User Guide 08092

IRMCS3043 System Overview/Guide

By

International Rectifier's iMOTION Team

Table of Contents

Introduction

The IRMCS3043 reference design platform supports development of dual motor drive applications in the 1200W – 1700W power range using the IRMCF343 digital control IC and the IRAMX16UP60A IRAM power module. The design kit includes a permanent magnet synchronous motor, a drive control board, a PFC inductor and the motor control development tools. The inclusion of the motor allows immediate use of the kit, while the motor control commissioning tools support rapid evaluation of the drive board running the target motor.

This document provides a high level description of the PFC and motor drive control algorithms, the hardware components and design tools. The *IRMCS3043 Quick Start Guide* provides instructions on how to set up the hardware and to start running the included motors. The *IRMCx300 Application Developers Guide* describes the various steps to configure and customize the design to match your motor and application requirements. The *IRMCx300 Software Developers Guide* describes application software development and the tools to generate code for the one time programmable version of the digital control IC. The *IRMCx300 Reference Manual* provides detailed descriptions of the digital control IC hardware, registers and control blocks.

IRMCF343 Application Circuit

In a typical IRMCF343 application such as an air conditioner compressor control circuit, shown in

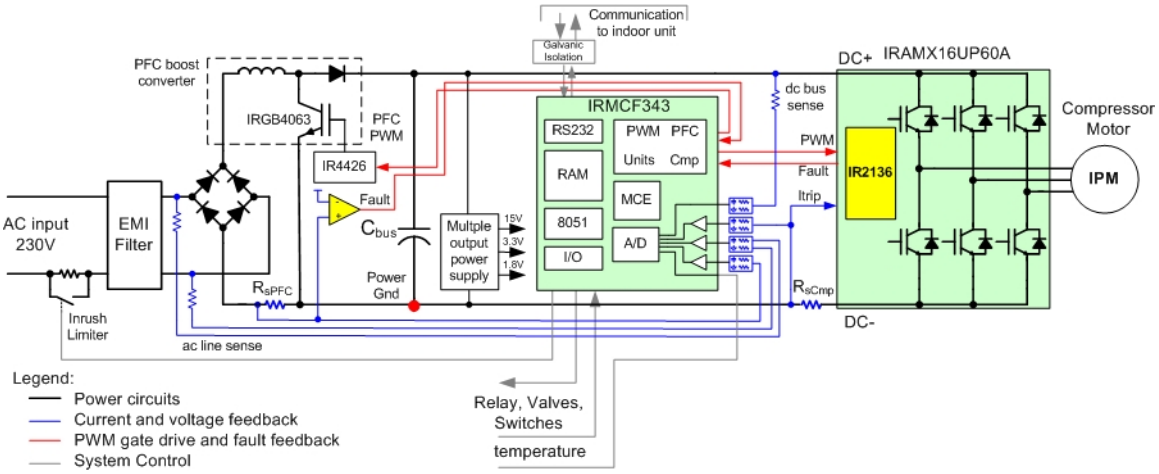

Figure 1, the drive circuit includes an input converter, two three-phase power inverters and the digital motor control IC. The input converter consists of an inrush current limiter, an EMI filter, a bridge rectifier, a PFC boost converter circuit and a dc bus capacitor. The three-phase power inverter uses a single IR appliance module (IRAM) and some passive components. The IRAMX16UP60A module integrates six IGBT transistors and diodes and the analog gate drive circuit. The IRAM module also includes inverter protection functions include over current trips, over temperature trips and the detection of gate drive voltage. The module requires external current sensing resistors. The IRMCF312 digital IC controls both ac motors and the PFC boost converter. It also manages application functions such as the motor speed profiles, load switching and communication with an external processor that handles the user interface. The IC integrates an 8-bit microcontroller (8051 MCU), a 16-bit Motion Control Engine (MCE™) and an Analog Signal Engine (ASE). This IC structure allows easy partitioning of the appliance application and motor control functions as shown in Figure 2.

