User's Guide

MSPM0 Sensorless FOC Tuning Guide

Venkata Pavan Mahankali
Aaron Barrera

ABSTRACT

This tuning guide provides step-by-step guidence to set up an MSPM0 MCU and supported DRV hardware board and to tune a 3-phase brushless DC motor.

Table of Contents

List of Figures

List of Tables

Trademarks

LaunchPad™ is a trademark of Texas Instruments.
Arm® and Cortex® are registered trademarks of Arm Limited.
All trademarks are the property of their respective owners.

1 Introduction

The MSPM0Gxxx family of 80MHz Arm®-Cortex® M0+ MCUs can commutate a 3-phase brushless DC (BLDC) motor with sensorless FOC control. The BLDC motor is driven by a three-phase brushless DC (BLDC) MOSFET gate driver or integrated MOSFET motor driver at 12V or 24V nominal DC rails or battery-powered applications. The driver typically integrates three current-sense amplifiers (CSAs) for sensing the three-phase currents of BLDC motors to achieve optimum FOC control.

Figure 1-1 shows a simplified schematic of an MSPM0Gxxx MCU and BLDC motor driver.

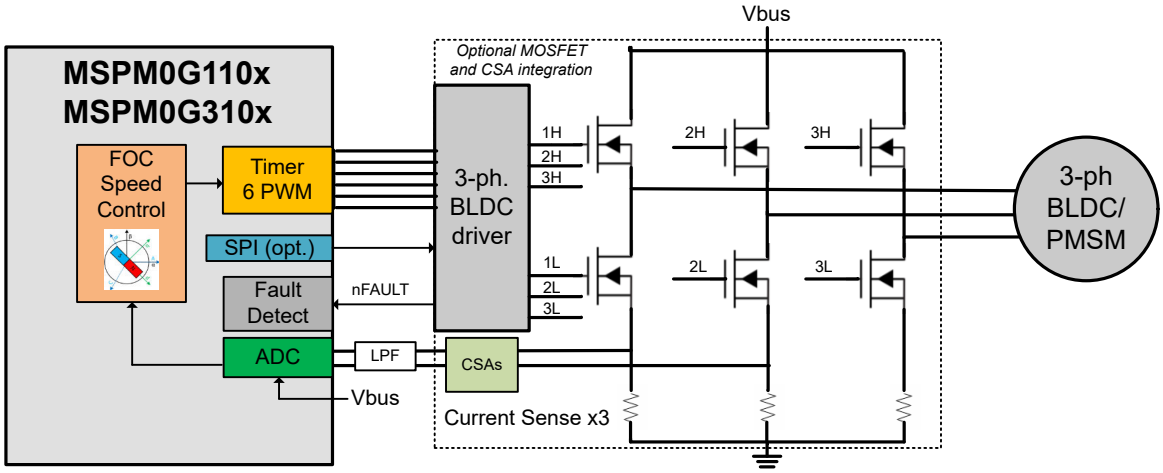

Figure 1-1. Simplified Schematic of MSPM0Gxxx + BLDC Motor Driver

This tuning guide provides the steps to tune a 3-phase BLDC motor using an MSPM0Gxxxx MCU. The tuning process is classified into four sections: **Hardware Setup, Software Setup, Basic Tuning** and **Advanced Tuning**.

- ***Hardware setup*** *:* Steps to set up TI-provided hardware or use a custom PCB for the tuning process.
- ***Software setup:*** Steps to set up TI-provided software for spinning and tuning a BLDC motor.
- ***GUI setup (optional)****:* Steps to use a graphical user interface (GUI) for spinning & tuning a BLDC motor.
- ***Basic tuning:*** Tuning steps to successfully spin the motor in closed loop.
- ***Advanced tuning:*** Tuning steps to conform to use-case and explore features in the device.

2 Hardware Setup

The following items are required to use this tuning guide:

- LP-MSPM0G3507 board
- Supported DRV83xx motor driver evaluation module (EVM)
  - BOOSTXL-DRV8323RS
  - DRV8316REVM
- Jumper wires for pin table connections
- A computer with the MSPM0 FOC software installed
- A BLDC motor to be tuned using this process. The motor data sheet is helpful but not mandatory.
- A DC power supply rated for the motor
- Basic lab equipment such as a digital multimeter (DMM), oscilloscope, current probe, and voltage probe

Figure 2-1 shows the block diagram connections for a sensorless FOC motor system. The system can be built using:

- TI-provided hardware (LP-MSPM0G3507 and DRV83xx EVM)
- Custom PCB hardware with an onboard MSPM0Gxxx MCU and a BLDC motor driver

The following sections describe how to configure the pins for each portion of the sensorless FOC block diagram.

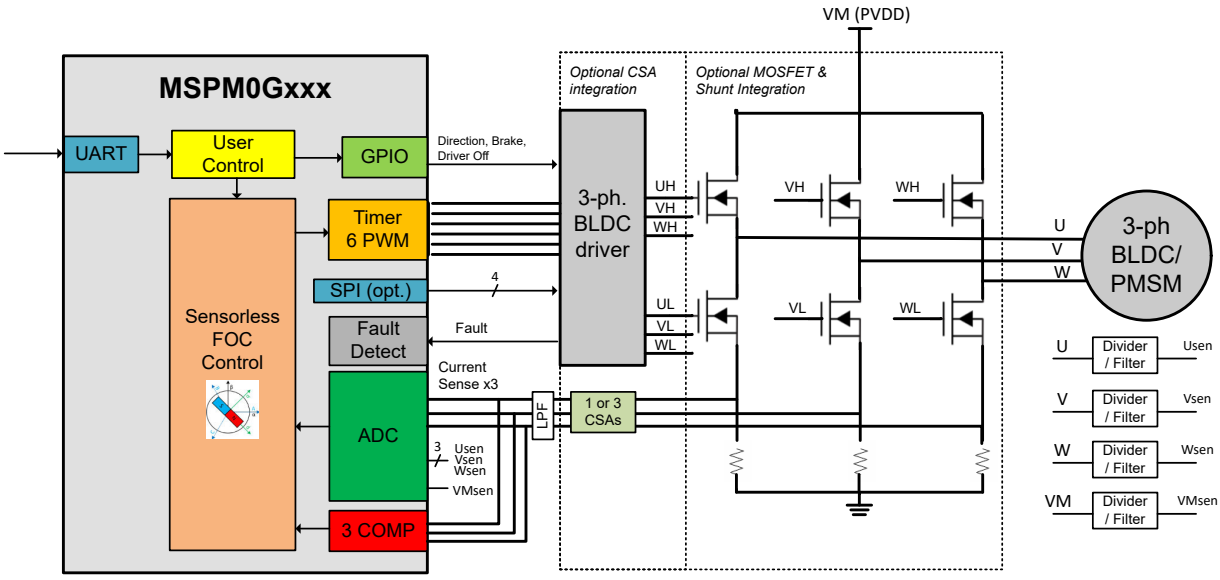

Figure 2-1. MSPM0Gxxx + BLDC Motor Driver - Sensorless FOC Block Diagram

The System Configuration tool (SysConfig) helps to configure the pins in a motor control system. The default pin configurations are provided for the EVM hardware setup to spin a motor, but pins can be remapped to other pins visually inside SysConfig. This is useful for reconfiguring different pins (such as PWM, ADC, or other control signals) on a custom PCB or for scaling to different packages across MSPM0 devices.

2.1 EVM Hardware Setup

TI provides LaunchPad™ development kits to evaluate MSPM0 Arm Cortex-M0+ micrcontrollers and evaluation modules (EVMs) to evaluate the DRV83xx family of brushless-DC motor drivers. These evaluation boards are available on ti.com and can be used as a system evaluation platform for sensorless FOC motor control.

For supported evaluation boards, refer to Section 2.1.1.

Note

The provided defaults have pre-configured pins that are intended to support hardware evaluation boards. If a custom PCB is used, refer to the following *Pin Configurations* sections to assign the supported pins for the 3-phase motor driver.

2.1.1 EVM Hardware Support

Table 2-1 shows the supported MSPM0 LaunchPad kits and EVMs and the connection guides for 3-phase sensorless FOC motor control.

Table 2-1. Supported Hardware for Sensorless FOC Using MSPM0

MSPM0Gxxx LaunchPad Kit	Motor Driver Hardware	Hardware User's Guide	Current Sense Amplifiers	SPI Driver Support	Recommended Motor Voltage Range	Recommended Motor Power
LP-MSPM0G3507	BOOSTXL-DRV8323RS	BOOSTXL-DRV8323RS Hardware User Guide	3	Yes	6V to 60V	< 1000W
	DRV8316REVM	DRV8316REVM Hardware User Guide	3	Yes	4.5V to 35V	< 80W

Note

Make sure that the jumper configurations for the LaunchPad kit and EVM are correct. For more information, see the user's guides for the LaunchPad kit and EVM.

2.2 Pin Configurations for IPD Usage

The default pin configurations for Initial Position Detection (IPD) are shown in Table 2-2. The required connections are the 3 current sense amplifier outputs to the positive inputs of the 3 integrated comparators of the MSPM0. The comparators monitor the phase current against a pre-set IPD threshold voltage (set by 8-bit DAC into the comparator's negative input).

Table 2-2. Pin Configurations for IPD

MSPM0 pin	MSPM0 Function	DRV connection	DRV Function
COMP0_INx+	Comparator 0 positive input	SOA	Phase A current sense output
COMP1_INx+	Comparator 1 positive input	SOB	Phase B current sense output
COMP2_INx+	Comparator 2 positive input	SOC	Phase C current sense output

2.3 Pin Configurations for PWM Outputs

The default pin configurations for PWM outputs are shown in Table 2-3. The required connections are six PWM output signals that send the commutation patterns for sensorless FOC motor control. TIMA includes features for motor control, such as complimentary PWM outputs with deadband, fault handling with <40ns response time, and repeat counters for configuring FOC loop rates.

TIMA0 is the preferred timer for motor control because it provides three complimentary pairs of PWM outputs from the same timer counter (such as TIMA0_C1 and TIMA0_C1N), but any TIMA0 or TIMA1 output pair can be used and cross-triggered to provide the six PWM output signals.

Table 2-3. Pin Configurations for PWM Outputs

MSPM0 Pin	Function	DRV Connection	DRV Function
TIMA0_C0	TIMA0 channel 0 output pin	INHA	Phase A high side PWM input
TIMA0_C0N	TIMA0 channel 0 complimentary output pin	INLA	Phase A low side PWM input
TIMA0_C1	TIMA0 channel 1 output pin	INHB	Phase B high side PWM input
TIMA0_C1N	TIMA0 channel 1 complimentary output pin	INLB	Phase B low side PWM input
TIMA0_C2	TIMA0 channel 2 output pin	INHC	Phase C high side PWM input
TIMA0_C2N	TIMA0 channel 2 complimentary output pin	INLC	Phase C low side PWM input

2.4 Pin Configurations for ADC Currents

The default pin configurations for ADC currents are shown in Table 2-4 and Table 2-5, depending on the DRV device used. The required connections are 3 ADC inputs connected to the 3 CSA outputs from the motor driver or external CSAs.

ADC0 and ADC1 are two simulaneous-sampling 4MSPS analog-to-digital converters that are used to measure phase currents and voltages. ADC0 and ADC1 measure phase currents simultaneously and bus voltage sequentially depending on the rotor angle under normal motor run conditions. ADC0 and ADC1 are sampled to measures phase voltages during Initial Speed Detection.

An optional low-pass RC filter can be placed in series from the CSA outputs to the ADC inputs to filter out any high frequency noise from the switching output signals for proper ADC sampling as shown in Figure 2-2.

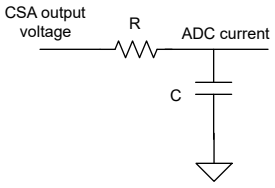

Figure 2-2. CSA Output Filter

Choose a filtering frequency f_c that it at least 10 times the PWM switching frequency (f_{PWM}). Use Equation 1 to calculate f_c based on the RC filter design.

$$f_c = \frac{1}{2\pi RC} \quad (1)$$

Table 2-4. Pin Configurations for ADC Currents With Simultaneous Sampling in DRV8316

MSPM0 Pin	Function	DRV Connection	DRV Function
A0_3	ADC0, channel 3 input	SOA	Phase A current sense output
A0_2	ADC0, channel 2 input	SOB	Phase B current sense output
A1_2	ADC1, channel 2 input	SOB	Phase B current sense output
A1_1	ADC1, channel 1 input	SOC	Phase C high side PWM input

Table 2-5. Pin Configurations for ADC Currents Without Simultaneous Sampling in DRV8323

MSPM0 Pin	Function	DRV Connection	DRV Function
A1_2	ADC1, channel 2 input	SOA	Phase A current sense output
A0_3	ADC0, channel 2 input	SOB	Phase B current sense output
A1_3	ADC1, channel 3 input	SOC	Phase C high side PWM input

2.5 Pin Configurations for ADC Voltages

The default pin configurations for ADC voltages are shown in the following tables. The required connections are four ADC inputs:

- Three ADC inputs connected to the three sensed phase voltages from the motor (VSENA,VSENB, VSENC)
- One ADC input connected to the sensed VM motor voltage (VSENVM)

The sensed voltages are realized using a resistor divider with an optional bypass filtering cap as shown in Figure 2-3. Size the resistors so any motor voltage transients do not exceed the maximum voltage of the ADC inputs. For more information on the resistor divider ratio, see Section 6.1.5.

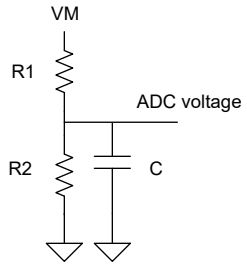

Figure 2-3. ADC Voltage Divider

Note

An optional center tap voltage (CTAP) can be used to sense the motor BEMF.

Table 2-6. Pin Configurations for ADC Phase Voltages

MSPM0 Pin	Function	DRV Connection	DRV Function
A1_6	ADC1, channel 6 input	VSENA	Phase A sensed voltage output
A0_7	ADC0, channel 7 input	VSENB	Phase B sensed voltage output
A1_5	ADC1, channel 5 input	VSENC	Phase C sensed voltage output

Table 2-7. Pin Configurations for ADC DC Bus Voltage Sensing for DRV8316

MSPM0 Pin	Function	DRV Connection	DRV Function
A1_3	ADC1, channel 3 input	VSEN -Vm	DC bus voltage output

Table 2-8. Pin Configurations for ADC DC Bus Voltage Sensing for DRV8323

MSPM0 Pin	Function	DRV Connection	DRV Function
A0_2	ADC0, channel 2 input	VSEN -Vm	DC bus voltage output

2.6 Pin Configurations for Faults

The default pin configurations for faults are shown in Table 2-9. Faults can be detected in hardware by the motor driver or MCU.

