Sensorless-FOC With Flux-Weakening and MTPA for IPMSM Motor Drives

Steve Lim

ABSTRACT

The interior permanent-magnet synchronous motor (IPMSM) for variable speed applications became popular with the sensorless Field Oriented Control (FOC) technique in industrial and automotive systems because of high power density, high efficiency and fast dynamic performance. This application report describes how to implement a flux-weakening logic and maximum torque per ampere (MTPA) logic in an InstaSPIN-FOC™ sensorless FOC solution based on the MotorWare™ software platform.

Contents

List of Figures

List of Tables

Trademarks

InstaSPIN-FOC, MotorWare, FAST are trademarks of Texas Instruments.
All other trademarks are the property of their respective owners.

1   Introduction

This document shows how to implement MTPA and Flux-Weakening algorithm for IPMSM with InstaSPIN-FOC, which is TI's high-performance sensorless FOC solution. MTPA logic is necessary to maximize efficiency of interior permanent magnet (IPM) synchronous motors that have high saliency. MotorWare is the software development package for InstaSPIN-FOC solution, but the standard example projects in MotorWare are not suitable for IPMSM that have high saliency. Also, the flux-weakening logic included in MotorWare should be revised when used with MTPA logic.

2   Fundamental Theories of IPMSM Motor Driver

2.1   *Basic Theory of FOC*

The torque in the synchronous machine is a vector cross product of the stator field or current vector and rotor magnetic field vector as shown in Equation 1 and Equation 2.

$$T_e = K\lambda_{rotor} \times I_{stator} \tag{1}$$

$$\left|T_e\right| = K\left|\vec{\lambda}_r\right|\left|\vec{I}_s\right|\sin\theta \tag{2}$$

The K is constant and the amplitude of torque is proportion to the amplitude of rotor flux vector (λ), the amplitude of stator current vector (I) and sin θ, which is angled between rotor flux and the stator current vector. This expression shows that stator current vector and rotor magnetic field vector should be orthogonal to get maximum torque with given stator current. The rotor position must be known to keep the angle at 90°. This can be achieved with a position sensor like encoder or resolver. Also, rotor position can be estimated by using sensorless technologies like InstaSPIN-FOC without these kinds of position sensors.

In brief, the goal of FOC is to align the stator flux to 90° of the rotor flux, that is, orthogonal to the rotor flux. To do this, a three-phase stator current is measured by shunts or in-line magnetic based current sensors. These measurements feed the Clarke transformation module. The outputs of this projection are designated i_{α_in} and i_{β_in}. These two components of the current are the inputs of the Park transformation that gives the current in the d,q rotating reference frame. The i_d and i_q components are compared to the references i_{d_ref} (the flux reference) and i_{q_ref} (the torque reference). At this point, this control structure shows an interesting advantage: it can be used to control either synchronous or asynchronous alternating current induction machines (ACIM) by simply changing the flux reference and obtaining rotor flux position. For the synchronous permanent magnet motor, the rotor flux is fixed and determined by the magnets. Hence, when controlling a PMSM, i_{d_ref} should be set to zero except when in flux-weakening control mode. On the other hand, ACIM motors need id current to create a rotor flux, so the flux reference must not be zero. This conveniently solves one of the major drawbacks of the "classic" control structures: the portability from asynchronous to synchronous drives. The torque command i_{q_ref} could be the output of the speed regulator when using a speed controller. The outputs of the current regulators are V_d and V_q; they are applied to the inverse Park transformation. The outputs of this projection are V_{α_out} and V_{β_out} which are the components of the stator vector voltage in the (α,β) stationary orthogonal reference frame. These are the inputs of the Space Vector PWM. The outputs of this SVM block are the signals that drive the inverter. Note that both Park and inverse Park transformations need the rotor flux position. So, knowledge of the rotor flux position is the core of the FOC.
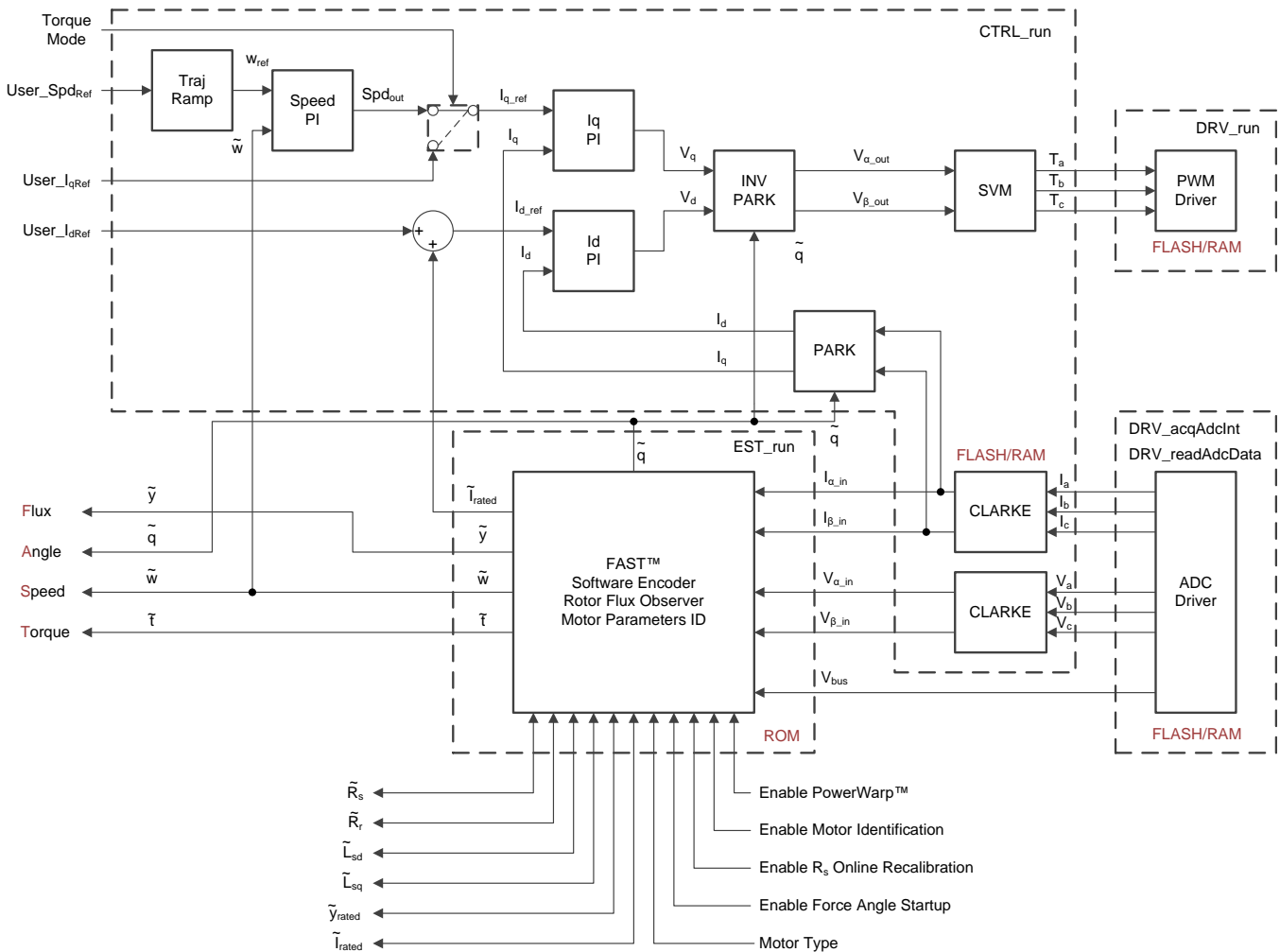
Figure 1. Block Diagram of FOC Algorithm With FAST™ Observer

