User's Guide

MCT8316A Tuning Guide

ABSTRACT

This Tuning guide provided step by step guidence to setup MCx8316AEVM, connect MCT8316A GUI to the EVM and tune a 3-phase Brushless DC motor using MCT8316A.

Table of Contents

List of Figures

List of Tables

1 Revision History

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

Changes from Revision * (August 2021) to Revision A (January 2022)	Page
• RTM updates	1

Trademarks

All trademarks are the property of their respective owners.

2 Introduction

The MCT8316A provides three half-H-bridge integrated MOSFET drivers with sensorless trapezoidal control to drive a three-phase brushless DC (BLDC) motor at 12-V/24-V DC rails or battery powered applications. The device integrates three current-sense amplifiers (CSA) with integrated current sense for sensing the three phase currents of BLDC motors to achieve optimum trapezoidal control. Simplified schematic of MCT8316A is shown in Figure 2-1.

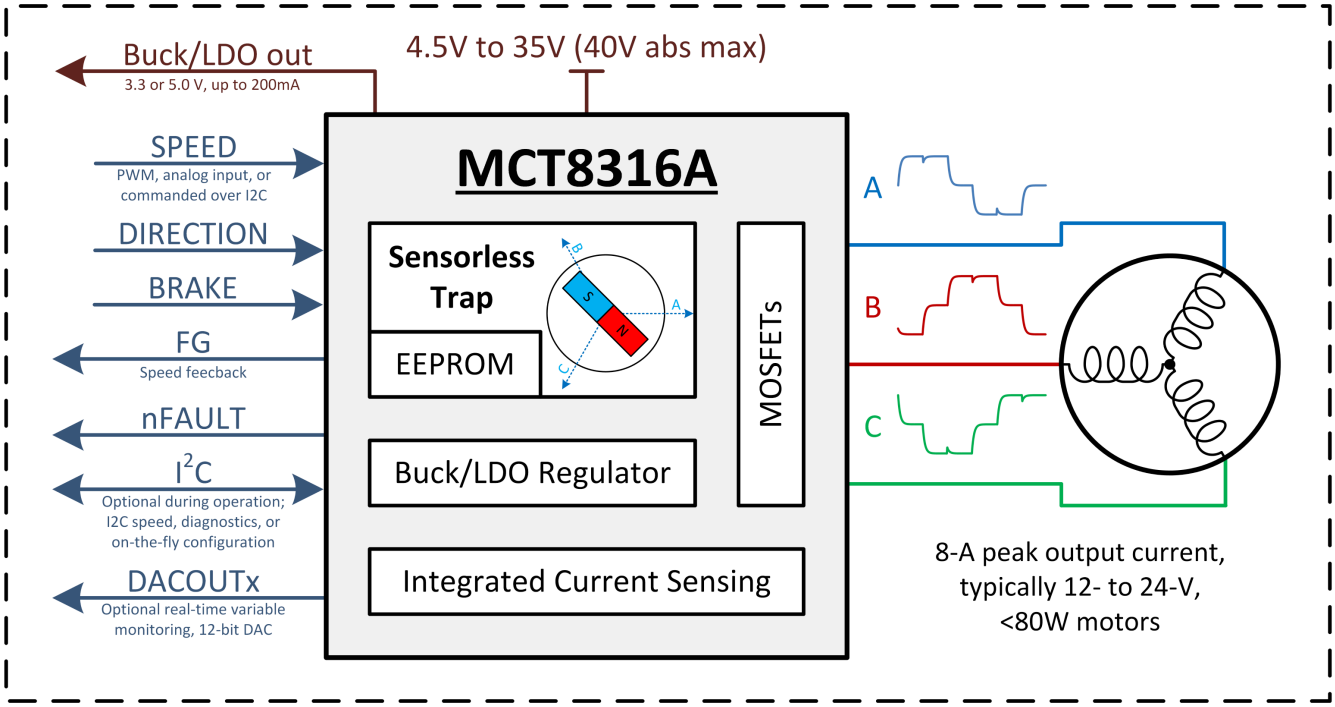

Figure 2-1. Simplified Schematic of MCT8316A

This tuning guide provides the steps to tune a 3-phase BLDC motor using the MCT8316A. The tuning process is classified into two sections: Essential controls and Basic controls. This process is also detailed in the Guided tuning section in the GUI.