Typically, a motor driver drives an active-low open-drain fault pin (nFAULT) when there is a detected fault in the system. Examples are MOSFET overcurrent, gate drive, or power supply-related faults connections in the driver.

MSPM0 MCUs can detect fault inputs with dedicated hardware paths to provide low latency and response times as fast as 40ns. This is faster than using a conventional GPIO interrupt with software latency. The fault input paths can be configured for fault handling using TIMA fault handler, such as shutting off the PWMs during an overcurrent condition. Examples of TIMA inputs include an external fault pin (such as TIMA_FLT0) and low-side overcurrent using comparators (such as COMP0_IN0+).

Table 2-9. Pin Configurations for Faults

MSPM0 Pin	Function	DRV Connection	DRV Function
TIMA0_C2	TIMA0 channel 2 input pin	nFAULT	Open-drain, active-low fault pin

2.7 Pin Configurations for GPIO Output Functions

Many GPIO output functions from the MSPM0 can be used for motor driver specific functions controlled by logic-level pins. Examples of motor driver functions are:

- Enable pin (ENABLE) / active-low sleep mode control (nSLEEP)
- Active high gate driver shutoff (DRVOFF)
- Active-high CSA Calibration (CAL)
- Active-high brake (BRAKE) / active-low brake (nBRAKE)
- Direction pin (DIR)

Note

See the motor driver data sheet and the user guide for GPIO configurable pins.

2.8 Pin Configurations for SPI Communication

The default pin configurations for SPI connections are shown in Section 2.8. Some motor drivers include an optional SPI that is used for configuring control registers and reading status registers for fault diagnosis. Some examples of SPI registers are:

- Configuring gate drive source/sink current strength
- Configuring CSA output behavior
- Running diagnostics
- Reading fault bits when the fault pin has been detected as active low
- Clearing fault status bits once the fault condition is removed
- Clearing watchdog timers

Note

See the motor driver data sheet if a SPI or hardware interface is used to configure system settings.

Table 2-10. Pin Configurations for SPI Connections

MSPM0 Pin	Function	DRV Connection	DRV Function
SPIx_CSy	SPI chip select (y = 0,1,2,3)	nSCS	SPI chip select
SPIx_SCK	SPI clock	SCLK	SPI clock
SPIx_POCI	SPI peripheral out controller in	SDO	SPI data out
SPIx_PICO	SPI peripheral in controller out	SDI	SPI data in

Note

See the motor driver data sheet to determine if the SDO pin is open-drain and requires a pullup resistor.

2.9 Pin Configurations for UART Communication

The default pin configurations for UART connections are shown in Section 2.9. UART can be used to receive commands to configure, spin, and control the motor. The commands are sent from a host MCU or GUI and can optionally be used for advanced protocols such as LIN communication.

Note

Use UART instance 0 (UART0_RX, UART0_TX) to configure the UART interface when used along with DMA and LIN interface.

Note

Use UART instance 3 (UART3_RX, UART3_TX) to configure the UART interface for GUI communication when used along with DMA.

Table 2-11. Pin Configurations for UART Connections

MSPM0 Pin	Function
UARTx_RX	UART receive
UARTx_TX	UART transmit

2.10 External Connections for Evaluation Boards

Follow the steps below when connecting an MSPM0 LaunchPad to a DRV83xx EVM:

1. Connect the motor the motor phase connection terminal black (phases A, B, and C). If the motor has a center tap connection or wires for Hall-effect sensors, leave these wires unconnected.
2. Make the inter-device connections from the MSPM0 LaunchPad to the DRV83xx EVM by mating the EVM to the LaunchPad or using jumper wires as shown in Figure 2-4. See Section 2.1.1 for hardware user guide connection details.
   a. If using a host MCU to communicate using UART to the MSPM0 device, connect the UART connections from the host MCU board to the MSPM0 LaunchPad kit.
   b. If using the GUI to communicate to the MSPM0 device using USB to backchannel UART, connect the backchannel UART connections to UART3_TX and UART3_RX as shown in Figure 2-5.
3. Connect a micro-USB cable from the MSPM0 LaunchPad kit to the PC.
   a. Remove GND and 3V3 isolation jumpers on the bridge if desired to isolate the PC from the motor system. If this step is done, 3V3 must be provided externally or from the DRV83xx EVM board, if available.
4. Supply a voltage compliant with the Power Supply Voltage (VM) range. See the board-specific user's guide or DRV-specific data sheet for recommended voltage range.

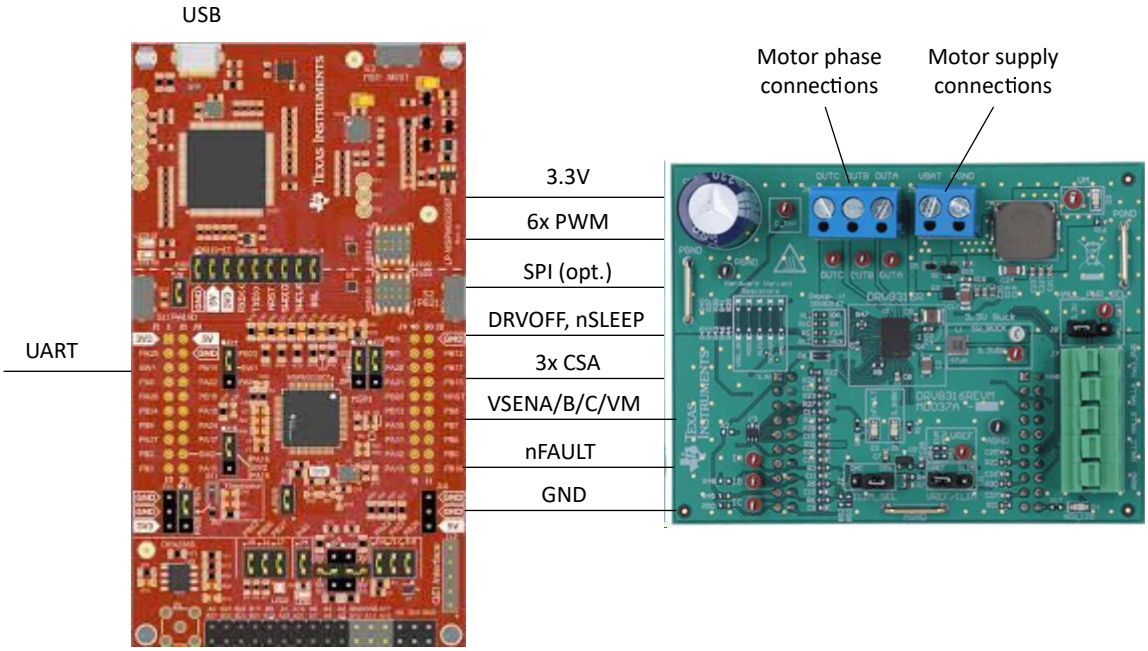

Figure 2-4. MSPM0 LaunchPad Kit and DRV83xx EVM External Configuration

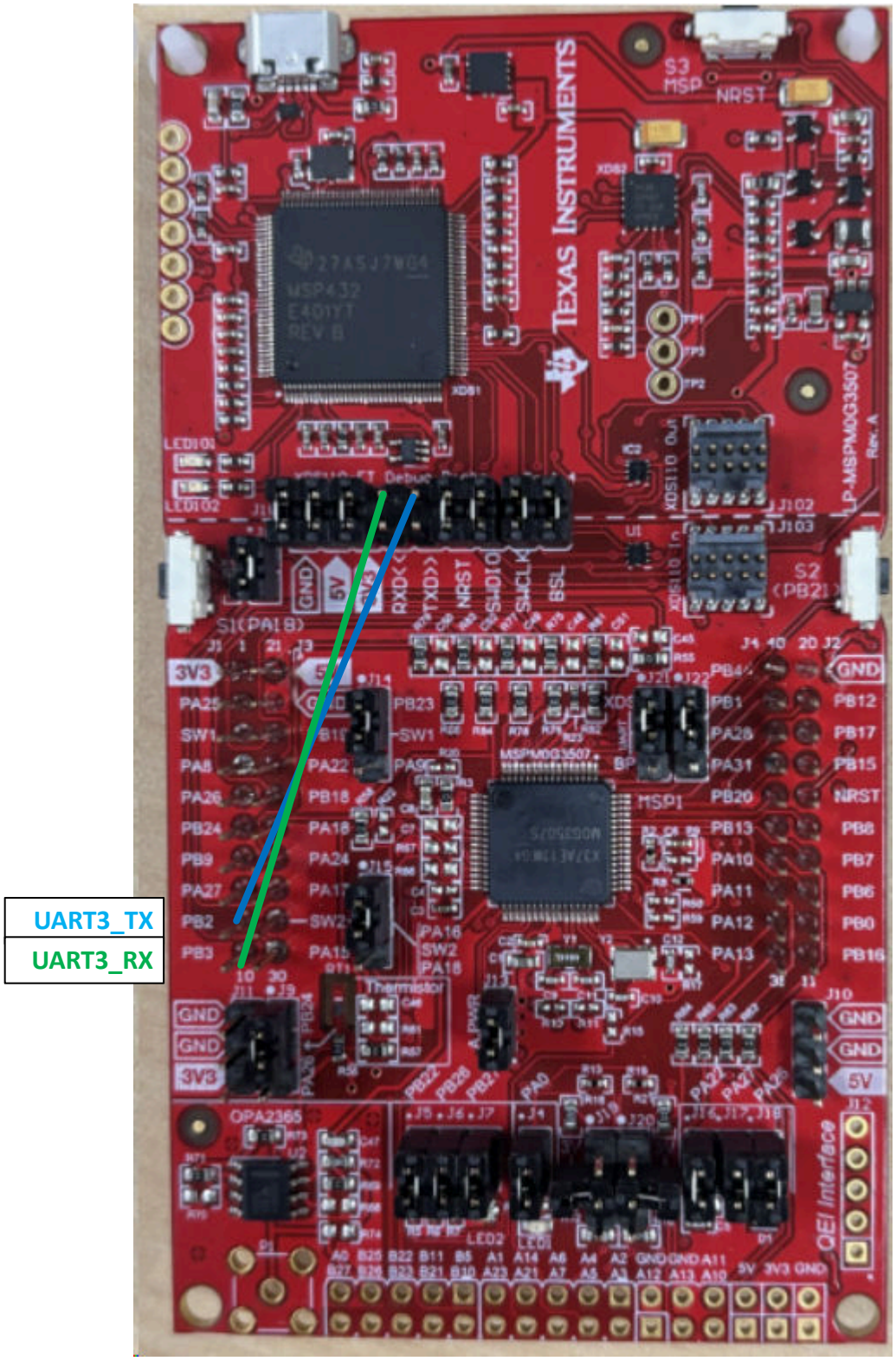

Figure 2-5. LP-MSPM0G3507 Backchannel Connection to UART3

3 Software Setup

Sensorless FOC software for MSPM0 MCUs are provided inside MSPM0-SDK and are available for evaluation with Code Composer Studio IDE.

For supported evaluation boards, see Section 3.1.

3.1 Software Support

Table 3-1 shows the software and documentation supported for Sensorless FOC control in TI Resource Explorer.

Table 3-1. Software Support for FOC Control

Sensorless FOC User's Guide(1)	Code Examples	GUI	Tuning User Guide
Sensorless FOC User's Guide	DRV8323RS DRV8316	MSPM0G Sensorless FOC GUI	MSPM0 Sensorless FOC Tuning Guide

(1) includes library overview, software setup, hardware setup, and more.

4 GUI Setup

The user can optionally use the MSPM0 Sensorless FOC GUI as a host to send commands to the MSPM0 MCU at the target to control the motor using either SWD or UART interface.

When SWD is used, the onboard XDS110 debugger on the MSPM0 LaunchPad kit sends the values from the modified GUI variables. These values are linked to variables and expressions in the motor control software and directly updates those values during run time. When UART is used, the GUI includes a USB-to-UART codec that can send UART commands as a host to the MSPM0 LaunchPad kit. The application software includes a configurable UART register map and data format that translates the UART data into simplified motor control commands.

Table 4-1. GUI Connection Types

Connection	Interface	Hardware Connections
GUI to target MSPM0 MCU	SWD (serial wire debug)	SWDIO, SWCLK
GUI to target MSPM0 MCU	UART	UART3_TX, UART3_RX

To set up the GUI:

1. Make sure that the hardware is connected as described in Section 2 for evaluation boards or a custom user PCB.
2. Run the MSPM0 Sensorless FOC GUI.
3. Configure and tune the motor as described in Section 6 and Section 7.

5 Register Map

5.1 User Control Registers (Base Address = 0x202000C8h)

User Control Registers are set of user configurable parameters to control the Motor in real time. These set of registers can be modified in the application code using pointer variable pUserCtrlRegs.

Table 5-1. User Control registers

Offset	Acronym	Register Name	Section
0h	SPEED_CTRL	Speed Control Register	Section 5.1.1
4h	ALGO_DEBUG_CTRL1	Algorithm Debug Control 1 register	Section 5.1.2
8h	ALGO_DEBUG_CTRL2	Algorithm Debug Control 2 register	Section 5.1.3

Complex bit access types are encoded to fit into small table cells as below.

Table 5-2. Register Configuration Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
Write Type		
W	W	Write
Reset or Default Value		
-n		Value after reset or the default value

5.1.1 Speed Control Register (Offset = 0h) [Reset = 00000000h]

Register to control Motor Speed

Table 5-3. SPEED_CTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
31 - 15	RESERVED	R	0h	Reserved
14-0	SPEED_CTRL	W	0h	Target Motor Speed/Torque value % of speed or Torque command × 32768 .

