\beta \end{bmatrix} = \begin{bmatrix} L_0 + \Delta L \cdot \cos(2\theta_r) & \Delta L \cdot \sin(2\theta_r) \\ \Delta L \cdot \sin(2\theta_r) & L_0 - \Delta L \cdot \cos(2\theta_r) \end{bmatrix} \cdot \begin{bmatrix} i_\alpha \\ i_\beta \end{bmatrix} + \psi_{PM} \cdot \begin{bmatrix} \cos(\theta_r) \\ \sin(\theta_r) \end{bmatrix}$$

Eqn. 3-13

After substitution, equation Equation 3-13 is derived when Equation 3-12 and Equation 3-13 are combined in such way that Equation 3-12 flux linkages are replaced with Equation 3-13. The equation described is:

$$\begin{bmatrix} u_\alpha \\ u_\beta \end{bmatrix} = R_S \cdot \begin{bmatrix} i_\alpha \\ i_\beta \end{bmatrix} + \begin{bmatrix} L_0 + \Delta L \cdot \cos(2\theta_r) & \Delta L \cdot \sin(2\theta_r) \\ \Delta L \cdot \sin(2\theta_r) & L_0 - \Delta L \cdot \cos(2\theta_r) \end{bmatrix} \cdot \frac{d}{dt}\begin{bmatrix} i_\alpha \\ i_\beta \end{bmatrix}$$
$$\omega_r \cdot \left(2\Delta L \begin{bmatrix} -\sin(2\theta_r) & \cos(2\theta_r) \\ \cos(2\theta_r) & \sin(2\theta_r) \end{bmatrix} + \psi_{PM} \cdot \begin{bmatrix} \cos(\theta_r) \\ \sin(\theta_r) \end{bmatrix} \right)$$

Eqn. 3-14

Here, the inductance matrix is represented by a PM motor in a rotating reference frame as:

$$\mathbf{L}(2\theta_r) = \begin{bmatrix} L_0 + \Delta L \cdot \cos(2\theta_r) & \Delta L \cdot \sin(2\theta_r) \\ \Delta L \cdot \sin(2\theta_r) & L_0 - \Delta L \cdot \cos(2\theta_r) \end{bmatrix}$$

Eqn. 3-15

Electromagnetic torque can be expressed in terms of the stator flux and stator current as below:

$$T_e = \frac{3}{2} \cdot pp \cdot (\psi_\alpha \cdot i_\beta - \psi_\beta \cdot i_\alpha)$$

Eqn. 3-16

The electromagnetic torque T_e developed by PM synchronous motors can be divided into two components. The first component is created by contribution of the permanent magnet flux ψ_{PM} and is called synchronous torque. The second component referred to as reluctance torque arises due to rotor saliency where the rotor tends to align with the minimum reluctance. So the resulting torque is given by:

$$T_e = T_{syn} + T_{rel}$$

Eqn. 3-17

Where, for the stationary reference frame, synchronous and reluctance torque components are described as follows:

$$T_{syn} = \frac{3}{2} \cdot pp \cdot \psi_{PM} \cdot (i_\beta \cos(\theta_\beta) + i_\alpha \sin(\theta_r))$$

Eqn. 3-18

$$T_{rel} = \frac{3}{2} \cdot pp \cdot \left(-\Delta L i_\alpha i_\beta \cos(2\theta_r) - \frac{1}{2}(i_\alpha^2 + i_\beta^2) \sin(2\theta_r) \right)$$

Eqn. 3-19

Although a quadrature axis PMSM model in a stationary reference frame is already simple, when compared to the full three phase model. It still contains a position variant inductance matrix. Both currents and voltages are also AC values that make it difficult for a control structure. Therefore an additional quadrature model is introduced.

3.1.3 PM Synchronous Motor in Rotating Reference Frame

The vector transformations of Equation 3-7 and Equation 3-9 are used to derive this model, assuming that the transformation argument θ is equal to the electrical rotor position θ_r. The stator voltage equation in the rotating reference frame fixed to the rotor is given as:

$$\begin{bmatrix} u_d \\ u_q \end{bmatrix} = \begin{bmatrix} R_S & 0 \\ 0 & R_S \end{bmatrix} \cdot \begin{bmatrix} i_d \\ i_q \end{bmatrix} + \begin{bmatrix} s & \omega_r \\ -\omega_r & s \end{bmatrix} \cdot \begin{bmatrix} \psi_d \\ \psi_q \end{bmatrix}$$ ***Eqn. 3-20***

Where *p* represents a derivative operator. The stator flux linkage is expressed as:

$$\begin{bmatrix} \psi_d \\ \psi_q \end{bmatrix} = \begin{bmatrix} L_d & 0 \\ 0 & L_q \end{bmatrix} \cdot \begin{bmatrix} i_d \\ i_q \end{bmatrix} + \psi_{PM} \cdot \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$ ***Eqn. 3-21***

In a salient PM synchronous machine, there is a difference between the rotor d-axis (main flux direction) and the rotor q-axis (main torque producing direction) inductances $L_d \neq L_q$. After substitution, the flux linkage equation into the stator voltage equation, then the interior permanent magnet motor expressed in the stationary reference frame can be derived forming the following equation.

$$\begin{bmatrix} u_d \\ u_q \end{bmatrix} = \begin{bmatrix} R_S & -\omega_e L_q \\ \omega_e L_d & R_S \end{bmatrix} \cdot \begin{bmatrix} i_d \\ i_q \end{bmatrix} + \begin{bmatrix} L_d & 0 \\ 0 & L_q \end{bmatrix} \cdot \frac{d}{dt} \begin{bmatrix} i_d \\ i_q \end{bmatrix} + \omega_e \Psi_{PM} \cdot \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$ ***Eqn. 3-22***

Electromagnetic torque can be expressed in terms of the stator flux and stator current as below:

$$T_e = \frac{3}{2} \cdot pp \cdot (\psi_d \cdot i_q - \psi_q \cdot i_d)$$ ***Eqn. 3-23***

Electromagnetic torque T_e developed by PM synchronous motors can be divided into two components. The first component is created by contribution of the permanent magnet flux ψ_{PM} and is called synchronous torque. The second component, referred to as reluctance torque, arises due to rotor saliency, where the rotor tends to align with minimum reluctance. So the resulting torque is given by:

$$T_e = T_{syn} + T_{rel}$$ ***Eqn. 3-24***

Where, for the stationary reference frame, synchronous and reluctance torque components are described as follows:

$$T_{syn} = \frac{3}{2} \cdot pp \cdot \psi_{PM} \cdot i_q$$ ***Eqn. 3-25***

$$T_{rel} = \frac{3}{2} \cdot pp \cdot \Delta L i_d i_q$$ ***Eqn. 3-26***

3.2 Vector Control of PM Synchronous Motor

3.2.1 Fundamental Principle of Vector Control

High-performance motor control is characterized by smooth rotation over the entire speed range of the motor, full torque control at zero speed, and fast acceleration and deceleration. To achieve such control, vector control techniques are used for PM synchronous motors. The vector control techniques are usually also referred to as field-oriented control (FOC). The basic idea of the vector control algorithm is to decompose a stator current into a magnetic field-generating part and a torque-generating part. Both components can be controlled separately after decomposition. The structure of the motor controller is then as simple for a separately excited DC motor.

Figure 3-4 shows the basic structure of the vector control algorithm for the PM synchronous motor. To perform vector control it is necessary to perform these steps:

1. Measure the motor quantities (phase voltages and currents).
2. Transform them into the two-phase system (α,β) using a Clarke transformation.
3. Calculate the rotor flux space vector magnitude and position angle.
4. Transform stator currents into the d, q reference frame using a Park transformation.

Also keep these points in mind:

- The stator current torque (i_{sq}) and flux (i_{sd}) producing components are separately controlled.
- The output stator voltage space vector is calculated using the decoupling block.
- The stator voltage space vector is transformed by an inverse Park transformation back from the d, q reference frame into the two-phase system fixed with the stator.
- The output three-phase voltage is generated using space vector modulation.

To be able to decompose currents into torque and flux producing components (i_{sd}, i_{sq}), the position of the motor-magnetizing flux must be known. This requires accurate sensing of the rotor position and velocity. Incremental encoders or resolvers attached to the rotor are naturally used as position transducers for vector control drives.

In some applications, the use of speed and position sensors are not desirable. In these applications the aim is not to measure the speed and position directly, but to instead employ some indirect techniques to estimate the rotor position. Algorithms that do not employ speed sensors are called “sensorless control.”

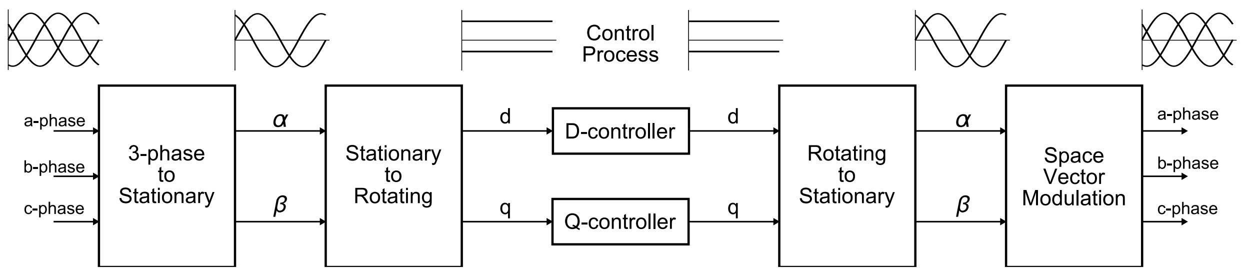


Figure 3-4. Vector control transformations

3.2.2 Description of the Vector Control Algorithm

The overview block diagram of the implemented control algorithm is illustrated in Figure 3-5. Similar to other vector-control-oriented techniques, it is possible to control the field and torque of the motor separately. The aim of this control is to regulate the motor speed at a predefined level. The speed command value is set by high level control. The algorithm is executed in two control loops. The fast inner control loop is executed with a hundred μsec period range. The slow outer control loop is executed with a period of msec range.

To achieve the goal of the PM synchronous motor control, the algorithm uses feedback signals. The essential feedback signals are three-phase stator current and stator voltage. For the stator voltage, the regulator output is used. For correct operation, the presented control structure requires a speed feedback on the motor shaft. In the case of the presented algorithm, a sensorless algorithm is used.

The fast control loop executes two independent current control loops. They are the direct and quadrature-axis current (i_{sd},i_{sq}) PI controllers. The direct-axis current (i_{sd}) is used to control the rotor-magnetizing flux. The quadrature-axis current (i_{sq}) corresponds to the motor torque. The current PI controllers' outputs are summed with the corresponding d and q axis components of the decoupling stator voltage. Thus, the desired space vector for the stator voltage is obtained and then applied to the motor. The fast control loop executes all the necessary tasks to be able to achieve an independent control of the stator current components. These include:

- Three-phase current reconstruction
- Forward Clark transformation
- Forward and backward Park transformations
- Rotor magnetizing flux position evaluation
- DC-bus voltage ripple elimination
- Space vector modulation (SVM)

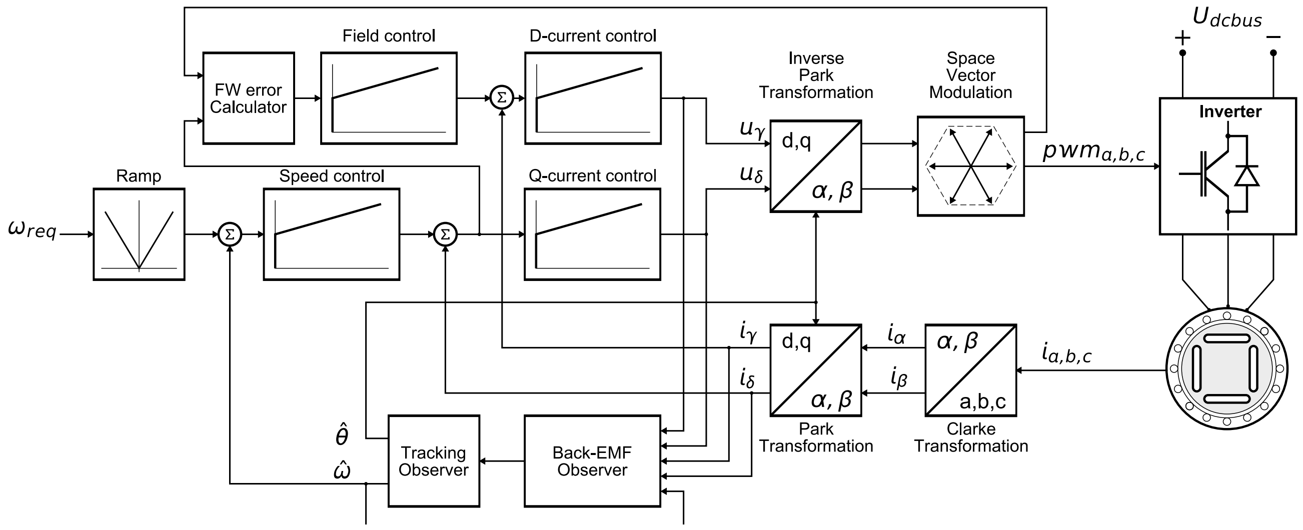


Figure 3-5. PMSM vector control algorithm overview

The slow control loop executes the speed controller and lower priority control tasks.The PI speed controller output sets a reference for the torque producing quadrature axis component of the stator current i_{q_ref} and flux producing current i_{d_ref}.

3.2.3 Current Control

Figure 3-5 shows the structure d-q current control. It is the most inner loop in the vector control of PM synchronous motor. In vector control techniques, both the cross-coupling terms as well as the back-EMF induced voltage term in Equation 3-22 are compensated using a feed-forward voltage, as follows:

$$u_d^{decoupling} = -\omega_r L_q i_q$$
$$u_q^{decoupling} = \omega_r L_d i_d + \omega_r \psi_{PM}$$

Eqn. 3-27

This helps to establish full decoupled d and q axis current loops, allows independent control of current d and q components, and simplifies the motor circuit. The resultant dynamics are is:

Eqn. 3-28

$$u_d = Ri_d + L_d\frac{di_d}{dt}$$

$$u_q = Ri_q + L_q\frac{di_q}{dt}$$

Both channels employ a PI controller with additional decoupling terms that reduce dynamic interactions. Both equations are structurally identical, therefore the same controller design approach can be applied for both d and q controllers. In the case that inductance parameter Ld (Lq) are not equal, then this results with the controllers having different Kp, Ki gains.To simplify the PI controllers design, the effects of AD and inverter transportation delay are neglected.

Considering a general form of a closed-loop with the PI controller and RL model as a plant, the closed-loop transfer function of the reference response is derived in a Laplace form, as follows:

$$G_C(s) = \frac{K_p s + K_i}{s} = K_p + \frac{K_i}{s}$$

Eqn. 3-29

Plant model is:

$$G_P = \frac{1}{Ls + R} = \frac{\frac{1}{R}}{\frac{L}{R}s + 1}$$

Eqn. 3-30

The characteristic polynomial of the transfer function G(s) form reveals in the closed-loop that it is a second order system.

$$G(s) = \frac{I(s)}{I(s)} = \frac{G_C(s)G_P(s)}{1 + G_C(s)G_P(s)} = \frac{\frac{K_P}{L}s + \frac{K_I}{L}}{s^2 + \left(\frac{K_P + R}{L}\right)s + \frac{K_I}{L}}$$

Eqn. 3-31

However, the PI controller introduces a zero to the closed-loop transfer function for commanded changes, and it is located at $-\frac{K_I}{K_P}$. This derivative characteristic of the loop increases the system overshoot by lowering the potential closed-loop bandwidth. It might be compensated by introducing a zero cancellation block in the feedforward path that has the following transfer function.

$$G_{ZC}(s) = \frac{\frac{K_I}{K_P}}{s + \frac{K_I}{K_P}}$$ **Eqn. 3-32**

The resulting transfer function with an implemented zero cancellation is then:

$$G(s) = \frac{\frac{K_I}{K_P}}{s + \frac{K_I}{K_P}} \cdot \frac{\frac{K_P}{L}s + \frac{K_I}{L}}{s^2 + \left(\frac{K_P + R}{L}\right)s + \frac{K_I}{L}}$$ **Eqn. 3-33**

The zero cancellation transfer function Gzc(s) has to be designed to compensate the closed-loop zero at $-\frac{K_I}{K_P}$, but with the unity DC gain. The unity DC gain ensures the closed-loop gain to be the same with or without zero cancellation, thus preserving the original loop gain. The DC gain can be verified by calculating Equation 3-32 with s=0.

By having the closed-loop zero canceled, the PI controller can be designed by comparing the closed-loop characteristic polynomial with that of a standard second order system as:

$$G_W(s) = \frac{\omega_0^2}{s^2 + 2\xi\omega_0 s + \omega_0^2}$$ **Eqn. 3-34**

Comparison can be made as follows:

$$s^2 + \left(\frac{K_P + R}{L}\right)s + \frac{K_I}{L} = s^2 + 2\xi\omega_0 s + \omega_0^2$$ **Eqn. 3-35**

Where ω0 is the natural frequency of the closed-loop system (loop bandwidth) and ξ is the loop attenuation. The proportional and integral gains of the PI controller can be calculated from Equation 3-35 as:

$$K_P = 2\xi\omega_0 L - R$$
$$K_I = \omega_0^2 L$$ **Eqn. 3-36**

Equation 3-36 describes a PI controller design in time domain. However, in the digital implementation, the control loop is calculated in discrete steps rather than continuously, therefore a discrete representation of the controller has to be used. Implementing the control system in a digital domain also inserts a transportation delay into the control loop. This delay is associated with the hold where each value of u(kT) is held until the next value is available.

3.2.4 Speed Control

The model of the plant in the speed loop is derived from the machine mechanical equation (2.8). Hence the model transfer function is given as follows:

Where kT is the torque constant representing the ratio between torque and current, similar to the current control loop, the speed loop is closed by a PI controller that enables the speed control with zero steady state error. The speed controller directly drives the required current that is the input to the current loop and hence the required torque when multiplied by kT. Because the mechanical time constants are bigger than the electrical time constants, the speed loop is sampled slower than the current loop. The current loop transients can be considered to be settled between the speed loop sampling instants that allow to substitute the whole current loop by a first order lag. The difference in mechanical and electrical time constants also implies that the speed controller will be saturated most of the time during the speed transient, this is because the current limit is reached much faster than the speed settling time. Therefore, it is crucial for the speed controller to have an anti-windup implemented.

In contrast to the current control loop, the parameters of the plant model in speed loop are usually difficult to determine. This is predominantly caused by variations in the system lumped inertia that changes with each mechanical arrangement of the system. The mechanical load on the motor shaft acts as a disturbance to the speed control loop, making the loop characteristic non-linear. Therefore, the mathematical derivation of the speed controller gains is difficult. However, it remains possible to determine the speed loop bandwidth through experiments. One possible approach is to use a sinusoidal reference to drive the commanded speed and observe the attenuation and phase shift of the measured speed.

3.2.5 Flux Weakening Control

The operation beyond the machine base speed requires the PWM inverter to provide output voltages higher than its output capability limited by its dc-link voltage. This in turn saturates the motor drive control system. By manipulating the -axis current into the machine it has the desired effect of weakening its field and the need for higher voltages as the machine exceeds its base speed. An anti-saturation control loop is then implemented in this software to regulate the duty-cycle to a desired maximum value under this condition, using as control variable the d-axis current.

3.2.6 Digital Controller

The PI algorithm in the continuous time domain Equation 3-29 can be rewritten into the discrete time domain by approximating the integral and derivative terms. The integral term is approximated with the Backward Euler Method, also known as backward rectangular or right-hand approximation as follows:

$$u_I(k) = u_I(k-1) + \Delta T_s e(k)$$ ***Eqn. 3-37***

The discrete time domain representation of the PI algorithms:

$$u(k) = K_P e(k) + u_I(k-1) + K_I e(k)$$ ***Eqn. 3-38***

Where:

- e(k)—Input error at step k; processed by the P and I terms
- u(k)—Controller output at step k
- Kp—Proportional gain
- Ki—Integral gain
- ΔT_S —Sampling time (period) [sec]

The discrete time domain representation of the PI algorithm scaled into the fractional range.

$$u_{FRAC}(k) = K_P^{FRAC} e_{FRAC}(k) + u_I^{FRAC}(k-1) + K_I^{FRAC} e_{FRAC}(k)$$

Eqn. 3-39

The PI algorithm in Equation 3-39 is implemented in the parallel (non-interacting) form allowing the user to define the P and I parameters independently without interaction. The PI controller parameters in fractional representation are determined as follows:

$$K_P^{FRAC} = K_P \frac{E_{MAX}}{U_{MAX}}$$
$$K_I^{FRAC} = K_I \Delta T_S \frac{E_{MAX}}{U_{MAX}}$$

Eqn. 3-40

Where:

- Emax—Input error maximal range, usually controller error range is the same as the controlled variable
- Umax—Controller output maximal range

The individual parameter (such as KIsc) of the PI algorithm is represented by two parameters in the processor implementation:

$$K_P^{FRAC} = f16PropGain \cdot 2^{i16PropGainShift}$$
$$K_I^{FRAC} = f16IntegGain \cdot 2^{i16IntegGainShift}$$

Eqn. 3-41

To determine parameters for the GFLIB_ControllerPIp function, first the calculation of the appropriate shift is derived as follows:

$$(1 > K_P^{FRAC} \cdot 2^{-i16PropGainShift} \geq 0.5) / \ln$$
$$(\ln(1) > \ln(K_P^{FRAC}) - i16PropGainShift \cdot \ln(2) \geq \ln(0.5))$$
$$\frac{\ln(K_P^{FRAC}) - \ln(1)}{\ln(2)} < i16PropGainShift \leq \frac{\ln(K_P^{FRAC}) - \ln(0.5)}{\ln(2)}$$
$$\ln(1) = 0$$

Eqn. 3-42

Due to simplification only the left side of Equation 3-42 is considered where ln(1) is omitted.

$$\frac{\ln(K_P^{FRAC})}{\ln(2)} < i16PropGainShift$$

Eqn. 3-43

$$i16PropGainShift = ceil\left(\frac{\log(K_P^{FRAC}) - \log(0.5)}{\log(2)}\right)$$
$$f16PropGain = K_P^{FRAC} \cdot 2^{-i16PropGainShift}$$

Eqn. 3-44

$$\text{i16IntegGainShift} = \text{ceil}\left(\frac{\log(K_I^{FRAC}) - \log(0.5)}{\log(2)}\right)$$
$$\text{f16IntegGainShift} = K_I^{FRAC} \cdot 2^{-\text{i16IntegGainShift}}$$

Eqn. 3-45

Division is usually a real number. Therefore "ceiling" returns the number rounded up, away from zero, and to the nearest multiple of significance.