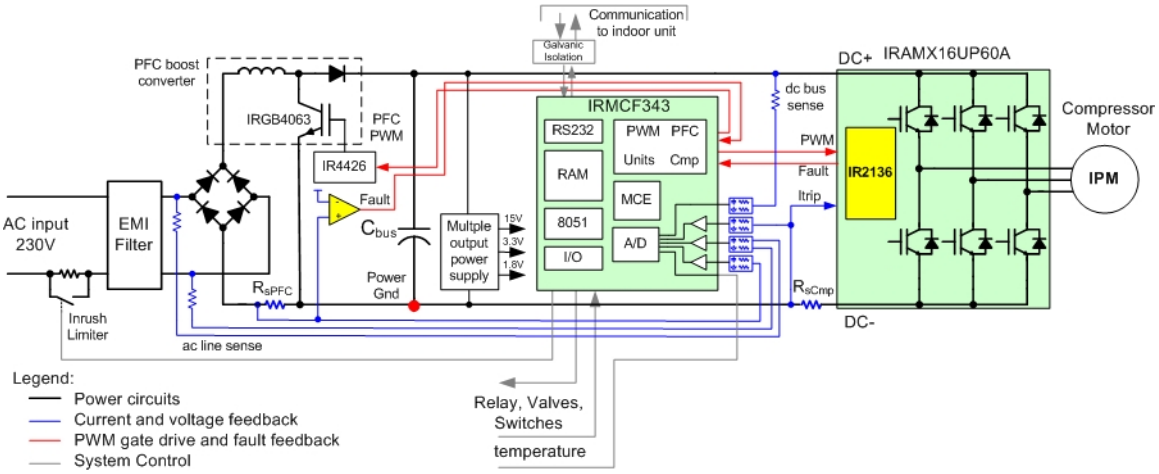

Figure 1 IRMCF343 Application Hardware

The 8051 microcontroller includes the timers, digital I/O and ports to communicate with the indoor unit controller. The 8051 is the master processor and has dedicated data and program memory space that loads automatically after power up from an external serial ROM. It communicates with the MCE via a shared memory interface that allows it to set control algorithm parameters and track variables. There is a version of the digital control IC, the IRMCK343, which integrates OTP ROM in place of the 8051 instruction RAM.

The Motion Control Engine controls both the compressor motor and the input PFC converter. The MCE implement a sensorless field oriented control algorithm that generates the inverter PWM signals based on dc bus voltage and dc link current samples. The MCE implements a digital power factor correction algorithm with feed forward control to minimize the converter switching frequency and maximize efficiency. The Analog Signal Engine integrates the 12-bit A/D converter, timing circuits and buffer amplifiers needed to determine the motor voltages and currents from the dc link shunt resistor. It includes the buffer amplifiers and timing circuits to sample the ac line voltage, the input converter current and the dc bus voltage. The only external components required for the ASE are the passive components needed to set the amplifier gains.

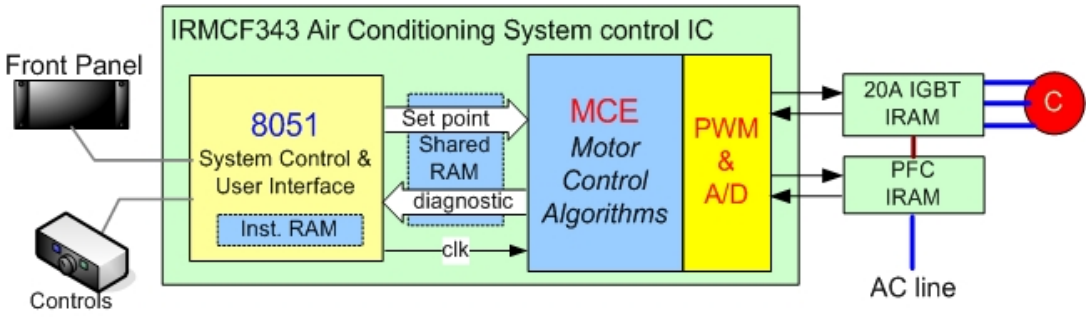

Figure 2 IRMCF343 Application Partitioning

The MCE™ implements the sensorless motor control algorithm using a hardware library of motor control modules such as vector transformations and proportional plus integral feedback compensators. The motion control sequencer schedules the execution of the library components required to implement the sensorless control and PFC algorithms. The MCE implements the algorithm in 16-bit fixed point arithmetic using a selection of macro control functions from the MCE library. The use of dedicated hardware macro functions enables a single fixed point control IC to simultaneously control two motors and the input PFC. A number of these block such as the PI compensator or the vector rotation ($e^{j\theta}$) are common motor control functions. Other blocks such as the rotor angle and speed estimator, are macro blocks that combine a number of motor control functions. There are also hardware interface functions that combine computational elements, timing functions and analog circuits. Each block includes control registers for parameters such as proportional loop gain or the PWM switching frequency. Control variables, gains and set points are stored in the shared RAM to allow the 8051 to change target speed and control system gains on the fly. The 8051 can also log control variables without interrupting the MCE operation. The 8051 initializes the MCE by loading the sequencer code and control parameters into the shared memory. The MCE operates almost completely independent of the 8051 reacting only to changes in the control set point or parameters.

Power Factor Correction Algorithm

The power factor controller forces the input line current waveform to follow that of the input line voltage. This ensures that the input power factor is unity and eliminates the higher order harmonic components usually present in the input current of a bridge rectifier. The minimization of current harmonics is a requirement in many regions and active control of power factor is typically used once the input power exceeds about 1kW. The power factor correction (PFC) loop structure is described in Figure 3 below. The input signals are the dc bus voltage v_{dc} , the input current i_{pfc} and the rectified ac voltage $|v_{ac}|$. The control loop adjusts the duty cycle of the boost converter transistor so that the boost converter input current i_{pf} follows the wave shape of the rectified ac voltage $|v_{ac}|$.

There are three major blocks in the controller: the voltage loop, the current loop and the voltage feed forward loop. The voltage loop calculates the input current magnitude I^* needed to maintain the dc bus voltage v_{dc} at the reference value V_{dc}^*. There is a rate limiter on the dc bus reference input to limit the peak input power after the PFC controller is initialized. The product of the input current magnitude I^* and the input ac voltage $|v_{ac}|$ provides a half wave sinusoidal reference input i_{pfc}^* for the current loop. The rectifier input current must follow this half wave sinusoidal waveform for the input line current to follow the input waveform of the ac line. However, a half wave rectified sine waveform is rich in harmonics and the current controller requires a high bandwidth to track them. The voltage feed forward loop minimizes the current loop bandwidth requirements by calculating the boost converter duty cycle needed to track the input voltage waveform. Therefore, the current loop only needs to respond to dynamic conditions allowing the A/D sample rate and PFC switching frequency to be minimized. In this reference design, the PFC switching frequency is 30 KHz, which is 3 or 4 times lower than would be required without feed forward control. The lower frequency lowers

the MCE processor calculation requirements and lowers the boost converter switching losses.