5.1.2 Algo Debug Control 1 Register (Offset = 4h) [Reset = 00000000h]

Register to control Algorithm debug functions

Table 5-4. Algorithm Debug Control 1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	CLEAR_FAULT	W	0h	Bit to clear set controller and Gate Driver Faults.Bit is automatically reset. 1h = Clear Fault Command.
30-22	FORCED_ALIGN_ANGLE	W	0h	9-bit value (in °) used during forced align state (FORCE_ALIGN_EN = 1) Angle applied (°) = FORCED_ALIGN_ANGLE % 360°
21-16	RESERVED	R	0h	
15	CLOSED_LOOP_DIS	W	0h	Use to disable closed loop 0h = Enable closed loop 1h = Disable closed loop, motor commutation in open loop

Table 5-4. Algorithm Debug Control 1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
14	FORCE_ALIGN_EN	W	0h	Force align state enable 0h = Disable force align state, device comes out of align state if MTR_STARTUP is selected as ALIGN or DOUBLE ALIGN 1h = Enable force align state, device stays in align state if MTR_STARTUP is selected as ALIGN or DOUBLE ALIGN
13	FORCE_SLOW_FIRST_CYCLE_EN	W	0h	Force slow first cycle enable 0h = Disable force slow first cycle state, device comes out of slow first cycle state if MTR_STARTUP is selected as SLOW FIRST CYCLE 1h = Enable force slow first cycle state, device stays in slow first cycle state if MTR_STARTUP is selected as SLOW FIRST CYCLE
12	FORCE_IPD_EN	W	0h	Force IPD enable 0h = Disable Force IPD state, device comes out of IPD state if MTR_STARTUP is selected as IPD 1h = Enable Force IPD state, device stays in IPD state if MTR_STARTUP is selected as IPD
11	FORCE_ISD_EN	W	0h	Force ISD enable 0h = Disable Force ISD state, device comes out of ISD state if ISD_EN is set 1h = Enable Force ISD state, device stays in ISD state if ISD_EN is set
10	FORCE_ALIGN_ANGLE_SRC_SEL	W	0h	Force align angle state source select 0h = Force Align Angle defined by ALIGN_ANGLE 1h = Force Align Angle defined by FORCED_ALIGN_ANGLE
9-0	FORCE_IQ_REF_SPEED_LOOP_DIS	W	0h	Sets Iq_ref when speed loop is disabled If SPEED_LOOP_DIS = 1b, then Iq_ref is set using IQ_REF_SPEED_LOOP_DIS Iq_ref = (FORCE_IQ_REF_SPEED_LOOP_DIS / 500) × 10, if FORCE_IQ_REF_SPEED_LOOP_DIS < 500 - (FORCE_IQ_REF_SPEED_LOOP_DIS - 512) / 500 × 10 if FORCE_IQ_REF_SPEED_LOOP_DIS > 512 Valid values are 0 to 500 and 512 to 1000

5.1.3 Algo Debug Control 2 Register (Offset = 8h) [Reset = 00000000h]

Register to control Algorithm Debug functions

Table 5-5. Algorithm Debug Control 2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31 - 28	RESERVED	R	0h	Reserved
27	STATUS_UPDATE_ENABLE	W	0h	This bit enables the continous update of user Status variables in real time.
26	CURRENT_LOOP_DIS	W	0h	Use to control the FORCE_VD_CURRENT_LOOP_DIS and FORCE_VQ_CURRENT_LOOP_DIS. If CURRENT_LOOP_DIS = 1b, current loop and speed loop are disabled 0h = Enable Current Loop 1h = Disable Current Loop
25-16	FORCE_VD_CURRENT_LOOP_DIS	W	0h	Sets Vd_ref when current loop and speed loop are disabled If CURRENT_LOOP_DIS = 1b, then Vd is controlled using FORCE_VD_CURRENT_LOOP_DIS Vd_ref = (FORCE_VD_CURRENT_LOOP_DIS / 500) if FORCE_VD_CURRENT_LOOP_DIS < 500 - (FORCE_VD_CURRENT_LOOP_DIS - 512) / 500 if FORCE_VD_CURRENT_LOOP_DIS > 512 Valid values: 0 to 500 and 512 to 1000

Table 5-5. Algorithm Debug Control 2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
15-6	FORCE_VQ_CURRENT_LOOP_DIS	W	0h	Sets Vq_ref when current loop speed loop are disabled If CURRENT_LOOP_DIS = 1b, then Vq is controlled using FORCE_VQ_CURRENT_LOOP_DIS Vq_ref = (FORCE_VQ_CURRENT_LOOP_DIS / 500) if FORCE_VQ_CURRENT_LOOP_DIS < 500 - (FORCE_VQ_CURRENT_LOOP_DIS - 512) / 500 if FORCE_VQ_CURRENT_LOOP_DIS > 512 Valid values: 0 to 500 and 512 to 1000
5-0	RESERVED	R	0h	Reserved

5.2 User Input Registers (Base Address = 0x20200000h)

User input registers are set of configurable registers to tune the motor performance in real time for various motor control features and save them in flash

Table 5-6. User Input Registers

Offset	Acronym	Register Name	Section
0h	SYSTEM_PARAMETERS	System Parameters	Section 5.2.1
28h	ISD_CFG	Initial Speed Detection Configuration	Section 5.2.2
2Ch	RVS_DRV_CONFIG	Reverse Drive Configuration	Section 5.2.3
30h	MOTOR_STARTUP1	Motor Startup 1 Configuration	Section 5.2.4
34h	MOTOR_STARTUP2	Motor Startup 2 Configuration	Section 5.2.5
38h	CLOSELOOP1	Close Loop1 Configuration	Section 5.2.6
3Ch	CLOSELOOP2	Close Loop2 Configuration	Section 5.2.7
40h	FAULT_CONFIG1	Fault Configuration 1	Section 5.2.8
44h	FAULT_CONFIG2	Fault Configuration 2	Section 5.2.9
48h	MISC_ALGO_CONFIG	Miscellaneous Algorithm Configuration	Section 5.2.10
4Ch	PIN_CONFIGURATION	Pin Configuration	Section 5.2.11
50h	PERI_CONFIG	Peripheral Configuration	Section 5.2.12

Complex bit access types are encoded to fit into small table cells as below.

Table 5-7. Register Configuration Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
Write Type		
W	W	Write
Reset or Default Value		
-n		Value after reset or the default value

5.2.1 SYSTEM_PARAMETERS (Offset = 0h)

Set of basic system configuration parameters essential for motor control system functionality

Table 5-8. Motor Resistance Configuration Registers (Offset = 4h)

Bit	Field	Type	Reset	Description
31-0	MTR_RESISTANCE	R/W	0h	Motor Resistance in milli ohms

Table 5-9. Motor Inductance Configuration (Offset = 8h)

Bit	Field	Type	Reset	Description
31-0	MTR_INDUCTANCE	R/W	0h	Motor Inductance in micro henry

Table 5-10. Motor BEMF Constant Configuration (Offset = 8h)

Bit	Field	Type	Reset	Description
31-0	MTR_BEMF_CONSTANT	R/W	0h	Motor BEMF constant in mV/Hz × 10.

Table 5-11. Base Voltage Configuration (Offset = Ch)

Bit	Field	Type	Reset	Description
31-0	VOLTAGE_BASE	R/W	0.0	Base voltage of the board calculated based on the voltage divider as (3.3V × voltage divider ratio) in volts. 3.3V is the full-scale value of the ADC.

Table 5-12. Base Current Configuration (Offset = 10h)

Bit	Field	Type	Reset	Description
31-0	CURRENT_BASE	R/W	0.0	Base current of the board calculated based on the CSA gain in as (1.65V / CSA Gain in volts/amp) in amps. 1.65V is the reference mid point voltage of the ADC for bidirectional current sensing. If the CSA gain is in V/V , multiply with current sense resistor value in ohms to compute CSA gain in volts/amp

Table 5-13. Motor Max Speed Configuration (Offset = 14h)

Bit	Field	Type	Reset	Description
31-0	MOTOR_MAX_SPEED	R/W	0h	Rated motor speed in Hz from the data sheet

Table 5-14. Speed Loop Proportional Gain (Offset = 18h)

Bit	Field	Type	Reset	Description
31-0	SPEED_LOOP_KP	R/W	0.0	Proportional gain for the closed loop speed control in float

Table 5-15. Speed Loop Integral Gain (Offset = 1Ch)

Bit	Field	Type	Reset	Description
31-0	SPEED_LOOP_KI	R/W	0.0	Integral gain for the closed loop speed control in float

Table 5-16. Torque Loop Proportional Gain (Offset = 20h)

Bit	Field	Type	Reset	Description
31-0	CURR_LOOP_KP	R/W	0.0	Proportional gain for the closed loop torque control in float

Table 5-17. Torque Loop Integral Gain (Offset = 24h)

Bit	Field	Type	Reset	Description
31-0	CURR_LOOP_KI	R/W	0.0	Integral gain for the closed loop torque control in float

5.2.2 ISD_CONFIG Register (Offset = 28h) [Reset = 00000000h]

Register to configure Initial Speed Detection

Table 5-18. ISD_CONFIG Register

Bit	Field	Type	Reset	Description
31-30	BEMF_RESYNC_THRESHOLD	R/W	0h	Minimum ratio of estimated BEMF with respect to actual BEMF for ISD resynchronization 0h = 0.75 1h = 0.80 2h = 0.85 3h = 0.90

Table 5-18. ISD_CONFIG Register (continued)

Bit	Field	Type	Reset	Description
29	ISD_EN	R/W	0h	ISD Enable 0h = Disable 1h = Enable
28	BRAKE_EN	R/W	0h	Brake enable 0h = Disable 1h = Enable
27	HIZ_EN	R/W		Hi-Z enable 0h = Disable 1h = Enable
26	RVS_DR_EN	R/W	0h	Reverse drive enable 0h = Disable 1h = Enable
25	RESYNC_EN	R/W	0h	Resynchronization Enable 0h = Disable 1h = Enable
24-21	FW_DRV_RESYN_THR	R/W	0h	Minimum Speed threshold to resynchronize to close loop (% of MAX_SPEED) 0h = 5% 1h = 10% 2h = 15% 3h = 20% 4h = 25% 5h = 30% 6h = 35% 7h = 40% 8h = 45% 9h = 50% Ah = 55% Bh = 60% Ch = 70% Dh = 80% Eh = 90% Fh = 100%
20	BRK_CONFIG	R/W	0h	Brake configuration 0h = Brake time is used to come out of brake state 1h = Brake current threshold is used to come out of brake state
16-19	BRK_TIME	R/W	0h	Brake time 0h = 10ms 1h = 50ms 2h = 100ms 3h = 200ms 4h = 300ms 5h = 400ms 6h = 500ms 7h = 750ms 8h = 1s 9h = 2s Ah = 3s Bh = 4s Ch = 5s Dh = 7.5s Eh = 10s Fh = 15s

Table 5-18. ISD_CONFIG Register (continued)

Bit	Field	Type	Reset	Description
15-12	HIZ_TIME	R/W	0h	Hi-Z time 0h = 10ms 1h = 50ms 2h = 100ms 3h = 200ms 4h = 300ms 5h = 400ms 6h = 500ms 7h = 750ms 8h = 1s 9h = 2s Ah = 3s Bh = 4s Ch = 5s Dh = 7.5s Eh = 10s Fh = 15s
11-9	STAT_DETECT_THR	R/W	0h	BEMF threshold to detect if motor is stationary 0h = 50mV 1h = 75mV 2h = 100mV 3h = 250mV 4h = 500mV 5h = 750mV 6h = 1000mV 7h = 1500mV
8-5	REV_DRV_HANDOFF_THR	R/W	0h	Speed threshold used to transition to open loop during reverse deceleration (% of MAX_SPEED) 0h = 2.5% 1h = 5% 2h = 7.5% 3h = 10% 4h = 12.5% 5h = 15% 6h = 20% 7h = 25% 8h = 30% 9h = 40% Ah = 50% Bh = 60% Ch = 70% Dh = 80% Eh = 90% Fh = 100%

Table 5-18. ISD_CONFIG Register (continued)

Bit	Field	Type	Reset	Description
4-0	REV_DRV_OPEN_LOOP_CURRENT	R/W	0h	Open loop current limit during speed reversal in % of CURRENT_BASE 0h = 7.5% 1h = 8.0% 2h = 8.5% 3h = 9.0% 4h = 9.5% 5h = 10% 6h = 11% 7h = 12% 8h = 13% 9h = 14% Ah = 15% Bh = 16% Ch = 17% Dh = 18% Eh = 20% Fh = 22.5% 10h = 25% 11h = 27.5% 12h = 30% 13h = 35% 14h = 40% 15h = 45% 16h = 50% 17h = 55% 18h = 60% 19h = 70% 1Ah = 75% 1Bh = 80% 1Ch = 85% 1Dh = 90% 1Eh = 95% 1Fh = 100%

5.2.3 RVS_DRV_CONFIG Register (Offset = 2Ch) [Reset = 00000000h]

Register to configure Reverse Drive

Table 5-19. RVS_DRV_CONFIG Register

Bit	Field	Type	Reset	Description
31-29	RESERVED	R	0h	Reserved
28	REV_DRV_CONFIG	R/W	0h	Chooses between forward and reverse drive setting for reverse drive 0h = Open loop current, A1, A2 based on forward drive 1h = Open loop current, A1, A2 based on reverse drive

Table 5-19. RVS_DRV_CONFIG Register (continued)

Bit	Field	Type	Reset	Description
27-24	REV_DRV_OPEN_LOOP_ACCEL_A1	R/W	0h	Reverse Drive Open loop acceleration coefficient A1 during reverse drive 0h = 0.01Hz/s 1h = 0.05Hz/s 2h = 1Hz/s 3h = 2.5Hz/s 4h = 5Hz/s 5h = 10Hz/s 6h = 25Hz/s 7h = 50Hz/s 8h = 75Hz/s 9h = 100Hz/s Ah = 250Hz/s Bh = 500Hz/s Ch = 750Hz/s Dh = 1000Hz/s Eh = 5000Hz/s Fh = 10000Hz/s
23-20	REV_DRV_OPEN_LOOP_ACCEL_A2	R/W		Reverse Drive Open loop acceleration coefficient A2 during reverse drive 0h = 0.0Hz/s2 1h = 0.05Hz/s2 2h = 1Hz/s2 3h = 2.5Hz/s2 4h = 5Hz/s2 5h = 10Hz/s2 6h = 25Hz/s2 7h = 50Hz/s2 8h = 75Hz/s2 9h = 100Hz/s2 Ah = 250Hz/s2 Bh = 500Hz/s2 Ch = 750Hz/s2 Dh = 1000Hz/s2 Eh = 5000Hz/s2 Fh = 10000Hz/s2
19-0	RESREVED	R	0h	Reserved

5.2.4 MOTOR_STARTUP1 Register (Offset = 30h) [Reset = 00000000h]

Register to configure motor startup settings1

Table 5-20. MOTOR_STARTUP1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	MTR_STARTUP_OPTION	R/W	0h	Motor start-up method 0h = Align 1h = Double align 2h = IPD 3h = Slow first cycle
29-26	ALIGN_SLOW_RAMP_RATE	R/W	0h	Align, slow first cycle and open loop current ramp rate 0h = 0.1A/s 1h = 1A/s 2h = 5A/s 3h = 10A/s 4h = 15A/s 5h = 25A/s 6h = 50A/s 7h = 100A/s 8h = 150A/s 9h = 200A/s Ah = 250A/s Bh = 500A/s Ch = 1000A/s Dh = 2000A/s Eh = 5000A/s Fh = No LimitA/s
25-22	ALIGN_TIME	R/W	0h	Align time 0h = 10ms 1h = 50ms 2h = 100ms 3h = 200ms 4h = 300ms 5h = 400ms 6h = 500ms 7h = 750ms 8h = 1s 9h = 1.5s Ah = 2s Bh = 3s Ch = 4s Dh = 5s Eh = 7.5s Fh = 10s