- ***Essential controls:*** Tuning steps to successfully spin the motor in closed loop
- ***Basic controls:*** Tuning steps to conform to use-case and explore features in the device

2.1 Hardware and GUI setup

Below are the items that are required to run through this tuning guide.

1. MCx8316AEVM Board.
2. A computer with the MCT8316_GUI installed.
3. A motor to be tuned using this process. It will be good to have motor datasheet but it is not mandatory.
4. A DC Power Supply rated for the motor.
5. Basic lab equipment such as a Digital Multimeter (DMM), Oscilloscope, current probe and voltage probes.

2.1.1 Jumper Configuration

Ensure the hardware is configured according to the jumper configuration as shown in Figure 2-2. Red boxes indicate where jumpers should be installed on the EVM.

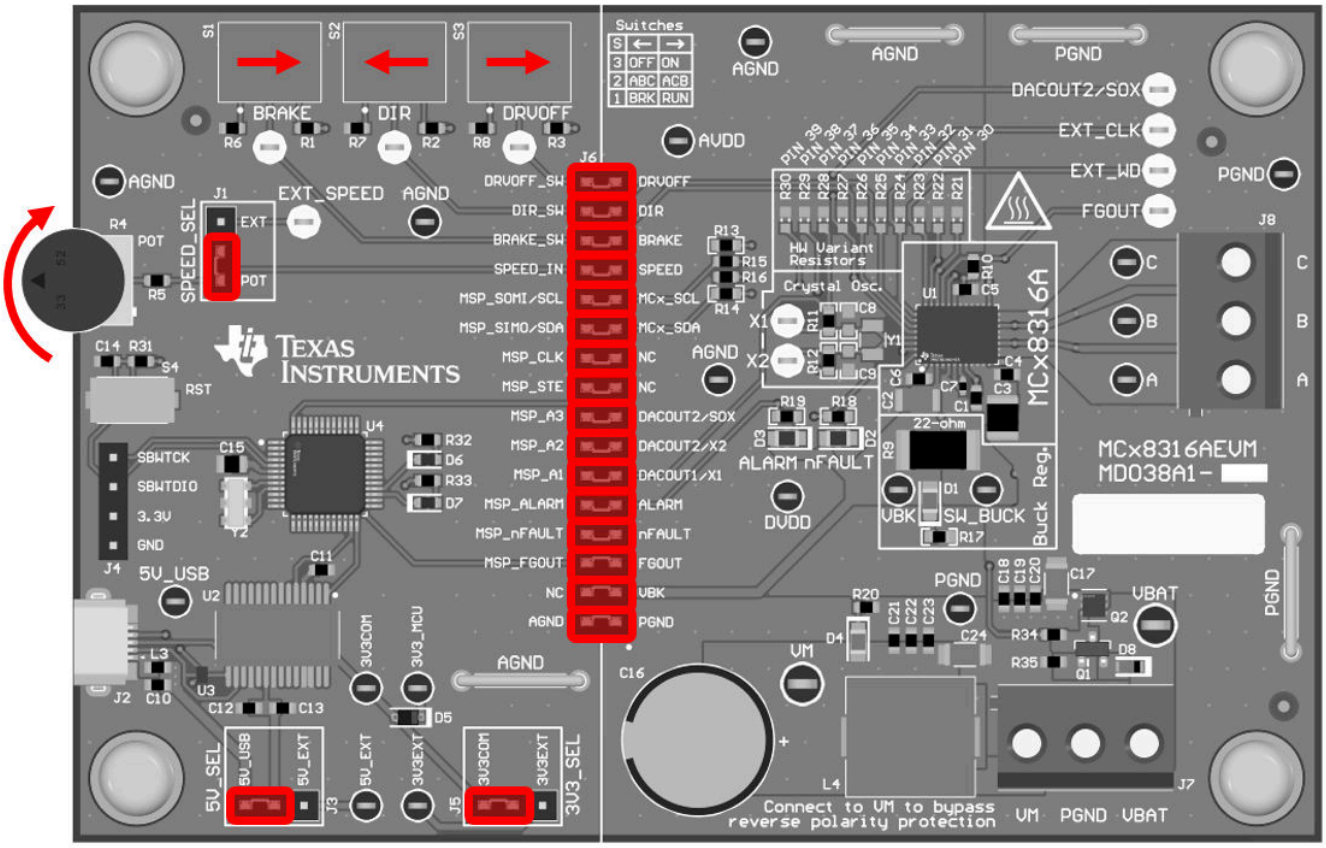

Figure 2-2. MCx8316A EVM Jumper Configuration

2.1.2 External Connections

Connect the motor to J8 (connections A, B, and C). If the motor has a center tap connection or wires for Hall-effect sensors, leave these wires unconnected. Supply a voltage compliant with the Power Supply Voltage (VM) range using J7. Recommended voltage range for the device is 4.5 V – 35 V.

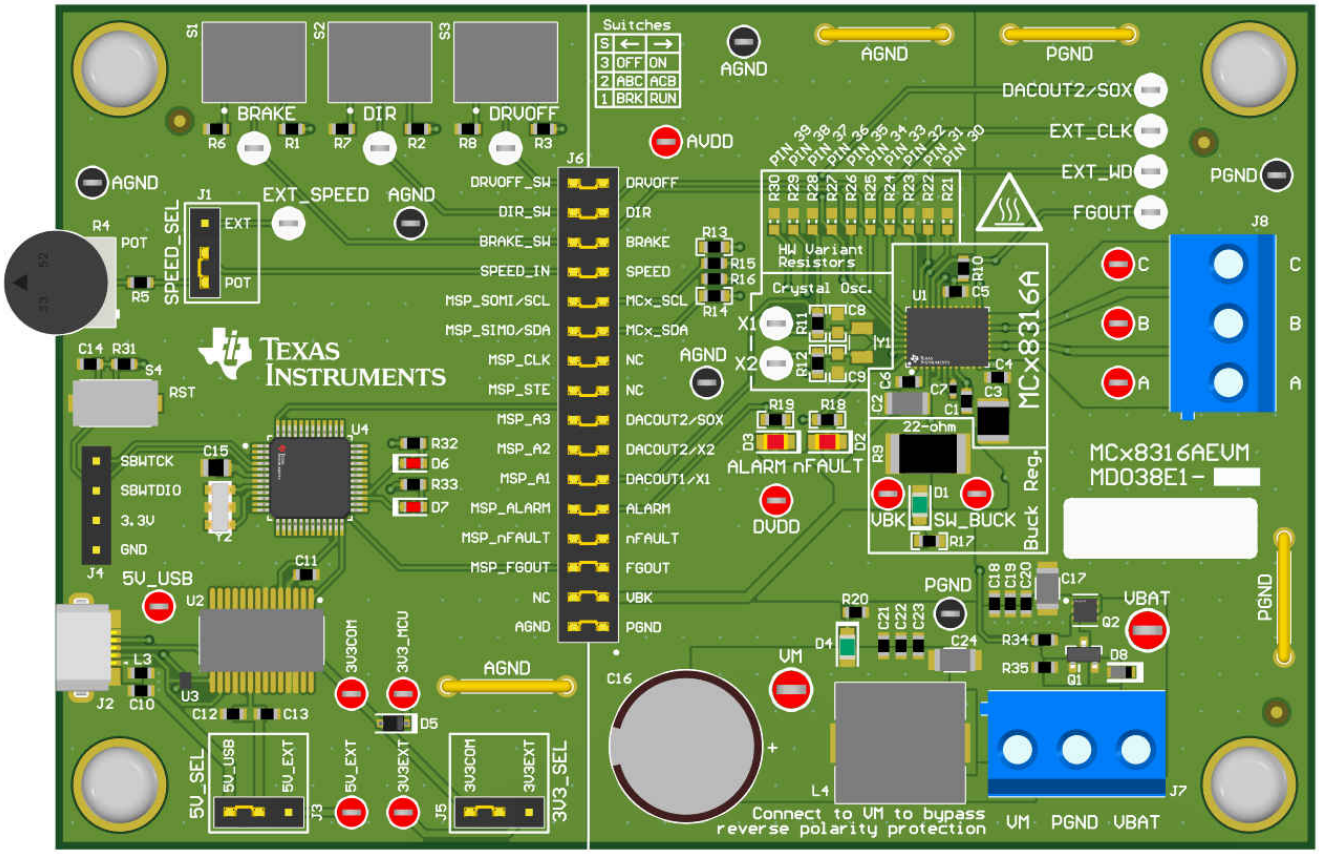

Figure 2-3. MCx8316A EVM External Configuration

2.1.3 Connecting to the GUI

2.1.3.1 Connect to computer

Ensure the EVM is powered on (VM) before connecting USB cable. Plug in the micro-USB, from the computer that holds the GUI, to the USB Connector (J2) on the EVM.