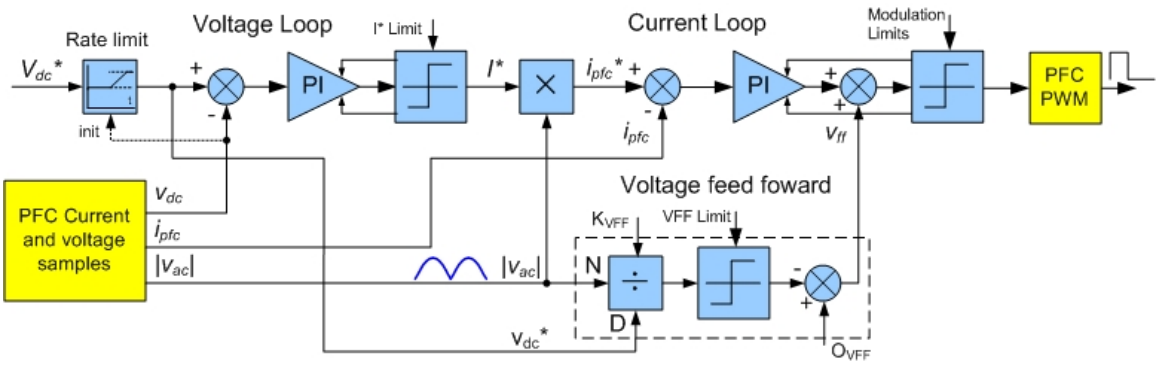

Figure 3 PFC Control Schematic

Sensorless Control Algorithm

The PMSM control algorithm structure is described in Figure 4 below. The main control loops are the velocity and current control loops that vary the motor winding voltages to drive the motor at the target speed. The current controller has a field oriented control (FOC) structure that maximizes motor efficiency. The field weakening and IPM control blocks extend the torque and speed range of the drive. The feedback signal processing functions determine rotor position, velocity, and stator winding currents all from the current flowing in the power inverter dc link. The start-up sequencer controls the transition from the zero and low speed control range to sensorless operation.

Velocity and Current Control

The controller has a cascaded structure with inner current control loops and an outer velocity control loop. The velocity controller calculates the motor torque (T*) required to follow the target velocity (ω*) while the current loops drive the motor currents needed to generate this torque. The proportional plus integral (PI) velocity loop compensator acts on the error between the target velocity (ω*) and the actual (estimated) velocity. The integral term forces the steady state error to zero while the proportional term improves the high frequency response. The PI compensator gains are adjusted depending on the motor and load characteristics to meet the target dynamic performance. The limiting function on the output of the PI compensator prevents integral windup and maintains the motor currents within the motor and drive capability. There is also a minimum speed setting and rate limiter on the target velocity to stay within the system mechanical limits.

The current loops calculate the inverter voltages to drive the motor currents needed to generate the desired torque. As in the case of the dc motor, the current magnitude determines the motor torque. However, in an ac machine the torque is also a function of the phase alignment between the stator currents and rotor flux. A field oriented control structure correctly aligns the ac motor winding currents with the rotor flux position to maximize torque production. Field oriented control (FOC) uses the Clarke transform and

a vector rotation ($e^{-j\theta}$) to transform the motor winding currents into two quasi dc components, an I_d component that reinforces or weakens the rotor field and an I_q component that generates motor torque. Two separate regulators control the I_d and I_q currents and a forward vector rotation ($e^{+j\theta}$) transforms the current loop V_d and V_q output voltages into the two phase ac components. The Space Vector Pulse Width Modulator (SVPWM) generates the three phase power inverter switching signals based on the α and β voltage inputs. Space vector modulation automatically inserts a third harmonic into the line to line voltage that maximizes bus voltage utilization using sinusoidal modulation. This FOC structure simplifies current control loop tuning by eliminating dependency on the motor electrical frequency and reducing the current loop tuning to a first order problem.