Table 5-20. MOTOR_STARTUP1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
21-17	ALIGN_OR_SLOW_CURRENT_ILIMIT	R/W	0h	Current limit during Align/Slow First Cycle in % of CURRENT_BASE 0h = 7.5% 1h = 8.0% 2h = 8.5% 3h = 9.0% 4h = 9.5% 5h = 10% 6h = 11% 7h = 12% 8h = 13% 9h = 14% Ah = 15% Bh = 16% Ch = 17% Dh = 18% Eh = 20% Fh = 22.5% 10h = 25% 11h = 27.5% 12h = 30% 13h = 35% 14h = 40% 15h = 45% 16h = 50% 17h = 55% 18h = 60% 19h = 70% 1Ah = 75% 1Bh = 80% 1Ch = 85% 1Dh = 90% 1Eh = 95% 1Fh = 100%
16-14	IPD_CLK_FREQ	R/W	0h	IPD clock frequency 0h = 50Hz 1h = 100Hz 2h = 250Hz 3h = 500Hz 4h = 1000Hz 5h = 2000Hz 6h = 5000Hz 7h = 10000Hz
13-7	IPD_CURR_THR	R/W	0h	7 bit value for IPD Current limt × CURRENT_BASE / 2^7
6	IPD_RLS_MODE	R/W	0h	IPD release mode 0h = Brake 1h = Tristate
5-4	IPD_ADV_ANGLE	R/W	0h	IPD advance angle 0h = 0° 1h = 30° 2h = 60° 3h = 90°

Table 5-20. MOTOR_STARTUP1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	IPD_REPEAT	R/W	0h	Number of times IPD is executed 0h = 1 time 1h = average of 2 times 2h = average of 3 times 3h = average of 4 times
1	OL_ILIMIT_CONFIG	R/W	0h	Open loop current limit configuration 0h = Open loop current limit defined by OL_ILIMIT 1h = Open loop current limit defined by ILIMIT
0	IQ_RAMP_EN	R/W	0h	Iq ramp down before transition to close loop 0h = Disable Iq ramp down 1h = Enable Iq ramp down

5.2.5 MOTOR_STARTUP2 Register (Offset = 34h) [Reset = 00000000h]

Register to configure motor startup settings2

Table 5-21. MOTOR_STARTUP2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-27	OL_ILIMIT	R/W	0h	Open loop current limit in % of CURRENT_BASE 0h = 7.5% 1h = 8.0% 2h = 8.5% 3h = 9.0% 4h = 9.5% 5h = 10% 6h = 11% 7h = 12% 8h = 13% 9h = 14% Ah = 15% Bh = 16% Ch = 17% Dh = 18% Eh = 20% Fh = 22.5% 10h = 25% 11h = 27.5% 12h = 30% 13h = 35% 14h = 40% 15h = 45% 16h = 50% 17h = 55% 18h = 60% 19h = 70% 1Ah = 75% 1Bh = 80% 1Ch = 85% 1Dh = 90% 1Eh = 95% 1Fh = 100%

Table 5-21. MOTOR_STARTUP2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
26-23	OL_ACC_A1	R/W	0h	Open loop acceleration coefficient A1 0h = 0.01Hz/s 1h = 0.05Hz/s 2h = 1Hz/s 3h = 2.5Hz/s 4h = 5Hz/s 5h = 10Hz/s 6h = 25Hz/s 7h = 50Hz/s 8h = 75Hz/s 9h = 100Hz/s Ah = 250Hz/s Bh = 500Hz/s Ch = 750Hz/s Dh = 1000Hz/s Eh = 5000Hz/s Fh = 10000Hz/s
22-19	OL_ACC_A2	R/W	0h	Open loop acceleration coefficient A2 0h = 0.0Hz/s2 1h = 0.05Hz/s2 2h = 1Hz/s2 3h = 2.5Hz/s2 4h = 5Hz/s2 5h = 10Hz/s2 6h = 25Hz/s2 7h = 50Hz/s2 8h = 75Hz/s2 9h = 100Hz/s2 Ah = 250Hz/s2 Bh = 500Hz/s2 Ch = 750Hz/s2 Dh = 1000Hz/s2 Eh = 5000Hz/s2 Fh = 10000Hz/s2
18	AUTO_HANDOFF_EN	R/W	0h	Auto handoff enable 0h = Disable Auto Handoff (and use OPN_CL_HANDOFF_THR) 1h = Enable Auto Handoff

Table 5-21. MOTOR_STARTUP2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
17-13	OPN_CL_HANDOFF_THR	R/W	0h	Open to close loop handoff threshold (% of MAX_SPEED) 0h = 1% 1h = 2% 2h = 3% 3h = 4% 4h = 5% 5h = 6% 6h = 7% 7h = 8% 8h = 9% 9h = 10% Ah = 11% Bh = 12% Ch = 13% Dh = 14% Eh = 15% Fh = 16% 10h = 17% 11h = 18% 12h = 19% 13h = 20% 14h = 22.5% 15h = 25% 16h = 27.5% 17h = 30% 18h = 32.5% 19h = 35% 1Ah = 37.5% 1Bh = 40% 1Ch = 42.5% 1Dh = 45% 1Eh = 47.5% 1Fh = 50%

Table 5-21. MOTOR_STARTUP2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
12-8	ALIGN_ANGLE	R/W	0h	Align angle 0h = 0° 1h = 10° 2h = 20° 3h = 30° 4h = 45° 5h = 60° 6h = 70° 7h = 80° 8h = 90° 9h = 110° Ah = 120° Bh = 135° Ch = 150° Dh = 160° Eh = 170° Fh = 180° 10h = 190° 11h = 210° 12h = 225° 13h = 240° 14h = 250° 15h = 260° 16h = 270° 17h = 280° 18h = 290° 19h = 315° 1Ah = 330° 1Bh = 340° 1Ch = 350° 1Dh = N/A 1Eh = N/A 1Fh = N/A
7-4	SLOW_FIRST_CYC_FREQ	R/W	0h	Frequency of first cycle in close loop startup (% of MAX_SPEED) 0h = 1% 1h = 2% 2h = 3% 3h = 5% 4h = 7.5% 5h = 10% 6h = 12.5% 7h = 15% 8h = 17.5% 9h = 20% Ah = 25% Bh = 30% Ch = 35% Dh = 40% Eh = 45% Fh = 50%
3	FIRST_CYCLE_FREQ_SEL	R/W	0h	First cycle frequency in open loop for align, double align and IPD startup options 0h = Defined by SLOW_FIRST_CYC_FREQ 1h = 0Hz

Table 5-21. MOTOR_STARTUP2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2-0	THETA_ERROR_RAMP_RATE	R/W	0h	Ramp rate for reducing difference between estimated theta and open loop theta 0h = 0.01 deg/ms 1h = 0.05 deg/ms 2h = 0.1 deg/ms 3h = 0.15 deg/ms 4h = 0.2 deg/ms 5h = 0.5 deg/ms 6h = 1 deg/ms 7h = 2 deg/ms

5.2.6 CLOSED_LOOP1 Register (Offset = 38h) [Reset = 00000000h]

Register to configure close loop settings1

Table 5-22. CLOSED_LOOP1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-27	RESERVED	R/W	0h	Reserved
26-22	ILIMIT	R/W	0h	Current limit in Closed loop Torque Mode and Closed loop Speed control in % of CURRENT_BASE 0h = 7.5% 1h = 8.0% 2h = 8.5% 3h = 9.0% 4h = 9.5% 5h = 10% 6h = 11% 7h = 12% 8h = 13% 9h = 14% Ah = 15% Bh = 16% Ch = 17% Dh = 18% Eh = 20% Fh = 22.5% 10h = 25% 11h = 27.5% 12h = 30% 13h = 35% 14h = 40% 15h = 45% 16h = 50% 17h = 55% 18h = 60% 19h = 70% 1Ah = 75% 1Bh = 80% 1Ch = 85% 1Dh = 90% 1Eh = 95% 1Fh = 100%
21-20	MTR_STOP	R/W	0h	Motor stop method 0h = Hi-z 1h = Active spin down 2h = Braking 3h = Reserved
19	OVERMODULATION_ENABLE	R/W	0h	Overmodulation enable 0h = Disable Over Modulation 1h = Enable Over Modulation

Table 5-22. CLOSED_LOOP1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18-14	CL_ACC	R/W	0h	Closed loop acceleration 0h = 0.5Hz/s 1h = 1Hz/s 2h = 2.5Hz/s 3h = 5Hz/s 4h = 7.5Hz/s 5h = 10Hz/s 6h = 20Hz/s 7h = 40Hz/s 8h = 60Hz/s 9h = 80Hz/s Ah = 100Hz/s Bh = 200Hz/s Ch = 300Hz/s Dh = 400Hz/s Eh = 500Hz/s Fh = 600Hz/s 10h = 700Hz/s 11h = 800Hz/s 12h = 900Hz/s 13h = 1000Hz/s 14h = 2000Hz/s 15h = 4000Hz/s 16h = 6000Hz/s 17h = 8000Hz/s 18h = 10000Hz/s 19h = 20000Hz/s 1Ah = 30000Hz/s 1Bh = 40000Hz/s 1Ch = 50000Hz/s 1Dh = 60000Hz/s 1Eh = 70000Hz/s 1Fh = No limit
13	CL_DEC_CONFIG	R/W	0h	Closed loop deceleration configuration 0h = Closed loop deceleration defined by CL_DEC 1h = Closed loop deceleration defined by CL_ACC

Table 5-22. CLOSED_LOOP1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
12-8	CL_DEC	R/W	0h	Closed loop deceleration. This register is used only if AVS is disabled and CL_DEC_CONFIG is set to '0' 0h = 0.5Hz/s 1h = 1Hz/s 2h = 2.5Hz/s 3h = 5Hz/s 4h = 7.5Hz/s 5h = 10Hz/s 6h = 20Hz/s 7h = 40Hz/s 8h = 60Hz/s 9h = 80Hz/s Ah = 100Hz/s Bh = 200Hz/s Ch = 300Hz/s Dh = 400Hz/s Eh = 500Hz/s Fh = 600Hz/s 10h = 700Hz/s 11h = 800Hz/s 12h = 900Hz/s 13h = 1000Hz/s 14h = 2000Hz/s 15h = 4000Hz/s 16h = 6000Hz/s 17h = 8000Hz/s 18h = 10000Hz/s 19h = 20000Hz/s 1Ah = 30000Hz/s 1Bh = 40000Hz/s 1Ch = 50000Hz/s 1Dh = 60000Hz/s 1Eh = 70000Hz/s 1Fh = No limit
7-8	PWM_FREQ_OUT	R/W	0h	Output PWM switching frequency 0h = 10kHz 1h = 15kHz 2h = 20kHz 3h = 25kHz 4h = 30kHz 5h = 35kHz 6h = 40kHz 7h = 45kHz 8h = 50kHz 9h = 55kHz Ah = 60kHz Bh = 65kHz Ch = 70kHz Dh = 75kHz Eh = N/A Fh = N/A
14	PWM_MODE	R/W	0h	PWM modulation 0h = Continuous Space Vector Modulation 1h = Discontinuous Space Vector Modulation

Table 5-22. CLOSED_LOOP1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3	AVS_EN	R/W	0h	AVS enable 0h = Disable 1h = Enable
2	DEADTIME_COMP_EN	R/W	0h	Deadtime compensation enable 0h = Disable 1h = Enable
1	SPEED_LOOP_DIS	R/W	0h	Speed loop disable 0h = Enable 1h = Disable

5.2.7 CLOSED_LOOP2 Register (Offset = 3Ch) [Reset = 00000000h]

Register to configure close loop settings2

Table 5-23. CLOSED_LOOP2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	ACT_SPIN_THR	R/W	0h	Speed threshold for active spin down (% of MAX_SPEED) 0h = 100% 1h = 90% 2h = 80% 3h = 70% 4h = 60% 5h = 50% 6h = 45% 7h = 40% 8h = 35% 9h = 30% Ah = 25% Bh = 20% Ch = 15% Dh = 10% Eh = 5% Fh = 2.5%
27-24.	BRAKE_SPEED_THRESHOLD	R/W	0h	Speed threshold for BRAKE pin and motor stop options (Low Side Braking or align braking) (% of MAX_SPEED) 0h = 100% 1h = 90% 2h = 80% 3h = 70% 4h = 60% 5h = 50% 6h = 45% 7h = 40% 8h = 35% 9h = 30% Ah = 25% Bh = 20% Ch = 15% Dh = 10% Eh = 5% Fh = 2.5%

Table 5-23. CLOSED_LOOP2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
23-19	BRK_CURR_THR	R/W	0h	Brake current limit in % of CURRENT_BASE 0h = 7.5% 1h = 8.0% 2h = 8.5% 3h = 9.0% 4h = 9.5% 5h = 10% 6h = 11% 7h = 12% 8h = 13% 9h = 14% Ah = 15% Bh = 16% Ch = 17% Dh = 18% Eh = 20% Fh = 22.5% 10h = 25% 11h = 27.5% 12h = 30% 13h = 35% 14h = 40% 15h = 45% 16h = 50% 17h = 55% 18h = 60% 19h = 70% 1Ah = 75% 1Bh = 80% 1Ch = 85% 1Dh = 90% 1Eh = 95% 1Fh = 100%

5.2.8 FAULT_CONFIG1 Register (Offset = 40h) [Reset = 00000000h]

Register to configure fault settings1

Table 5-24. FAULT_CONFIG1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R/W	0h	Reserved
5-2	LCK_RETRY	R/W	0h	Lock detection retry time 0h = 100ms 1h = 500ms 2h = 1s 3h = 2s 4h = 3s 5h = 4s 6h = 5s 7h = 6s 8h = 7s 9h = 8s Ah = 9s Bh = 10s Ch = 11s Dh = 12s Eh = 13s Fh = 14s
1-0	MTR_LCK_MODE	R/W	0h	Motor Lock Mode 0h = Motor lock detection causes latched fault; nFAULT active; 1h = Fault automatically cleared after LCK_RETRY time. 2h = Motor lock in report only mode. 3h = Motor lock detection is disabled