2.2 *Maximum Torque Per Ampere (MTPA)*

IPMSM has a permanent-magnet inside the rotor that generates magnetic saliency. This saliency produces a reluctance torque from the difference between the direct-d axis inductance and the quadrature q-axis inductance. The maximum efficiency of the IPM motors can be achieved by proper selection of the current vector ratio between magnetic torque current and reluctance torque current in the total current. The torque on IPMSM can be expressed as shown in Equation 3. From this equation, the Id current should be a negative value to achieve positive reluctance torque because direct inductance (L_d) is larger than quadrature inductance (L_q). PP is the number of pole pairs.

$$T_e = 1.5PP\left(\varnothing_f i_q + \left(L_d - L_q\right)i_d i_q\right)$$

(3)

Also, i_d and i_q current vectors are can be expressed with phase term like that shown in Figure 2.

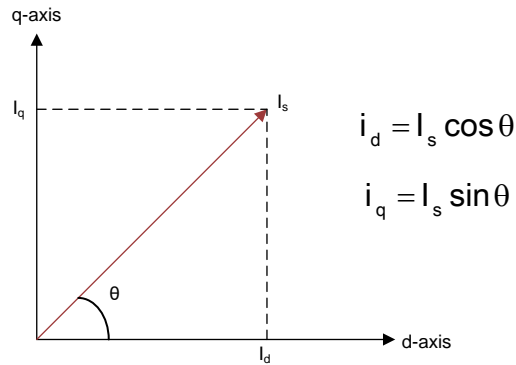$$i_d = I_s \cos\theta$$

$$i_q = I_s \sin\theta$$

Figure 2. Current Vector on d,q Rotating Reference Frame

The above torque output can be revised with the updated current vectors.

$$T_e = 1.5PP\left(\varnothing_f i_s \sin\theta + 0.5\left(L_d - L_q\right)I_s^2 \sin 2\theta\right) \tag{4}$$

The total torque (magnetic torque + reluctance torque) is positively maximized at 90° < θ < 135° and negatively maximized at 225° < θ <270°, that is, when the maximum positive torque per current is required, i_d current should be negative value that is located at 90° < θ < 135° from the d-axis, in terms of beta angle, 0° < β < 45° as shown in Figure 3. There are so many rotor flux angles that can meet the commanded torque. The MTPA concept is to get the minimum current space vector for the constant torque like red trajectory on Figure 3.
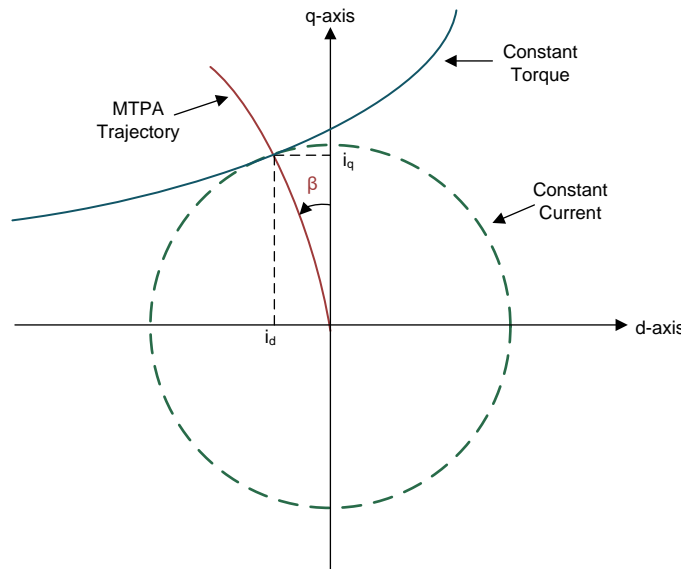

Figure 3. Maximum Torque per Ampere Trajectory

Once the current command I_s is generated from the speed controller, the value of the current components (i_{d_ref} and i_{q_ref}) satisfying that constraint is:

$$i_{d\_ref} = \frac{\varnothing_f - \sqrt{\varnothing_f^2 + 8\left(L_q - L_d\right)^2 I_s^2}}{4\left(L_q - L_d\right)} \tag{5}$$

$$i_{q\_ref} = \text{sign}\left(I_s\right)\sqrt{I_s^2 - I_{d\_ref}^2} \tag{6}$$

$$\text{sign}\left(I_s\right) = 1 \text{ if } I_s \geq 0$$

$$\text{sign}\left(I_s\right) = -1 \text{ if } I_s < 0 \tag{7}$$

Figure 4 shows a block diagram of speed controller incorporating MTPA trajectory generation. The output of the speed controller is a current command I_s^* and feeds the MTPA block to calculate both current references (i_{d_ref}, i_{q_ref}) in the d,q rotating reference frame to get the optimal current vectors.
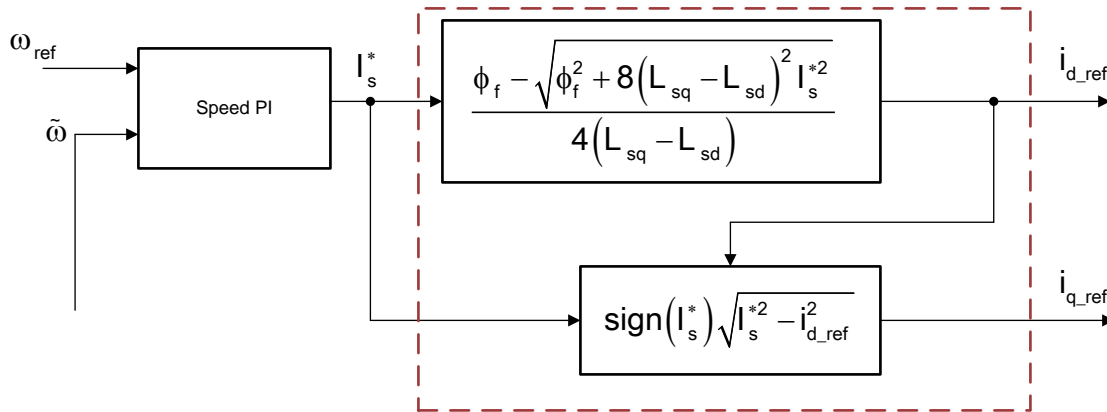