2.1.3.2 Connect to the GUI

The GUI should automatically connect to the EVM, as shown on the status bar located on the bottom of this GUI.

If the GUI doesn't connect automatically, please ensure that the "FTDI" serial port is selected by going to the top menu "Options -> Serial Port ...". Then click on the chain link icon on the left of the status bar to connect to the EVM.

2.1.3.3 Verify Hardware Connection

After the GUI establishes connection with the EVM through USB, you can verify the communication with the MCx8316 device by clicking on the "Read All Registers" button on the top right of this GUI. After a few seconds, the GUI should indicate with a pop-up notification that all registers have been read back successfully.

3 Essential Controls

The goal of this section is to help customers spin their motor successfully in closed loop. This section provides standardized steps to tune parameters for successful Motor spin-up in closed loop.

3.1 Recommended Default Values based on application

Launch the MCx8316A EVM GUI. GUI will load the recommended default values based on the application.

If your application requires ultra-slow acceleration such as pedestal fans, load the default values listed in Table 3-1.

Table 3-1. Ultra-Slow Acceleration Default Values

Address Name	Address	Recommended Value
ISD_CONFIG	0x00000080	0x6F04C1C0
MOTOR_STARTUP1	0x00000082	0x2BA89190
MOTOR_STARTUP2	0x00000084	0x1A10B990
CLOSED_LOOP1	0x00000086	0x08220200
CLOSED_LOOP2	0x00000088	0x02A6E4B0
CLOSED_LOOP3	0x0000008A	0x4CC40100
CLOSED_LOOP4	0x0000008C	0x000CE944
CONST_SPEED	0x0000008E	0x00A00504
CONST_PWR	0x00000090	0x5DC04C84
150_DEG_TWO_PH_PROFILE	0x00000096	0x36DB6DA6
150_DEG_THREE_PH_PROFILE	0x00000098	0x36DB6D80
TRAP_CONFIG1	0x0000009A	0x054BA106
TRAP_CONFIG2	0x0000009C	0x62880000
FAULT_CONFIG1	0x00000092	0x78F43025
FAULT_CONFIG2	0x00000094	0x7447A009
GD_CONFIG1	0x000000AC	0x1C440000
GD_CONFIG2	0x000000AE	0x14000000
PIN_CONFIG1	0x000000A4	0x2D720600
PIN_CONFIG2	0x000000A6	0x080C0000
DEVICE_CONFIG	0x000000A8	0x7FFF0000
PERI_CONFIG	0x000000AA	0x00000000

If your application requires slow acceleration such as blowers, load the default values listed in Table 3-2.

Table 3-2. Slow Acceleration Defalt Values

Address Name	Address	Recommended Value
ISD_CONFIG	0x00000080	0x6F04C180
MOTOR_STARTUP1	0x00000082	0x2F889192
MOTOR_STARTUP2	0x00000084	0x1A1CF990
CLOSED_LOOP1	0x00000086	0x09220200
CLOSED_LOOP2	0x00000088	0x02A6E4B0
CLOSED_LOOP3	0x0000008A	0x4CC40100
CLOSED_LOOP4	0x0000008C	0x000CE944
CONST_SPEED	0x0000008E	0x00A00504
CONST_PWR	0x00000090	0x5DC04C84
150_DEG_TWO_PH_PROFILE	0x00000096	0x36DB6DA6
150_DEG_THREE_PH_PROFILE	0x00000098	0x36DB6D80
TRAP_CONFIG1	0x0000009A	0x054BA106
TRAP_CONFIG2	0x0000009C	0x52880000

Table 3-2. Slow Acceleration Defalt Values (continued)

Address Name	Address	Recommended Value
FAULT_CONFIG1	0x00000092	0x78F43025
FAULT_CONFIG2	0x00000094	0x7447A009
GD_CONFIG1	0x000000AC	0x1C440000
GD_CONFIG2	0x000000AE	0x14000000
PIN_CONFIG1	0x000000A4	0x2D720600
PIN_CONFIG2	0x000000A6	0x080C0000
DEVICE_CONFIG	0x000000A8	0x7FFF0000
PERI_CONFIG	0x000000AA	0x00000000

If your application requires faster acceleration such as robotic vacuums, load the default values listed in Table 3-3.