5.2.9 FAULT_CONFIG2 Register (Offset = 44h) [Reset = 00000000h]

Register to configure fault settings2

Table 5-25. FAULT_CONFIG2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-27	RESERVED	R/W	0h	Reserved
26	LOCK1_EN	R/W	0h	Lock 1 : Abnormal speed enable 0h = Disable 1h = Enable
25	LOCK2_EN	R/W	0h	Lock 2 : Abnormal BEMF enable 0h = Disable 1h = Enable
24	LOCK3_EN	R/W	0h	Lock 3 : No motor enable 0h = Disable 1h = Enable
23-21	LOCK_ABN_SPEED	R/W	0h	Abnormal speed lock threshold (% of MAX_SPEED) 0h = 130% 1h = 140% 2h = 150% 3h = 160% 4h = 170% 5h = 180% 6h = 190% 7h = 200%
20-18	ABNORMAL_BEMF_THR	R/W	0h	Abnormal BEMF lock threshold (% of expected BEMF) 0h = 10% 1h = 20% 2h = 30% 3h = 40% 4h = 50% 5h = 60% 6h = 70% 7h = 80%

Table 5-25. FAULT_CONFIG2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
17-13	NO_MTR_THR	R/W	0h	No Motor current limit in % of CURRENT_BASE 0h = 7.5% 1h = 8.0% 2h = 8.5% 3h = 9.0% 4h = 9.5% 5h = 10% 6h = 11% 7h = 12% 8h = 13% 9h = 14% Ah = 15% Bh = 16% Ch = 17% Dh = 18% Eh = 20% Fh = 22.5% 10h = 25% 11h = 27.5% 12h = 30% 13h = 35% 14h = 40% 15h = 45% 16h = 50% 17h = 55% 18h = 60% 19h = 70% 1Ah = 75% 1Bh = 80% 1Ch = 85% 1Dh = 90% 1Eh = 95% 1Fh = 100%
12-8	RESERVED	R/W	0h	Reserved.
7-5	MIN_VM_MOTOR	R/W	0h	Minimum voltage for running motor in % of BASE_VOLTAGE 0h = No Limit 1h = 5% 2h = 10% 3h = 12% 4h = 15% 5h = 18% 6h = 20% 7h = 25%
4	MIN_VM_MODE	R/W	0h	Undervoltage fault mode 0h = Latch on Undervoltage 1h = Automatic clear if voltage in bounds

Table 5-25. FAULT_CONFIG2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-1	MAX_VM_MOTOR	R/W	0h	Maximum voltage for running motor in % of BASE_VOLTAGE 0h = 60% 1h = 65% 2h = 70% 3h = 75% 4h = 80% 5h = 85% 6h = 90% 7h = Max Voltage
0	MAX_VM_MODE	R/W	0h	Overvoltage fault mode 0h = Latch on Overvoltage 1h = Automatic clear if voltage in bounds

5.2.10 MISC_ALGO Register (Offset = 48h) [Reset = 00000000h]

Register to multiple miscellaneous Algorithm Configuration,

Table 5-26. MISC_ALGO Register Field Descriptions

Bit	Field	Type	Reset	Description
31-20	RESERVED	R/W	0h	Reserved
19-16	CL_SLOW_ACC	R/W	0h	Close loop acceleration when estimator is not yet fully aligned 0h = 0.1Hz/s 1h = 1Hz/s 2h = 2Hz/s 3h = 3Hz/s 4h = 5Hz/s 5h = 10Hz/s 6h = 20Hz/s 7h = 30Hz/s 8h = 40Hz/s 9h = 50Hz/s Ah = 100Hz/s Bh = 200Hz/s Ch = 500Hz/s Dh = 750Hz/s Eh = 1000Hz/s Fh = 2000Hz/s
15	IPD_HIGH_RESOLUTION_EN	R/W	0h	IPD high resolution enable 0h = Disable 1h = Enable
14	FAST_ISD_EN	R/W	0h	Fast initial speed detection enable 0h = Disable Fast ISD 1h = Enable Fast ISD
13-12	ISD_STOP_TIME	R/W	0h	Persistence time for declaring motor has stopped 0h = 1ms 1h = 5ms 2h = 50ms 3h = 100ms
11-10	ISD_RUN_TIME	R/W	0h	Persistence time for declaring motor is running 0h = 1ms 1h = 5ms 2h = 50ms 3h = 100ms
9-8	ISD_TIMEOUT	R/W	0h	Timeout in case ISD is unable to reliably detect speed or direction 0h = 500ms 1h = 750ms 2h = 1000ms 3h = 2000ms
7-5	AUTO_HANDOFF_MIN_BEMF	R/W	0h	Minimum BEMF for handoff 0h = 0mV 1h = 50mV 2h = 100mV 3h = 250mV 4h = 500mV 5h = 1000mV 6h = 1250mV 7h = 1500mV

Table 5-26. MISC_ALGO Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4-3	BRAKE_CURRENT_PERSIST	R/W	0h	Persistence time for current below threshold during brake 0h = 50ms 1h = 100ms 2h = 250ms 3h = 500ms
2-0	REV_DRV_OPEN_LOOP_DEC	R/W	0h	% of open loop acceleration to be applied during open loop deceleration in reverse drive 0h = 50% 1h = 60% 2h = 70% 3h = 80% 4h = 90% 5h = 100% 6h = 125% 7h = 150%

5.2.11 PIN_CONFIG Register (Offset = 4Ch) [Reset = 00000000h]

Register to configure hardware pins

Table 5-27. PIN_CONFIG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-20	RESERVED	R/W	0h	Reserved
19	VDC_FILT_DIS	R/W	0h	Vdc Filter Disable 0h = Enabled 1h = Disabled
18-3	RESERVED	R/W	0h	Reserved
2	BRAKE_PIN_MODE	R/W	0h	Brake pin mode 0h = Low side Brake 1h = Align Brake
1-0	BRAKE_INPUT	R/W	0h	Brake pin override 0h = Hardware Pin BRAKE 1h = Override pin and brake / align according to BRAKE_PIN_MODE 2h = Override pin and do not brake / align 3h = Hardware Pin BRAKE

5.2.12 PERI_CONFIG Register (Offset = 50h) [Reset = 00000000h]

Register to peripheral

Table 5-28. PERI_CONFIG1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-13	RESERVED	R	0h	Reserved
12-9	MCU_DEAD_TIME	R/W	0h	Dead time applied between the High Side and Low side switches = 50ns × MCU_DEAD_TIME

Table 5-28. PERI_CONFIG1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
8-4	BUS_CURRENT_LIMIT	R/W	0h	Bus Current Limit in % of CURRENT_BASE 0h = 7.5% 1h = 8.0% 2h = 8.5% 3h = 9.0% 4h = 9.5% 5h = 10% 6h = 11% 7h = 12% 8h = 13% 9h = 14% Ah = 15% Bh = 16% Ch = 17% Dh = 18% Eh = 20% Fh = 22.5% 10h = 25% 11h = 27.5% 12h = 30% 13h = 35% 14h = 40% 15h = 45% 16h = 50% 17h = 55% 18h = 60% 19h = 70% 1Ah = 75% 1Bh = 80% 1Ch = 85% 1Dh = 90% 1Eh = 95% 1Fh = 100%
3	BUS_CURRENT_LIMIT_ENABLE	R/W	0h	Bus current limit enable 0h = Disable 1h = Enable
2-1	DIR_INPUT	R/W	0h	DIR pin override 0h = Hardware Pin DIR 1h = Override DIR pin with clockwise rotation OUTA-OUTB-OUTC 2h = Override DIR pin with counter clockwise rotation OUTA-OUTC-OUTB 3h = Hardware Pin DIR
0	DIR_CHANGE_MODE	R/W	0h	Response to change of DIR pin status 0h = Follow motor stop options and ISD routine on detecting DIR change 1h = Change the direction through Reverse Drive while continuously driving the motor

6 Basic Tuning

The goal of this section is to help spin user motors successfully in closed loop with minimal configurations. This section provides standardized mandatory steps to tune parameters for successful Motor spin-up in closed loop. "**Closed loop"** is defined as sensorless closed loop Field-oriented control where the motor spins at the commanded Speed/Torque reference.

6.1 System Configuration Parameters

The system configuration defines the primary parameters associated with the motor control system to start the motor spinning in closed loop torque/speed control modes.

6.1.1 Configuring System Parameters from GUI

Configure the system parameters using the **System Configuration** page in the GUI as shown below. If the parameters are already programmed in the firmware for a given system, the GUI page displays the default programmed values upon pressing READ ALL REGS. These parameters to be updated accordingly from the below steps.

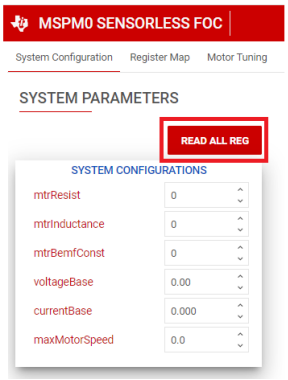

Figure 6-1. GUI System Parameter Configuration

6.1.2 Motor Resistance in Milliohms (mΩ)

Using the motor data sheet, the user can input the motor phase resistance in milliohms (mΩ) using the *mtrResist* parameter in the **System Configuration** page. If the motor does not have a data sheet, then measure the phase-to-phase resistance across any two phases using a digital multimeter and calculate the phase resistance by dividing the phase-to-phase resistance by 2 as shown in Equation 2.

$$\text{Phase resistance} = \text{Measured Phase to Phase Resistance} \times (0.5) \quad (2)$$

The motor phase resistance refers to the equivalent phase to center tap resistance, R_{PH}, as shown in Figure 6-2. This measurement is valid for both star wound and delta wound motors.

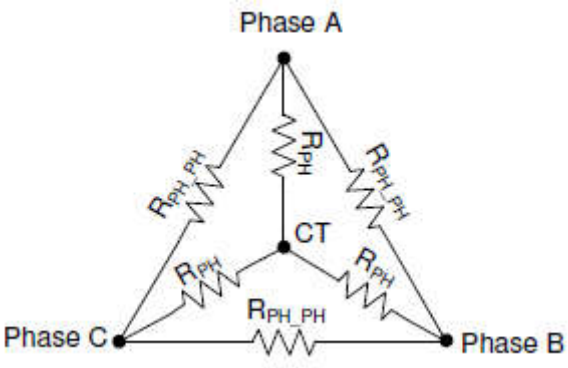

Figure 6-2. Resistance Measurement

6.1.3 Motor Inductance in Microhenries (µH)

From the motor data sheet, input the motor phase inductance in microhenry (µH) using the *mtrInductance* parameter in the **System Configuration** page. To know the motor inductance, measure the phase-to-phase inductance at 1kHz across any two phases using an LCR meter. Calculate the phase inductance by dividing the phase to phase inductance by 2 as shown in Figure 6-3.

$$\text{Phase Inductance} = \text{Measured Phase to Phase Inductance} \times (0.5) \quad (3)$$

Motor phase inductance refers to the inductance from the phase output to the center tap, L_{PH} , as shown in Figure 6-3. For motors with different phase to phase inductances, measure all three phase to phase inductances and calculate the average and use this value as the phase to phase inductance. This measurement is valid for both star wound and delta wound motors.

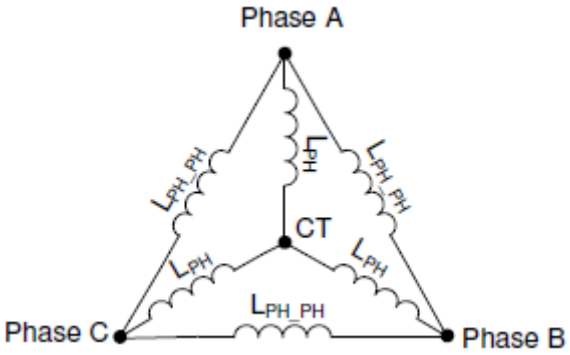

Figure 6-3. Inductance Measurement

6.1.4 Motor BEMF Constant

Using the motor's data sheet, the user can input the motor's BEMF constant Ke in mV/Hz and program *mtrBEMFConst* in the **System Configuration** Page as Ke × 10.

Equation 4 and Equation 5 below can be used to convert Ke in mV/rpm, mV*sec/rad and torque constant Kt to Ke in mV/Hz.

$$\text{BEMF Constant}\left[\frac{\text{mV}}{\text{Hz}}\right] = \frac{\text{Ke}\left[\frac{\text{mV}}{\text{RPM}}\right]*60}{\#\ pole\ pairs} \quad (4)$$

$$\text{BEMF Constant}\left[\frac{\text{mV}}{\text{Hz}}\right] = \frac{\text{Kt}\left[\frac{\text{mN}\cdot\text{m}}{\text{A}}\right]*2\pi}{\#\ pole\ pairs} \quad (5)$$

If the motor does not have a data sheet, measure the voltage across any two phases of the motor using an oscilloscope by manually spinning the motor. A sinusoidal or trapezoidal voltage appears on the oscilloscope. Measure the peak voltage Ep in milli-volts and time period Tp in seconds. Calculate BEMF constant Ke as shown in Equation 6.

$$\text{Bemf Constant Ke} = \text{Ep} * \text{Tp} / \sqrt{3} \quad (6)$$

6.1.5 Base Voltage (V)

Base voltage represents the maximum measurable bus voltage and phase voltages in the motor control system. Input the system base voltage (in volts) in the *voltageBase* parameter of the **System Configuration** page in GUI. The user can compute the system base voltage based on the Voltage scaling resistor divider bridge values R1 and R2 and the Full Scale ADC voltage (FSV) of 3.3V as shown in Equation 7. See Figure 2-3 for hardware configuration of the voltage divider scaling ratio.

$$\text{Base}_{\text{Voltage}} = \frac{\text{ADC Full Scale Value}}{\text{Voltage Divider Scaling Ratio}} = \frac{3.3V}{\frac{R1}{R1+R2}} \quad (7)$$

For example, in a system with resistor divider scaling ratio of 1/20 from DC supply voltage to ADC input, the base voltage or maximum measurable system voltage by the ADC is 3.3V × (1/20) = **66V**.