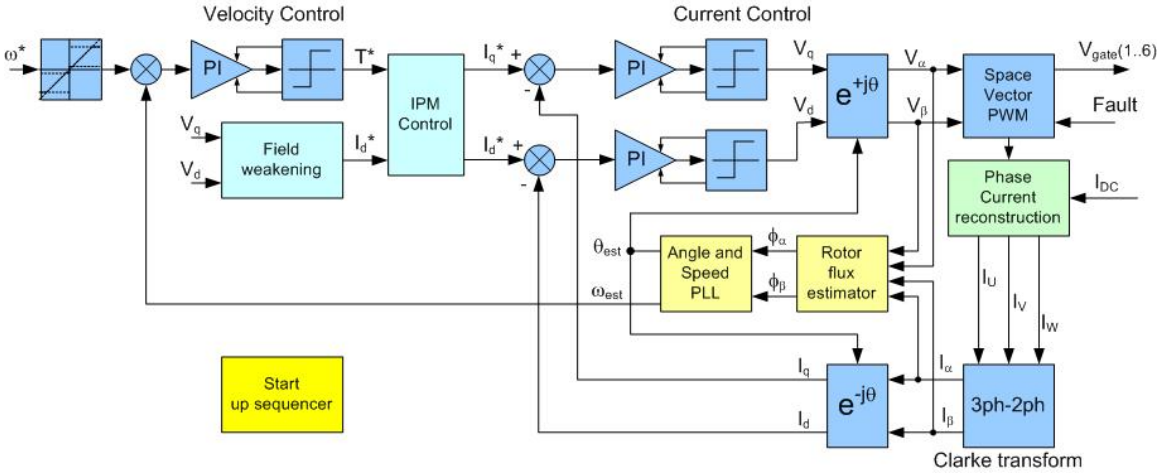

Figure 4 Sensorless Field Oriented Control Algorithm

Field Weakening and IPM control

Typically, the I_q controller input is the torque reference from the velocity controller and the I_d reference current is set to zero. However, above a certain speed, known as the base speed, the inverter output voltage becomes limited by the dc bus voltage. In this situation, the field weakening controller generates a negative I_d to oppose the rotor magnet field that reduces the winding back EMF. This enables operation at higher speeds but at a lower torque output. The controller includes a compensator that adjusts the I_d current to maintain the motor voltage magnitude within the bus voltage limit. The compensator operates on the square of the voltage magnitude since this is easy to calculate from the sum of $(V_d)^2$ and $(V_q)^2$.

When driving an interior permanent magnet (IPM) motor the rotor saliency can generate a reluctance torque component to augment the torque produced by the rotor magnet. The motor torque function, in equation below, has a cylindrical torque term that is a function of I_q and a reluctance torque term that is a function of both I_d and I_q.

$$Torque = \frac{p}{2}\left(\Psi_{magnet}.I_q + \left(L_d - L_q\right)I_d.I_q\right) \qquad \textbf{Equation 1}$$

where,
p is the number of poles
ψ_{magnet} is the permanent magnet linked flux
L_d and L_q are the direct and quadrature axis inductances

When driving a surface magnet motor, there is zero saliency (L_d=L_q) and I_d is set to zero for maximum efficiency. In the case of IPM motor which has saliency (L_d< L_q) a negative I_d will produce positive reluctance torque. The most efficient operating point is when the total torque is maximized for a given current magnitude. This is found by transforming Equation 1 into a form with current magnitude (I_m) and phase advance (β) terms by substituting I_q with I_m.cos(β) and I_d with I_m.sin(β).

$$Torque = \frac{p}{2}\left(\Psi_{magnet}.I_m.\cos(\beta) + \tfrac{1}{2}\left(L_d - L_q\right)(I_m)^2.\sin(2\beta)\right) \qquad \textbf{Equation 2}$$

The significance of the reluctance torque increases and so the phase advance is increased with increasing current. The plot of the total torque as a function of angle and current in Figure 5, shows that the optimum angle advance can be approximated by a linear function. The IPM control block uses the optimum phase advance at rated current to define the linear approximation.

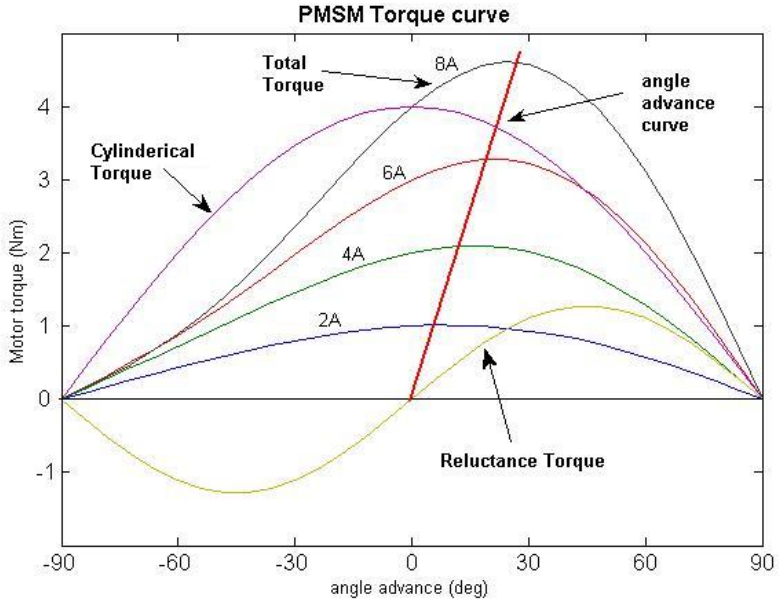

Figure 5 Motor torque as a function of current and angle advance

Position and velocity estimation

In a PMSM, the rotor flux is locked to the rotor position and so it can be measured directly using a shaft mounted sensors. It can also be measured indirectly from the motor back EMF since this is a function of the rotor position and speed. The position estimation algorithm, described in Figure 6, has two stages. The rotor flux function is first derived from a circuit model and then a phase locked loop estimates the flux angle and frequency.