Table 3-3. Faster Acceleration Default Values

Address Name	Address	Recommended Value
ISD_CONFIG	0x00000080	0x6F04C100
MOTOR_STARTUP1	0x00000082	0x38C8D197
MOTOR_STARTUP2	0x00000084	0x1A399990
CLOSED_LOOP1	0x00000086	0x11466200
CLOSED_LOOP2	0x00000088	0x02A6E4B0
CLOSED_LOOP3	0x0000008A	0x4CC40101
CLOSED_LOOP4	0x0000008C	0x000CE944
CONST_SPEED	0x0000008E	0x00A00504
CONST_PWR	0x00000090	0x5DC04C84
150_DEG_TWO_PH_PROFILE	0x00000096	0x36DB6DA6
150_DEG_THREE_PH_PROFILE	0x00000098	0x36DB6D80
TRAP_CONFIG1	0x0000009A	0x054BA106
TRAP_CONFIG2	0x0000009C	0x2A880000
FAULT_CONFIG1	0x00000092	0x78F43025
FAULT_CONFIG2	0x00000094	0x7947A009
GD_CONFIG1	0x000000AC	0x1C440000
GD_CONFIG2	0x000000AE	0x14000000
PIN_CONFIG1	0x000000A4	0x2D720600
PIN_CONFIG2	0x000000A6	0x080C0000
DEVICE_CONFIG	0x000000A8	0x7FFF0000
PERI_CONFIG	0x000000AA	0x00000000

If your application requires ultra fast acceleration such as fuel pumps, load the default values listed in Table 3-4.

Table 3-4. Ultra Fast Acceleration Default Values

Address Name	Address	Recommended Value
ISD_CONFIG	0x00000080	0x6EC4C000
MOTOR_STARTUP1	0x00000082	0x3E07118B
MOTOR_STARTUP2	0x00000084	0x3B52239C
CLOSED_LOOP1	0x00000086	0x1F326200
CLOSED_LOOP2	0x00000088	0x02A224B0
CLOSED_LOOP3	0x0000008A	0x4D640111
CLOSED_LOOP4	0x0000008C	0x000CE944
CONST_SPEED	0x0000008E	0x00A00504

Table 3-4. Ultra Fast Acceleration Default Values (continued)

Address Name	Address	Recommended Value
CONST_PWR	0x00000090	0x5DC04C84
150_DEG_TWO_PH_PROFILE	0x00000096	0x36DB6DA6
150_DEG_THREE_PH_PROFILE	0x00000098	0x36DB6D80
TRAP_CONFIG1	0x0000009A	0x050BA106
TRAP_CONFIG2	0x0000009C	0x221C0000
FAULT_CONFIG1	0x00000092	0x60F43025
FAULT_CONFIG2	0x00000094	0x7487A009
GD_CONFIG1	0x000000AC	0x1C440000
GD_CONFIG2	0x000000AE	0x14000000
PIN_CONFIG1	0x000000A4	0x2D720600
PIN_CONFIG2	0x000000A6	0x08000000
DEVICE_CONFIG	0x000000A8	0x7FFF0000
PERI_CONFIG	0x000000AA	0x00000000

If you are not sure of your application, load the default values listed in Table 3-5