6.1.6 Base Current (A)

Base current represents the maximum measurable motor phase current in the motor control system. The user inputs the system base current (in Amps) in the *currentBase* parameter of the **System Configuration** page in GUI. The user can compute the system base current based on the current sense amplifier gain (CSAGAIN) in volts/amp and the full-scale ADC voltage (FSV) of 3.3V as shown in Equation 8. There is a factor of 2 considered to support bidirectional current sensing with 1.65V as the zero-current offset.

$$\text{Base}_{\text{Current}} = \frac{\text{ADC Full Scale Value}}{2{*}CSAGAIN\left[\frac{V}{A}\right]} = \frac{3.3\text{V}}{2{*}CSAGAIN\left[\frac{V}{A}\right]} \tag{8}$$

For example, in a system with CSAGAIN = 0.15V/A , the base current or maximum measurable system current by the ADC is 3.3V / (2 × 0.15V/A) = **11A**.

Note

In some driver devices, CSAGAIN can be set as a register over I2C or SPI or by hardware using a resistor value. See the driver data sheet for how to configure the driver CSAGAIN setting.

If the system uses a current sense resistor (R_{SENSE}) with CSAGAIN units mentioned in volts/volt (V/V), the CSA gain in volts/amp can be computed using Equation 9.

$$CSAGAIN\left[\frac{V}{A}\right] = \text{R}_{\text{SENSE}} \times CSAGAIN\left[\frac{V}{V}\right] \tag{9}$$

6.1.7 Maximum Motor Electrical Speed (Hz)

Using the motor's data sheet, the user can input the maximum motor electrical speed in Hz using the *maxMotorSpeed* parameter in the **System Configuration** Page. If this data is not available, the user can input the number of pole pairs and motor mechanical speed in RPM. The user can convert the motor mechanical speed in RPM to motor electrical speed in Hz using Equation 10.

$$f_{Electrical} = \frac{n_{PolePairs} \cdot \omega_{Mechanical}}{60} \tag{10}$$

Where:

- $\omega_{\textbf{Mechanical}}$ is the mechanical speed in units revolutions per minute (RPM)
- $f_{\textbf{Electrical}}$ is the electrical speed in units of hertz (Hz)
- $n_{\textbf{PolePairs}}$ is the number of motor pole pairs

Note

To determinine the number of motor poles without a motor data sheet:

1. Use a lab power supply and make sure the current limit is set to less than the motor rated current. Do not turn on the supply.
2. Connect V+ of the supply to phase A and V- of the supply to phase B of the motor. Any 2 of the 3 phases can be chosen at random if the phases not labeled.
3. Turn on supply, The rotor settles at one position by injecting current.
4. Manually rotate the rotor until rotor snaps to another settle position. Rotor settle down at various positions around one mechanical cycle.
5. Count the number of settle-down positions for one fully mechanical cycle, which is the number of pole pairs. Multiplying by two calculates the number of poles.

Be careful of gearing systems within a motor. The gearing ratio determines how many rotor revolutions correlate to the shaft's mechanical revolution.

6.2 Control Configurations for Basic Motor Spinning

After configuring the system parameters in the GUI, the user can go to the **Register Map** page and configure the Register Map Tuning parameters as shown in Figure 6-4. By default, the firmware has the recommended settings, which can be read into the GUI by pressing the "READ ALL REG" button.

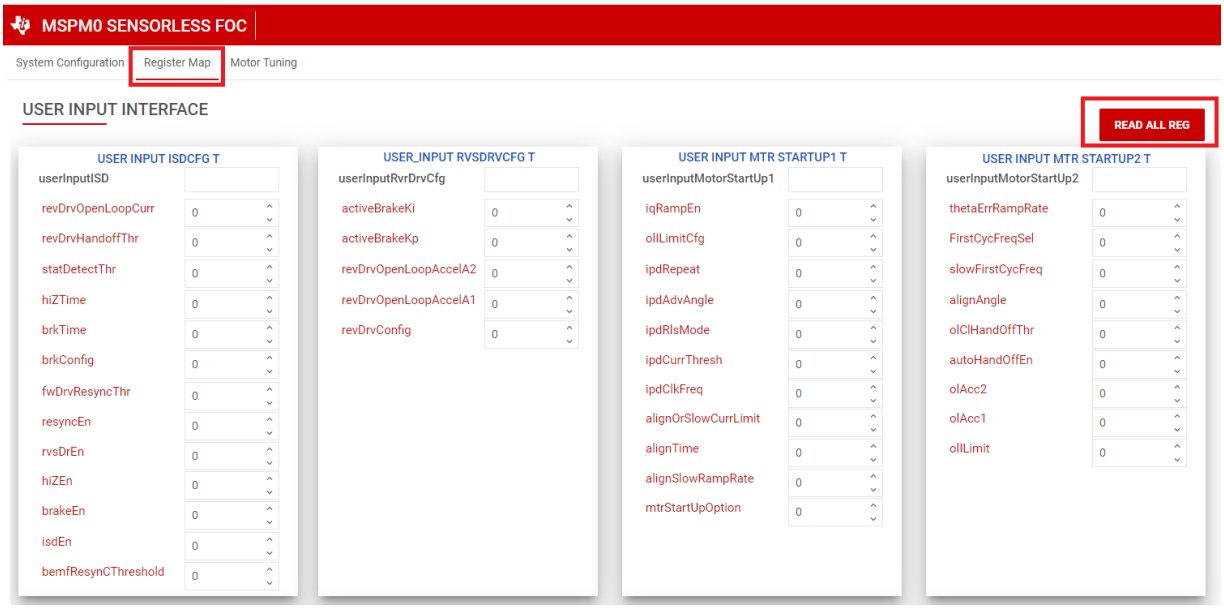

Figure 6-4. Register Map GUI Page

6.2.1 Basic Motor Startup

Sensorless FOC relies on the position detection estimated from BEMF to accurately drive the motor. At startup, since the motor can be at standstill or the motor can be spinning with unknown speeds, the rotor position is unknown.

The FOC algorithm has various startup algorithms to reliably start the motor and ramp the motor with sufficient speeds or estimate the position of an already spinning motor before switching to the estimator for continuous rotor position tracking. The following sections describe the basic configuration needed to start and ramp the motor from standstill until open-loop spin up. If any motor faults are observed during basic open-loop spin up, see Section 6.3.

6.2.1.1 Disable ISD

Initial Speed Detection (ISD) is a feature to enter the motor automatically when the motor is already spinning. This is also called Headwind/Tailwind startup, or Catch-on-the-fly startup. For basic spinning for motor, the ISD feature is disabled by default by setting *isdEn* = 0 on the **Register Map** page. For basic motor tuning, the motor is expected to be at standstill before giving the speed command. To tune the motor for ISD, refer to Section 7.

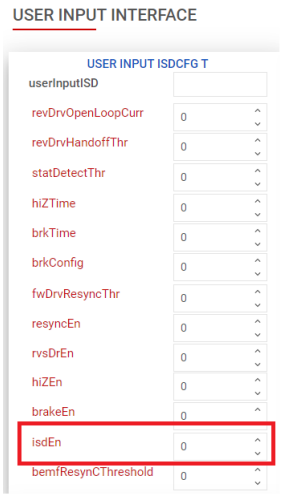

Figure 6-5. Disabling ISD in GUI

6.2.1.2 Motor Start Option - Align

When the motor is ramping up from standstill, the Motor Align startup algorithm forces the rotor to align to a fixed ALIGN_ANGLE with a defined current limit acting as a torque reference for a predefined ALIGN_TIME. By default, the motor startup option is set as Align (*mtrStartUpOption* = 0b) in the MTR_STARTUP of MOTOR_STARTUP1 configuration. For basic spinning, use the default parameters which works for most of the motors.

If the motor fails to align for the given load setup, increase the *alignOrSlowCurrentLimit* parameter on the **Register Map** page. For fine tuning the Motor Align configuration, refer to Section 7.

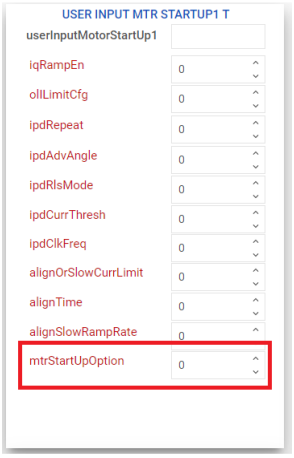

Figure 6-6. Motor Startup in GUI

6.2.1.3 Motor Open Loop Ramp

To estimate the rotor position accurately, the motor needs to build sufficient BEMF before switching to closed loop. During startup, the FOC algorithm accelerates the motor with a second order open loop ramp profile to increase the speed until sufficient BEMF is built. By default, for basic spin up of motor, the open loop ramp up parameters are configured with a linear first order configuration with sluggish acceleration, which works for most of the motors. Disable switching to closed loop control to verify the appropriate functionality of open loop , using *closedloopDis* = 1b in ALGO_DEBUG_CTRL on the **Motor Tuning** page. To further optimize the startup time, refer to Section 7 to fine tune the startup performance.

Depending on the load of the motor, tune the OL_ILIMIT to the lowest possible value for smooth handoff once the closed loop is enabled (*closedloopDis* = 0b).

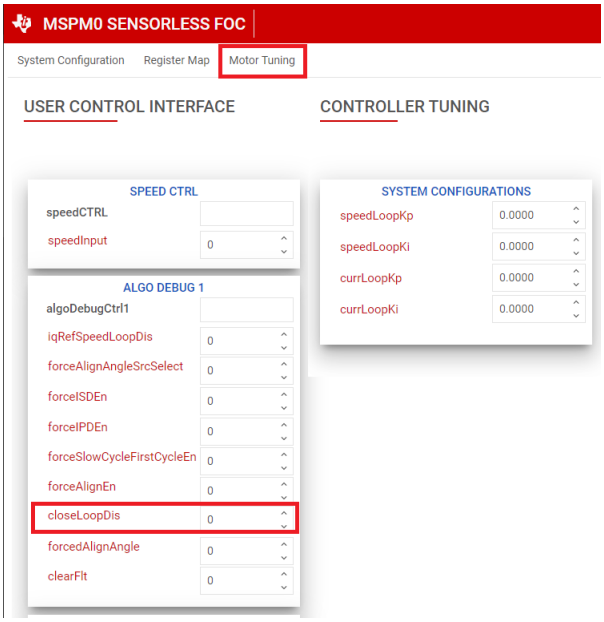

Figure 6-7. Setting ClosedLoop Disable in GUI

6.2.1.4 Motor Open Loop Debug

If motor fails to spin in open loop continuously or if motor current oscillates without spinning the motor, the user can fine tune the open loop configurations in the RAM ALGO DEBUG 2 parameters.

To verify the signal path or check the motor parameters accuracy, the user can disable the current loop by setting the *currLoopDis* bit and setting the *forceVQCurrLoopDis* and *forceVDCurrLoopDis* on the **Register Map** page.

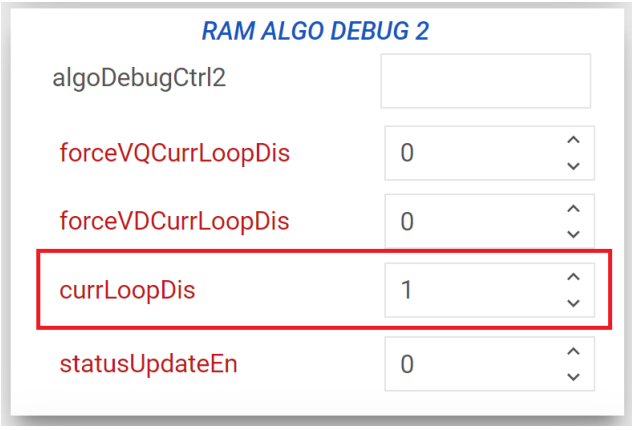

Figure 6-8. Disabling Current loop in GUI

Adjust Vd and Vq values to spin the motor. Once the motor is spinning at a constant speed, observe the Id and Iq outputs in the User Outputs section of the **Motor Tuning** page to check the stability of the current loop.

6.2.2 Controller Configuration for spinning the Motor in Closed Loop

After tuning the open loop with the motor spinning continuously in the open loop state, the user can switch to closed loop by clearing *closedloopDis* = 0b in ALGO_DEBUG_CTRL on the **Motor Tuning** page. Follow the below steps to achieve closed loop speed control.

6.2.2.1 PI Controller Tuning for Closed Loop Speed Control

Current Controller Tuning

The FOC Algorithm uses two current PI controllers: one each for Id and Iq to control flux and torque separately. Kp and Ki coefficients are the same for both PI controllers and are configured through *currLoopKp* and *currLoopKi* in the **Motor Tuning** page.

For the basic tuning, configure *currLoopKp* and *currLoopKi* parameters as "0" so that these values are auto-computed based on the motor parameters and reflected in the User Outputs section of the **Motor Tuning** page in the GUI. These values can be further updated for fine tuning the performance and controlling the dynamics of the system.

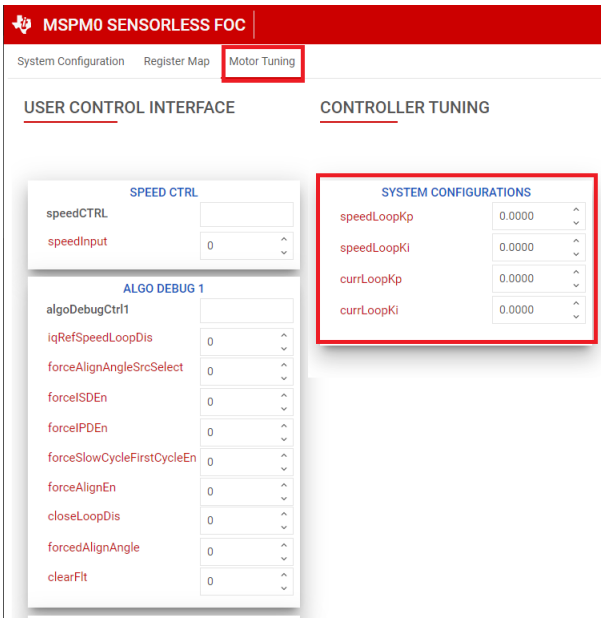

Figure 6-9. PI Loop Tuning in GUI Motor Tuning page

Speed Controller Tuning

The FOC Algorithm uses an integrated speed control loop that helps maintain a constant speed over varying operating conditions. The Kp and Ki coefficients are configured through *speedLoopKp* and *speedLoopKi* in the "System Configurations" section on the **Motor Tuning** page. The output of the speed loop is used to generate the current reference for torque control. The output of the speed loop is limited by configuring *iLIMIT* in the closedLoop1 configuration in the Register Map page of GUI. When output of the speed loop saturates, the integrator is disabled to prevent integral wind-up.