In the case of the recurrent form of the Proportional-Integral (PI) controller, different techniques are used to convert the continuous PI controller function into the discrete representation.

NOTE

The continuous function can only be approximated and the discrete representation can never be exactly equivalent. The resulting difference equation derived by the discretization method is in the form as reported below:

$$u(k) = u(k-1) + CC1 \cdot e(k) + CC2 \cdot e(k-1)$$

Eqn. 3-46

The transition from the continuous to the discrete time domain reveals the following controller coefficients

Table 3-1. Controller coefficients

Controller Coefficients	Bilinear transformation	Backward rectangular	Forward rectangular
CC1	$K_P + \frac{K_I \Delta T_S}{2}$	$K_P + K_I \Delta T_S$	K_P
CC2	$-K_P + \frac{K_I \Delta T_S}{2}$	$-K_P$	$-K_P + K_I \Delta T_S$

Where:

- K_P—Proportional gain
- K_I—Integral gain
- T_S—Sampling period

The discrete time domain representation of the recurrent PI algorithm scaled into the fractional range is:

$$\text{f16Uk} = \text{f32Acc} + \text{f16CC1} \cdot \text{f16Error} + \text{f16CC2} \cdot \text{f16ErrorK\_1}$$

Eqn. 3-47

Where f32Acc is the accumulated controller portion over time and is used as the internal variable of this algorithm. The f16CC1 and f16CC2 are recurrent controller coefficients that are adapted as:

$$\mathrm{f16CC1} = \mathrm{CC1} \cdot \frac{\mathrm{Error\_max}}{\mathrm{U\_max}}$$

Eqn. 3-48

$$\mathrm{f16CC2} = \mathrm{CC2} \cdot \frac{\mathrm{Error\_max}}{\mathrm{U\_max}}$$

Eqn. 3-49

For proper operation of the recurrent PI controller implemented on the 16 and 32-bit DSC care must be taken due to the fixed point representation of individual values. A scaling shift ui16NShift is introduced to be scaled to the 16-bit fixed point format. Then, fractional representation on the DSC of the recurrent PI controller is calculated by the following formula as:

$$\mathrm{f16Uk} = \mathrm{f16Acc} + \mathrm{f16CC1Sc} \cdot \mathrm{f16Error} + \mathrm{f16CC2Sc} \cdot \mathrm{f16ErrorK1}$$

Eqn. 3-50

Where:

$$\mathrm{f16CC1Sc} = \mathrm{f16CC1} \cdot 2^{-\mathrm{ui16NShift}}$$

Eqn. 3-51

$$\mathrm{f16CC2Sc} = \mathrm{f16CC2} \cdot 2^{-\mathrm{ui16NShift}}$$

Eqn. 3-52

ui16NShift is chosen so that the coefficients reside within the common range <-1, 0.9999>. In addition, the ui16NShift is chosen as a power of 2. The final de-scaling is a simple shift operation.

$$\mathrm{ui16NShift} = \max\left(\mathrm{ceil}\left(\frac{\log(\mathrm{abs}(\mathrm{f16CC1}))}{\log 2}\right), \mathrm{ceil}\left(\frac{\log(\mathrm{abs}(\mathrm{f16CC2}))}{\log 2}\right)\right)$$

Eqn. 3-53

The zero cancellation transfer function can also be transformed into Z-domain using the Backward Euler Method as follows:

$$s \approx \frac{1 - z^{-1}}{\Delta T_S}$$

$$\frac{1}{s} = \frac{z}{z-1} \cdot \Delta T_S$$

Eqn. 3-54

$$a = \frac{K_I}{K_P}$$

$$G_{ZC}(z) = \frac{a}{\frac{1-z^{-1}}{\Delta T_S} + a} = \frac{a\Delta T_S}{1-z^{-1}+a\Delta T_S} = \frac{\frac{a\Delta T_S}{1+a\Delta T_S}}{1-\frac{1}{1+a\Delta T_S}z^{-1}} \qquad \textbf{Eqn. 3-55}$$

$$G_{ZC}(z) = \frac{\frac{\frac{K_I}{K_P}\Delta T_S}{1+\frac{K_I}{K_P}\Delta T_S}}{1-\frac{1}{1+\frac{K_I}{K_P}\Delta T_S}z^{-1}} = \frac{\frac{K_I\Delta T_S}{K_P+K_I\Delta T_S}}{1-\frac{K_P}{K_P+K_I\Delta T_S}z^{-1}}$$

The discrete implementation is therefore given by:

$$y(k) = \frac{K_I\Delta T_S}{K_P+K_I\Delta T_S}\cdot x(k) + \frac{K_P}{K_P+K_I\Delta T_S}\cdot y(k-1) \qquad \textbf{Eqn. 3-56}$$

It is clear that Equation 3-56 is a simplified form of the first order low-pass filter in IIR implementation. The zero cancellation block physically behaves as a low pass filter, smoothing the input command.

The GDFLIB_FilterIIR1 function calculates the first order infinite impulse response (IIR) filter. The IIR filters are also called recursive filters because both the input and the previously calculated output values are used for calculation. The first order IIR filter in the Z-domain is given as:

$$H(z) = \frac{b_1 + b_2 z^{-1}}{1 + a_2 z^{-1}} \qquad \textbf{Eqn. 3-57}$$

Transformed into a time domain difference equation:

$$y(k) = b_1\cdot x(k) + b_2\cdot x(k-1) + a_2\cdot y(k-1) \qquad \textbf{Eqn. 3-58}$$

This function is used to perform a zero cancellation transfer function with the following parameters:

$$b_1 = \frac{K_I\Delta T_S}{K_P+K_I\Delta T_S}$$

$$b_2 = 0 \qquad \textbf{Eqn. 3-59}$$

$$a_2 = \frac{K_P}{K_P+K_I\Delta T_S}$$

3.2.7 Space Vector Modulation (SVM)

The space vector modulation (SVM) can directly transform the stator voltage vectors from the two-phase α,β-coordinate system into the pulse-width modulation (PWM) signals (duty cycle values).

The standard technique of output voltage generation uses an inverse Clarke transformation to obtain three-phase values. Using the phase voltage values, the duty cycles needed to control the power stage switches are then calculated. Although this technique gives good results, space vector modulation is more straightforward (valid only for transformation from the α,β-coordinate system).

The basic principle of the standard space vector modulation technique can be explained with the help of the power stage schematic diagram given in Figure 3-6. Eight possible switching states (vectors) are feasible regarding the three-phase power stage configuration. See Figure 3-6. These states are given by combinations of the corresponding power switches. A graphical representation of all the combinations is the hexagon shown in Figure 3-7. There are six non-zero vectors, U_0, U_{60}, U_{120}, U_{180}, U_{240}, U_{300}, and two zero vectors, O_{000} and O_{111}, defined in α,β coordinates.

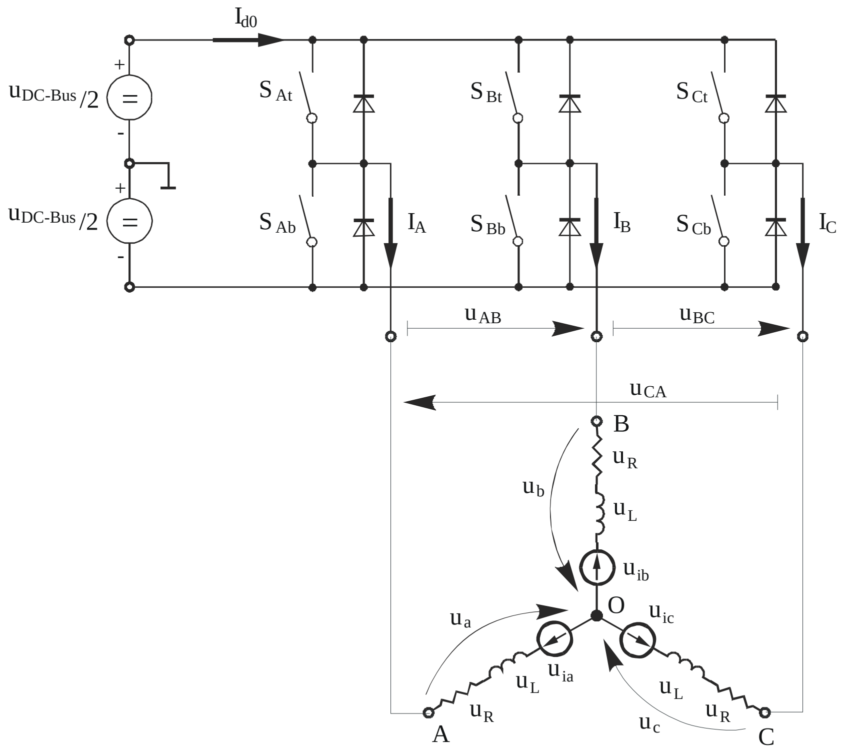


Figure 3-6. Power stage schematic diagram

The combination of ON/OFF states in the power stage switches for each voltage vector is coded by the three-digit number in parentheses. See Figure 3-7. Each digit represents one phase. For each phase, a value of one means that the upper switch is ON and the bottom switch is OFF. A value of zero means that the upper switch is OFF and the bottom switch is ON. These states, together with the resulting instantaneous output line-to-line voltages, phase voltages, and voltage vectors, are listed in Table 3-2.

Table 3-2. Switching patterns and resulting instantaneous

a	b	c	U_a	U_b	U_c	U_{AB}	U_{BC}	U_{CA}	Vector
0	0	0	0	0	0	0	0	0	O_{000}
1	0	0	$2U_{DC\text{-}Bus}/3$	$-U_{DC\text{-}Bus}/3$	$-U_{DC\text{-}Bus}/3$	$U_{DC\text{-}Bus}$	0	$-U_{DC\text{-}Bus}$	U_0
1	1	0	$U_{DC\text{-}Bus}/3$	$U_{DC\text{-}Bus}/3$	$-2U_{DC\text{-}Bus}/3$	0	$U_{DC\text{-}Bus}$	$-U_{DC\text{-}Bus}$	U_{60}
0	1	0	$-U_{DC\text{-}Bus}/3$	$2U_{DC\text{-}Bus}/3$	$-U_{DC\text{-}Bus}/3$	$-U_{DC\text{-}Bus}$	$U_{DC\text{-}Bus}$	0	U_{120}
0	1	1	$-2U_{DC\text{-}Bus}/3$	$U_{DC\text{-}Bus}/3$	$U_{DC\text{-}Bus}/3$	$-U_{DC\text{-}Bus}$	0	$U_{DC\text{-}Bus}$	U_{240}
0	0	1	$-U_{DC\text{-}Bus}/3$	$-U_{DC\text{-}Bus}/3$	$2U_{DC\text{-}Bus}/3$	0	$-U_{DC\text{-}Bus}$	$U_{DC\text{-}Bus}$	U_{300}
1	0	1	$U_{DC\text{-}Bus}/3$	$-2U_{DC\text{-}Bus}/3$	$U_{DC\text{-}Bus}/3$	$U_{DC\text{-}Bus}$	$-U_{DC\text{-}Bus}$	0	U_{360}
1	1	1	0	0	0	0	0	0	O_{111}

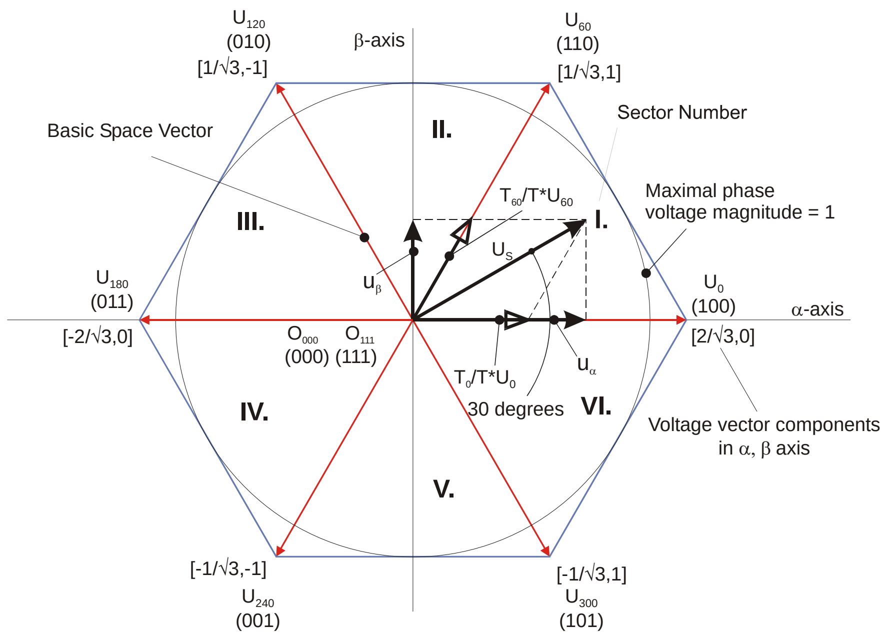


Figure 3-7. Basic space vectors and voltage vector projection

The SVM is a technique used as a direct bridge between the vector control (voltage space vector) and the PWM.

The SVM technique consists of several steps:

1. Sector identification
2. Space voltage vector decomposition into directions of sector base vectors U_x, and $U_{x\pm60}$
3. PWM duty cycle calculation

The principle technique of the SVM is the application of the voltage vectors UXXX and OXXX and in some instances the mean vector of the PWM period t_{PWM} is equal to the desired voltage vector.

This method gives the greatest variability in arranging the zero and non-zero vectors during the PWM period. You can arrange these vectors to lower switching losses, or to reach a different result such as center-aligned PWM, edge-aligned PWM, minimal switching, and so on.

For the chosen SVM, this rule is defined:

- The desired space voltage vector is created only by applying the sector base vectors: the non-zero vectors on the sector side (U_x, $U_{x\pm60}$), and the zero vectors (O_{000} or O_{111}).

The following define the principle technique of the SVM:

$$t_{PWM} \times U_{S[\alpha,\beta]} = t_1 \times U_x + t_2 \times (U_{x\pm60} + t_0 \times (O_{000} \vee O_{111}))$$ **Eqn. 3-60**

$$t_{PWM} = t_1 + t_2 + t_0$$ **Eqn. 3-61**

To solve the time periods t_0, t_1, and t_2, it is necessary to decompose the space voltage vector $U_{S[\alpha,\beta]}$ into the directions of the sector base vectors U_x, $U_{x\pm60}$. The Equation 3-60 splits into equations Equation 3-62 and Equation 3-63.

$$t_{PWM} \times U_{Sx} = t_1 \times U_x$$ **Eqn. 3-62**

$$t_{PWM} \times U_{S(x\pm60)} = t_2 \times U_{x\pm60}$$ **Eqn. 3-63**

By solving this set of equations, we can calculate the necessary duration for the application of the sector base vectors U_x, $U_{x\pm60}$ during the PWM period T_{PWM} to produce the correct stator voltages.

$$t_1 = \frac{|U_{Sx}|}{|U_x|} t_{PWM} \quad \textbf{for vector } \mathbf{U_x}$$ **Eqn. 3-64**

$$t_2 = \frac{|U_{Sx}|}{|U_{x\pm60}|} t_{PWM} \quad \textbf{for vector } \mathbf{U_{x\pm60}}$$ **Eqn. 3-65**

$$t_0 = t_{PWM} - (t_1 + t_2) \quad \textbf{either for } \mathbf{O_{000}} \textbf{ or } \mathbf{O_{111}}$$ **Eqn. 3-66**

3.3 Position Sensorless Elimination

The first stage of the proposed overall control structure is the rotor PM alignment algorithm to set an accurate initial position. This allows applying a full start-up torque to the washing machine's motor. In the second stage, the FOC is in open-loop mode to move the motor up to a speed value where the observer provides sufficiently accurate speed and position estimations. As soon as the observer provides appropriate estimates, the rotor speed and position calculation is based on the estimation of a back-EMF in the stationary reference frame using a Luenberger observer.

3.3.1 Rotor Alignment

In this design, there is a no motor position or a speed sensor used, this is because in the beginning the initial position needs to be determined before the motor is started. One of the possibilities and an easily implemented method is to align the rotor to a predefined position. The motor is powered by a selected static voltage pattern (usually the zero position in the sine wave table) and the rotor aligns to the predefined position. The alignment is executed once, only during the first motor start. Figure 3-8 shows the motor alignment. Before the constant current vector is applied to the stator, the rotor position is not known. After a stabilization period, the rotor flux must be aligned to the stator flux. In practice, this is true when the external load torque is low enough compared to the torque produced by the alignment vector.

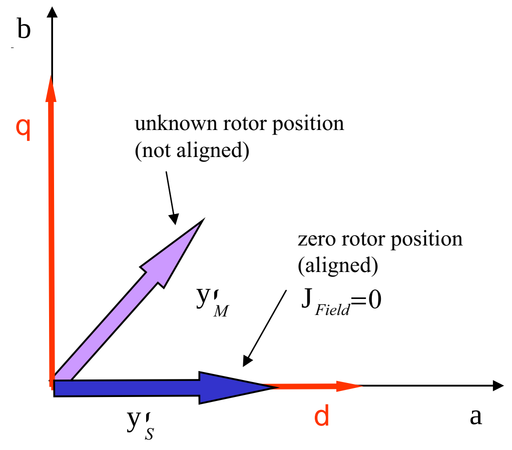

b
q
unknown rotor position (not aligned)
zero rotor position (aligned)
ψ'_M
$\vartheta_{Field}=0$
ψ'_S
d
a

Figure 3-8. Rotor alignment stabilization — PMSM starting mode

3.3.2 Open Loop Startup

After identifying the initial rotor position, the field oriented control is used in an open-loop mode. The current set-point is determined by the speed controller. This generates the torque reference current iQref and the proportional integral controller of the speed control loop is initialized to maximum allowable current. The angular speed feedback FBCK is kept at zero level during the open loop operation. The vector transformations are fed by a time varying reference position signal obtained by integrating the speed ramp reference. This strategy moves the motor up to the speed values where the observer provides sufficiently accurate speed and position estimates.

To correctly switch on the sensorless PMSM control, the open-loop estimates and high speed sensorless algorithm have to be evaluated and merged. Merging an algorithm based on a cross-over function can ensure a smooth transition to close the position and speed feedbacks.

The merge function with the weight coefficient a_M is used to determine the speed and position feedback signals based on a speed ramp command. The weighting function is divided into three subsequent regions named open-loop state, merge state, and closed-loop state, respectively. The weighting coefficient a_M is kept at a zero level during the open loop state. The merge state is determined between the lower and upper speed limits M1 and M2 where weighting coefficient a_M increases up to reaching a closed-loop state. The weighting function is shown in Figure 3-9.

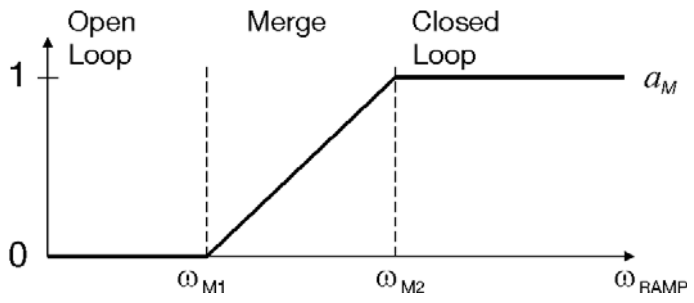


Figure 3-9. Crossover function with a_M weight coefficient

The upper and lower speed limits of the crossover function M1 and M2 are found from the estimation accuracy limits through experiments. Angular speed feedback W_FBCK is determined by the following formula:

$$\omega_{FBCK} = a_M \cdot \omega_{OBSRV} \quad \textit{Eqn. 3-67}$$

Where estimated speed OBSRV is multiplied by weighting coefficient a_M.

The vector transformations are fed by position feedback signal FBCK that is determined by combination of open-loop position signal OL and position calculated by back-EMF observer OBSRV. The resulting feedback position signal FBCK is calculated using crossover function (Equation 3-68) with weighting coefficient a_M:

$$\theta_{FBCK} = \theta_{OL} \cdot a_M + (1 - a_M) \cdot \theta_{OBSRV} \quad \textit{Eqn. 3-68}$$

Where weighting coefficient a_M is multiplied by both the estimated position OBSRV and open-loop position OL signals. The products are then added to form the feedback position signal FBCK.