The two-phase stator circuit model described by Equation 3 forms the basis for the flux estimation. The two phase stator voltages are inputs to the SVPWM function and the two phase current measurements are in the current feedback path. Integration of the equation followed by simple manipulation yields the terms that are sine and cosine functions of the rotor flux angle. In the hardware implementation, the voltage integrator includes has a low frequency cut off to prevent dc saturation.

$$v_\alpha = R_S.i_\alpha + L_S.\frac{di_\alpha}{dt} + \frac{d}{dt}\left(-\psi_r.\cos(\theta_r)\right)$$

$$v_\beta = R_S.i_\beta + L_S.\frac{di\beta}{dt} + \frac{d}{dt}\left(-\psi_r.\sin(\theta_r)\right)$$

Equation 3

The angle and frequency phase locked loop (PLL) estimates the flux angle and speed from the rotor sine and cosine flux functions. The vector rotation calculates the error (ε) between the rotor flux angle (θ_r) and the estimated angle (θ_{est}). The PI compensator and integrator in the closed loop path force angle and frequency estimate (ω_{est}) to track the angle and frequency of the rotor flux. This second order feedback loop has zero error when the rotor flux is changing at a constant frequency. The motor velocity is derived from the rotor frequency according to the number of number of rotor poles. The PLL startup function supports the motor startup sequencer. At low speeds, the back EMF signal is unreliable and so the PLL is driven in open loop at fixed frequency ramp (α_{start}). Once the frequency reaches a minimum threshold (ω_{thr}), the PLL takes its inputs from the flux estimator.

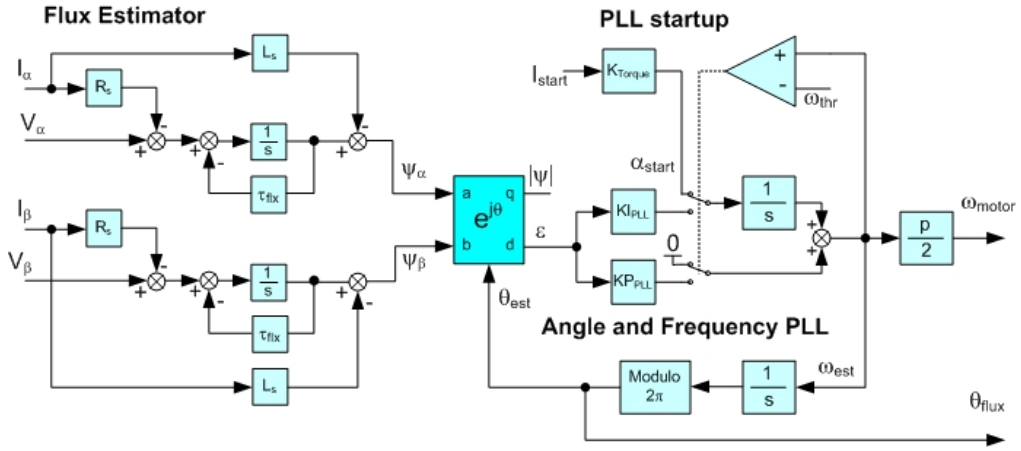

Figure 6 Rotor flux estimator and rotor angle and speed PLL.

Start up sequencing

A special mode of operation is required for starting because the motor back EMF is swamped by circuit noise when the speed is close to zero. There are two modes in the start up sequence as described in Figure 7 below. In the first mode, known as parking, the controller applies dc current to the motor coils to align the rotor at a known rotor electrical angle. In the startup mode, the controller drives the motor at a constant current magnitude (I_{start}) and the rotor angle and frequency PLL runs in open loop with a fixed frequency ramp. The controller switches over to full closed loop control when the PLL reaches the switch over frequency threshold. In the closed loop mode, the motor current is driven by the velocity and current loops and reacts to changes in the command velocity input.

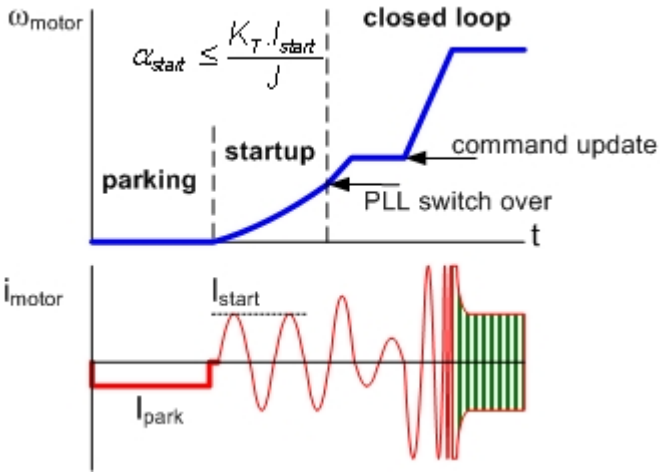

Figure 7 Starting sequence

In the ideal setup, the PLL frequency ramp matches the motor starting acceleration, which is determined from the starting torque and the mechanical system inertia. However, when there is load friction the starting acceleration will be less than the ideal value. Setting the PLL frequency ramp lower than the ideal value compensates for the load friction. This system is self correcting since if the actual starting acceleration is higher than the frequency ramp then the rotor will advance in phase causing the generated torque to drop until the acceleration rates match.