Figure 4. Speed Controller With MTPA Controller

2.3 Flux-Weakening Control for High Speed Operation

IPMSM has a mechanically robust rotor structure that is effective in flux-weakening operation because it has low effective air gap. So, IPMSM can be operated not only in the constant torque region under the base speed but also in the constant power region over the base speed, which is normally rated speed, but it can be varied according to current and voltage limitation. IPMSM is operated in constant torque region with MTPA, so all currents (i_d, i_q) contribute to torque generation in this region and torque is also constant if the motor current is constant. In speed control mode, IPMSM can generate max torque within total current limit up to base speed.
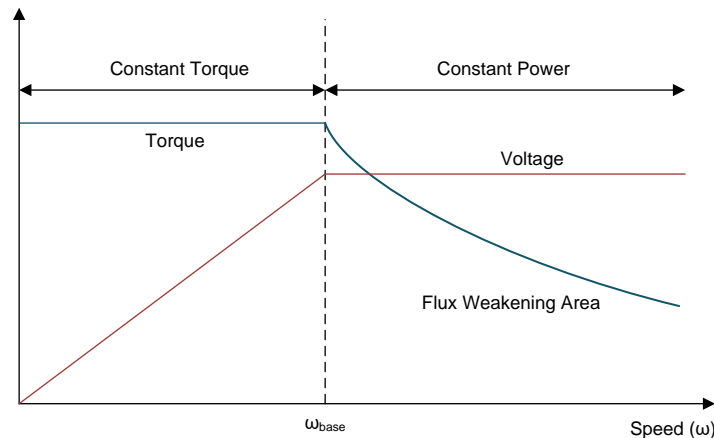

Figure 5. Speed Range on IPMSM

Over the base speed, d-axis and q-axis output voltage is limited by the max voltage that the inverter can output based on the DC bus voltage, as shown in Equation 8. Generally, V_{smax} is $V_{dc}/\sqrt{3}$ based on voltage space vector.

$$v_{d\_ref}^2 + v_{q\_ref}^2 \leq v_{smax}^2 \tag{8}$$

Also, the maximum current limit is determined by the inverter and the motor's current max peak rating as shown in Equation 9.

$$i_{d\_ref}^2 + i_{q\_ref}^2 \leq i_{s\,max}^2 \tag{9}$$

In this region, flux-weakening needs to increase motor speed, and flux-weakening region is determined by both of voltage limit and current limits.

Many flux-weakening strategies have been proposed for optimal design to minimize phase current magnitude. This solution implements a voltage feedback controller that is running on the synchronous frame, and d-axis current is controlled based on the difference between V_{s_ref} and feedback voltage. The current vector reference coming from MTPA calculation is modified by applying the output of voltage amplitude of feedback regulator.
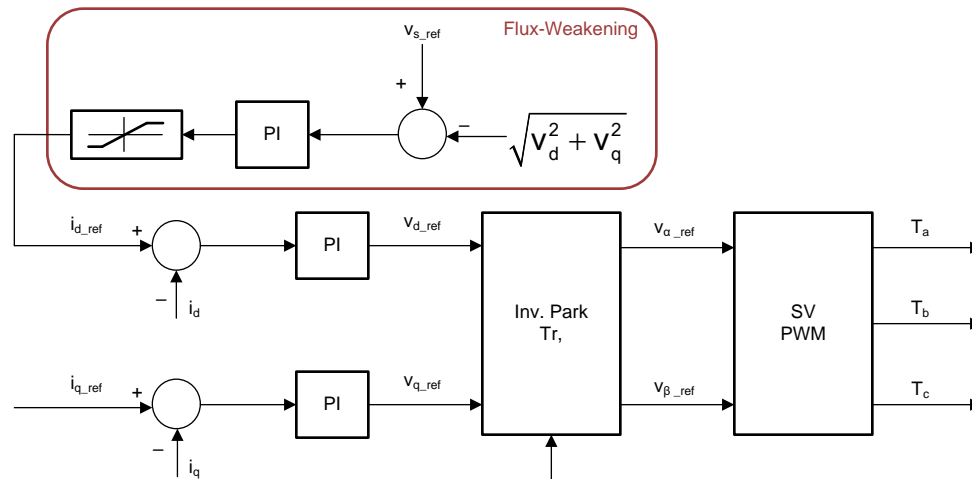

Figure 6. Flux-Weakening Block Diagram

The V_{s_ref} is limited to $V_{dc}/\sqrt{3}$ in this voltage feedback control method, but this algorithm is very effective in real applications because the implementation of this logic is relatively simple and robust in most of the operation range.