To tune the Kp and Ki values for speed loop:

1. Configure the motor to spin continously in open loop by setting *closedloopDis* to 1b. Disable the automatic handoff by setting *autoHandOffEn* to *0b*.
2. Set the closed loop hand off threshold to around 50% of maximum speed using *olClHandOffThr*.
3. Set the *iqRampEn* bit to *1b* in the userInputMotorStartUp1 register.
4. The current reference gradually decreases and settles down to the lowest possible Iqref to run at the given threshold speed.
5. Speed loop Kp [SPD_LOOP_KP] is calculated using this equation: SpeedLoop K_p = Current Reference at olClHandOffThr in Amps / olClHandOffThr in Hz
6. Speed loop Ki [SPD_LOOP_KI] is calculated using this equation: Speed Loop K_i = Speed Loop K_p × 0.1
7. Enable closed loop by clearing the closedloopDis to (0b) in the configuration in the **Register Map** page of the GUI.

Note

The tuning of speed loop Kp and Ki is experimental. If the above recommendation does not work, manually tune speed loop Kp and Ki until the desired results are achieved.

The following table shows general guidelines to change controller gains.

Parameter	Rise Time	Overshoot	Settling Time	Stead State Error	Stability
Kp	Decreases	Increases	Small Change	Decreases	Degrades
Ki	Decreases	Increases	Increases	Eliminates	Degrades

6.2.2.2 Testing for Successful Startup into Closed Loop

1. **Apply a nonzero speed command**

   Change the "Speed Input Command" to a nonzero value. When the speed command is issued, the device starts to commutate and the motor spins at a speed that is proportional to Speed Command × MAXIMUM MOTOR SPEED / 32767.

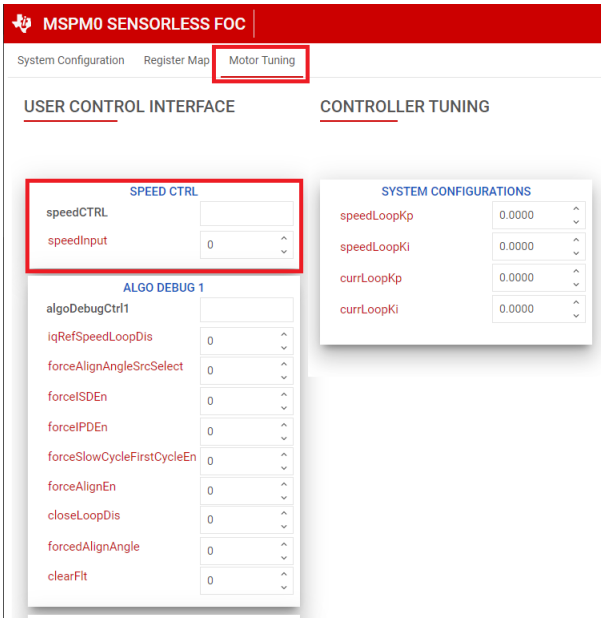

Figure 6-10. Setting Speed Input From GUI

2. **Check if motor spins in closed loop at commanded speed.**

   Enable the "Continous Read status" toggle button towards the bottom right corner of the GUI and monitor the Fault Status register. If no faults is triggered, move to the Section 7 section.

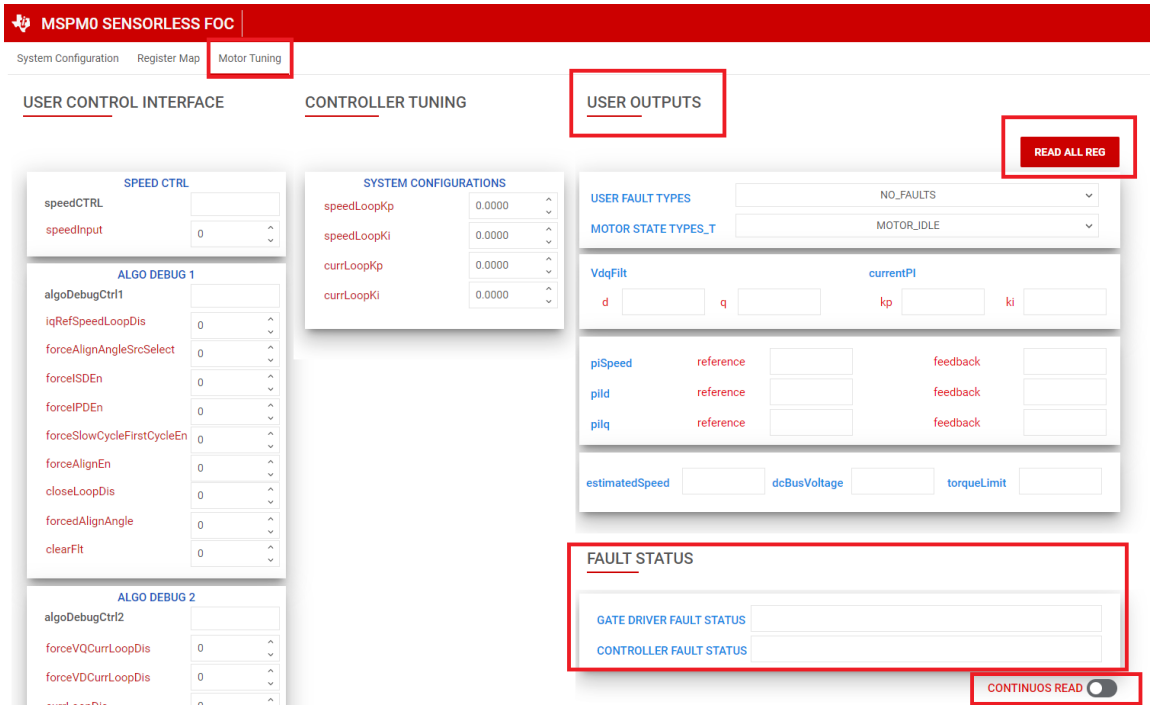

Figure 6-11. Reading Fault Status From GUI

3. If any fault is triggered, tune the configuration for fault handling using these steps:
   a. Set zero speed command by setting the Speed Input command to 0.
   b. Clear the fault status registers by setting the clear fault bit (*ClearFlt)* bit in the ALGO DEBUG CTRL1 register.
   c. Check Section 6.3 for steps to debug faults.

6.3 Fault Handling

The following sections describe faults that can be triggered based on the default register configuration.

6.3.1 Abnormal BEMF Fault [ABN_BEMF]

This fault is triggered when the estimated BEMF voltage drops below the programmed Abnormal BEMF threshold percentage [ABNNORMAL_BEMF_THR]. For example, if the estimated or measured Ke is 5mV/Hz and the programmed Abnormal BEMF threshold is 40%, this fault is triggered when the estimated Ke drops below 2mV/Hz. This fault can also be triggered when the programmed Ke is inaccurate.

There are two cases for Abnormal BEMF threshold:

Case 1: Estimated BEMF voltage drops when the motor speed drops. Motor speed can drop due to load dynamics (sudden change in load). For applications with load dynamics, the speed is expected to drop and recover back. Because the speed drops, the BEMF voltage also drop and can trigger this fault. For such applications, the recommended value of 10% is to be set for the Abnormal BEMF threshold to avoid triggering this fault.

Case 2: This fault can be triggered if the programmed Ke is inaccurate. Follow steps recommended in section MOTOR_BEMF_CONSTANT to obtain accurate Ke.

6.3.2 Monitoring Power Supply Voltage Fluctuations for Voltage Out of Bound Faults

In applications where the power supply fluctuates, user needs to specify the minimum and maximum power supply voltage range. During an undervoltage condition, the motor operates in overmodulation region to achieve the target speed leading to current distortion, inefficiency or noise. During an overvoltage condition, the MOSFETs and motor are stressed with continued operation in high voltage.

Tuning Under Voltage Limit 1: Keep decreasing the supply voltage until there is a drop in the speed. Measure the bus voltage at which the speed drops and set MIN_VM_MOTOR to that value. Range of minimum bus voltage that can be configured is between 0 to 25% of Maximum BASE_VOLTAGE.

Tuning Over Voltage Limit: Keep increasing the bus voltage to a point where the motor phase voltage reaches the maximum rated voltage of the motor. MAX_VM_MOTOR is the bus voltage at which motor phase voltage reaches the maximum rated voltage of the motor. Range of maximum bus voltage that can be configured is between 60% to Maximum BASE_VOLTAGE.

Note

The FOC Algorithm provides an undervoltage recovery mode [MIN_VM_MODE] and an overvoltage recovery mode [MAX_VM_MODE]. Undervoltage recovery mode can be configured to either automatically clear Undervoltage fault [MTR_UNDER_VOLTAGE] or latch on Undervoltage fault. Overvoltage recovery mode can be configured to either automatically clear Overvoltage fault [MTR_OVER_VOLTAGE] or latch on Overvoltage fault.

6.3.3 No Motor Fault [NO_MTR]

This fault is triggered when the phase current is below the No motor lock threshold % of base current.

Step 1: Make sure the motor phases are connected to the terminals as shown in the Hardware User Guide.

Step 2: If the fault persists, decrease the No-Motor lock current threshold [NO_MTR_THR].

7 Advanced Tuning

This section helps you spin motors successfully in closed loop with minimal configurations. This section provides standardized mandatory steps to tune parameters for successful motor spin-up in closed loop. Closed loop is defined as the sensorless closed loop where motor spins at the commanded speed and torque reference.

7.1 Control Configurations Tuning

7.1.1 Initial Speed Detection of the Motor for Reliable Motor Resynchronization

The Initial Speed Detection (ISD) function is used to identify the initial condition of the motor. It is important to know the initial condition of the motor for reliable resynchronization. Motor resynchronization failures can occur when the device attempts to start the motor while the motor is coasting or spinning in the direction opposite to the intended direction of spin. Motors can coast in applications that require frequent motor starts and stops, if the motor is being forced externally, or if there is a power interruption. Motors can spin in the direction opposite to the intended direction of spin if motor phase wires are connected to OUTA, OUTB, and OUTC in wrong sequence or when the wrong direction command is issued. Motors with higher inertia coast for a longer period of time. Enable ISD in applications that require frequent motor starts and stops, and use higher inertia motors.

For example, ceiling fan motors have higher inertia due to the fan blades and can coast for a long time before stopping.

Step 1: Enable ISD [ISD_EN]

Step 2: Enable Motor ISD Resynchronize [RESYNC_EN]

Note

If the motor fails to start:

1. Increase the Motor Stationary BEMF Threshold [STAT_DETECT_THR].
2. Increase the Motor Stationary Persistence Time [ISD_STOP_TIME].
3. Increase the Motor Run Persistence Time [ISD_RUN_TIME].
4. Increase the minimum speed threshold to resynchronize to closed loop.

7.1.2 Unidirectional Motor Drive Detecting Backward Spin

For applications that require spinning the motor in a specific direction, it is important to know if the motor is coasting or spinning in the direction opposite to the intended direction of spin. The MSPM0 FOC Algorithm reverse drive function acts to reverse decelerate the motor through zero speed and to accelerate after changing direction until it transitions into closed loop as shown in Figure 7-1.

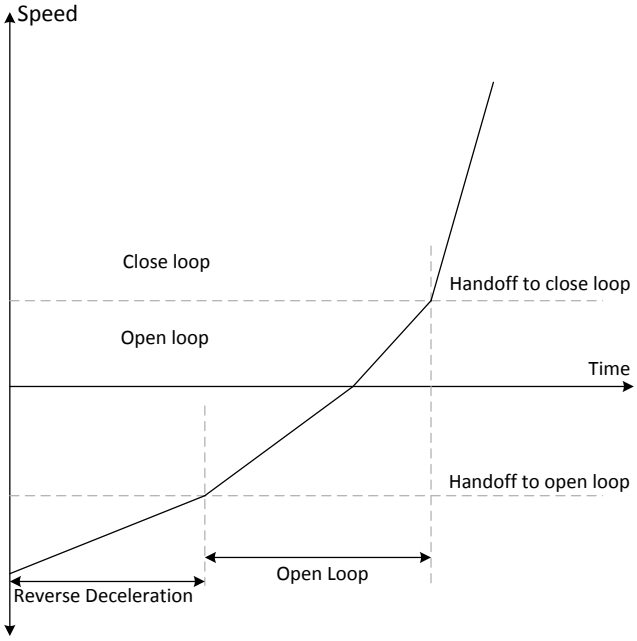

Figure 7-1. Reverse Drive Function

The MSPM0 FOC algorithm provides an option to apply brakes and stop the motor while the motor is coasting or spinning in reverse direction and then accelerate into closed loop after changing the direction.

In applications such as ceiling fans and pumps, it is required to spin the motor in a specific direction for desired results. For such applications, follow these recommendations:

Step 1: Enable ISD [ISD_EN]

Step 2: Enable Motor ISD Reverse drive [RVS_DR_EN]

Step 3: Enable reverse resynchronization [RESYNC_EN]

Note

Follow these recommendations if the motor fails to resynchronize in reverse direction:

1. Increase the reverse deceleration speed threshold to transition to open loop
2. Enable Open loop reverse drive configuration [REV_DRV_CONFIG]
3. Increase the Reverse Drive Open Loop Current Reference [REV_DRV_OPEN_LOOP_CURRENT]
4. Decrease open loop acceleration coefficient A1 and A2 during reverse drive

7.1.3 Preventing Back Spin of Rotor During Startup

For applications where a reverse spin is not acceptable, the Initial Position Detection algorithm (IPD) function is an alternative way to start up the motor. With the proper IPD setting, the motor startup can be faster than using align. While this function is suitable for motors with high inertia, such as heavy blades (for example: a ceiling or appliance fan), it is not suitable for motors with low inertia, such as small blades (for example: a computer fan), because the current injection will cause the motor to shake, resulting in the IPD not being accurate.

In applications where the acoustic noise ("chirp") generated by IPD is not acceptable during startup, select "Slow first cycle" as the startup method.

Option 1: IPD

Step 1: If IPD is chosen as startup method, select IPD in the Motor startup option [MTR_STARTUP] in "USER INPUT MTR_STARTUP1_T configuration of REGISTER MAP" tab in the GUI.

Step 2: Select the IPD Current threshold [IPD_CURR_THR]. IPD current threshold is selected based on the inductance saturation point of the motor. A higher current has better chance to accurately detect the initial position. However, higher current might result in rotor movement, vibration, and noise. Start with 50% of the rated current of the motor. If the motor startup is unsuccessful, increase the threshold until the motor starts successfully. Note that the IPD current threshold should not be higher than the rated current of the motor.

Step 3: Select IPD clock value [IPD_CLK_FREQ]. IPD clock defines how fast the IPD pulses are applied. Higher inductance motors and higher current thresholds need a longer time to settle the current down, so we need set the clock at a slower time. However, a slower clock makes the IPD noise louder and it lasts longer, so we suggest setting the clock as fast as possible as long as IPD current is able to settle down completely.