The open-loop position OL is a time varying reference signal derived by integrating the speed ramp reference. The reference speed command ramp is carefully chosen to ensure a safe starting with minimum oscillation up to the maximum torque. The speed reference ramp is chosen such that the start-up torque is greater than the one required by the washing machine. Therefore, the actual rotor position advances the 'forced' open-loop position START.

By initializing the integrator part of the speed controller to the maximum allowable current value and simultaneously knowing the correct rotor position from the beginning allows delivering the full start-up torque. This accelerates the washing machine at the rate equal to the angular speed output of the back-EMF observer. After the upper speed limit M2 is attained, the back-EMF observer is switched on-line and the closed-loop speed control is achieved.

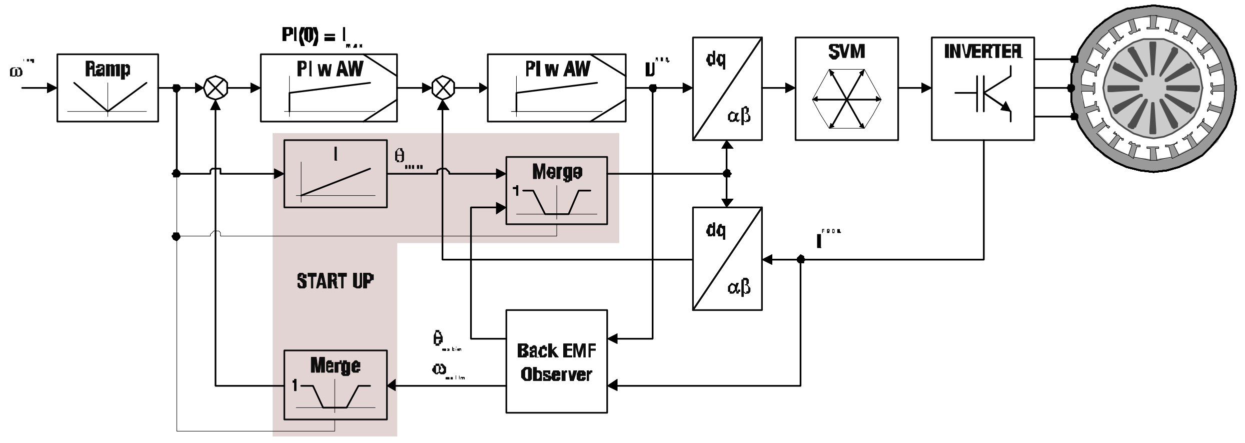


Figure 10. Sensorless speed control with open loop startup and back-EMF observer

3.3.3 Back-EMF Observer

When the minimum operating speed is reached a measurable level of back-EMF is generated by the rotor permanent magnets. The back-EMF observer then gradually transitions into the closed-loop mode. The feedback loops are then controlled by the estimated angle and estimated speed signals from the back-EMF observer.

This estimation method for the position and angular speed is based on the motor mathematical model with an extended electro-motive force function [4][5]. This extended back-EMF model includes both position information from the conventionally defined back-EMF and the stator inductance. This allows to extract the rotor position and velocity information by estimating only the extended back-EMF.

$$\begin{bmatrix} u_\alpha \\ u_\beta \end{bmatrix} = R\begin{bmatrix} i_\alpha \\ i_\beta \end{bmatrix} + \begin{bmatrix} pL_d & (L_d - L_q)\omega_e \\ -(L_d - L_q)\omega_e & pL_d \end{bmatrix}\begin{bmatrix} i_\alpha \\ i_\beta \end{bmatrix} + \{(L_d - L_q)(\omega_e i_d - i'_q) + k_e\omega_e\}\begin{bmatrix} -\sin(\theta_e) \\ \cos(\theta_e) \end{bmatrix} \qquad \textit{Eqn. 3-69}$$

Where:

- R—Stator resistance
- Ld, Lq—d-axis and q-axis inductance
- k_e—Back-EMF constant
- ω_e—Angular electrical speed
- uα, uβ—Stator voltages
- iα, iβ—Stator currents
- p—Operator of derivative
- i'_q—First derivative of the iq current

The observer is applied to the PMSM motor with an estimator model excluding the extended back-EMF term. Then the extended back-EMF term can be estimated using the observer as depicted in Figure 3-2 that uses a simple observer of the PMSM motor stator current. Presented here the back-EMF observer is realized within a stationary reference frame, see Figure 3-11. The estimator of the α,β-axis consists of the stator current observer based on the RL motor circuit with estimated motor parameters. This current observer is fed by the sum of the actual applied motor voltage, cross-coupled rotational term that corresponds to the motor saliency (Ld-Lq), and the compensator corrective output. The observer provides back-EMF signals as disturbance because the back-EMF is not included in observer model.

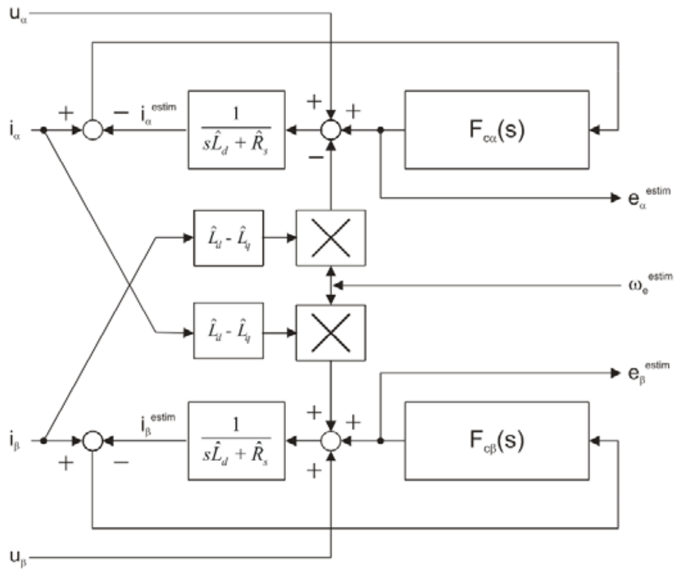


Figure 3-11. Diagram of extended back-EMF and Angle Tracking Observer

3.3.4 Speed and Position Extraction from the Estimated EEMF

Several extraction methods for attaining the electrical rotor position and its rotor speed based on an extended EMF rotating vector have been proposed. These known approaches use either a direct calculation of the arctangent trigonometric function with two input variables or the use of a tracking observer.

The rotor electrical position, meaning rotor shaft angle, can be determined by an inverse tangent function with two inputs of the extended EMF estimates. This determination can be expressed as follows:

$$\hat{\theta}_e = \operatorname{atan}\left(-\frac{\hat{E}_\alpha}{\hat{E}_\beta}\right)$$

Eqn. 3-70

The method with the arctangent trigonometric function (atan) shown in Figure 3-12 yields values of the unfiltered rotor angle without any speed information. Therefore, the targeted application often requires a speed calculation with a need to add some smoothing capability through an extra low pass filter. Moreover, if the noise is presented in measured back-EMF signals, then it results in a noisy position estimate θ_{estim} when using direct calculation methods such as the arctangent function. Minimization of the noise problem is highly desired and further filtering needs to be applied either to the estimated position or to the input signals, but an applied filter might add a lag property to the rotor position estimate θ_{estim}.

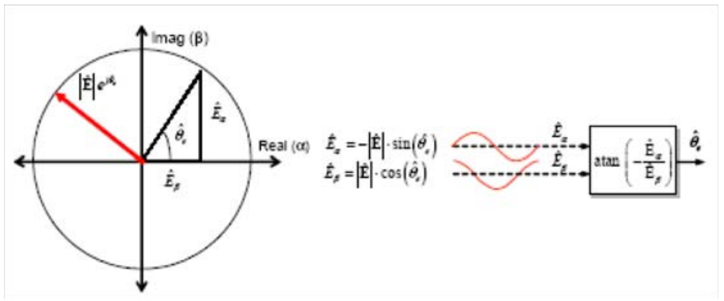


Figure 3-12. Extraction rotor position by inverse tangent function

This inherited drawback of arctangent function can be avoided by proposing the angle tracking observer. The angle tracking observer shown in Figure 3-13 is widely used for the estimation of the rotor angle and speed, and is generally known as an angle tracking observer (ATO). By employing the tracking observer, noise on the position estimate can be filtered out without adding lag to the estimate within its bandwidth.

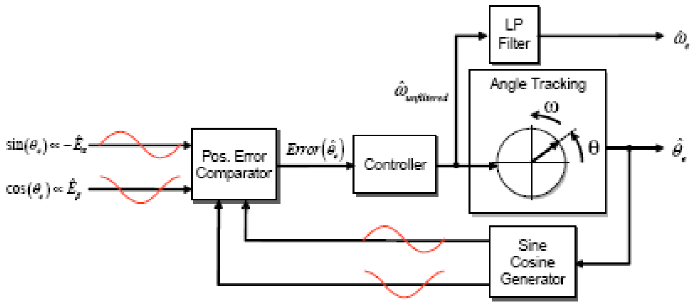


Figure 3-13. Basic principle of angle tracking observer

The angle tracking observer compares input values of the estimated back-EEMF signals (Eα and Eβ), with their corresponding estimated outputs sin(θ_{estim}) and cos(θ_{estim}). As in any common closed-loop system, the intention is to minimize the observer error as shown in Figure 3-14.

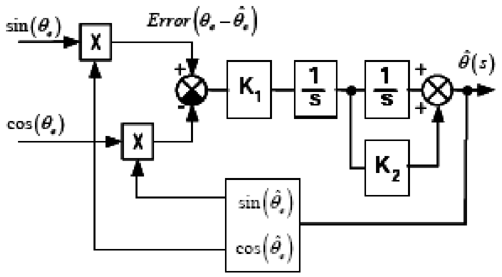


Figure 3-14. Basic implementation of angle tracking observer

NOTE

The mathematical expression of the observer error is well known as the formula of the difference of two angles.

The primary benefit of the angle tracking observer, in comparison with the trigonometric method, is its smoothing capability. This filtering is achieved by the integrator and PI controller that are connected in series and closed by a unit feedback loop as shown in block diagram Figure 3-15. This block diagram tracks actual rotor angle and speed and continuously updates their estimations.

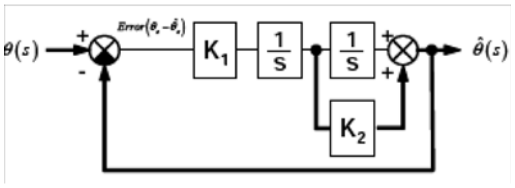


Figure 3-15. Simplified block diagram of angle tracking observer

Chapter 4 System Concept

4.1 System Specifications

The system can drive a three-phase PM synchronous motor without a mechanical position sensor.

The application meets these performance specifications:

- Targeted at the MC56F8013/23/25 digital signal controller
- Running on the MC56F8013/23/25 controller board and three-phase high voltage power stage
- Control technique incorporating:
  - Vector control of a three-phase PM synchronous motor
  - Position and speed detection using a dq back-EMF observer and tracking observer
  - Closed-loop speed control
  - Speed range extended using field weakening
  - Unidirectional rotation
  - Closed-loop current control
  - Independent control of flux and torque
  - Startup with alignment
  - Three-phase motor currents measured in each phase leg
  - 100 μs sampling period on the MC56F8023
- FreeMASTER software control interface (motor start/stop, speed setup)
- FreeMASTER software monitor
  - FreeMASTER software graphical control page (required speed, actual motor speed, start/stop status, DC-bus voltage level, motor current, system status)
  - FreeMASTER software speed scope (observes actual and desired speeds, DC-bus voltage, and motor current)
  - FreeMASTER software high-speed recorder (reconstructed motor currents, vector control algorithm quantities)
- DC-bus over-voltage and under-voltage, over-current protection

4.2 Application Description

A standard system concept is chosen for the drive (see Figure 4-1). The system incorporates these hardware boards:

- Power supply 90 V–260 V AC RMS, 5 A
- Three-phase, high-voltage power stage
- Three-phase PM synchronous motor (default configuration for TG Drives 3-phase PMSM motor)
- MC56F8013/23/25 controller board

The MC56F8023 populated on the controller board executes the control algorithm. In response to the user interface and feedback signals, it generates PWM signals for the three-phase high-voltage power stage. High-voltage waveforms generated by the DC to AC inverter are applied to the motor.

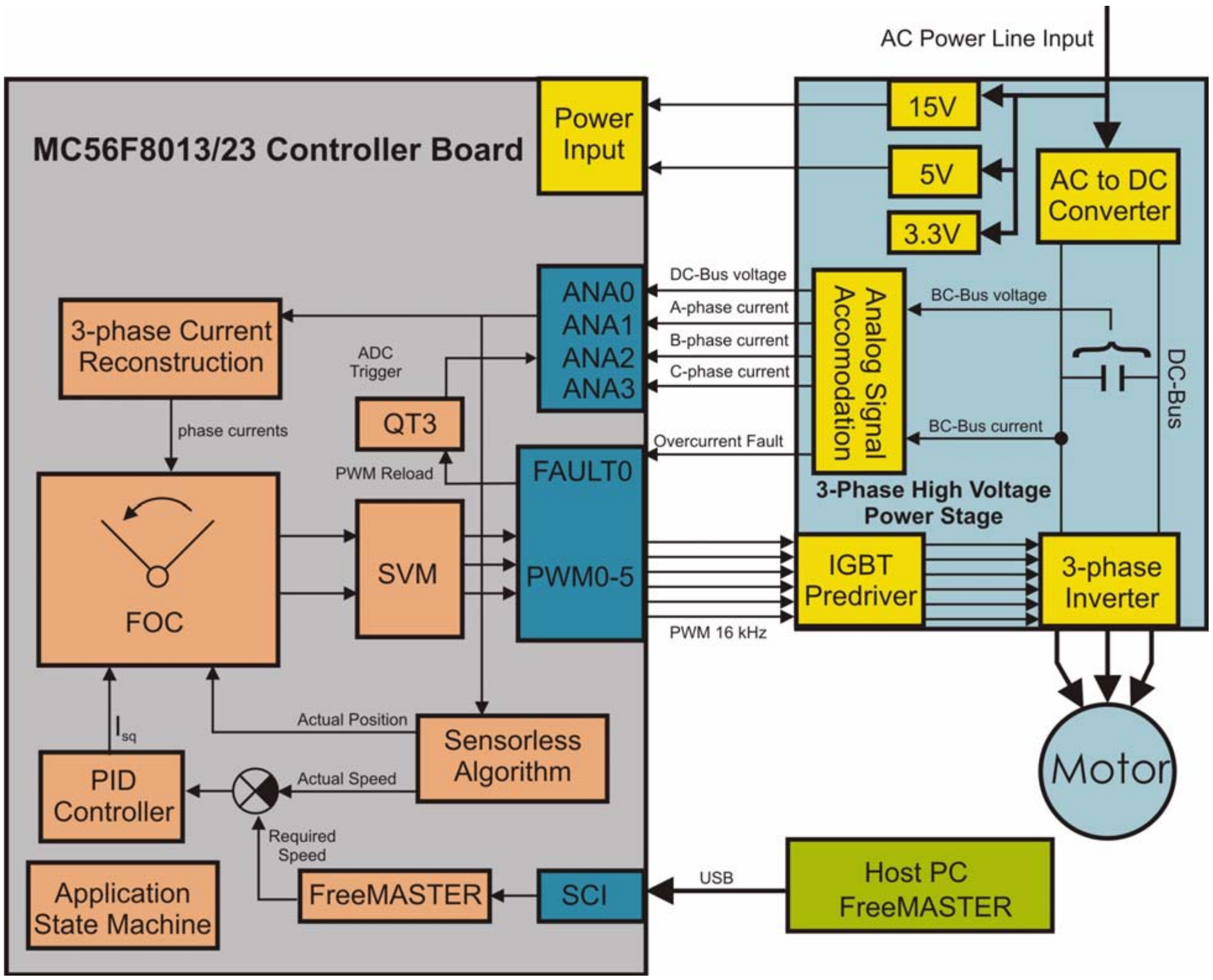


Figure 4-1. System concept

4.3 Control Process

The state of the user interface is scanned periodically while the actual speed of the motor, DC-bus voltage, and the phase currents are sampled. The speed command is calculated according to the state of the control signals (start/stop, required speed from FreeMASTER). The speed command is then processed by means of the speed ramp algorithm. The comparison between the actual speed command obtained from the ramp algorithm output and the measured speed generates a speed error. The speed error is input to the speed PI controller generating a new desired level of reference for the torque-producing component of the stator current.

The DC-bus current and voltage are sampled with ADC. The ADC sampling is triggered by QuadTimer channel 3 and synchronized to the PWM signal. A digital filter is applied to the sampled values. The three-phase motor current is reconstructed from samples taken from the DC-link shunt resistor. The reconstructed three-phase current is then transformed into space vectors and used by the FOC algorithm.

The rotor position and speed are estimated by the back-EMF and angle tracking observers. Based on the estimated signals, the FOC algorithm performs a vector control technique directed towards the rotor-magnetizing flux space vector, as described in 3.2, “Vector Control of PM Synchronous Motor.” Two independent current PI control loops are executed to achieve the desired behavior of the motor. Output from the FOC is a stator voltage space vector that is transformed by means of space vector modulation into PWM signals. The three-phase stator voltage is generated by means of a three-phase voltage source inverter and applied to the motor that is connected to the power stage terminals.

The application can be controlled via a FreeMASTER control page from a host PC. The FreeMASTER communicates via USB. The USB is opto-isolated to achieve safety isolation from high voltage.

The drive’s application state machine manages the operating states of the drive. There are five states of the drive. The actual operating state is indicated by the FreeMASTER control page. In the case of over-voltage, under-voltage, or over-current, the signals for the three-phase inverter are disabled and the fault state is displayed.

Chapter 5 Hardware

5.1 Hardware Implementation

This application can drive a three-phase PM synchronous motor. It consists of these modules:

- Host PC
- MC56F8013/23/25 controller board
- Three-phase AC/BLDC high voltage power stage
- Three-phase PM synchronous motor

The application hardware system configuration is shown in Figure 5-1.

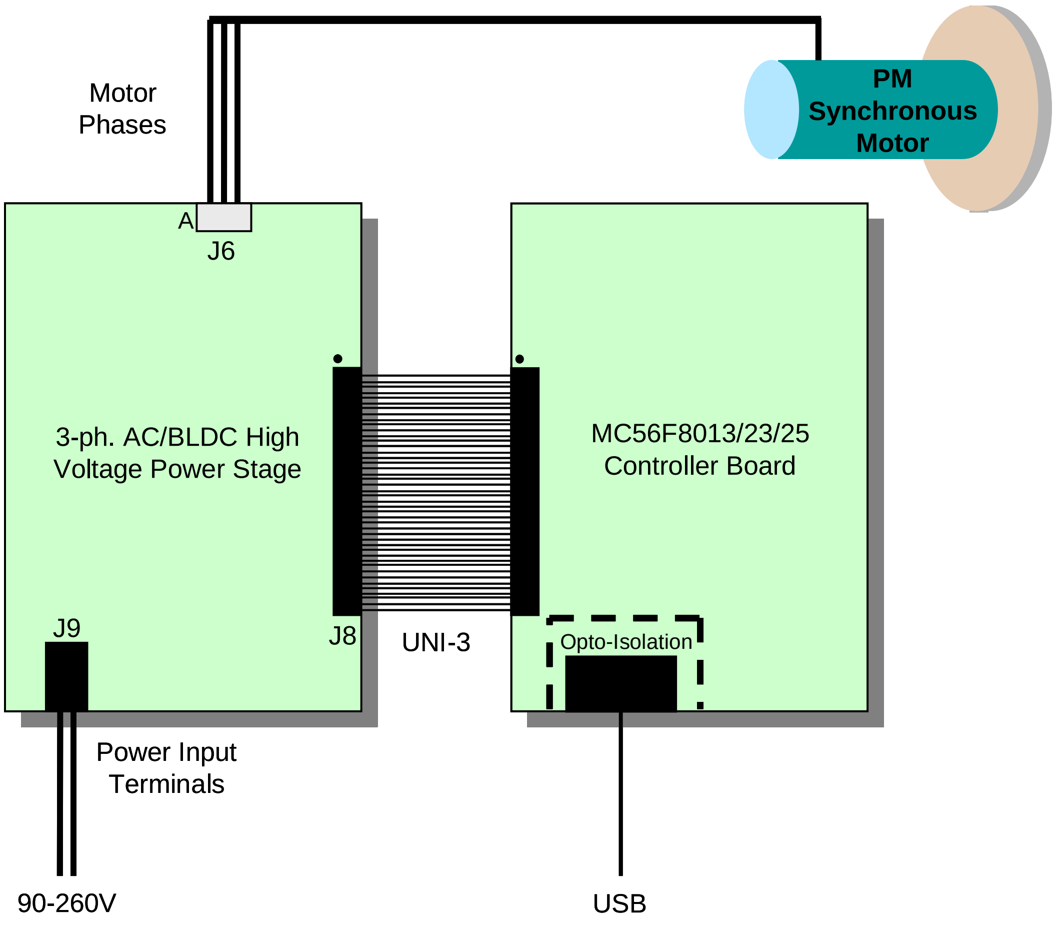


Figure 5-1. Hardware system configuration

All system parts are supplied and documented:

- The MC56F8013/23/25 controller board:
  - Uses Freescale's MC56F8023 as the controller
  - Described in the *MC56F8013/23/25 Controller Board Hardware User's Manual*
- Three-phase AC/BLDC high-voltage power stage:
  - High-voltage, three-phase power stage with single-phase input 115/230 V AC, and 750 VA variable voltage three-phase IGBT bridge output
  - Described in the *3-Phase AC/BLDC High Voltage Power Stage Board Users Guide*

A detailed description of each individual board can be found in the appropriate user manual, or on the Freescale web site (www.freescale.com). The user manuals include a schematic of the board, a description of individual function blocks, and a bill of materials (parts list).