3 Key Considerations for Using InstaSPIN-FOC Solution

3.1 Hardware

InstaSPIN-FOC depends on motor phase currents and phase voltage feedbacks for rotor angle and speed estimation in FAST observer. Hence, these feedback signals should be designed very carefully to achieve the desired sensorless performance. For more details about managing motor signals, see the *InstaSPIN-FOC™ and InstaSPIN-MOTION™ User's Guide*.
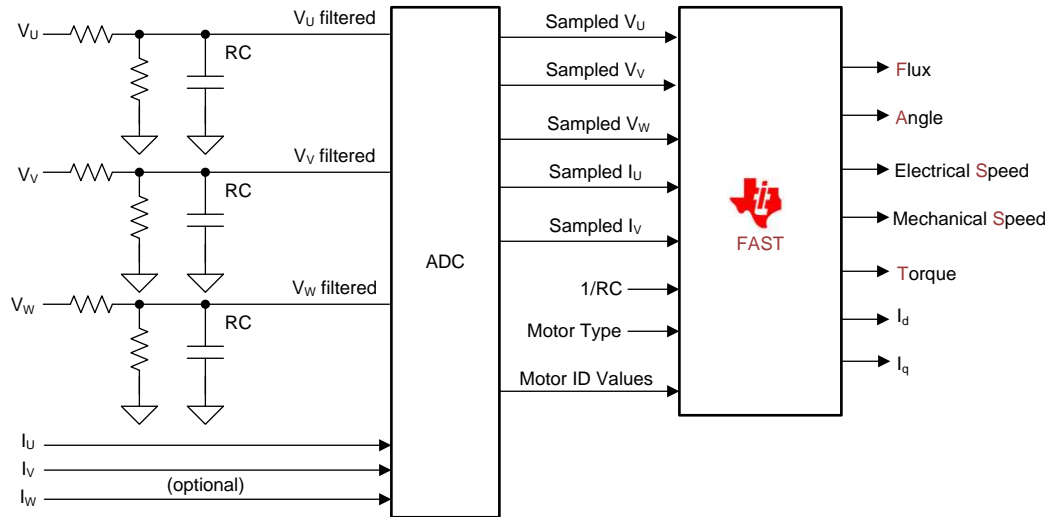

Figure 7. Hardware Interface for InstaSPIN-FOC Solution

However, many PCB layouts get overlooked related with feedback signals that are very sensitive to sensorless performance. Figure 8 shows how to layout these feedback signals. Both grounds of current feedback circuit and voltage feedback circuit should be located into signal ground not to power ground. In other words, G1 and G2 should be connected to signal ground plane. If not, the feedback signals might considerably capture the switching noise. In worst case, the sensorless observer may not work correctly.

Also, current feedback signals should be connected to shunt resistors with kelvin connection to remove a common mode noise.
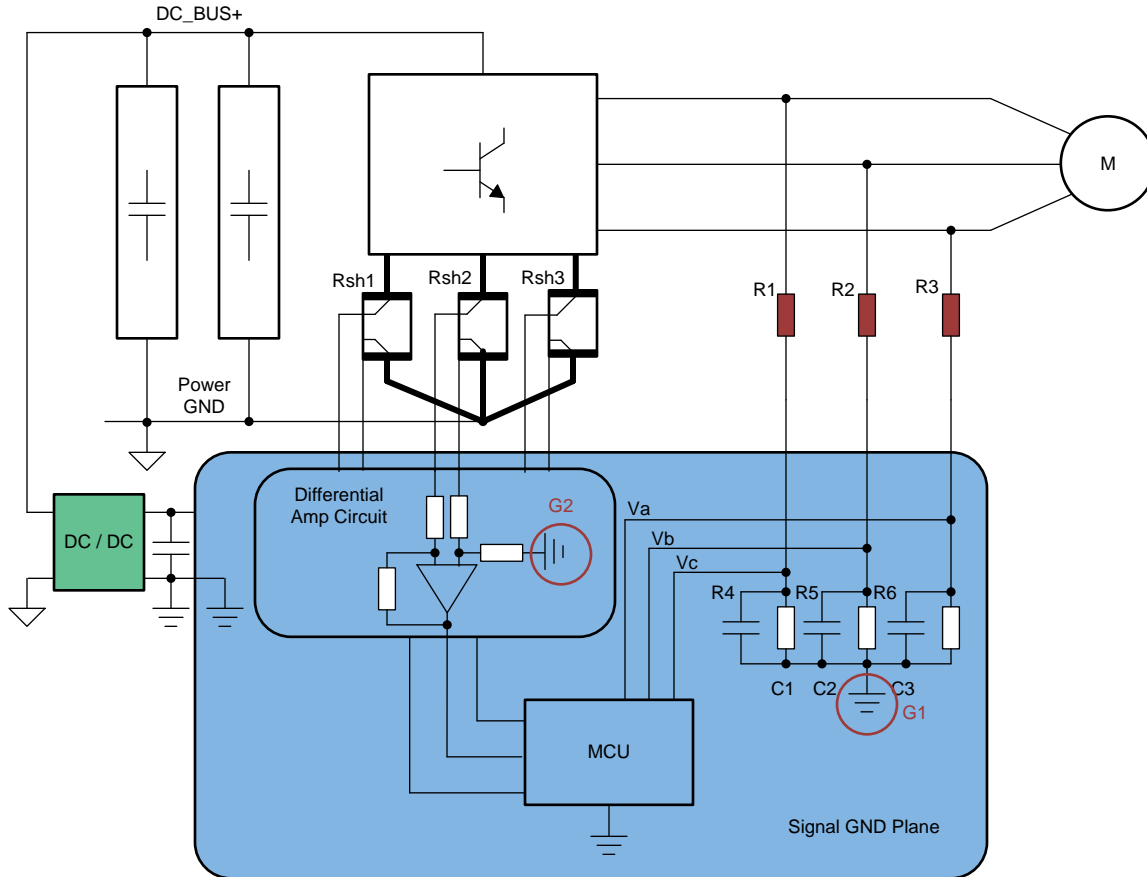

**Figure 8. PCB Layout Guideline for InstaSPIN-FOC Solution**

3.2 *Software*

InstaSPIN-FOC contains many public parameters in user.h files for controller and estimator modules. For more details about the parameters, see the *InstaSPIN-FOC™ and InstaSPIN-MOTION™ User's Guide*. There are three key parameters related with the user's hardware circuit as discussed in the following sections. These parameters should be rearranged according to the user's hardware circuit.

3.2.1 USER_ADC_FULL_SCALE_VOLTAGE_V

This module defines the maximum voltage at the input to the AD converter. The value that will be represented by the maximum ADC input (3.3 V) and conversion (0FFFh). Hardware dependent, this should be based on the voltage sensing and scaling to the ADC input.
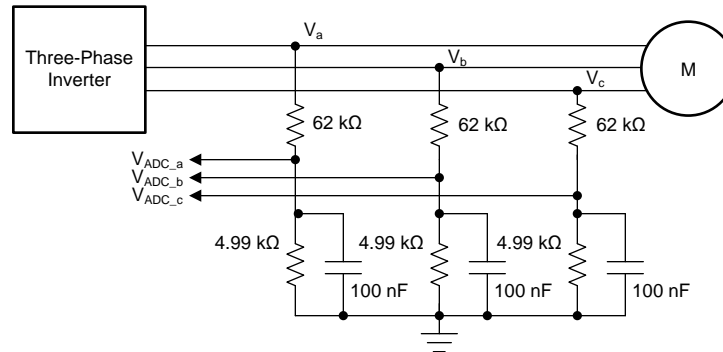
Figure 9. Example Circuit of Phase Voltage Feedback

For example, the voltage feedback from motor phases are as shown in Figure 9.