The parking process is described in Figure 8. The controller drives the stator windings at a constant current to set the stator field at a fixed angle. Any arbitrary parking angle can be selected by driving the appropriate combination of u,v and w winding currents. The rotor tends to align itself with stator field regardless of the initial position. This defines the initial values for the angle and frequency PLL to maximize the starting torque. When driving a static load, the rotor will not be correctly aligned after parking. However, the system can tolerate static loads of up to 50% of the rated torque. For example, if the static load is 50% of the parking torque then the alignment error will be 30^o electrical ($\cos(30^o)=0.5$). In this case, the starting torque will be 87% ($\cos(60^o)=0.866$) of the maximum value. There is a possibility of a start failure when driving static loads if the

initial rotor position is almost completely misaligned with the parking angle. In this case, the initial alignment torque can be lower than the static load and the rotor will not move. This problem is overcome by adding a second parking stage with a parking angle shifted by approximately 60° to move away from the misaligned position.

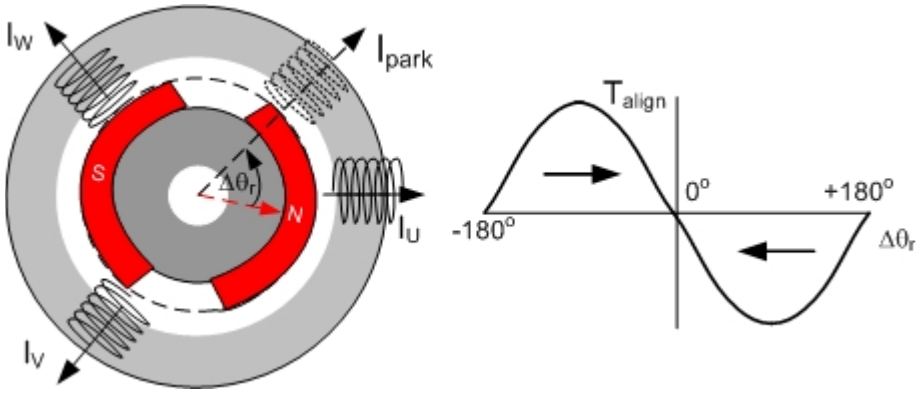

Figure 8 Parking

Phase Current Measurement

Direct measurement of the motor winding current requires isolation circuits to handle the high common mode voltage and switching frequency at the motor windings. The phase current reconstruction circuit avoids the isolation requirement by measuring current in the dc link. The motor winding currents are measured by synchronizing the sampling of the current in the dc link shunt with the power inverter switching. In every PWM cycle, there are two active states where the motor windings are connected between the two dc bus rails. In each active state, one dc bus rail connects to a single motor winding and the other bus rail connects to the other two motor windings. The motor flowing from the dc bus rail flows through one winding and returns from the remaining two windings via the dc link shunt. The current sampled in the dc link shunt during this period is equal to the current in the single motor winding. A second winding current is sampled during the second active PWM state. The third winding current is calculated from the sum of the first two currents since they all must sum to zero.

This can be seen by examining the current flow in the power circuit in Figure 9 as it relates to the state of the power inverter switches. The first inverter state is a zero vector state where all windings are shorted to the lower dc bus rail. The second state is an active state where the U phase is connected to the positive rail and the V and W phases are connected to the negative rail. The third inverter state is also an active state but now only the W phase is connected to the negative rail. The fourth inverter state is a zero vector state where the windings are shorted to the positive rail. The second half of the PWM cycle is a mirror image of the first half of the cycle. In this complete PWM cycle, there are two states when the dc link current equals the U phase current and two states when the dc link current equals the negative W phase current.

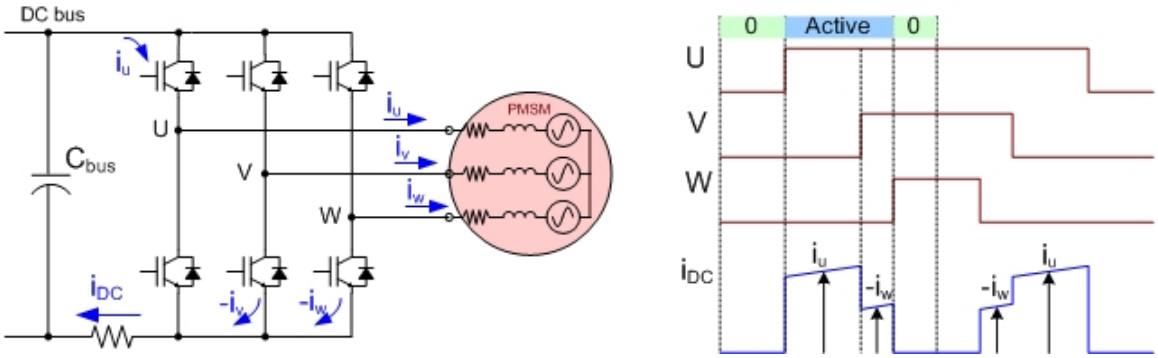

Figure 9 Phase current reconstruction

The Space vector modulator generates the PWM switching signals and the dc link current sample timing signals. The current reconstruction circuits include the A/D converter and the analog amplifier to bring the current shunt signal within the range of the converter. Successful implementation requires careful circuit board layout and fine tuning of the sample timing to avoid the significant circuit noise generated by the power device switching. This topic is covered in detail in the *IRMCx300 Application Developers Guide*.