Note

The device triggers the IPD timeout fault IPD_FAULT_CLOCK_TIMEOUT for motors with very high inductance, or if the motor is not connected. If this fault is triggered, make sure that the motor is connected to the device. If the fault still persists, set the IPD release mode [IPD_RLS_MODE] to Tri-state if any overshoot in DC bus voltage is acceptable.

The device triggers the IPD Frequency fault IPD_FAULT_DECAY_TIME if the IPD clock frequency is set too high. If this fault is triggered, decrease the IPD Clock value [IPD_CLK_FREQ].

Step 4: Select IPD Advance Angle [IPD_ADV_ANGLE]. Start with 90º for maximum startup torque. If there is sudden jerk observed during startup, reduce the angle to 60º or 30º for a smoother startup.

Option 2: Slow first cycle

Follow these steps if Slow first cycle is chosen as the startup method:

Step 1: Select Slow first cycle in the Motor startup option [MTR_STARTUP] in "Control Configuration – Motor Startup Stationary" tab in the GUI.

Step 2: Select align or slow first cycle current reference [ALIGN_OR_SLOW_CURRENT_ILIMIT]. Lower current reference may lose synchronization of motor. Higher current may lead to sustained oscillations for high inertia motors, or sudden jerky motion for low inertia motors. It is recommended to start with 50% of the rated current of the motor. In applications where the startup torque is high, the motor might lose synchronization. In such applications, increase the current reference. In applications where sustained oscillations or sudden jerks are observed, decrease the current reference.

Step 3: Select align or slow first cycle current ramp rate [ALIGN_SLOW_RAMP_RATE]. Current reference is ramped to avoid reverse rotation of the motor. Lower current ramp rate may lose synchronization of motor. A higher current ramp rate may lead to sustained oscillations for high inertia motors, or sudden jerking motion for low inertia motors. Start with setting the ramp time to 0.5 seconds to ramp to rated current of the motor. In applications where the startup torque is high, the motor can lose synchronization. In such applications, increase the current ramp rate. In applications where sustained oscillations or sudden jerks are observed, decrease the current ramp rate.

Step 4: Select the frequency of the first cycle [SLOW_FIRST_CYC_FREQ]. Lower frequency can give a jerky motion at startup. Higher frequency might not be able to synchronize the motor. Start with 20% of the maximum speed of the motor. In applications where the startup torque is high, the motor might lose synchronization. In such applications, decrease the frequency. In applications where jerky motions are observed, increase the frequency.

7.1.4 Gradual and Smooth Start up Motion

For applications that require slow and gradual startup with lower speed overshoots during handoff, follow the below recommendations:

Step 1: Decrease Open loop acceleration coefficient A1 [OL_ACC_A1] and Open loop acceleration coefficient A2 [OL_ACC_A2].

Step 2: Enable Iq ramp down after transition to closed loop [IQ_RAMP_EN]

If there is speed overshoots, decrease ramp rate for reducing difference between estimated theta and open loop theta [THETA_ERROR_RAMP_RATE].

7.1.5 Faster Startup Timing

Startup time is the time taken for the motor to reach the target speed from zero speed. For applications that require quick startup time, we recommend choosing either Initial Position Detection (IPD) or Slow first cycle as the startup method.

Option 1: Initial Position Detection (IPD)

Step 1: Select IPD [MTR_STARTUP] as the motor startup method.

Step 2: Increase IPD current threshold [IPD_CURR_THR] to rated current of the motor.

Step 3: Increase IPD clock value [IPD_CLK_FREQ] to higher frequency up to a value where the device does not trigger IPD frequency fault. Check Section 7.1.3 (Step 3) for more details.

Step 4: Select IPD repeating times [IPD_REPEAT] to 1 time.

Step 5: Select Open loop current limit [OL_ILIMIT] to be the same as Current limit for Torque PI Loop [ILIMIT].

Note

Configuring current Limits to a value higher than motor stall current overheats or damages the motor.

Step 6: Increase Open loop acceleration coefficient A1 [OL_ACC_A1] and Open loop acceleration coefficient A2 [OL_ACC_A2].

Step 7: Select Minimum BEMF for handoff [AUTO_HANDOFF_MIN_BEMF] to 0mV.

If the device triggers Abnormal BEMF [ABN_BEMF] fault, then recommended to increase the [AUTO_HANDOFF_MIN_BEMF].

Step 8: Keep increasing ramp rate for reducing difference between estimated theta and open loop theta to 2 deg/ms.

Step 9: Increase Closed loop acceleration rate [CL_ACC]

Option 2: Slow first cycle

Step 1: Select Slow first cycle as the motor startup method in [MTR_STARTUP].

Step 2: Select Align or slow first cycle current limit [ALIGN_OR_SLOW_CURRENT_ILIMIT] to be the same as Current limit for Torque PI loop [ILIMIT].

Step 3: Keep increasing Align or slow first cycle current ramp rate [ALIGN_SLOW_RAMP_RATE] until the open loop current reaches 100% of the rated current of the motor.

Step 4: Follow Step 5 to Step 9 in Option 1.

7.1.6 Stopping Motor Quickly

For applications that require stopping the motor quickly, configure Motor stop options [MTR_STOP] to either Low side braking:

Step 1: Configure Motor stop options [MTR_STOP] to Low side braking.

Step 2: Select Speed threshold for BRAKE pin and Motor STOP options. Setting speed threshold to higher speed can result in FETs carrying large current. Setting speed threshold to lower speed results in increase in the stop time of the motor. Recommended to start with 50% of the maximum speed, If the motor phase current exceeds the FET's maximum current rating, then decrease the threshold. If the stop time is too high, then recommended to increase the threshold without hitting the maximum current limit.

7.1.7 Preventing Supply Voltage Overshoot During Motor Stop.

For applications that require preventing supply voltage overshoots during motor stop, select active spin down as Motor stop options. Active spin down can be used as a motor stop option in applications where fast stop is not required but some amount of inductive energy going back to power supply is acceptable

Step 1: Configure Motor stop options [MTR_STOP] to Active spin down

Step 2: Configure active spin down speed threshold [ACT_SPIN_THR]. It is recommended to set the ACT_SPIN_THR to 50% of the maximum speed. If there is voltage overshoot seen on the power supply, decrease the ACT_SPIN_THR till the voltage overshoot reaches acceptable limit.

7.1.8 Protecting the Power supply

Protecting the power supply from drawing higher current or potential voltage overshoots is important in battery operated applications or applications that do not have an internal overcurrent or overvoltage protection built into the power supply.

Step 1: When the load on the motor increases, the device draws higher current from the power supply. To limit the current drawn from the power supply, enable bus current limit [BUS_CURRENT_LIMIT_ENABLE] and configure the bus current limit [BUS_CURRENT_LIMIT] to protect the power supply from drawing higher current.

For example, limiting the current drawn from power supplies such as batteries is required because the battery life depends on the charge and discharge cycles. Enabling the bus current limit limits the power supply current by limiting the speed of the motor.

Step 2: When a command is issued for the motor to decelerate, based on the deceleration rate, energy from the motor can be pumped back to the power supply, increasing the supply voltage to levels that are possibly unsafe for electronics. Enable the antivoltage surge [AVS] to protect the power supply from voltage overshoots that override any deceleration limit set by any other register and automatically apply a safe deceleration rate.

Figure 7-2 shows overshoot in power supply voltage when AVS is disabled. Motor decelerates from 100% duty cycle to 10% duty cycle at a deceleration rate of 70000Hz/s. Figure 7-3 shows no overshoot in power supply voltage when AVS is enabled.

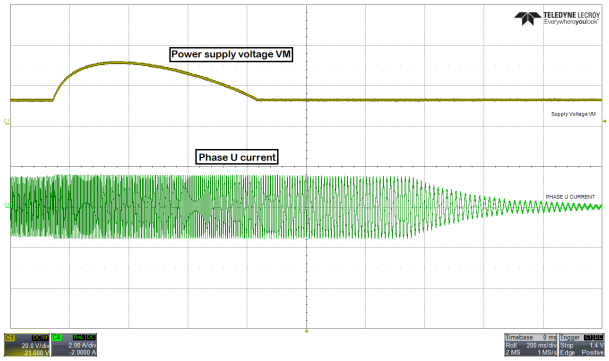

Figure 7-2. Power Supply Voltage and Phase Current Waveform When AVS is Disabled

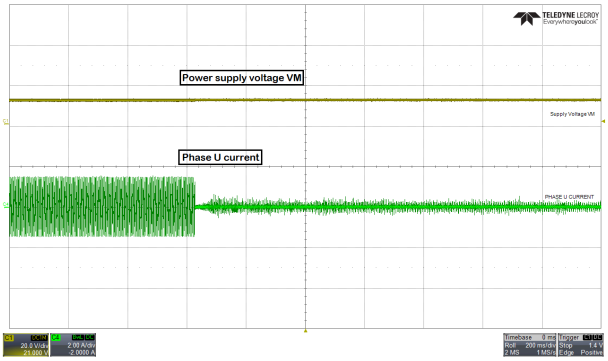

Figure 7-3. Power Supply Voltage and Phase Current Waveform When AVS is Enabled

7.2 Hardware Configurations

7.2.1 Direction Configuration

The FOC algorithm lets you set the direction of the motor using a register-based direction configuration:

- **Register-based direction configuration**

  The direction of motor spin can be set based on register setting as shows below.

  DIR_INPUT 01b: apply phase sequence OUT A → OUT B →OUT C .

  DIR_INPUT 10b: apply phase sequence OUT A → OUT C → OUT B.

7.2.2 Brake Configuration

FOC Algorithm enables user to brake the motor under various scenarios. Brake state can be configured to either low side braking (Low-Side Braking) or align brake (Align Braking) through *BRAKE_PIN_MODE*. FOC Algorithm decreases output speed to value defined by *BRAKE_SPEED_THRESHOLD* before entering brake state. As long as BRAKE is driven 'High', motor stays in brake state. Brake functionality can be achieved the following way:

- **Register based Brake configuration**.

  User can configure the *BRAKE_INPUT* in *PIN_CONFIG* register to apply the brakes using register settings as below.

  – *BRAKE_INPUT - 1b*: Override pin and brake / align according to *BRAKE_PIN_MODE*
  – *BRAKE_INPUT - 10b*: Override pin and do not brake / align

7.2.3 Real Time Variable Tracking

32-bit algorithm variables can be output in real time from the MCU through the DAC. DAC output is enabled by setting DAC_EN = 1. The DAC in MSPM0 is 12 bit, thus a scaling needs to be applied before output. User has two ways to scale the variable before output.

- **For variables in global IQ format(IQ27):**

  DAC_OUTPUT_VOLTAGE = (VARIABLE_VALUE × DAC_SCALING_FACTOR + 1) × 1.65V

  In the above eqaution setting the DAC_SCALING_FACTOR to 1 enables user to represent a data of IQ(1.0) to IQ(-1.0) in between 0 - 3.3V. To represent the data exceeding the value 1.0 use higher DAC_SCALING_FACTOR.

  For Example: To represent a data from -2.0 to +2.0 , set the DAC_SCALING_FACTOR to 0.5.
- **For variables in other IQformat:**

  For output of any other IQ, user can left shit or right shift the variable to bring the data in a 12 bit range before output.This mode is selected by setting DAC_SCALING_FACTOR to 0.

  If variable value is less than a 12-bit value, set DAC_SCALE to positive, the DAC output follows as below:

  DAC_OUTPUT_VOLTAGE = (VARIABLE_VALUE << DAC_SCALE) × 3.3V

  If variable value is greater than a 12-bit value, set DAC_SCALE to negative, the DAC output follows as below

  DAC_OUTPUT_VOLTAGE = (VARIABLE_VALUE >> DAC_SCALE) × 3.3V

Note

Settings DAC_EN = 1 feeds the variable output to the DAC registers, but user needs to enable the DAC peripheral in TI Sysconfig for the DAC peripheral to function. Also make sure the DAC output pin is not loaded by any other peripheral.

Table 7-1. Address Table for DAC Monitoring

Variable	Address
A phase current	0x202001B0
B phase current	0x202001B4
C phase current	0x202001B8
A phase current raw ADC value	0x202001BC
B phase current raw ADC value	0x202001C0
C phase current raw ADC value	0x202001C4
A phase voltage	0x20200208
B phase voltage	0x2020020C
C phase voltage	0x20200210
A phase voltage raw ADC value	0x20200214
B phase voltage raw ADC value	0x20200218
C phase voltage raw ADC value	0x2020021C

Table 7-1. Address Table for DAC Monitoring (continued)

Variable	Address
D axis current	0x202002D0
Q axis current	0x202002D4
D axis voltage	0x202002D8
Q axis voltage	0x202002DC
Estimated motor velocity filtered	0x20200700
Estimated rotor angle	0x20200708
SVM ouput duty A phase	0x202002A4
SVM ouput duty B phase	0x202002A8
SVM ouput duty C phase	0x202002AC

8 Revision History

NOTE: Page numbers for previous revisions may differ from page numbers in the current version.

DATE	REVISION	NOTES
March 2024	*	Initial Release

IMPORTANT NOTICE AND DISCLAIMER

TI PROVIDES TECHNICAL AND RELIABILITY DATA (INCLUDING DATA SHEETS), DESIGN RESOURCES (INCLUDING REFERENCE DESIGNS), APPLICATION OR OTHER DESIGN ADVICE, WEB TOOLS, SAFETY INFORMATION, AND OTHER RESOURCES “AS IS” AND WITH ALL FAULTS, AND DISCLAIMS ALL WARRANTIES, EXPRESS AND IMPLIED, INCLUDING WITHOUT LIMITATION ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT OF THIRD PARTY INTELLECTUAL PROPERTY RIGHTS.

These resources are intended for skilled developers designing with TI products. You are solely responsible for (1) selecting the appropriate TI products for your application, (2) designing, validating and testing your application, and (3) ensuring your application meets applicable standards, and any other safety, security, regulatory or other requirements.

These resources are subject to change without notice. TI grants you permission to use these resources only for development of an application that uses the TI products described in the resource. Other reproduction and display of these resources is prohibited. No license is granted to any other TI intellectual property right or to any third party intellectual property right. TI disclaims responsibility for, and you will fully indemnify TI and its representatives against, any claims, damages, costs, losses, and liabilities arising out of your use of these resources.

TI’s products are provided subject to TI’s Terms of Sale or other applicable terms available either on ti.com or provided in conjunction with such TI products. TI’s provision of these resources does not expand or otherwise alter TI’s applicable warranties or warranty disclaimers for TI products.

TI objects to and rejects any additional or different terms you may have proposed.

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265