The maximum phase voltage feedback measurable by the microcontroller can be calculated as shown in Equation 10, considering the maximum voltage for the ADC input is 3.3 V.

$$v_a^{max} = v_{ADC\_a}^{max} \times \frac{\left(4.99\ k\Omega + 62\ k\Omega\right)}{4.99\ k\Omega} = 3.3\ V \times \frac{\left(4.99\ k\Omega + 62\ k\Omega\right)}{4.99\ k\Omega} = 44.3 \tag{10}$$

Hence, the following setting is done in user.h:

```
#define USER_ADC_FULL_SCALE_VOLTAGE_V        (44.3)
```

3.2.2 USER_VOLTAGE_FILTER_POLE_Hz

The FAST estimator needs the motor phase voltage feedbacks to get the motor input voltages. Because the command voltage is not accurate, especially at low speed, due to the dead-band and inverter switch's voltage drop, the InstaSPIN-FOC directly measures the phase voltages to minimize the voltage feedback error. The low-pass filters are required to get the sine waveform from PWM waveform of inverter output voltages. This low-pass filter's cutoff frequency information feeds to the FAST estimator to allow an accurate detection of the motor phase voltage feedback from filtered signals.

As a general guideline, a cutoff frequency of a few hundred Hz is enough to filter out a PWM frequency of 10 to 20 kHz.

In the above example, the filter pole setting can be calculated as shown in Equation 11.

$$F_{filter\_pole} = \frac{1}{2\pi \times R_{parallel} \times C} = \frac{1}{2\pi \times \left(\dfrac{62\ k\Omega \times 4.99\ k\Omega}{62\ k\Omega + 4.99\ k\Omega}\right) \times 100\ nF} = 344.62\ Hz \tag{11}$$

The following code example shows how this is defined:

```
#define USER_VOLTAGE_FILTER_POLE_Hz      (344.62)
```

3.2.3 USER_ADC_FULL_SCALE_CURRENT_A

This parameter defines the maximum current at the AD converter. This value is represented by the maximum ADC input (3.3 V) and conversion (0FFFh). The value is hardware dependent and should be based on the current sensing and scaling to the ADC input.
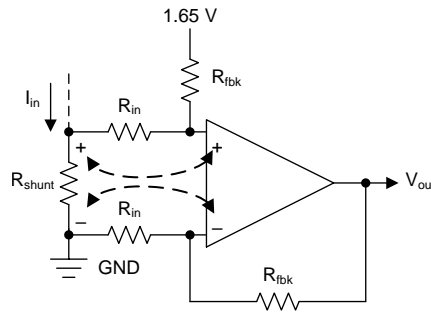

Figure 10. Example Circuit of Current Feedback

For example, if shunt resistance (Rshunt) is 0.01Ω, input resistance (Rin) is 1kΩ, and feedback resistance (Rfdk) is 16.5kΩ, input peak current can be calculated as shown in Equation 12.

$$I_{in} = \frac{V_{out}}{R_{shunt} \times R_{fdk} / R_{in}} - 1.65 \text{ V} = \frac{3.3 \text{ V}}{0.01 \, \Omega \times \dfrac{16.5 \text{ k}\Omega}{1 \text{ k}\Omega}} - 1.65 \text{ V} = 10 \text{ A}$$

(12)

As shown in this example, the maximum peak to peak current measureable by the microcontroller is 20A, which is the peak to peak value of ±10A. So, the following code should be defined in user.h:

```
#define USER_ADC_FULL_SCALE_CURRENT_A      (20.0)
```

4 How to Implement MTPA and Flux-Weakening With InstaSPIN-FOC

The MotorWare library is a cohesive set of software and technical resources designed to minimize motor control system development time. From device-specific drivers and support software to complete system examples, MotorWare software provides support for every stage of development and evaluation.

Especially, the example projects (labs) integrated in MotorWare are used not only to experiment with InstaSPIN but also to use as reference for user's design. There are so many lab projects in the MotorWare, but lab11a is the best project for implementing MTPA and flux-weakening logic since the lab provides the most flexibility to modify the project without too many levels of abstraction. Also, the lab11a that does not use the CTRL module can save the MCU's execution cycle about 5% more than other labs that use CTRL module.

Figure 11 shows how the MTPA and Flux-Weakening can be located in the FOC algorithm structure. The output of the speed controller should be changed to the total torque current instead of the q-axis current. The total torque current feeds to MTPA controller that outputs id and iq current reference according to the MTPA beta trajectory angle. In the flux-weakening region for high speed, the negative output of flux-weakening module is added to the d-axis current reference of the MPTA module.
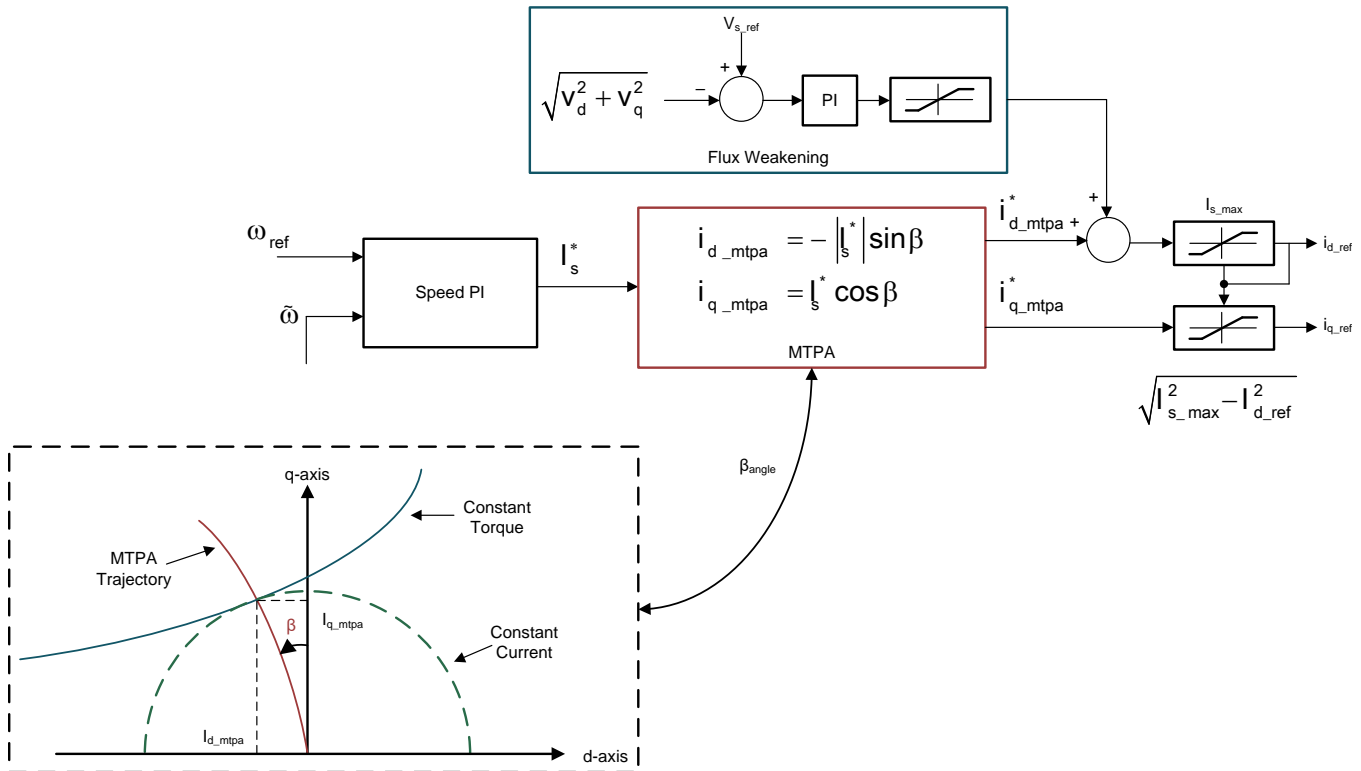