5.2 MC56F8013/23/25 Controller Board

The MC56F8013/23/25 controller board is based on an optimized PCB and power supply design. It demonstrates the abilities of the MC56F8013/23/25 and provides a hardware tool to help in the development of applications that use the MC56F8013/23/25 and are targeted at motor control applications.

The MC56F8013/23/25 controller board can be populated with either the MC56F8013, MC56F8023 or MC56F8025 parts.

The controller board is an evaluation module type of board; it includes an MC56F8013, MC56F8023 or MC56F8025 part, encoder interface, tacho-generator interface, communication options, digital and analog power supplies, and peripheral expansion connectors. The expansion connectors are for signal monitoring and allow expandability of user features. Test pads are provided for monitoring critical signals and voltage levels.

The MC56F8013/23/25 controller board is designed for these purposes:

- New users can become familiar with the MC56F801x/802x architecture features .
- The board can serve as a platform for real-time software development.
  The tool suite allows you to develop and simulate routines, download the software to an on-chip memory, run the software, and debug it using a debugger via the JTAG/OnCE port. The breakpoint features of the OnCE port let you easily specify complex break conditions and execute your software at full speed until the break conditions are satisfied. The ability to examine and modify all user-accessible registers, memory, and peripherals through the OnCE port considerably simplifies the task of the developer.
- To serve as a platform for hardware development.
  The hardware platform enables external hardware modules to be connected. The OnCE port's unobtrusive design means all of the memory on the digital signal controller chip is available to the user.

The board facilitates an evaluation of various features present in the MC56F8013/8023/25, and can be used to develop real-time software and hardware products based on either device. It provides the features necessary to write and debug software, demonstrate the functionality of that software, and to interface with the customer's application specific device(s).

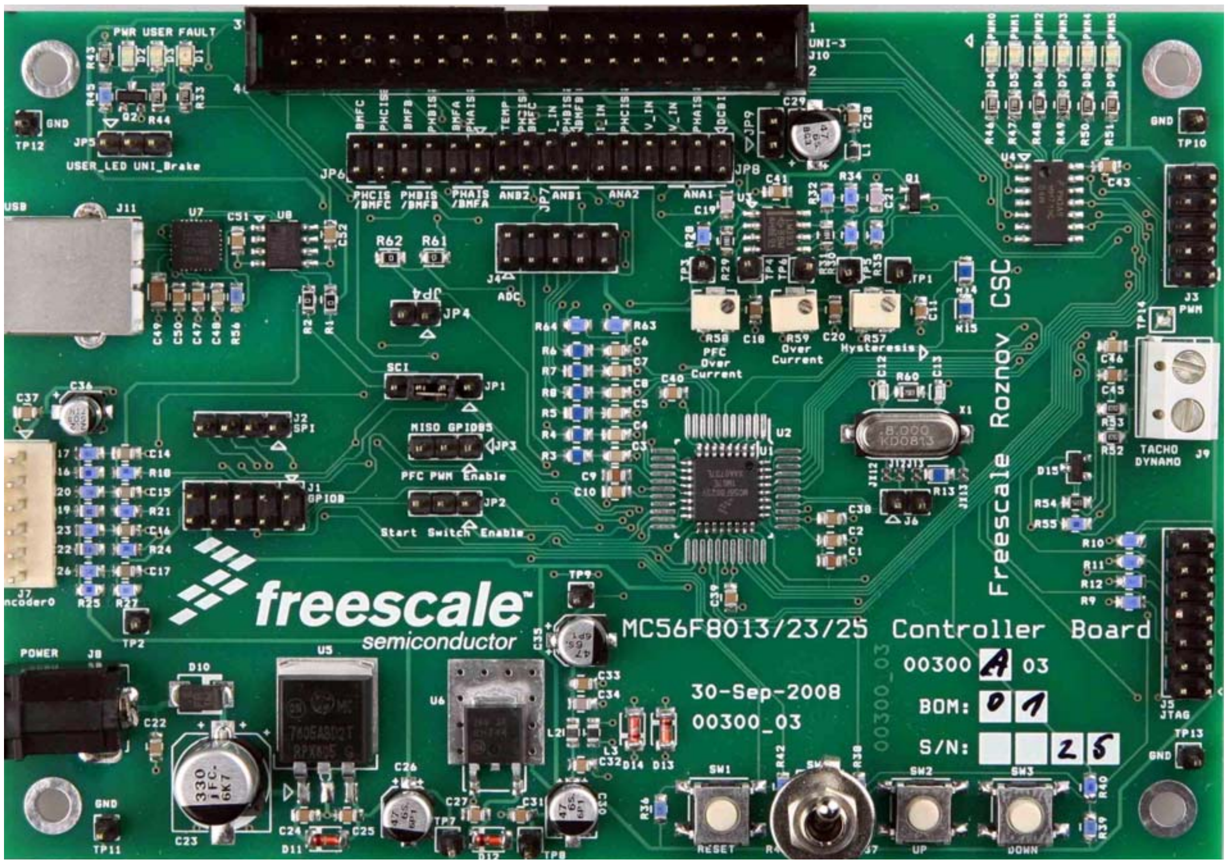


Figure 5-2. MC56F8013/23 controller board top view

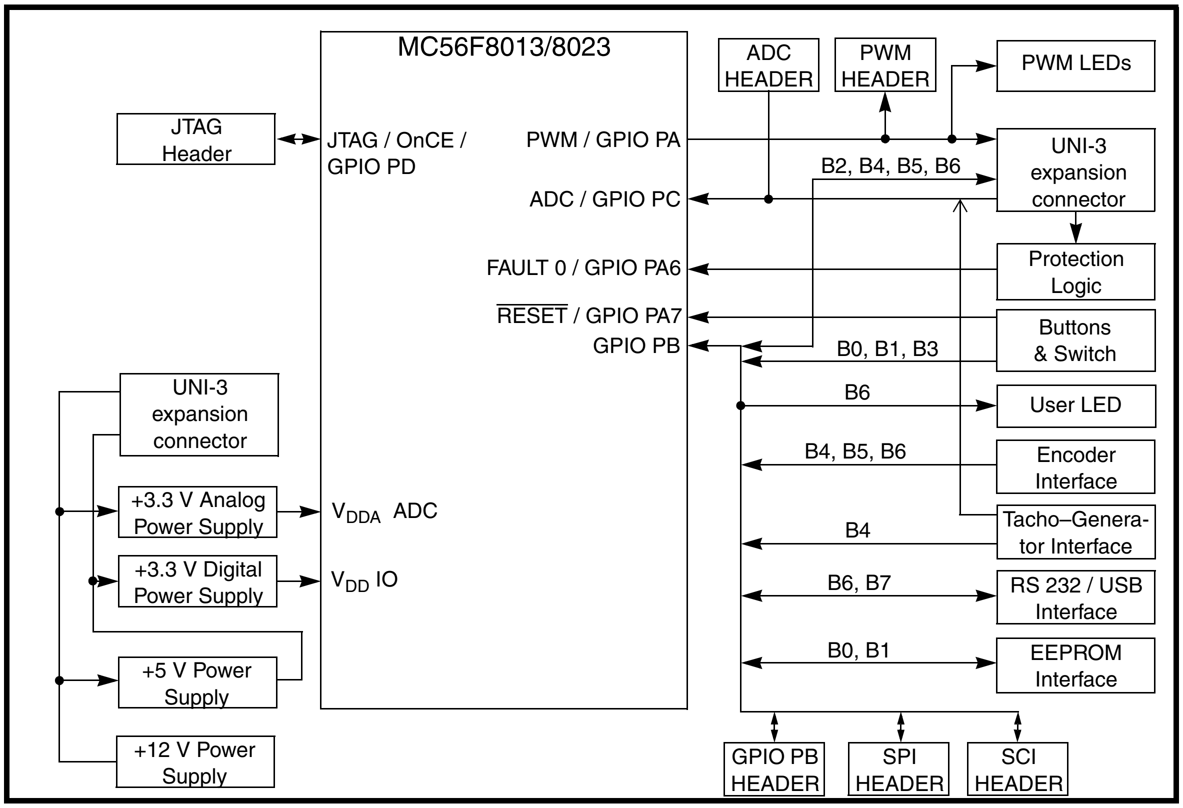


Figure 5-3. Block diagram of the MC56F8013/23/25 controller board

The MC56F8013/23/25 controller board is flexible enough to allow full exploitation of the MC56F8013/8023's features to optimize the performance of the user's end product. See Figure 5-2 and Figure 5-3.

5.3 Three-Phase AC/BLDC High-Voltage Power Stage

Freescale's three-phase high-voltage (HV) AC power stage is a 750-voltamps (one horsepower), the three-phase power stage that operates off DC input voltages from 140 to 325 volts, and AC line voltages from 100 to 240 V. In combination with one of the controller boards, it provides a software development platform that allows algorithms to be written and tested without the need to design and build a power stage. It supports a wide variety of algorithms for both AC induction and brushless DC (BLDC) motors.

The high-voltage AC power stage has a printed circuit board. The printed circuit board contains an input rectifier, brake IGBT and diode, bridge IGBTs, IGBT gate drive circuits, analog signal conditioning, low-voltage power supplies, and some large, passive, power components. All of the power devices that need to dissipate heat and a temperature sensor are mounted on a heatsink underneath the printed circuit board (see Figure 5-4).

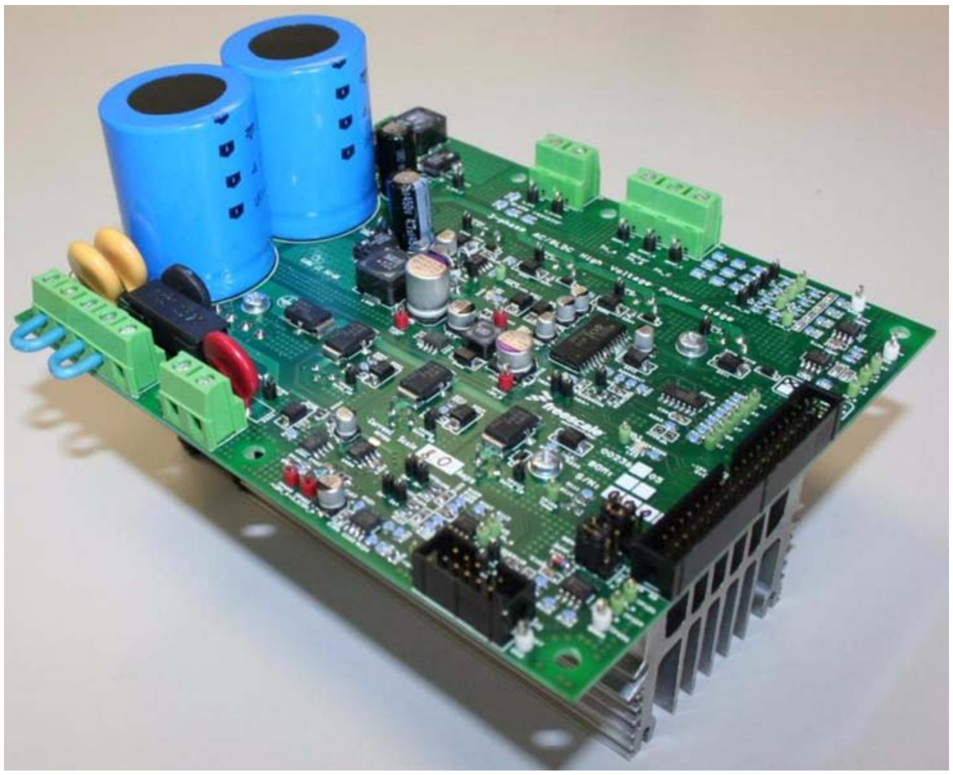


Figure 5-4. Three-phase AC/BLDC high voltage power stage

Figure 5-5 shows a block diagram. Input connections are made via the 40-pin ribbon cable connector J8. Power connections to the motor are made on output connector J6. Phase A, phase B, and phase C are labelled Ph_A, Ph_B, and Ph_C on the board. Power requirements are met by a single external 140 to 325 volt DC power supply or an AC line voltage. Either input is supplied through connector J9. An external brake resistor can be connected via connector J4. The power stage can be extended by an external PFC board. The PFC board connection is made via power connector J5 and signal connector J2.

Current measuring circuitry can be set-up from 2 to 8 amps full scale modifying resistor values. Both bus and phase leg currents are measured. An over-current trip point is set at 10 amps.

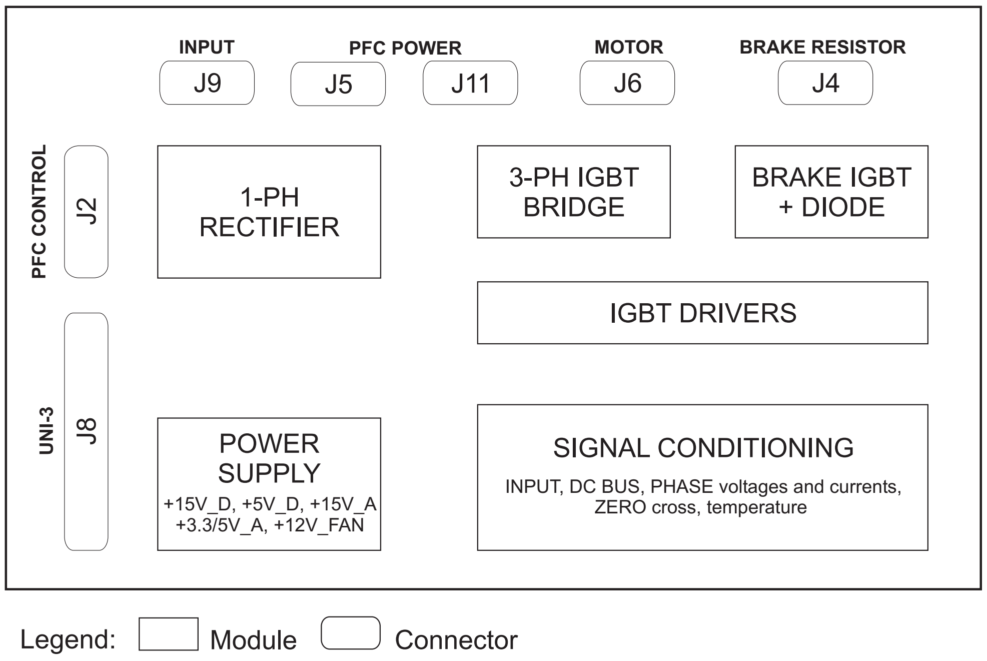


Figure 5-5. Three-phase AC/BLDC power stage block diagram

Table 5-1. Electrical characteristics of three-phase AC/BLDC power stage

Characteristic	Symbol	Min	Typ	Max	Units
DC input voltage	V_{dc}	140	—	325	V
AC input voltage	V_{ac}	100	—	240	V
Logic 1 input voltage	V_{IH}	1.5	—	1.7	V
Logic 0 input voltage	V_{IL}	0.9	—	1	V
Input resistance	R_{In}	—	10	—	kΩ
Analog output range*	V_{Out}	0	—	3.3	V
Bus current sense voltage	I_{Sense}	—	206.25	—	mV/A
Bus current sense offset	I_{Offset}	—	$+V_{REF}$	—	V
Bus voltage sense voltage	V_{Bus}	—	8.09	—	mV/V
Bus voltage sense offset	V_{Offset}	—	0	—	V
Continuous output current**	I_C	—	—	10	A
Deadtime (built in IR2133)	t_{Off}	—	250	—	ns

* Range set according +3.3 VA / +5 VA power supply

** The values are measured at 25°C; for other temperatures the values may be different.

5.4 Motor Specifications — Example

The motor used in this application is a standard production three-phase PM synchronous motor with an incremental encoder mounted on the shaft. The motor is start-connected. The motor and sensor have these specifications:

Table 5-2. Specifications of the motor and incremental sensor

	Motor Type	**Three-Phase PM Synchronous Motor TG drives TGT2-0040-45-320/T0PS1K-1M**
Motor Specification	Nominal Voltage Un	200 V RMS
	Nominal Speed nn	4500 RPM
	Nominal Current In	0.88 A RMS
	Nominal Torque	0.36 Nm
Motor Model Parameters	Stator Winding Resistance	12.7 Ohm
	Stator Winding Inductance d axis	11.1 mH
	Stator Winding Inductance q axis	12.5 mH
	Number of Pole-Pairs	3

Chapter 6
Software Design

6.1 Introduction

This section describes the software design of the PM synchronous motor vector control application. First, the numerical scaling in fixed-point fractional arithmetic of the DSC is discussed. Then, particular issues such as current control, speed control, position estimation, voltage, and current sensing are explained. Finally, the control software implementation is described. The aim of this chapter is to help to understand the included software.

6.2 Scaling Application Variables

6.2.1 Fractional Numbers Representation

The PM synchronous motor vector control application uses a fractional representation for all real quantities except time. The N-bit signed fractional format is represented using 1.[N–1] format (1 sign bit, N–1 fractional bits). Signed fractional numbers (SF) lie in the following range:

$$-1.0 \le SF \le +1.0 - 2^{-[N-1]} \qquad \textit{Eqn. 6-1}$$

For words and long-word signed fractions, the greatest negative number that can be represented is –1.0, whose internal representation is \$8000 and \$80000000, respectively. The greatest positive word is \$7FFF or $1.0 - 2^{-15}$, and the greatest positive long word is \$7FFFFFFF or $1.0 - 2^{-31}$.

6.2.2 Scaling Analog Quantities

Analog quantities such as voltage, current, and frequency are scaled to the maximum measurable range that is dependent on the hardware. The following equation shows the relationship between a real and a fractional representation:

$$\text{Fractional Value} = \frac{\text{Real Value}}{\text{Real Quantity Range}} \qquad \textit{Eqn. 6-2}$$

Where:

- Fractional Value—Is a fractional representation of the real value [Frac16]
- Real Value —Is the real value of the quantity [V, A, rpm, and so on.]
- Real Quantity Range —Is the maximum range of the quantity, defined in the application [V, A, rpm, and so on.]

As an example, the above scaling can be demonstrated on a DC-bus voltage and motor phase voltage. All variables representing voltage are scaled the same in the application. They are scaled to the maximum measurable voltage range of the power stage. For the demo hardware board, the range is V_{max} = 407 V. Variable values in fractional format are defined by the following equation:

$$(\text{Frac16})\text{voltage\_variable} = \frac{V_{\text{Measured}}}{V_{\text{max}}} \qquad \textbf{Eqn. 6-3}$$

The fractional variables are internally stored as signed 16-bit integer values, and their values can be evaluated as follows:

$$(\text{Int16})\text{voltage\_variable} = (\text{Frac16})(\text{voltage\_variable} \times 2^{15}) \qquad \textbf{Eqn. 6-4}$$

The maximum range of analog quantities used by the application is defined by #define statements in the application configuration files. The default scaling ranges for the reference design hardware setup are as follows:

```
#define XLS_PMSM_U_MAX              235                        /* Volts */
#define XLS_PMSM_I_MAX              2*4.0                      /* Amps */
```

Notice that the XLS_PMSM_I_MAX corresponds to a full range of the ADC converter input voltage (0 – 3.3 V). For motor phase-current sensing, the zero current level is shifted into the middle of this range (= 1.65 V). Thus, the maximum positive and negative phase current that can be sensed is Max_Current = XLS_PMSM_I_MAX/2. In other words, if the current sensing range of the power stage is from – 4.0 A to + 4.0 A, the value of XLS_PMSM_I_MAX is set to the value 8.0 A. And if the current sensing range of the power stage is from –8.0 A to + 8.0 A, the value of XLS_PMSM_I_MAX is set to the value 16.0 A.