Control Algorithm Implementation and tuning

The PMSM control algorithm described in Figure 4 is implemented by the MCE with the control variables scaled within the 16 bit fixed point data range. The control schematic in Figure 10 provides further details on the PMSM controller implementation and some of the important control parameters. A key feature of the design is the scaling of variables to avoid data overflow or excessive rounding. Hardware limitations such as the A/D converter resolution and SVPWM resolution set the data range for current samples and voltage outputs. All frequency dependant parameters such as integral gain and filter time constants need to be defined as a function of the PWM frequency that sets the sample rate. Some control blocks modify the scaling between different parts of the algorithm to maximize dynamic range. For example, the current and velocity loop controllers scale current so that 4095 represents rated drive current while the current sampling and vector rotation block must scale current where 2047 represents the maximum controllable drive current. Control registers on each of the blocks allow gain setting or feature selection for each of the blocks. Registers and input variables can be modified at any time but typically, only the target speed, drive limits and compensator gains are adjusted while the motor is running. A full description of each control block is in the *IRMCx300 Reference Manual*.

Drive system commissioning involves the calculation of controller parameters to match the hardware configuration, motor characteristics and drive performance specifications. Control loop gain is a function of the compensator gain and the gain of all elements in the

loop such the power inverter gain and the current feedback gain. Control loop tuning and drive commissioning is supported by the *MCEWizard* and *MCEDesigner*. *MCEWizard* is an interactive design tool that calculates control IC parameters in digital counts based on the system specifications expressed in engineering units. It also embeds pole-zero cancellation design rules for PI loop tuning. *MCEDesigner* is the drive evaluation software that communicates with the digital IC and allows on the fly tuning of drive parameters. The *MCEWizard* and *MCEDesigner* software supports both motor control loop tuning and configuration of the PFC controller. Further details on controller implementation and tuning are described in detail in the *IRMCx300 Application Developers Guide*.

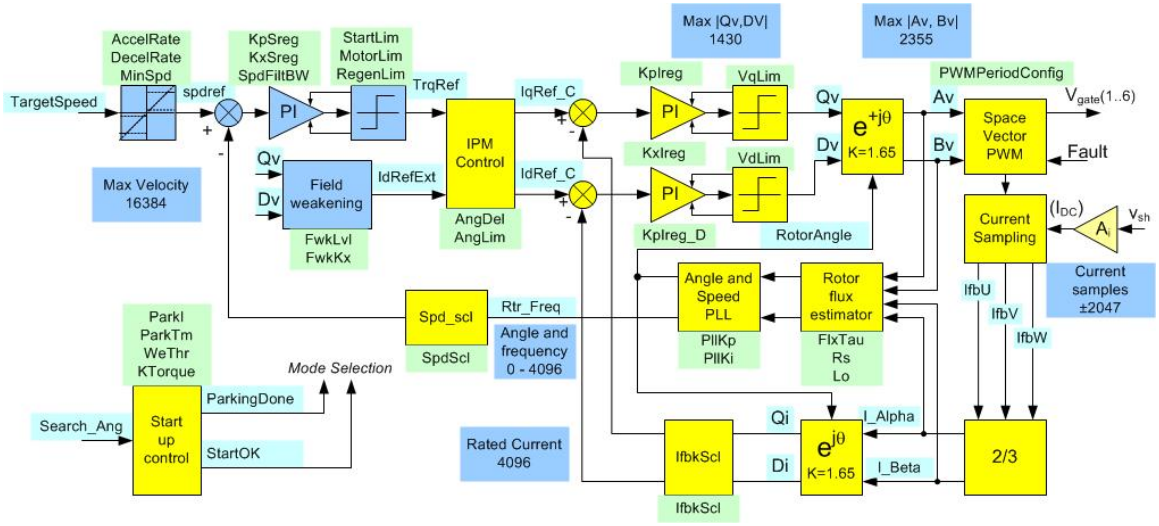

Figure 10 Control Algorithm Implementation

IRMCS3043 Reference Design Tools and Application Development

The IRMCS3043 reference design kit includes a permanent magnet synchronous motor, a PFC inductor, a drive control board, the motor control development tools and the full set of design documentation. The supplied motor control development tools enable drive commissioning, performance evaluation and algorithm customization. The supplied design files include 8051 firmware, drive configuration files, algorithm schematics and circuit board design files. The recommended 8051 application software tools need to be purchased from third party vendors.

Figure 11 describes the major tools components needed to develop the complete drive application. This process includes demonstrating the technology, evaluating the drive performance in the end application, developing appliance control software and designing of the final hardware. The reference design board serves as the initial evaluation platform and drive development tool. The MCE installation software loads the motor control design tools, design files and user documentation on to your PC. The boot EEPROM on the board is loaded with the *MCEDesigner* agent so the control IC is able to communicate

with the *MCEDesigner* software running on the PC when the board is first powered up. The serial connection between the PC and the control IC is isolated so the drive control board can be connected directly to the ac line. Configuration files are included for the motor supplied with the reference design so the motor can be run right away. Details on the set up and safe use of the hardware are described in the *IRMCS3043 Quick Start Guide*.