Figure 11. Block Diagram of MTPA and Flux-Weakening Algorithm

The following four steps show how to modify lab11a in Motorware v18 for MTPA and flux-weakening. If the fixed Ld and Lq are used, Step 1 and Step 2 can be skipped.

1. Build the Ld and Lq inductance table with IQ format (optional).

   a. If an inductance table is ready, according to the motor phase current, a look-up table of Ld and Lq can be calculated by using Equation 13 and Equation 14.

   b. Ldq_pu value should be IQ30 format for Piccolo InstaSPIN-FOC devices that has a fixed-point ROM code because FAST observer rom code was implemented by IQ30 format.

$$\text{IQ\_FULL\_SCALE\_INDUCTANCE} = \frac{\text{USER\_IQ\_FULL\_SCALE\_VOLTAGE\_V}}{\left(\text{USER\_IQ\_FULL\_SCALE\_CURRENT\_A} \times \text{USER\_VOLTAGE\_FILTER\_POLE\_rps}\right)}$$

(13)

$$L_{dq\_}pu = \frac{L_{dq\_}H}{\left(\text{IQ\_FULL\_SCALE\_INDUCTANCE} \times 2^{\wedge}\left(30 - L_{dq\_}Qfmt\right)\right)}$$

(14)

2. Function call the inductance look-up table as the following example code (optional).

```
// inductance look-up table
{
    _iq Is_pu_abs = _IQsqrt(_IQmpy(gIdq_pu.value[0],gIdq_pu.value[0])
                        + _IQmpy(gIdq_pu.value[1],gIdq_pu.value[1]));

    _iq20 Is_A_abs = _IQtoIQ20(_IQmpy(Is_pu_abs, _IQ(USER_IQ_FULL_SCALE_CURRENT_A)));
    uint16_t Ldq_table_index = _IQ20int(_IQ20mpy(Is_A_abs, _IQ20(IND_TABLE_SCALE)));

    if(Ldq_table_index > (INDUCTANCE_TABLE_SIZE - 1))
    {
        Ldq_table_index = INDUCTANCE_TABLE_SIZE - 1;
    }

    Ls_dq_pu.value[0] = Ld_table_pu[Ldq_table_index];
    Ls_dq_pu.value[1] = Lq_table_pu[Ldq_table_index];

    EST_setLs_dq_pu(estHandle, &Ls_dq_pu);
    EST_setLs_qFmt(estHandle, inductance_qfmt);
}
```

3. Change the speed controller so that MTPA output is based on the beta angle.

```
// when appropriate, run the PID speed controller
 if((pidCntSpeed++ >= USER_NUM_CTRL_TICKS_PER_SPEED_TICK) && (!gMotorVars.Flag_enableRsRecalc))
 {
    // clear counter
    pidCntSpeed = 0;

    // Set new min and max for the speed controller output
    PID_setMinMax(pidHandle[0], -gIs_Max_pu, gIs_Max_pu);

    // run speed controller
    PID_run_spd(pidHandle[0],TRAJ_getIntValue(trajHandle_spd),speed_pu,&gIs_ref_pu);

    //Id,Iq reference calculation based on MTPA
    {
        _iq sin_beta = _IQsinPU(gBeta_Angle_pu);
        _iq cos_beta = _IQcosPU(gBeta_Angle_pu);
        gIdq_mtpa_pu.value[0] = -_IQmpy(_IQabs(gIs_ref_pu),sin_beta);
        gIdq_mtpa_pu.value[1] = _IQmpy(gIs_ref_pu,cos_beta);
    }
 }
```

4. Change the Flux-Weakening logic that has Id and Iq current limitation.

```
if(FW_getFlag_enableFw(fwHandle) == true)
{
    FW_incCounter(fwHandle);

    if(FW_getCounter(fwHandle) > FW_getNumIsrTicksPerFwTick(fwHandle))
    {
        FW_clearCounter(fwHandle);
        refValue = gMotorVars.VsRef;
        fbackValue =_IQmpy(gMotorVars.Vs,EST_getOneOverDcBus_pu(estHandle));
        FW_run(fwHandle, refValue, fbackValue, &gId_fw_ref_pu);
    }
    // summation of MTPA and FW output with limitation
    refValue = _IQsat(gIdq_mtpa_pu.value[0] + gId_fw_ref_pu,gIs_Max_pu,-gIs_Max_pu);

    // calculate Id reference(MTPA+FW) squared
    Id_ref_squared_pu = _IQmpy(refValue,refValue);

    // Take into consideration that Iq^2+Id^2 = Is^2
    gIdq_ref_pu.value[1] = _IQsqrt(gIs_Max_squared_pu - Id_ref_squared_pu);

}
else
{
    gIdq_ref_pu.value[0] = gIdq_mtpa_pu.value[0];
    refValue = gIdq_ref_pu.value[0] + TRAJ_getIntValue(trajHandle_Id);
}
```