6.2.3 Scaling Angles

The angles, such as rotor flux position are represented as 16-bit signed fractional values in the range [–1,1) that corresponds to the angle in the range [–pi, pi). In a 16-bit signed integer value the angle is represented as follows:

$$-\text{pi} \approx 0\text{x}8000 \qquad \textbf{Eqn. 6-5}$$

$$\text{pi} \times (1.0 - 2^{-15}) \approx 0\text{x}7\text{FFF} \qquad \textbf{Eqn. 6-6}$$

6.2.4 Scaling Parameters

Real-value parameters in equations such as the SM observer, decoupling voltage, and so on, are represented as 16-bit signed fractional values in the range [–1,1). The real parameter value (ohms, henries) has to be adjusted to correspond to the scaling range of the analog value that forms a particular equation. The adjusted value is then split into a fractional range of [–1,1) and an N-bit scale. The scaling process can be explained by a simple example of the Ohm’s law equation.

$$V_{\text{Real}} = R \times I_{\text{Real}} \qquad \textbf{Eqn. 6-7}$$

$$V_{\text{Frac16}} \times \text{V\_MAX} = R \times I_{\text{Frac16}} \times \text{I\_MAX} \qquad \textbf{Eqn. 6-8}$$

$$V_{Frac16} = \left(R \times \frac{I\_MAX}{V\_MAX}\right) \times I_{Frac16} = R_{Adjusted} \times I_{Frac16} \qquad \textbf{Eqn. 6-9}$$

Substitute the following values:

R = 300 Ohms, I_MAX = 8 Amps, V_MAX = 407 Volts

The $R_{Adjusted}$ can be evaluated:

$$R_{Adjusted} = R \times \frac{I\_MAX}{V\_MAX} = 300 \times \frac{8}{407} = 5.8968 \qquad \textbf{Eqn. 6-10}$$

The $R_{Adjusted}$ is out of range of the signed fractional number. Right shifting the value by R_{Scale}-bits to fit into the desired range is needed. Therefore, a scale (shift) part of the parameter is introduced. For this example shift the result of R_{Scale} = 3 bits. The resistor value scaled to the signed fractional range is as follows:

$$R_{Frac16} \times 2^{R_{Scale}} = R_{Adjusted} \qquad \textbf{Eqn. 6-11}$$

$$R_{Frac16} = R \times \frac{I\_MAX}{V\_MAX} \times 2^{-R_{Scale}} = 300 \times \frac{8}{407} \times 2^{-3} = 0.7371 \qquad \textbf{Eqn. 6-12}$$

The Ohm’s law equation scaled into signed fractional arithmetic is evaluated as follows:

$$V_{Frac16} = \left(\left(R \times \frac{I\_MAX}{V\_MAX} \times 2^{-R_{Scale}}\right)(I_{Frac16})\right) \times 2^{R_{Scale}} \qquad \textbf{Eqn. 6-13}$$

NOTE

The final multiplication result has to be left-shifted back by R_{Scale} bits to stay within the proper range of the V_{Frac16} variable.

All algorithm and motor parameters are scaled to their 16-bit or possibly 32-bit fractional representation. For most parameters, there are two definitions. One evaluates the parameter fractional representation, and the other defines the required N-bit shift = scale. Here is the encoder position scale parameter as an example:

```
#define ENC_POSITION_SCALE_SHIFT    -6
#define ENC_POSITION_SCALE          FRAC16(0.75)
```

The N-bit shift constant is defined as positive, while a left shift is applied to the scaled variable. Then the real value of the speed controller gain is:

$$cPositionScaleReal = \frac{EncPositioScale}{2^{EncPositionScaleShift}} = \frac{0.75}{2^{-6}} = 48 \qquad \textbf{Eqn. 6-14}$$

To evaluate the fractional and shift constants of the parameters, use the Excel spreadsheet included in the software files.

Also introduced are several scaling factors for some of the complicated constant calculations (in the Excel spreadsheet included with the software). Their purpose is to transform integer constants into the defined fractional range [–1, 1) and vice versa. Some of the equations can be made universal with these scaling factors, and can potentially be reused for system quantities that are not represented by the fractional range [–1,1).

The voltage scaling factor is:

$$S_V = \frac{V_{sysrange}}{V_{max}} \qquad \text{Eqn. 6-15}$$

Where:

- S_V is the scaling factor
- $V_{sysrange}$ is the range of system representation voltage
- V_{max} is the range of real voltage

In this application, the system representation is a fractional number of the range, as in Equation 6-1. Therefore, the scaling coefficient is usually:

$$S_V = \frac{1}{V_{max}} \qquad \text{Eqn. 6-16}$$

Then the real variable is:

$$V_{Real} = \frac{(Frac16)Voltage\_Variable}{S_V} \qquad \text{Eqn. 6-17}$$

The system parameters are then calculated accordingly:

$$R_{Adjusted} = R \times \frac{S_V}{S_I} = 300 \times \frac{1/407}{1/8} = 5.8968 \qquad \text{Eqn. 6-18}$$

6.3 Software Flowchart

The main software flow chart (see Figure 6-1) incorporates the main routine entered from the reset state. The main routine includes the digital signal controller initialization, FreeMASTER, and super state machine. The digital signal controller enters the reset processing state when a hardware reset signal is asserted. Upon exiting the reset state, the core enters the normal processing state and starts to execute the internal code. The power on reset (POR) procedure is followed by the standard microprocessor start-up sequence. It is necessary to set-up the occupied peripherals to their correct application initial values.

Application control is a software mechanism incorporating a state machine that governs the overall application operation, and upon available data, runs appropriate software states. In run-time conditions, the hardware status is monitored, such as temperature, IC devices status, and so on.

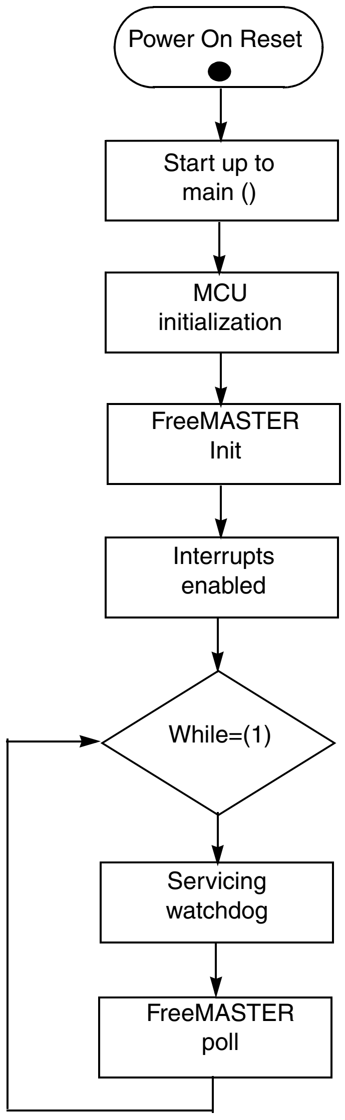


Figure 6-1. Main software flowchart

The application software is interrupt-driven when running in real time. There is a single periodic interrupt service routine executing the major motor control tasks Figure 6-2. The background loop is executed in the application main(). It manages non-critical timing tasks, such as FreeMASTER communication polling.

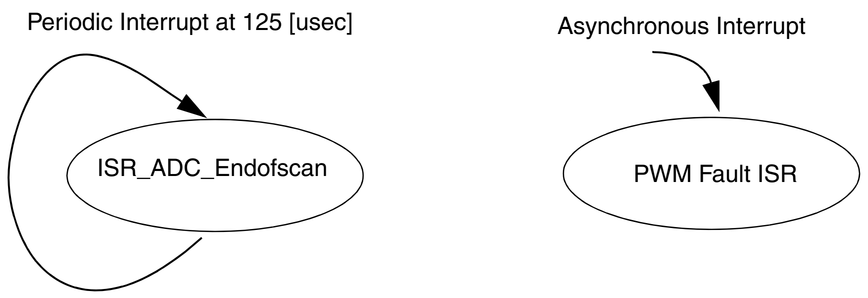


Figure 6-2. Main software interrupts

The MC56F8025 microcontroller provides programmable fault protection where a fault protection can disable any combination of PWM pins. These faults are generated by logic one on any of the fault pins. When the fault protection hardware disables the PWM pins, the PWM generator continues to run, and only the output pins are deactivated. If a fault is latched in, it must be cleared. The fault state is invoked prior to enabling the PWM, to prevent an unexpected interrupt.

6.4 Application Software States

Application control is a software mechanism incorporating a state machine that governs the overall application operation and upon available data, runs appropriate software states. In run-time conditions, the hardware status is monitored, such as temperature, IC devices status, and so on.

The flowchart diagram of the main interrupt in (Figure 6-3) is created to show the main functional elements that occur within developed software and to the information that flows among these functional elements. This includes the information that is either is coming from or going to the environment entities. This diagram shows the big picture of the data processing functions within the developed software.

The benefits of the structured modular software design are well perceived. This is especially true for complex motor control systems with many interacting software sub-blocks. Due to these reasons the Freescale library set for 56800E has been developed including GFLIB, GDFLIB, MCLIB, and ACLIB.

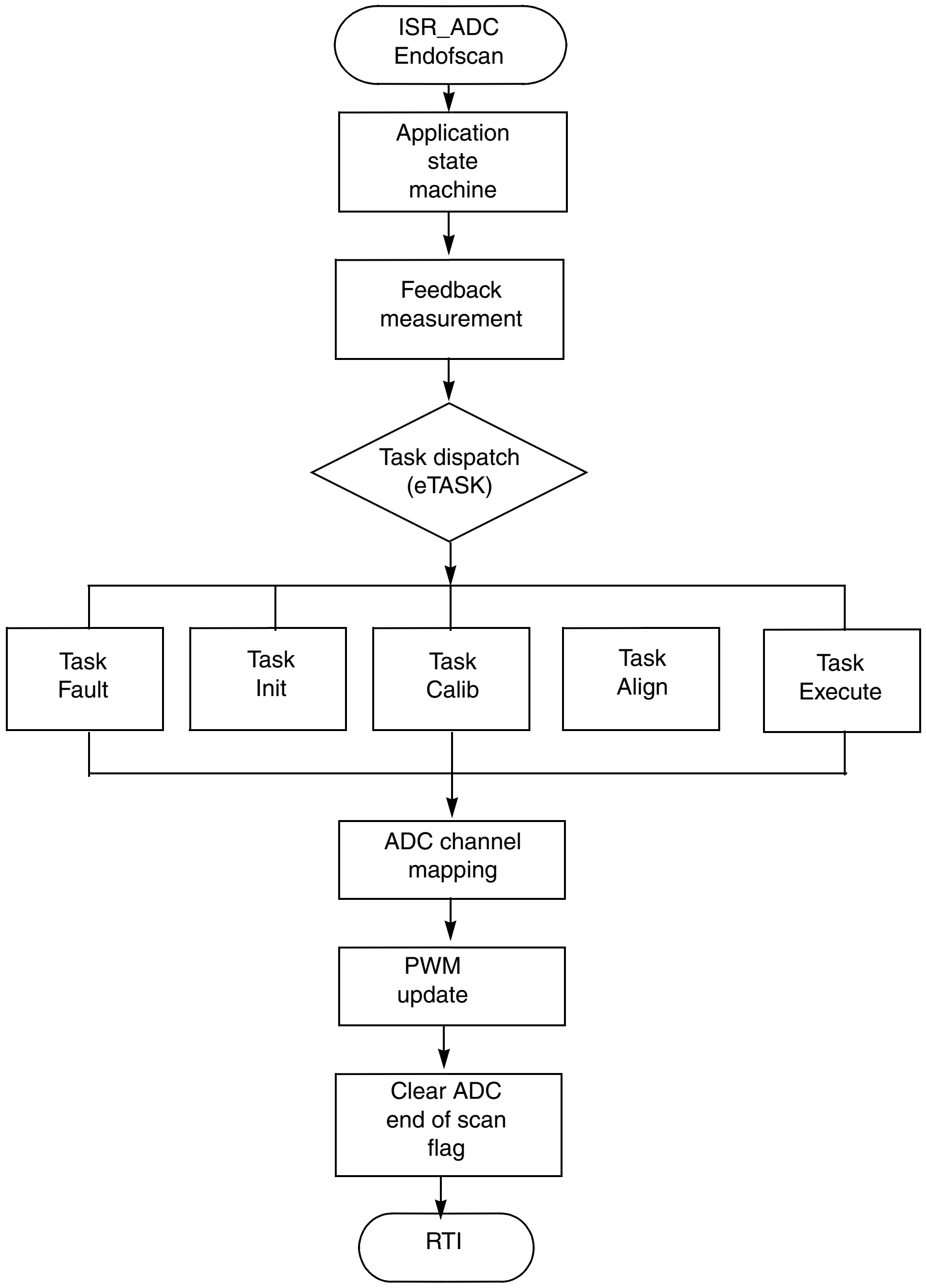


Figure 6-3. Flowchart of main interrupt ISR_ADC_Endofscan

The ISR_ADC_Endofscan interrupt service routine is called upon completion of the AD conversion and is executed at every second PWM period. The ISR_ADC_Endofscan is called once per two PWM reloads with a period set to 125 [usec]. It can also be set on each PWM reload 62.5 [usec]. This routine implements the function caller for motor control software and is executed every second PWM cycle period. The main software elements of this interrupt service routine are the following:

- Application state machine
- Measurement of feedback analog signals

- Dispatching tasks based on processing status
  - Fault task
  - Initialization task
  - Calibration task
  - Alignment task
  - Execution task
- ADC channel reconfiguration
- Update of PWMs

6.5 Feedback Measurement

The phase currents are measured by a shunt resistor in each phase. A voltage drop in the shunt resistor is amplified by an operational amplifier, and shifted up by 1.65 V. The resultant voltage is converted by an A/D converter. See Figure 6-4.

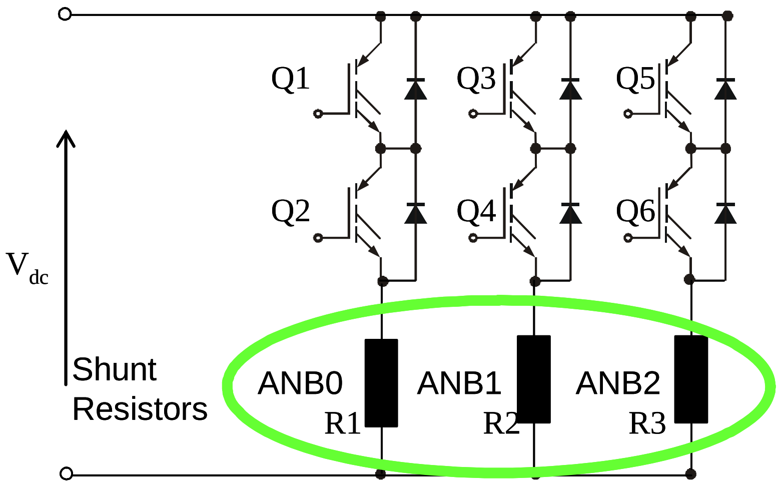


Figure 6-4. Measurements of phase currents using shunt resistors

As shown in Figure 6-4, the currents cannot be measured at each instant. For example, the current flows through Phase A (and shunt resistor R1) only if transistor Q2 is switched on. Likewise, the current in Phase B can be measured if transistor Q4 is switched on, and the current in Phase C can be measured if transistor Q6 is switched on. To get an actual instant of current sensing, voltage shape analysis must be performed.

The voltage shapes of two different PWM periods are shown in Figure 6-7. The voltage shapes correspond to center-aligned PWM sinewave modulation. As shown, the best instant of current sampling is in the middle of the PWM period, where all bottom transistors are switched on.

To set the exact instant of sampling, the DSP56F80x family offers the ability to synchronize ADC and PWM modules via the SYNC signal. This exceptional hardware feature, patented by Motorola, is used for current sensing. The PWM outputs a synchronization pulse that is connected as an input to the synchronization module TC2 (Quad Timer C, counter/timer 2). A high-true pulse occurs for each reload of the PWM, regardless of the state of the LDOK bit. The intended purpose of TC2 is to provide a user-selectable delay between the PWM SYNC signal and the updating of the ADC values. A conversion process can be initiated by the SYNC input which is an output of TC2.

The timing diagram in Figure 6-5 shows how the ADC sampling to the PWM update is performed. The events are executed in the following steps:

1. After the PWM (system) restart, the PWM is set for every opportunity reload with both full and half cycle reload. Therefore, when the PWM counter reaches zero a PWM reload occurs and the PWM_reload_sync is generated
2. The PWM Reload ISR is first time entered.
3. The second PWM reload occurs and PWM_reload_sync is generated at the PWM counter modulo value (half reload). The TMR channel 3 count is triggered by the PWM_reload_sync signal, which is connected to its secondary source input. The timer starts counting up from zero.
4. The PWM Reload ISR is second time entered. The PWM Reload ISR sets the PWM module for every fourth opportunity reload. This trick gives the functionality of PWM_reload_sync being generated each second half cycle reload.

A compare on T3 register occurs. A rising edge on the SYNC0 input of the ADC module starts an ADC conversion.

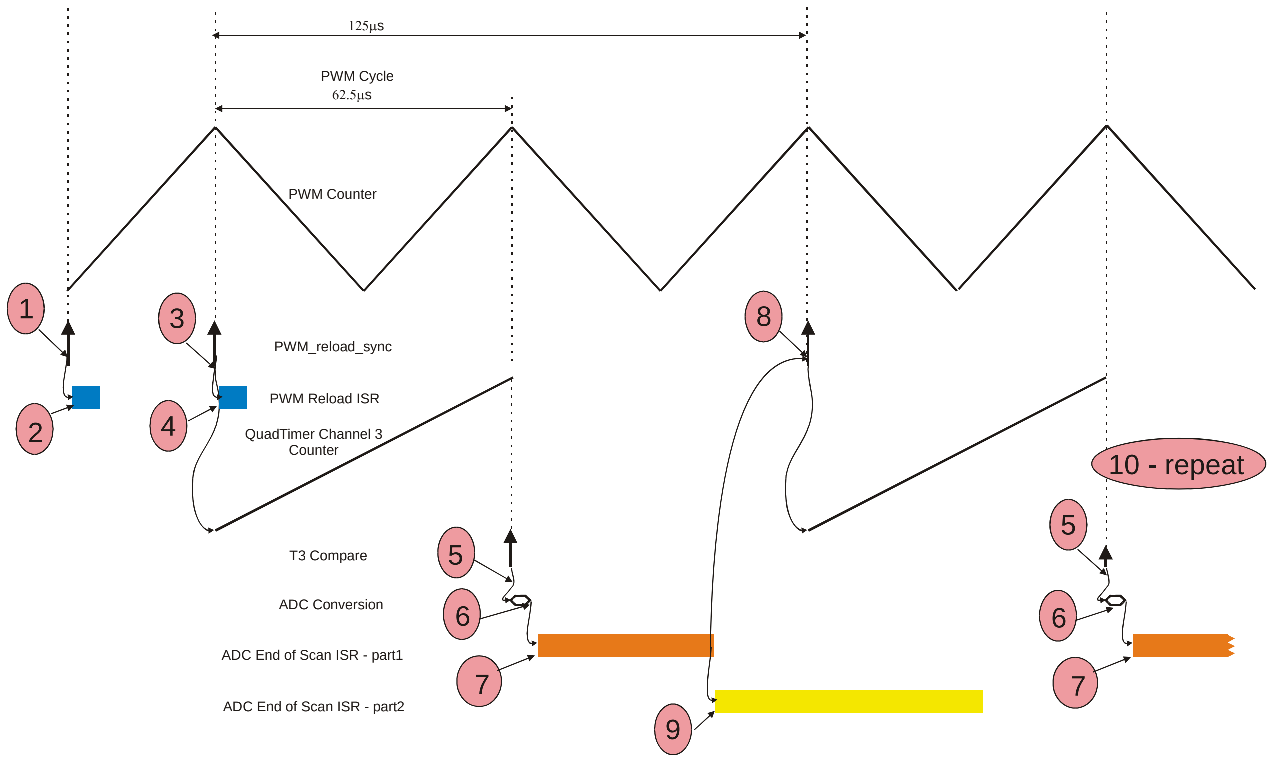


Figure 6-5. Time diagram of PWM and ADC synchronization and ADC End Of Scan interrupts

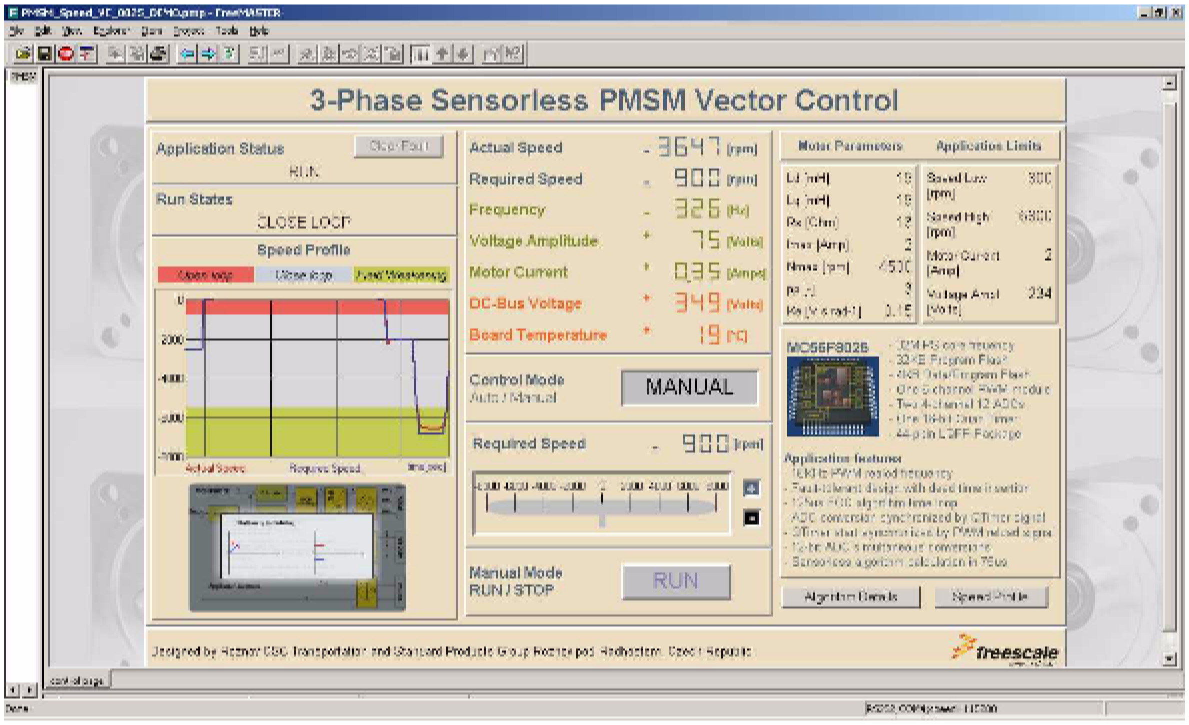
5. The ADC conversion is finished. The ADC End of Scan 1 flag (EOSI1) is set.
6. The ADC End of Scan ISR is entered. The ADC End of Scan ISR – part 1 calculates the current regulator. At the end of the ADC End of Scan ISR – part 1, the new values are stored in the PWM value registers (VAL0-5).