The MCEWizard drive commissioning tool enables customization of motor drive parameters to match motor and system specifications. It asks a series of questions, supported by explanation and graphics, to determine the motor characteristics and system specifications. The tool checks the data for consistency and generates the full set of digital control parameters. The MCEDesigner tool imports the control parameters and downloads them to the control IC over the serial link. This tool also allows the generation of user defined speed and application control profiles to exercise the drive in the end application. The MCEDesigner plotting function allows tracing of control system variables to support system performance evaluation.

The MCEWizard supports customization of the drive circuit board including changes to the input voltage, power stage, current feedback and gate drive circuits. Circuit board layout can have a significant effect on current feedback circuit performance so recommendations in the *IRMCx300 Application Developers Guide* should be carefully followed.

Customizing the drive algorithm requires the Matlab™ Simulink tool to edit the control algorithm schematic. A web based MCECompiler tool generates the MCE algorithm executable files that define the user algorithm along with a register map. The MCEDesigner tool downloads this file to the boot EEPROM on the reference design kit. The register map enables the generation of MCEDesigner user functions to access any new control registers. Changes to the reference design blocks supplied may invalidate parameters calculated by the MCEWizard.

Application software development requires the Keil™ µVision3 C and assembly tools and the FS2™ JTAG interface to download the code to the control IC. The JTAG interface to the control IC is also isolated so the FS2™ pod can be safely connected to the board when powered from the ac line. The *IRMCx300 Software Developers Guide* describes how to configure these tools for use with the controller along with the sample application code supplied with the design kit. When the user 8051 application code is complete, the *MCEProgrammer* tool generates binary files to program the boot EEPROM or internal OTP for the final application board.

Next Steps

At this stage, you should have some idea on how the digital control IC operates and how the reference design kit can help you develop your own drive system. The next step is to go to the *IRMCS3043 Quick Start Guide* to start running the motor supplied with your kit and become familiar with the *MCEDesigner* drive evaluation software. Run *MCEWizard* to see how you customize control parameters to match your own motor. The *IRMCx300*

Application Developers Guide describes the commissioning steps in detail including tips on how to test motor parameters. *MCEDesigner* allows you to create your own speed profiles so you can test your motor in your target application. You can use the *MCEDesigner* plotting tool to evaluate the performance of the drive and to help you tune control parameters. Once you are ready to develop your own drive, you will need to purchase third party 8051 tools. The *IRMCx300 Software Developers Guide* describes the sample code supplied and explains how to configure the tools to work with control IC. The reference design schematics and PCB layout are supplied to support your hardware development but also refer to the hardware design section of the *IRMCx300 Application Developers Guide.* You may need to refer to the *IRMCx300 Reference Manual* from time to time as this is the source of all information on the control IC but for most of the time, the *Developers* guides will lead you through your design. Good luck with your project!

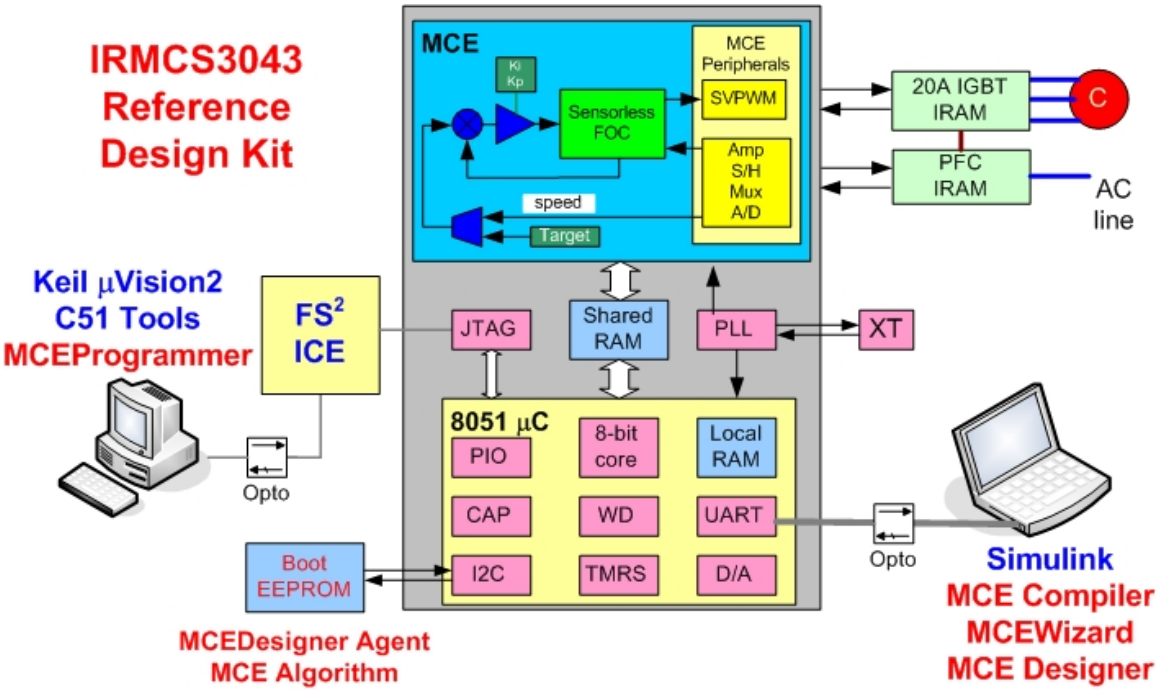

Figure 11 IRMCS3043 Reference Design Kit