Figure 12 shows the flow chart definition of the Main ADC interrupt service routine (ISR) for FOC and speed control.
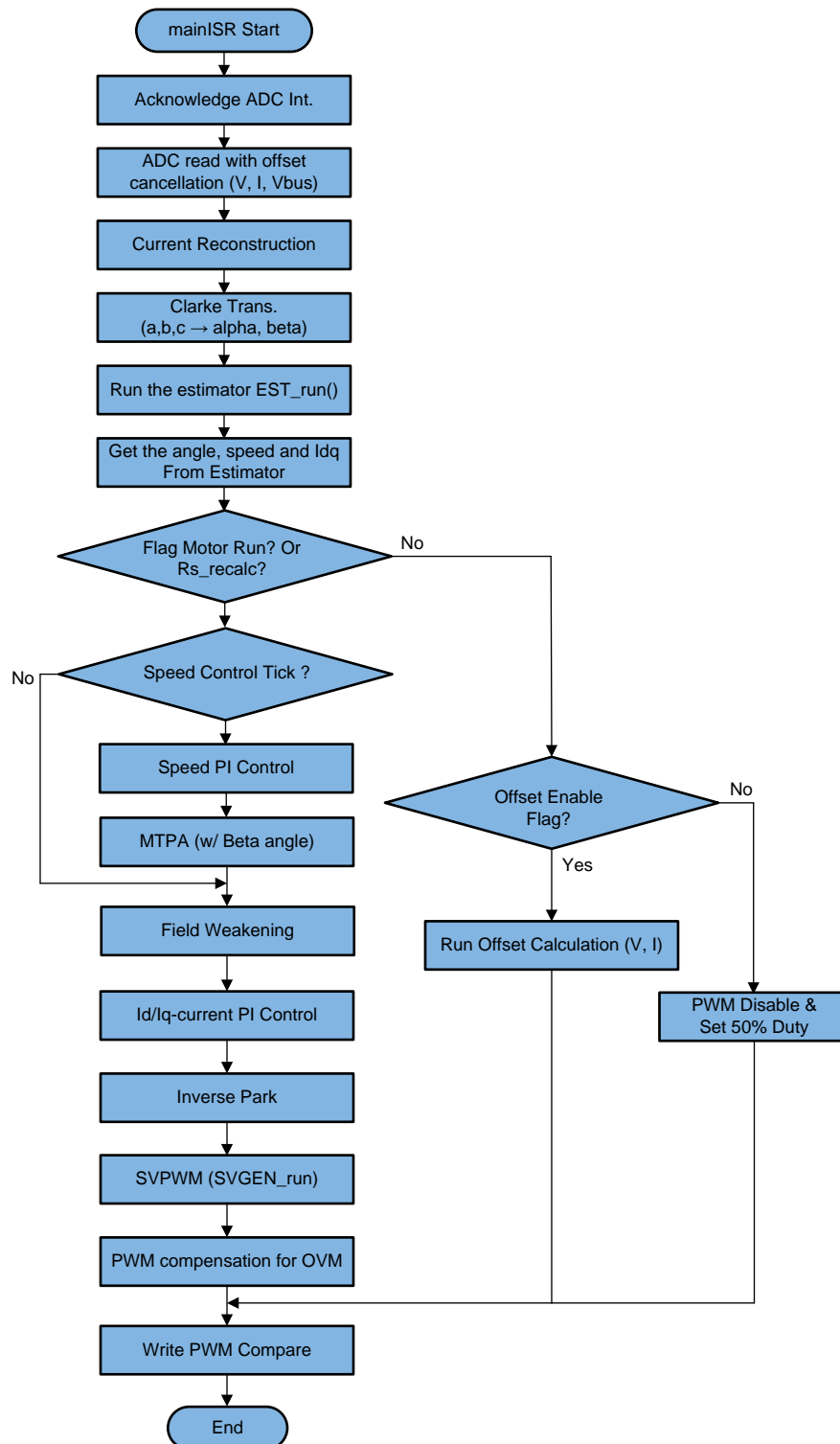
```
                    ┌─────────────────┐
                    │  mainISR Start  │
                    └────────┬────────┘
                             │
                  ┌──────────────────────┐
                  │  Acknowledge ADC Int. │
                  └──────────┬───────────┘
                             │
                  ┌──────────────────────┐
                  │  ADC read with offset │
                  │  cancellation (V, I, Vbus) │
                  └──────────┬───────────┘
                             │
                  ┌──────────────────────┐
                  │ Current Reconstruction│
                  └──────────┬───────────┘
                             │
                  ┌──────────────────────┐
                  │   Clarke Trans.       │
                  │  (a,b,c → alpha, beta)│
                  └──────────┬───────────┘
                             │
                  ┌──────────────────────┐
                  │ Run the estimator EST_run() │
                  └──────────┬───────────┘
                             │
                  ┌──────────────────────┐
                  │ Get the angle, speed and Idq │
                  │  From Estimator       │
                  └──────────┬───────────┘
                             │
              ◇ Flag Motor Run? Or Rs_recalc? ◇──No──→ ◇ Offset Enable Flag? ◇──No──→ ┌ PWM Disable & Set 50% Duty ┐
                             │                                │Yes
              ◇ Speed Control Tick ? ◇──No──┐     ┌ Run Offset Calculation (V, I) ┐
                             │              │
                  ┌──────────────────────┐ │
                  │   Speed PI Control    │ │
                  └──────────┬───────────┘ │
                             │              │
                  ┌──────────────────────┐ │
                  │  MTPA (w/ Beta angle) │ │
                  └──────────┬───────────┘ │
                             │←─────────────┘
                  ┌──────────────────────┐
                  │    Field Weakening    │
                  └──────────┬───────────┘
                             │
                  ┌──────────────────────┐
                  │ Id/Iq-current PI Control │
                  └──────────┬───────────┘
                             │
                  ┌──────────────────────┐
                  │     Inverse Park      │
                  └──────────┬───────────┘
                             │
                  ┌──────────────────────┐
                  │   SVPWM (SVGEN_run)   │
                  └──────────┬───────────┘
                             │
                  ┌──────────────────────┐
                  │ PWM compensation for OVM │
                  └──────────┬───────────┘
                             │
                  ┌──────────────────────┐
                  │  Write PWM Compare    │
                  └──────────┬───────────┘
                             │
                    ┌────────────────┐
                    │      End       │
                    └────────────────┘
```