7. The PWM Reload occurs and the PWM is loaded with the buffered values from the PWM value registers (VAL0-5). TMR channel 3 count is triggered by the PWM_reload_sync signal, which is connected to its secondary source input. The timer starts counting up from zero.
8. The ADC End of Scan ISR – part 2 calculates SMO for position and speed estimation
9. The steps to 8 are repeated

The timing scheme where the current sensing alternates with PWM reload is used to get a best sampling performance (and dynamic) for the PMSM current control. The ADC End of Scan ISR – part 1 duration is low enough to calculate current regulator with the PWM update between two half cycle reloads. The ADC End of Scan ISR – part 1 and part 2 duration is low enough to get between two PWM cycles.

However, all three currents can not be measured at an arbitrary voltage shape. The PWM period II in Figure 6-7 shows an instant when the bottom transistor of Phase A is switched on for a very short time. If the on-time is shorter than a critical time, the current can not be accurately measured. The critical time is given by hardware configuration, transistor commutation times, response delays, of the processing electronics, and so on. Therefore, only two currents are measured and a third current is calculated from the following equation:

$$0 = i_A + i_B + i_C \qquad \textbf{\textit{Eqn. 19}}$$


3-Phase Sensorless PMSM Vector Control

Application Status: RUN

Run States: CLOSE LOOP

Speed Profile: Open loop | Close loop | Field Weakening

Actual Speed - 3647 [rpm]
Required Speed - 900 [rpm]
Frequency - 325 [Hz]
Voltage Amplitude + 75 [Volts]
Motor Current + 0.35 [Amps]
DC-Bus Voltage + 349 [Volts]
Board Temperature + 19 [°C]

Control Mode Auto / Manual: MANUAL

Required Speed - 900 [rpm]

Manual Mode RUN / STOP: RUN

Motor Parameters		Application Limits	
Ld [mH]	15	Speed Low [rpm]	300
Lq [mH]	15	Speed High [rpm]	6800
Rs [Ohm]	12	Motor Current [Amp]	2
Imax [Amp]	2	Voltage Ampl [Volts]	234
Nmax [rpm]	4500		
pp [-]	3		
Ke [V s rad-1]	0.15		

MC56F8025
- 32MHz core frequency
- 32KB Program Flash
- 4KB Data/Program Flash
- One 6-channel PWM module
- Two 4-channel 12 ADCs
- One 16-bit Quad Timer
- 44-pin LQFP Package

Application features
- 16KHz PWM reload frequency
- Fault-tolerant design with dead time insertion
- 125us FOC algorithm time loop
- ADC conversion synchronized by QTimer signal
- QTimer start synchronized by PWM reload signal
- 12-bit ADC simultaneous conversions
- Sensorless algorithm calculation in 75us

Designed by Roznov CSC Transportation and Standard Products Group Roznov pod Radhostem, Czech Republic

Figure 6-6. Voltage shapes of two different PWM periods

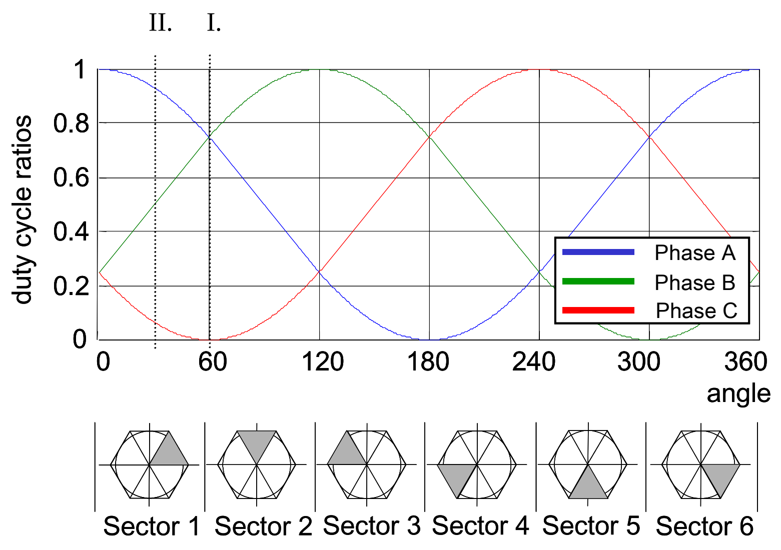


Figure 6-7. 3-phase Sinewave voltages and corresponding sector value

A decision must be made about what phase current must be calculated. The simplest technique is to calculate the current of the most positive voltage phase. For example, Phase A generates the most positive voltage within section 0–60°, Phase B within section 60°–120°, and so on; see Figure 6-7.

In this case, the output voltages are divided into six sectors, as shown in Figure 6-7. The current calculation is then made according to the actual sector value.

Sectors 1 and 6:

$$i_A = -i_B - i_C \qquad \textit{Eqn. 20}$$

Sectors 2 and 3:

$$i_B = -i_A - i_C \qquad \textit{Eqn. 21}$$

Sectors 4 and 5:

$$i_C = -i_B - i_A \qquad \textit{Eqn. 22}$$

NOTE

The sector value is used for current calculation only, and has no other meaning in the sinewave modulation. But if you use any type of space vector modulation, you can get the sector value as part of space vector calculation.

6.6 Application State Machine

The application state machine stores the status at a given time and can operate on input to change the status or cause an action to take place for any given change. This application state machine determines what state the application is in and the task are dispatched based on this. This state machine has five application states: FAULT, INIT, CALIB, ALIGN and EXECUTE. The transition between the states is shown in Figure 6-8

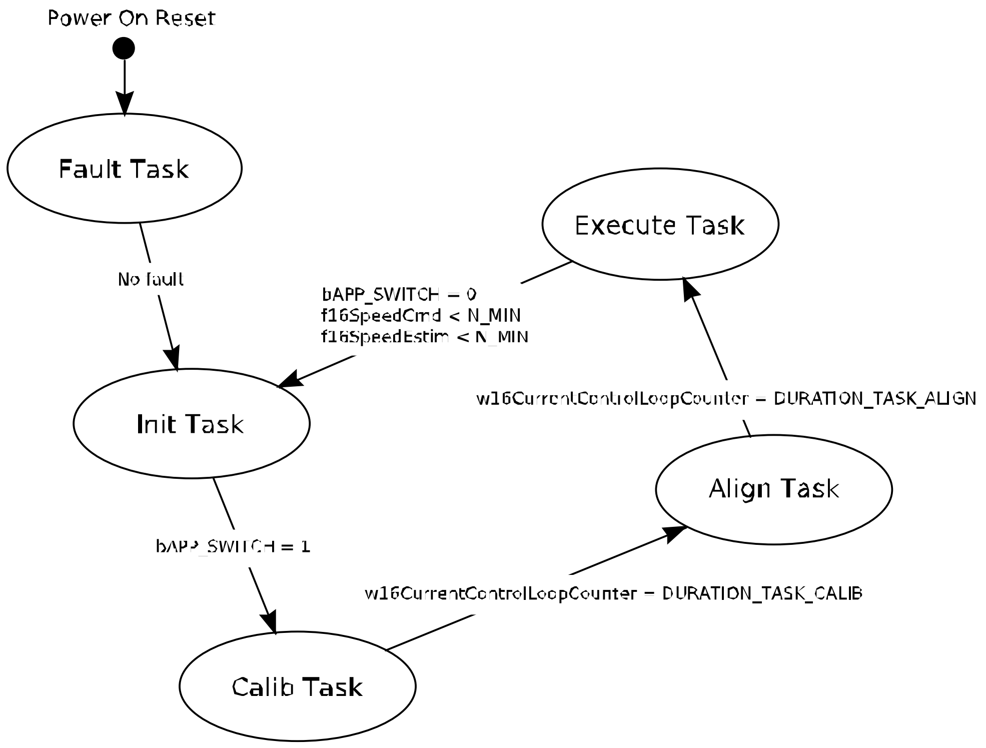


Figure 6-8. Application state machine

After reset, the application is set to the fault state where over-current, over-voltage, and under-voltage fault bit states are tested. If there is no fault detected, the application goes to the next state. After any of them are set, the fault state is immediately entered. The application states are described in the following sections.

6.7 Task Dispatching

This task dispatcher invokes defined functions such as

— TaskFault
— TaskInit
— TaskCalibration
— TaskAlignment
— TaskExecute

The individual tasks of the control routines are described in the following sections.

6.7.1 Task FAULT

The application goes to this state immediately after reset or when a fault is detected. The system allows all the states to pass on to the FAULT state. Faults detected in the application (called function TaskFault())are:

- Over-current
- Over-voltage
- Under-voltage

The next state (INIT) is set once all fault bits are cleared. This means that no fault has been detected.

Moreover, this task can be entered asynchronously when PWM_Fault_ISR is invoked. The PWM_Fault_ISR routine (Figure 6-9) generates asynchronous events and disables the PWM output pad. It is executed on power drive protect interrupt that occurs when an inverter output short-circuit happens. If a fault is latched in, it must be cleared and the fault state is invoked prior to enabling the PWM, to prevent an unexpected interrupt.

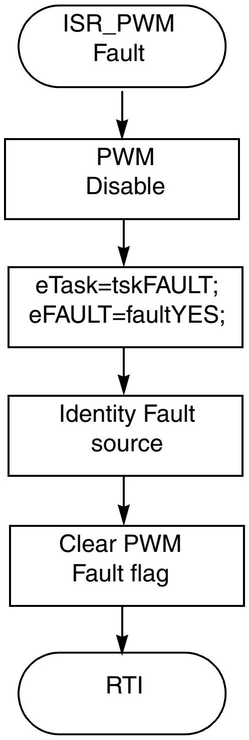


Figure 6-9. Fault interrupt service routine

6.7.2 Task INIT

This transient state is entered from the FAULT state. The application variable initialization is performed, calling function TaskInit() where all variables are initialized to their default values; speed PI controller, speed ramp increments, phase increment, and so on).

6.7.3 Task CALIB

This transient state, between the INIT and ALIGN states enables the PWM pads to be set with a fifty-percent duty cycle, and measure the phase current quantity to set the offset value performed in the function (TaskCalibration()).

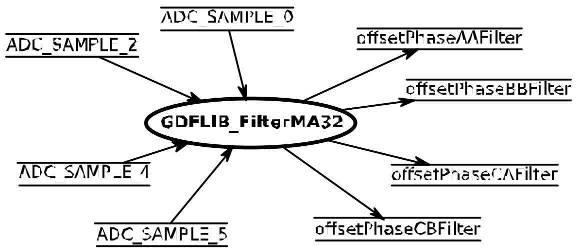


Figure 6-10. Dataflow diagram during calibration

6.7.4 Task ALIGN

In this design, there is a no motor position or speed sensor used. Because in the beginning, the initial position needs to be determined before the motor is started. One of the possibilities and easily implemented, is a method to align the rotor to a predefined position. The motor is powered by a selected static voltage pattern (usually the zero position in the sine wave table). The rotor aligns to the predefined position.

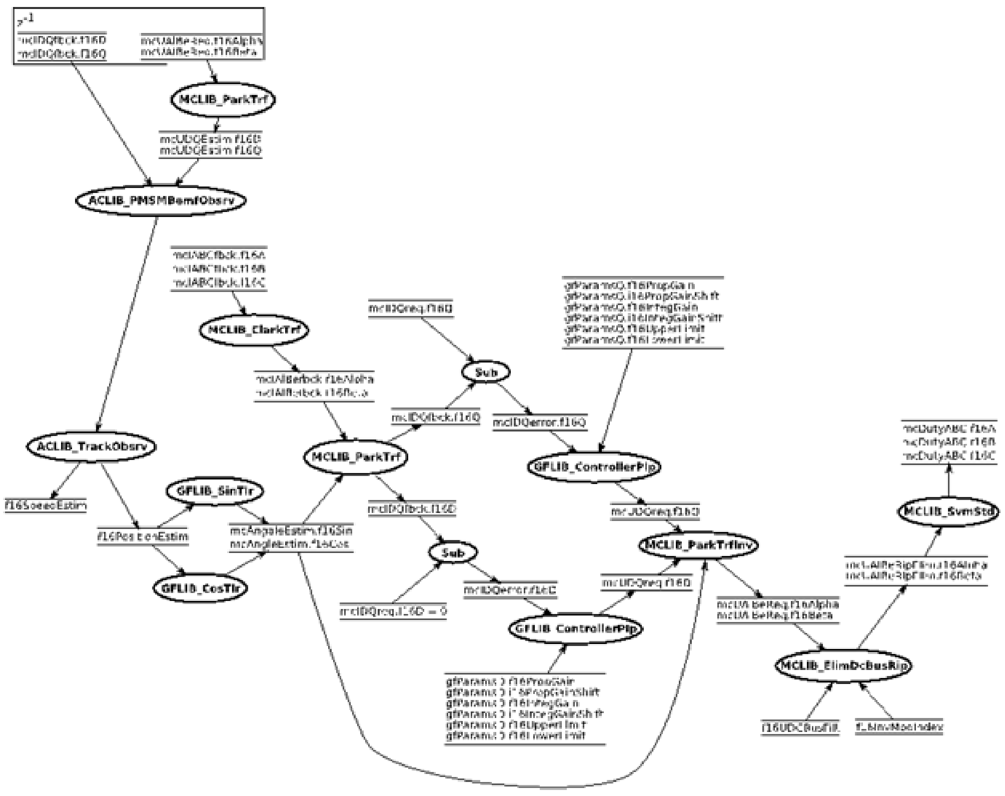


Figure 11. Dataflow diagram during alignment

The alignment is done only once during first motor start. The constant current vector is applied to the stator in terms of amplitude and its position. After a stabilization period the rotor flux must be aligned to the stator flux. In practice, this is true when the external load torque is low enough compared to the torque produced by the alignment vector. The dataflow diagram shows the implemented alignment procedure in Figure 11.

6.7.5 Task EXECUTE

This is the most complex state and performs the majority of tasks. All necessary procedures for motor startup and rotation are processed in this state. When the speed command is set to a non-zero value, the motor starts to run. Otherwise, the motor is in the INIT state and the PWM signals are disabled.

As indicated by the flowchart of the ISR_ADC_Endofscan (see Figure 6-3), the sampling of analog values is processed to acquire appropriate information for the overall control process and monitoring.

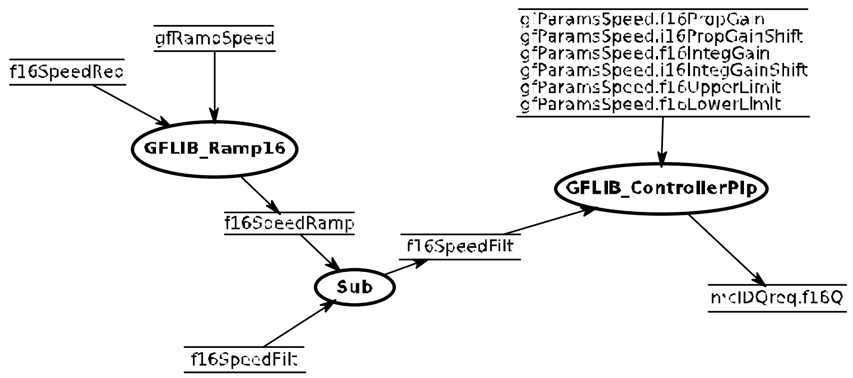

gfRampSpeed
f16SpeedReq
gfParamsSpeed.f16PropGain
gfParamsSpeed.i16PropGainShift
gfParamsSpeed.f16IntegGain
gfParamsSpeed.i16IntegGainShift
gfParamsSpeed.f16UpperLimit
gfParamsSpeed.f16LowerLimit
GFLIB_Ramp16
GFLIB_ControllerPIp
f16SpeedRamp
f16SpeedFilt
Sub
mcIDQreq.f16Q
f16SpeedFilt

Figure 6-12. Dataflow of speed loop

The speed command is calculated according to the state of the control signals and means that the task EXECUTE is called. The speed command is then processed by means of the speed ramp algorithm. The comparison between the actual speed command obtained from the ramp algorithm output and the estimated speed generates a speed error. The speed error is input to the speed PI controller, generating a new desired level of reference for the torque producing component of the stator current. The dataflow diagram of the speed control is shown in Figure 6-12.

This procedure is followed by calculating the digital values of the rotor position.Speed measurement is preformed by software computation by the estimation algorithm. Knowing the appropriate control information, the PMSM current control is followed and individual software functional blocks are executed. When the PMSM current control process is performed, the required voltage vector is generated and is ready to apply by updating the PWM value registers.

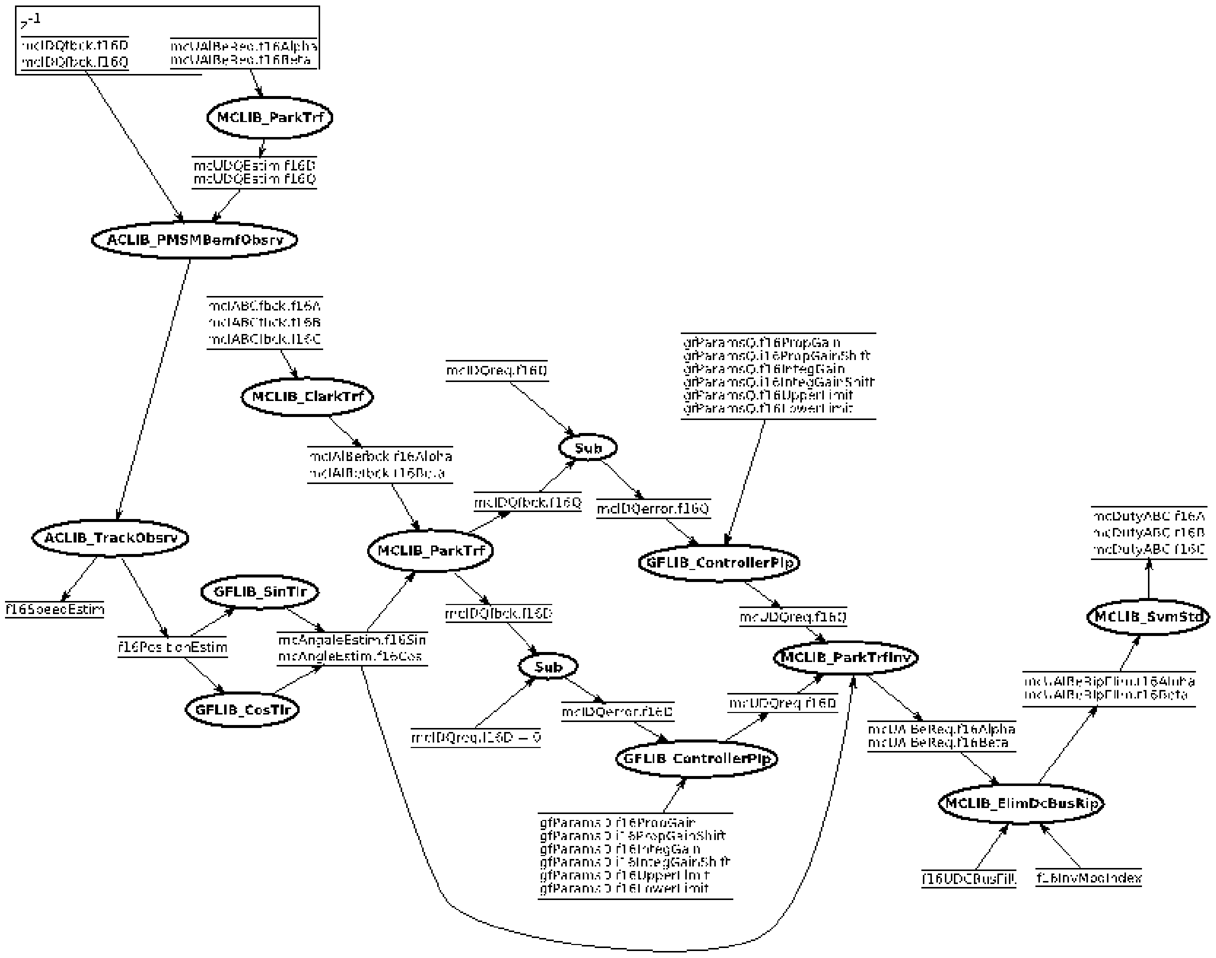


Figure 6-13. Dataflow of current control loop

Additionally, there might be a Freemaster software recorder FMSTR_Recorder() routine used to monitor the application.