Figure 12. Main ISR Flow Chart

5 Test Results

To verify the revised source code from lab11a for the IPM motors, tests were run with an electric compressor motor and dynamo system as shown in Figure 13. The electric compressor motor for the HVAC system is a good motor for testing MTPA and flux-weakening algorithm because it has high saliency characteristic. In this test, MTPA was tested on a partial load instead of a full load condition due to the power rating limitation of TI's high voltage kit.
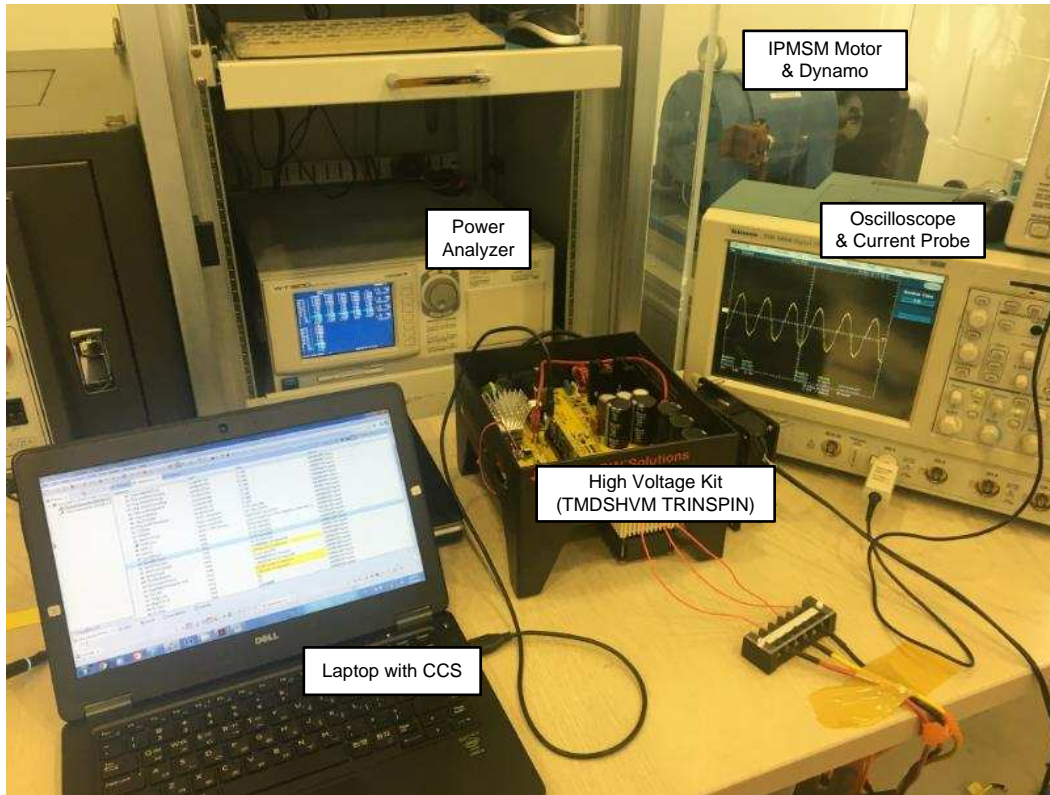
Figure 13. Test Setup for MTPA and Flux-Weakening Tests

Table 1. Motor Specifications

Motor Parameters	Value	Description
Nominal Input Voltage	200 Vdc	Inverter Input Voltage
Rated Output Power	2800 Watt	
Resistance (Rs)	0.130185 Ω	Average Value of Line to Neutral
d-axis Inductance (Ls_d)	0.007324 H	
q-axis Inductance (Ls_q)	0.001532 H	
Rated Flux	0.2084 V/H	Motor Flux Constant
Poles	6	3 Pole Pairs
Beta Angle	25°	MTPA Trajectory Angle at rated design point

Figure 14 shows the U-phase motor current at around 3000 rpm (150 Hz) with MTPA beta angle around 20° and 1 N·m load torque conditions.
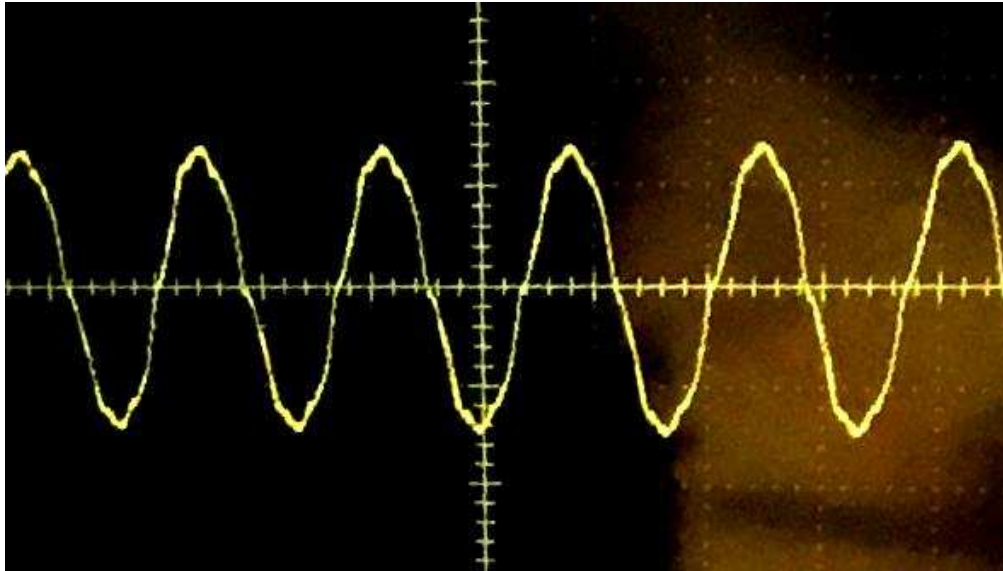
Figure 14. 50 Hz, 5 Arms Current Waveform With MTPA

6   Summary

This applications report discusses how to merge MTPA and Flux-Weakening algorithm into InstaSPIN-FOC for IPM motor applications like electric compressor. Also this document lists the hardware and software key guidance for an InstaSPIN-FOC sensorless solution that needs high SNR on feedback signals to get high sensorless performance. The MTPA and Flux-Weakening algorithms implemented on this report can be applied to most of IPM motor variable speed applications like electric compressor, fan and pump that do not need high dynamics.

7   References

- *InstaSPIN-FOC™ and InstaSPIN-MOTION™ User's Guide*
- *MotorWare™ Software*
- *High Voltage Motor Control Kit with InstaSPIN-FOC Enabled Piccolo MCU*