6.8 FreeMASTER Software

FreeMASTER software was designed to provide a debugging, diagnostic and demonstration tool for the development of algorithms and applications. Moreover, it is useful for tuning the application for different power stages and motors. Almost all the application parameters can be changed via the FreeMaster interface. This consists of a component running on a PC and another part running on the target DSC, connected via an RS-232 serial port. A small program is resident in the DSC that communicates with the FreeMASTER software to parse commands, return status information to the PC, and process control information from the PC. FreeMASTER software executing on the PC uses Microsoft Internet Explorer as the user interface.

6.8.1 FreeMASTER Serial Communication Driver

The presented application includes the FreeMASTER Serial Communication Driver. The FreeMASTER Serial Communication Driver fully replaces the former PC master driver. The new FreeMASTER driver remains fully compatible with the communication interface provided by the old PC master drivers. It has many useful enhancements and optimizations.

The main advantage of the new driver is a unification across all supported Freescale processor products, as well as several new features that were added. One of the key features implemented in the new driver is the target-side addressing (TSA), which enables an embedded application to describe the memory objects it grants the host access to. By enabling the TSA safety option, the application memory can be protected from illegal or invalid memory accesses.

To include the new, The FreeMASTER Serial Communication Driver in the application, the user has to manually include the driver files in the CodeWarrior project. For the presented application, the driver has already been included.

The FreeMASTER driver files are located in following folders:

{Project}support\freemaster\56F8xxx—Contains platform-dependent driver C-source and header files, including a master header file **freemaster.h.**

{Project}support\freemaster\common—Contains common driver source files shared by the driver for all supported platforms.

All C files included in the freemaster folders are added to the project for compilation and linking (see **support** group in the project). The master header file **freemaster.h** declares the common data types, macros and prototypes of the FreeMASTER driver API functions. This should be included in your application (using `#include` directive), wherever you need to call any of the FreeMASTER driver API functions.

The FreeMASTER driver does not perform any initialization or configuration of the SCI module it uses to communicate. It is the user’s responsibility to configure the communication module before the FreeMASTER driver is initialized by the FMSTR_Init() call. The default baud rate of the SCI communication is set to 9600 Bd.

NOTE

Higher communication speeds than 9600 Bd is not supported by the MC56F8013/23 Controller Board due to the limited speed of opto-couplers.

FreeMASTER uses a poll-driven communication mode. It does not require the setting of interrupts for the SCI. Both communication and protocol decoding are managed in the application background loop. The polling-mode requires a periodic call of the `FMSTR_Poll()`function in the application main.

The driver is configured using the **appconfig.h** header file. Changes to the file are preferably made through the provided Quick Start Graphical Configuration Tool (in CodeWarrior toolbar Project/Configuration Tool). The user has to modify this file to configure the FreeMASTER driver. The FreeMASTER driver C-source files include the configuration file, and use the macros defined there for conditional and parameter compilation.

A detailed description of the FreeMASTER Serial Communication Driver is provided in the *FreeMASTER Serial Communication Driver User's Manual*.

6.8.2 FreeMASTER Recorder

Part of the FreeMASTER software is also a recorder that is able to sample the application variables at a specified sample rate. The samples are stored in a buffer and read by the PC via an RS232 serial port. The sampled data can be displayed in a graph or stored. The recorder behaves like a simple on-chip oscilloscope with trigger and pretrigger capabilities. The size of the recorder buffer and the FreeMASTER recorder time base can be defined in the **appconfig.h** configuration.

The recorder routine must be called periodically from the loop where you want to take the samples. The following line must be added to the loop code:

```
FMSTR_Recorder(); /* FreeMASTER recorder routine call */
```

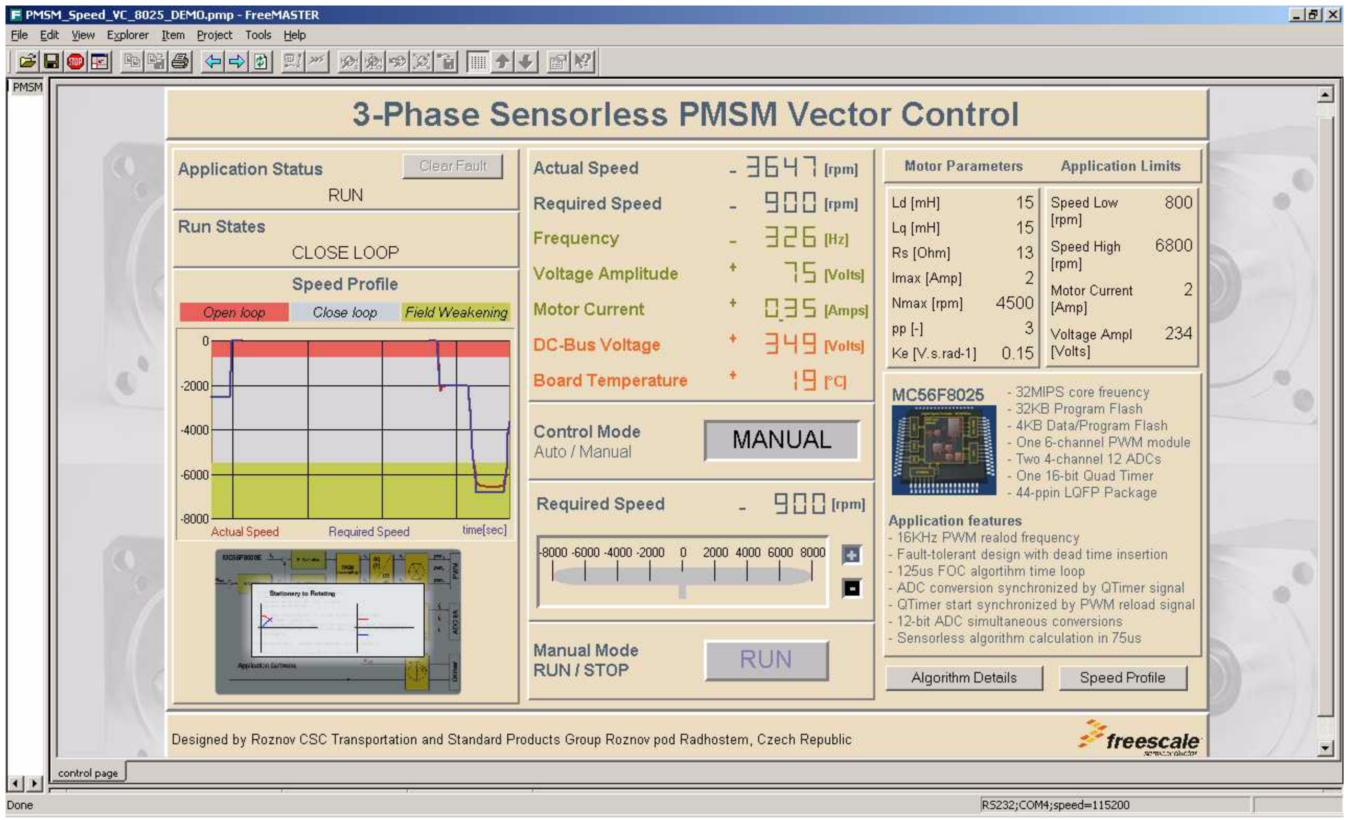
In this application, the FreeMaster recorder is called from the ADC Complete EOS interrupt that creates a 125µs time-base for the recorder function. A detailed description of the FreeMASTER software is provided in the *FreeMASTER Software User Manual*.

6.8.3 FreeMASTER Control Page

The FreeMASTER control page creates a graphical user interface (GUI) for the sensorless PMSM vector control application. Start the FreeMASTER software window’s project by clicking on the **PMSM_Sinusodial_Sensorless_Tuning.pmp** file**.** Figure 6-14 illustrates the FreeMASTER software control window after this project has been launched. To switch to the control page, click on the **control page** tag.

A user is able to monitor all the important quantities of the motor by clicking the Speed gauge, the motor is started and the desired speed is then set. The actual motor speed, motor currents, and voltages are displayed on the control page gauges.

The application status is displayed. A status fault LED indicates the occurrence of an application fault.



3-Phase Sensorless PMSM Vector Control

Application Status: RUN

Run States: CLOSE LOOP

Speed Profile

Actual Speed	- 3647	[rpm]
Required Speed	- 900	[rpm]
Frequency	- 326	[Hz]
Voltage Amplitude	+ 75	[Volts]
Motor Current	+ 035	[Amps]
DC-Bus Voltage	+ 349	[Volts]
Board Temperature	+ 19	[°C]

Control Mode Auto / Manual: MANUAL

Required Speed - 900 [rpm]

Manual Mode RUN / STOP: RUN

Motor Parameters		Application Limits	
Ld [mH]	15	Speed Low [rpm]	800
Lq [mH]	15	Speed High [rpm]	6800
Rs [Ohm]	13	Motor Current [Amp]	2
Imax [Amp]	2	Voltage Ampl [Volts]	234
Nmax [rpm]	4500		
pp [-]	3		
Ke [V.s.rad-1]	0.15		

MC56F8025
- 32MIPS core freuency
- 32KB Program Flash
- 4KB Data/Program Flash
- One 6-channel PWM module
- Two 4-channel 12 ADCs
- One 16-bit Quad Timer
- 44-ppin LQFP Package

Application features
- 16KHz PWM realod frequency
- Fault-tolerant design with dead time insertion
- 125us FOC algortihm time loop
- ADC conversion synchronized by QTimer signal
- QTimer start synchronized by PWM reload signal
- 12-bit ADC simultaneous conversions
- Sensorless algorithm calculation in 75us

Designed by Roznov CSC Transportation and Standard Products Group Roznov pod Radhostem, Czech Republic

freescale semiconductor

Figure 6-14. FreeMASTER control screen

The FreeMASTER software control page actions that are supported:

- Setting the required speed of the motor
- Reset the application fault status

The FreeMASTER software control page displays:

- DC bus current and voltage
- d, q axis currents and voltages
- Application fault status

For software tuning and demonstration, the following sub-blocks are prepared

- Demonstration measurements
- Control regulators—For adjusting PI controller parameters
- Observer tuning
    - — Adjusting the SMO parameters
    - — Adjusting the SMO feedback
    - — Adjusting start-up parameters
- Other tuning—Code execution measurement, DC bus brake

Chapter 7
Application Setup

The PM synchronous motor vector control application is targeted at the MC56F8013/23/25 device. The design of the PM synchronous motor vector control drive incorporates these hardware components:

- MC56F8013/23/25 controller board
- Three-phase AC/BLDC high-voltage power stage board
- Three-phase PM synchronous motor (default configuration for motor TG drives TGT3-0040-45-320/T0PS1K-1M)

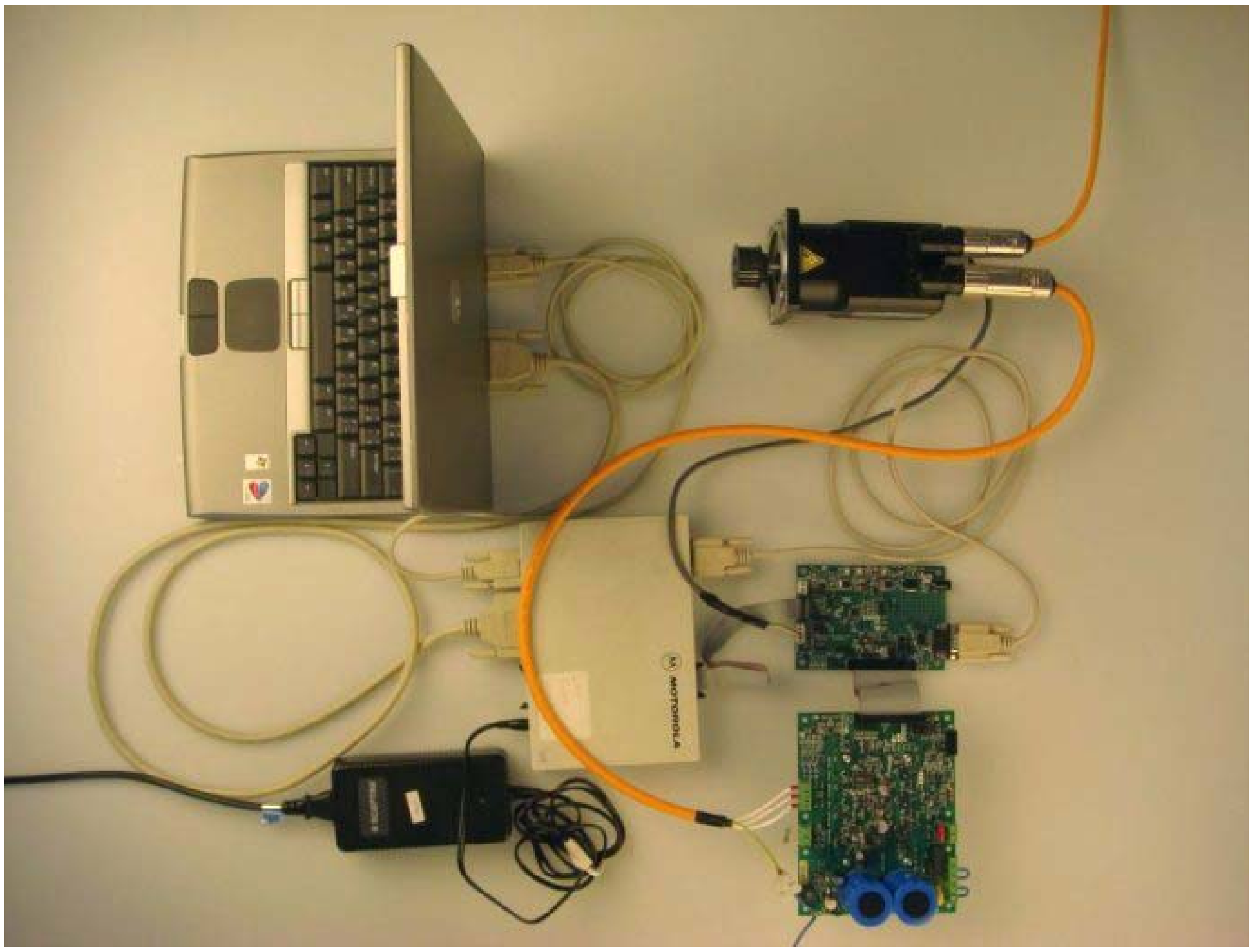


Figure 7-1. Demo application setup

7.1 MC56F8013/23/25 Controller Board Setup

Prior to the MC56F8013/23/25 controller board being connected to the power stage, it needs to be configured for correct operation. The demo application code also has to first be programmed into the flash memory.

Table 7-1. Jumper configuration

MC56F80xxEVB	
Jumper	**Setting**
JP3	1 2–3
JP9	1–2
JP6 PHBIS	1–2
JP6 PHAIS	1–2
JP8 TEMP	1–2
JP8 PHBIS / BEMF_B	1–2
JP6PHCIS	1–2
JP6 PHAIS	1–2

For the MC56F8013/23/25 controller board configuration, follow these steps:

1. Set the jumper configuration on the MC56F8013/23/25 controller board as shown in Table 7-1.
2. Connect a +12 V power supply to the J12 power connector on the MC56F8013/23/25 controller board.
3. Set the over-current and PFC over-current thresholds.
4. Trim potentiometer R58 sets the PFC over-current threshold. Use a voltmeter to measure the threshold level at test point TP4. Turn the R58 trim potentiometer to set the threshold level. Voltage level on TP4 must be > 3.2 V.
5. Trim potentiometer R59 sets the over-current threshold. Use a voltmeter to measure the threshold level at test point TP6. Turn the R59 trim potentiometer to set the threshold level. The voltage level on TP6 must be > 3.2 V.
6. Connect the parallel JTAG command converter or the USB-TAP to the host PC to the J5 header on the MC56F8013/23/25 controller board.
7. Compile your project and program it into the device.
8. Disconnect the +12 V power supply from the J12 power connector on the MC56F8013/23/25 controller board.
9. Unplug the JTAG command converter (or the USB-TAP) from the J5 header on the MC56F8013/23/25 controller board.

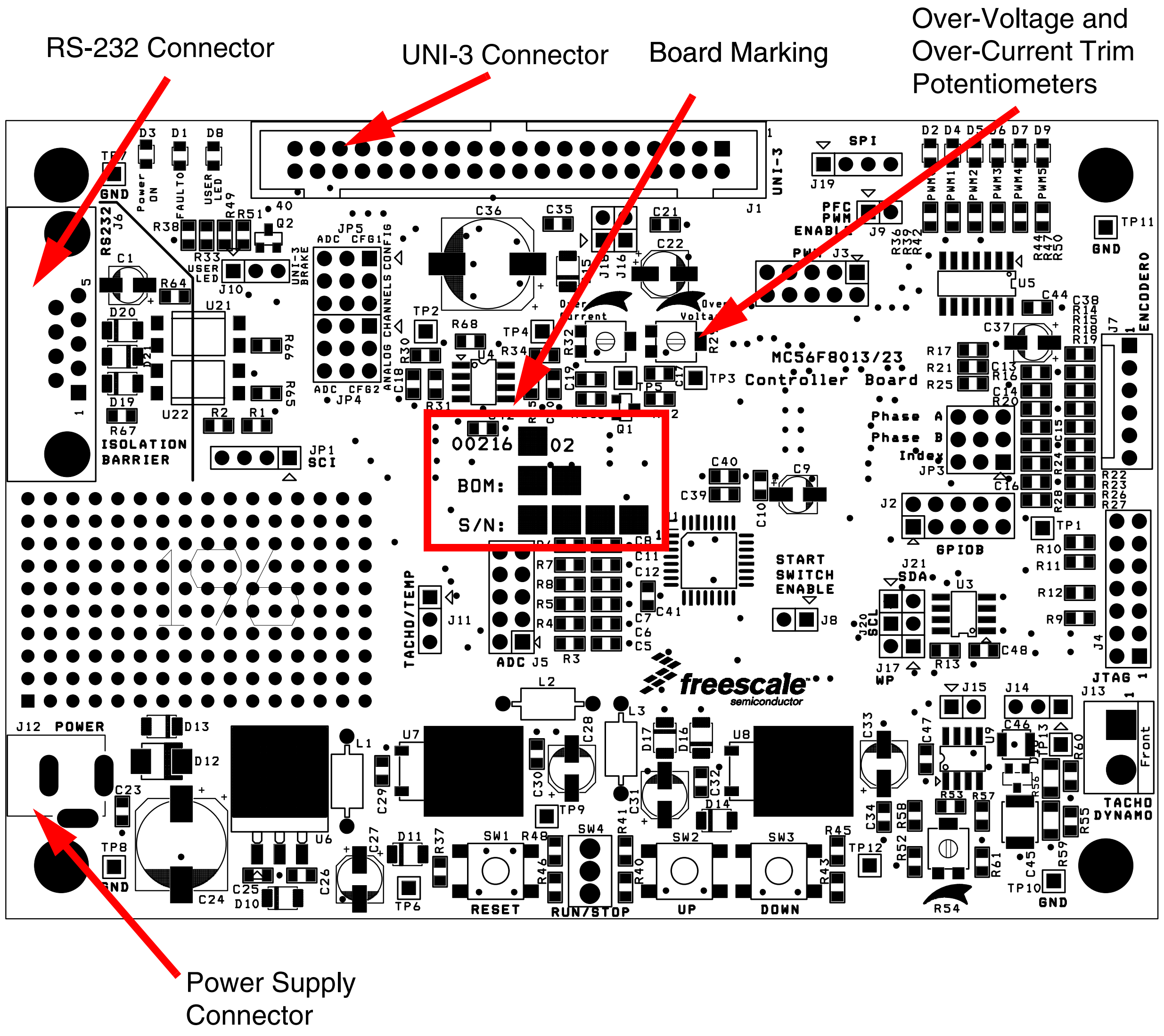


Figure 7-2. MC56F8013/23 controller board view

7.2 Demo Hardware Setup

When the MC56F8013/23/25 controller board is configured it can be connected to the power stage and the whole demo hardware setup can be built. The complete application setup, is shown in Figure 7-1 and Figure 5-1. To build the demo application setup perform these steps:

1. Connect the UNI-3 connector (J1) on the MC56F8013/23/25 controller board to the UNI-3 counterpart connector on the three-phase AC/BLDC high voltage power stage board via a 40-pin ribbon cable.
2. Connect a serial cable to an open COM port on the host PC and to the J6 DB-9 connector on the MC56F8013/23 controller board, for FreeMASTER remote control.
3. Connect an incremental encoder cable to the J7 connector on the MC56F8013/23/25 controller board.

4. Connect the motor phases to terminal J6 on the three-phase AC/BLDC high voltage power stage board.
5. Connect a 115/230 V AC power supply or a 100–320 V DC power supply to terminal J6 on the three-phase AC/BLDC high voltage power stage board.

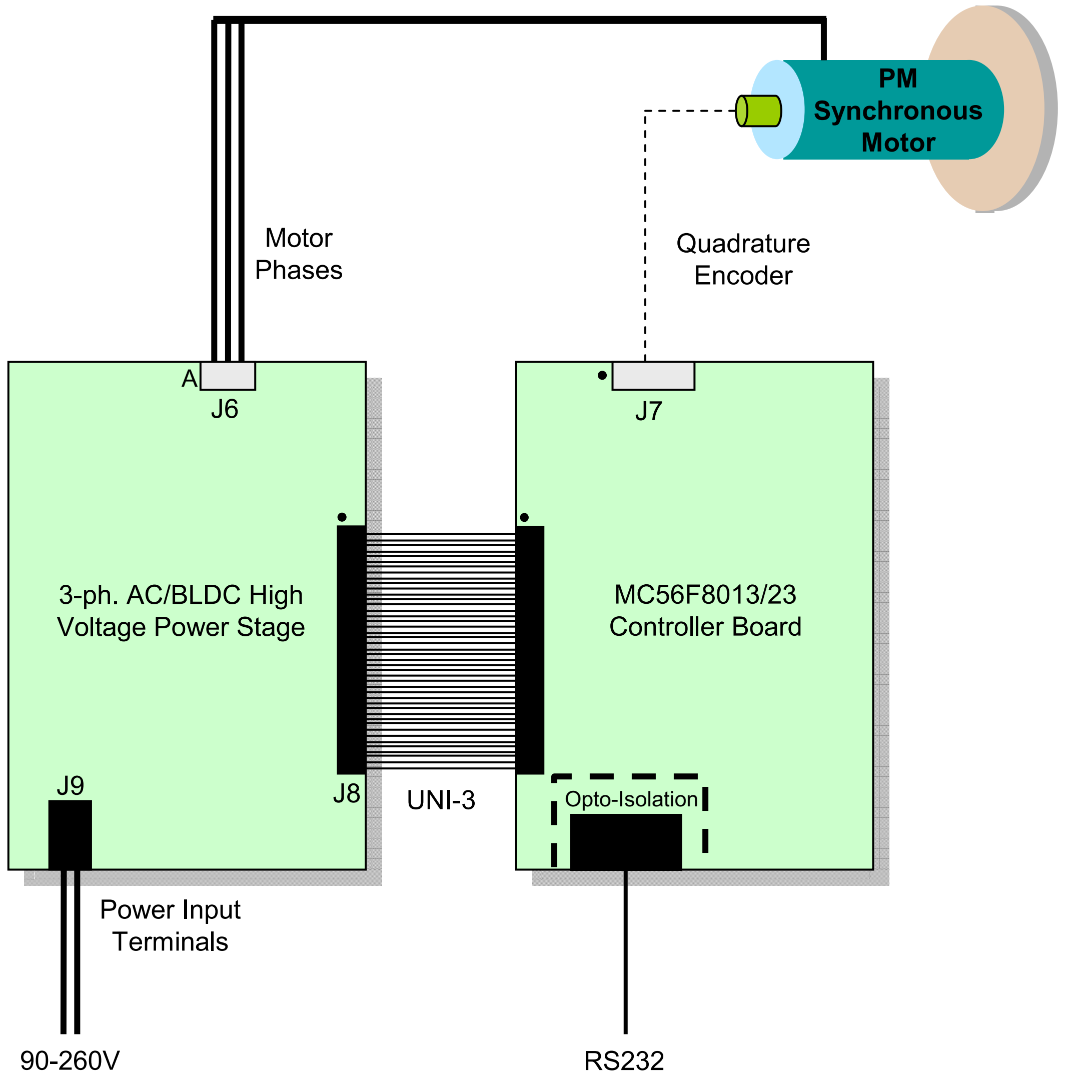


Figure 7-3. Demo application connection overview

6. Switch on the high-voltage power supply.
7. Start the FreeMASTER project and switch to the “control page” panel.

WARNING

There is a risk of electric shock. Both the MC56F8013/23/25 controller board and the three-phase AC/BLDC high voltage power stage board are connected to a high voltage. The start/stop toggle switch and the up/down buttons on the MC56F8013/23/25 controller board are disabled in this application. The user must avoid touching them. The only safe mode of control is by remotely controlling the application using the host PC running the FreeMASTER application. The USB port is the only interface that provides galvanic isolation.

WARNING

The JTAG connector does not provide galvanic isolation. The demo hardware setup must be disconnected from high voltage if a JTAG connection is required. To debug the application, or to download code to the device flash memory using JTAG, use a low voltage +12 V power supply connected to the J8 connector on the MC56F8013/23/25 controller board.

Chapter 8
Results and Measurements

All results presented in this section have been obtained with sensorless operation. To test the accuracy of the position and speed estimation, the motor has been equipped with an incremental encoder giving the exact rotor position. All algorithms initial position detection, start up sequence, and back-EMF state observer have been implemented using a 16-bit fixed point digital signal controller, Freescale’s MC56F8023. Motor phase currents were measured at the shunt resistors placed in each of the inverter legs and each phase current is reconstructed according to.

8.1 System and Measurement Conditions

8.1.1 Hardware Setup

The application measurements were provided using a TGT3-0130-30-320 motor and the default hardware setup described in Chapter 7, “Application Setup”.

8.1.2 Start-up Performance

After identifying the initial rotor position, the FOC is started in open-loop mode. The current set-point is determined by the speed controller that generates the torque reference current iqREF. Moreover, a certain amount of id_{REF} current is generated that improves torque producing capability of the motor by adding a reluctance torque.

Figure 8-1 shows experimental results of the startup sequence where the time profile of the reference open-loop position OL and the reference currents iq_{REF}, and id_{REF} are displayed. This sequence is accurately chosen to ensure safe starting with minimum oscillation up to the maximum torque. The weighting function parameters for the lower limit M1 was set to 100 [rpm] and the upper limit M2 was set to 200 [rpm], respectively.

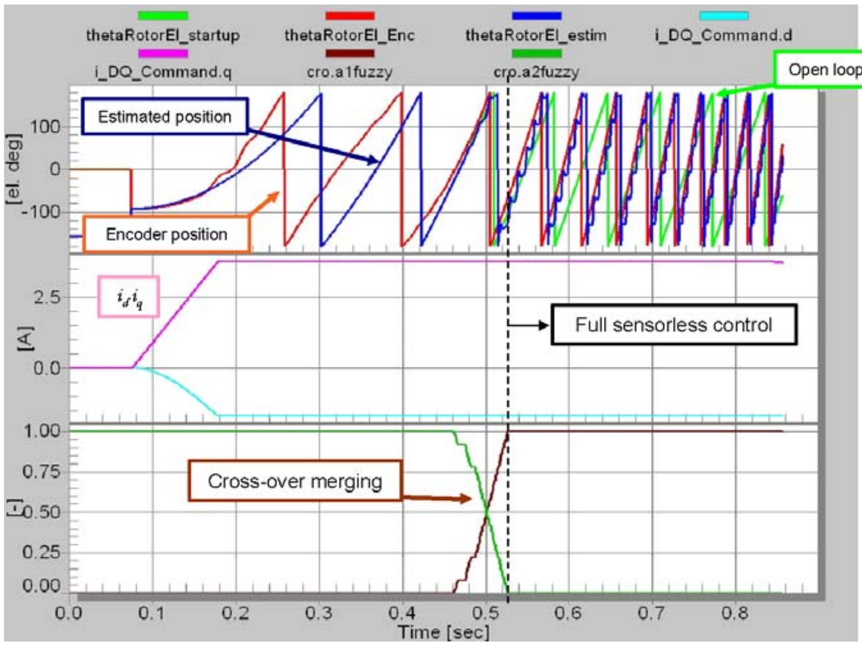


Figure 8-1. Startup sequence experimental results

Motor phase currents during the start-up sequence consist of initial position detection, open-loop startup, and transition to sensorless closed loop operation are depicted in Figure 8-2.

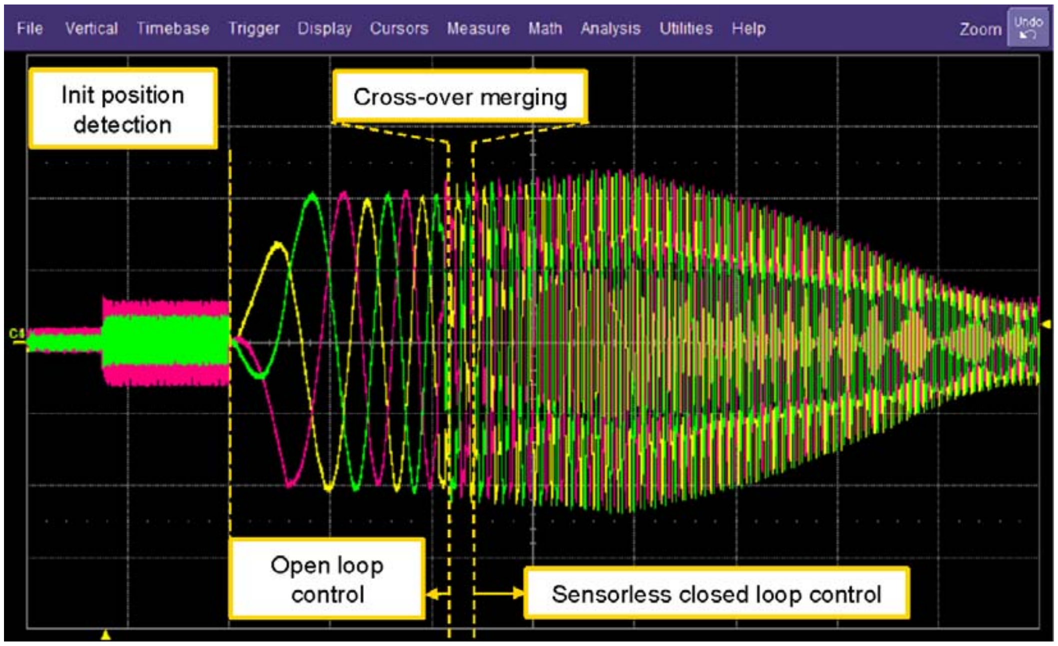


Figure 8-2. Motor phase currents of startup sequence

8.1.3 Wash Cycle Performance

During the wash cycle operation the motor runs at a relatively low speed (300 up to 1000[rpm]) and it might generate high torque (up to 2.5[Nm]) depending on the loading condition that is given by the weight of clothes, wash program, amount of the water, and so on. Figure 8-3 shows sensorless operation at constant 300 [rpm] where speed and torque ripple can be identified under the load condition. The torque ripples were caused by an unbalanced load.

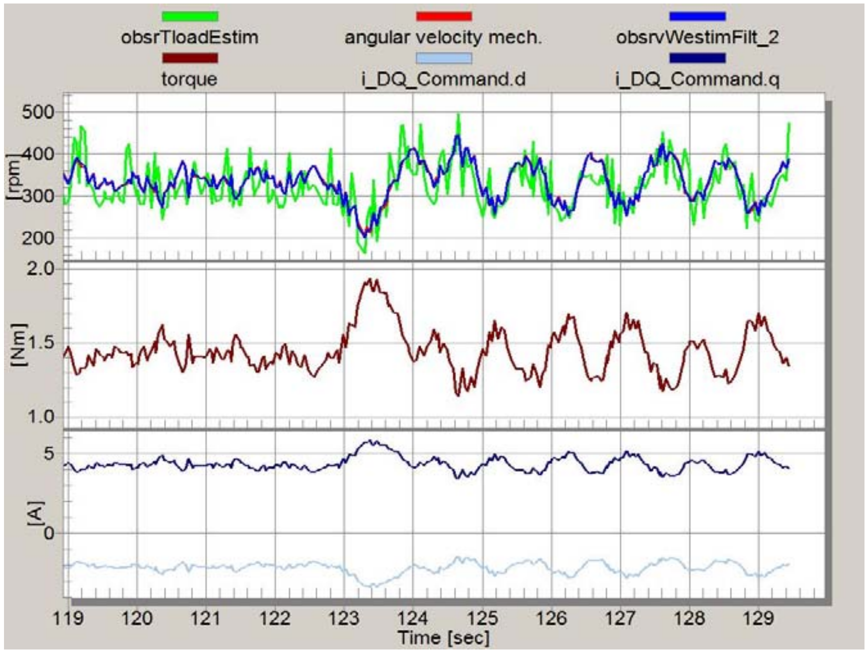


Figure 8-3. Sensorless operation during steady state spin at 300 [rpm]

Figure 8-4 shows the speed profile diagram where initially the motor runs at 300 [rpm] with a fully loaded washing machine. The torque variation visible in Figure 8-4 is caused by agitation of heavily soaked clothes. The wash program commands acceleration to 750 [rpm] speed level, then there is a certain time interval of a constant spin at 750 [rpm].

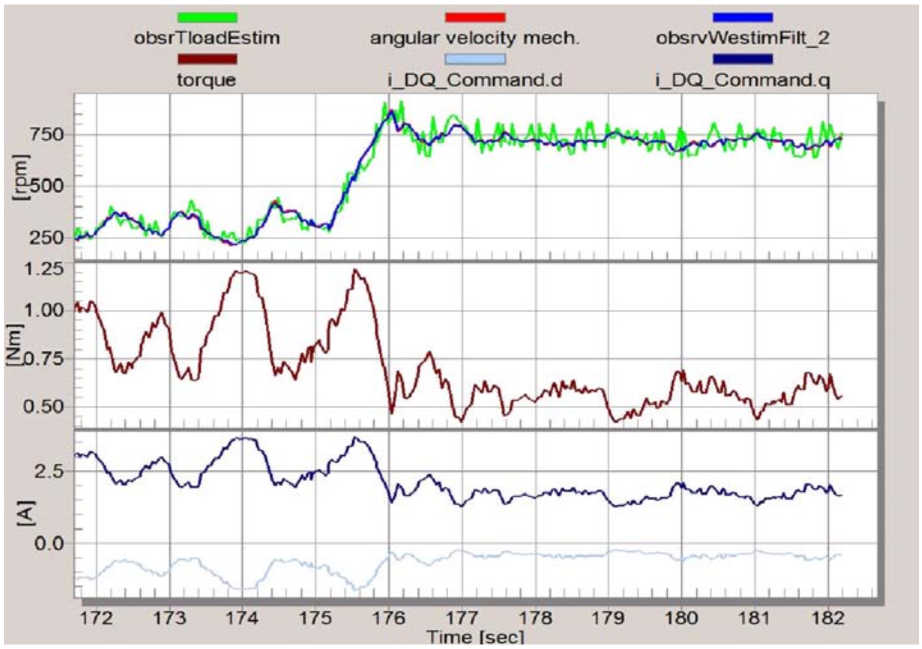


Figure 8-4. Speed operation during wash cycle with acceleration

Motor phase currents of the sensorless speed control during the wash cycle operation when the motor accelerates from 300 [rpm] to 750 [rpm] are depicted in Figure 8-5.

Motor Phase Currents:

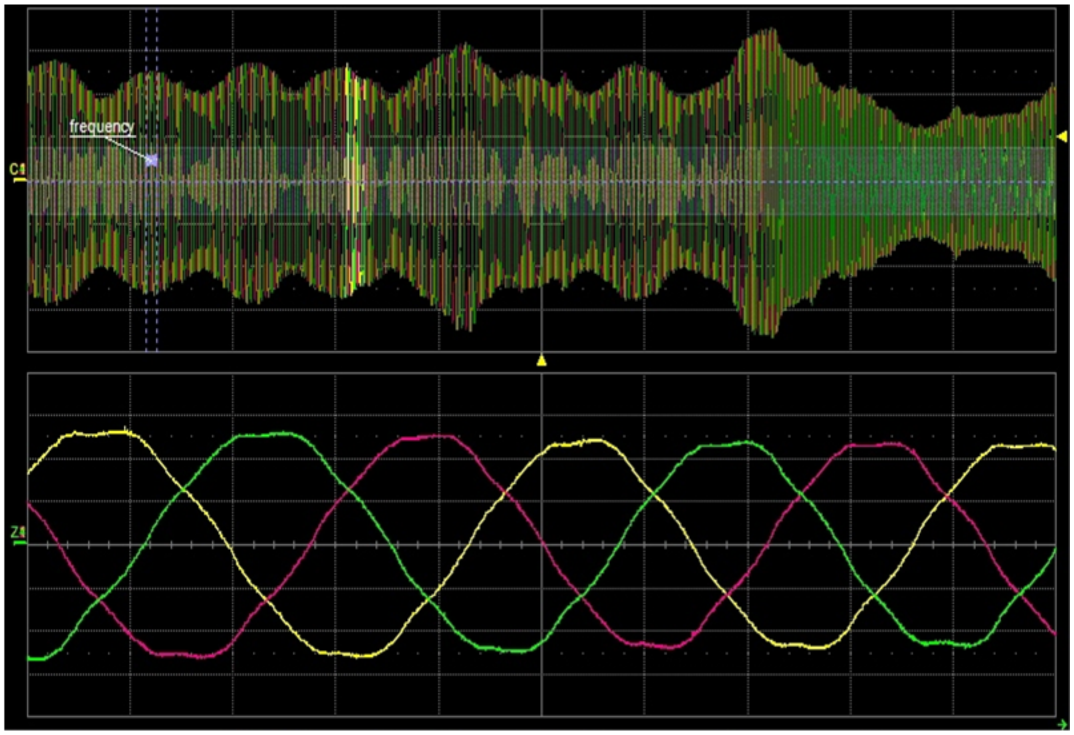


Figure 8-5. Motor phase currents during wash cycle with acceleration

8.1.4 Spin Cycle Performance

After the wash cycle completion, the washing machine starts operation in the spin-dry cycle shown in Figure 8-6. The estimated torque was used to detect out-of-balance in the wash load before entering the spin cycle. The washing machine accelerates to the pre-defined high speed level to 11000 [rpm] for a spin dry where it remains for a short time. Next, the washtub decelerates down to 9000 [rpm], after deceleration it proceeds to a low speed level. This cycle might be repeated several times depending on the wash program chosen.

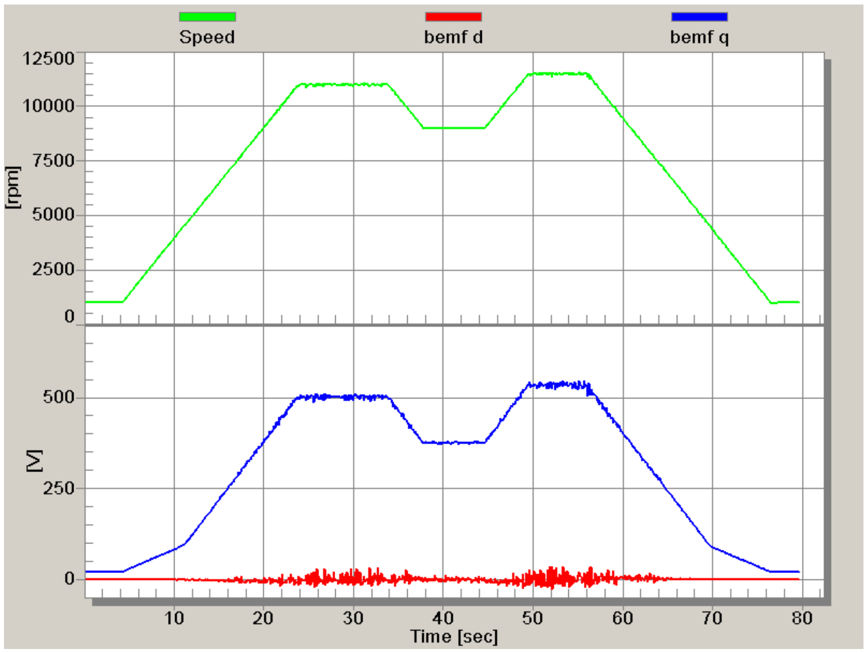


Figure 8-6. Spin dry cycle

The motor mechanical characteristic is one of the most important characteristics of the electric motor for washing machine drives. This characteristic shows the motor torque generation capability across the operational speed range. Knowing the shape of the motor mechanical characteristic allows the application designer to select the correct motor drive. Hence, to validate the proposed sensorless algorithm, the motor mechanical characteristic was measured in both sensored and full sensorless mode, Figure 8-6. As can be seen from the results, the torque generation capability of the motor in sensorless mode is almost the same as when the motor is operated in sensored mode. The measured torque error is less than 0.02 Nm across the measured range.

8.2 Conclusion

A sensorless speed control of a permanent magnet synchronous motor using a back-EMF mode observer with an angle tracking observer has been developed. The experimental results show the performance of the proposed algorithm implemented on a DSP controller. The MC56F8023 were presented. The proposed sensorless algorithm was applied to the horizontal belted drive washer and its driving performance for washing and spinning was verified through experiments. It has been proved that the proposed algorithm is suitable for this type of the washing machine.

How to Reach Us:

Home Page:
www.freescale.com

E-mail:
support@freescale.com

USA/Europe or Locations Not Listed:
Freescale Semiconductor
Technical Information Center, CH370
1300 N. Alma School Road
Chandler, Arizona 85224
+1-800-521-6274 or +1-480-768-2130
support@freescale.com

Europe, Middle East, and Africa:
Freescale Halbleiter Deutschland GmbH
Technical Information Center
Schatzbogen 7
81829 Muenchen, Germany
+44 1296 380 456 (English)
+46 8 52200080 (English)
+49 89 92103 559 (German)
+33 1 69 35 48 48 (French)
support@freescale.com

Japan:
Freescale Semiconductor Japan Ltd.
Headquarters
ARCO Tower 15F
1-8-1, Shimo-Meguro, Meguro-ku,
Tokyo 153-0064
Japan
0120 191014 or +81 3 5437 9125
support.japan@freescale.com

Asia/Pacific:
Freescale Semiconductor Hong Kong Ltd.
Technical Information Center
2 Dai King Street
Tai Po Industrial Estate
Tai Po, N.T., Hong Kong
+800 2666 8080
support.asia@freescale.com

For Literature Requests Only:
Freescale Semiconductor Literature Distribution Center
P.O. Box 5405
Denver, Colorado 80217
1-800-441-2447 or 303-675-2140
Fax: 303-675-2150
LDCForFreescaleSemiconductor@hibbertgroup.com