Table 3-5. Default Values

Address Name	Address	Recommended Value
ISD_CONFIG	0x00000080	0x6F04C140
MOTOR_STARTUP1	0x00000082	0x35489195
MOTOR_STARTUP2	0x00000084	0x1A295990
CLOSED_LOOP1	0x00000086	0x0E3A0200
CLOSED_LOOP2	0x00000088	0x02A6E4B0
CLOSED_LOOP3	0x0000008A	0x4CC40101
CLOSED_LOOP4	0x0000008C	0x000CE944
CONST_SPEED	0x0000008E	0x00A00504
CONST_PWR	0x00000090	0x5DC04C84
150_DEG_TWO_PH_PROFILE	0x00000096	0x36DB6DA6
150_DEG_THREE_PH_PROFILE	0x00000098	0x36DB6D80
TRAP_CONFIG1	0x0000009A	0x054BA106
TRAP_CONFIG2	0x0000009C	0x3A880000
FAULT_CONFIG1	0x00000092	0x78F43025
FAULT_CONFIG2	0x00000094	0x7447A009
GD_CONFIG1	0x000000AC	0x1C440000
GD_CONFIG2	0x000000AE	0x14000000
PIN_CONFIG1	0x000000A4	0x2D720600
PIN_CONFIG2	0x000000A6	0x080C0000
DEVICE_CONFIG	0x000000A8	0x7FFF0000
PERI_CONFIG	0x000000AA	0x00000000

3.2 Device and Pin Configuration

3.2.1 Speed input mode

MCT8316A provides four options to configure the input speed command.

- PWM input on SPEED pin by varying duty cycle of input signal
- Frequency input on SPEED pin by varying frequency of input signal
- Analog input on SPEED pin by varying amplitude of input signal
- Over I^2C by configuring SPEED_CTRL

Configure the speed input [SPD_CTRL_MODE] register to select the appropriate speed input mode.

If PWM input on SPEED pin is chosen as the speed input, configure SPD_PWM_RANGE_SELECT to select the PWM frequency range.

If Frequency input on SPEED pin is chosen as the speed input, configure INPUT_MAX_FREQUENCY to select the maximum PWM frequency. For example, if 10 kHz is chosen as maximum PWM frequency, 10 kHz PWM signal on SPEED pin will correspond to 100% speed command.

To command the speed via I²C, toggle the “Enable Speed Control via I²C” in the I²C Control Section in GUI.

3.3 Algorithm configuration – Motor parameters

3.3.1 Maximum motor electrical speed (Hz)

Using the motor’s datasheet, the user can input the maximum motor electrical speed in Hz. If this data is not available, the user can input the number of pole pairs and motor mechanical speed in RPM. The GUI will convert the motor mechanical speed in RPM to motor electrical speed in Hz using Equation 1.

$$f_{Electrical} = \frac{n_{PolePairs} \cdot \omega_{Mechanical}}{60} \quad (1)$$

Where:

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

Configure the abnormal speed threshold [LOCK_ABN_SPEED] to 150% of maximum speed [MAX_SPEED].

Note

Determining number of motor poles without a motor datasheet:

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 they are not labeled
3. Turn on supply The rotor should have settled at one position with the injecting current.
4. Manually rotate the rotor until rotor snaps to another settle position. It will have several settle-down 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 will determine how many rotor revolutions correlate to the shaft’s mechanical revolution.

3.4 Fault Configuration

3.4.1 Cycle by cycle current limit (ILIMIT)

Cycle by cycle current limit provides a means of controlling the amount of current delivered to the motor. This is useful when the system must limit the amount of current pulled from the power supply during motor start-up. The cycle by cycle current-limit limits the current applied to the motor from exceeding the configured threshold. Configure Cycle by cycle current limit [ILIMIT] to the rated peak phase current of the motor. Check the motor datasheet for the rated peak phase current in Amps. Use Equation 2 to choose the correct ILIMIT in volts.

$$ILIMIT\ (V) = Cycle\ by\ cycle\ current\ limit(A) \times \left[CSA\_GAIN\right] \left(\frac{v}{A}\right) \quad (2)$$

If the motor is unable to reach the maximum speed, increase ILIMIT in steps until the motor reaches maximum speed.

3.5 Testing for successful startup into closed loop

1. *Apply a nonzero speed command*
   a. If the Speed input mode is I2C, change the “I2C Speed Command Percentage” to a non-zero value. Once the speed command is issued, the device will start commutating and the motor should start spinning that is proportional to the I2C Speed Command Percentage.
   b. If the speed input mode is Analog, rotate the Speed Control potentiometer (R4) clockwise to control the motor speed. Once the speed command is issued, the device will start commutating and the motor should start spinning that is proportional to the voltage on the Speed Pin.
   c. If the speed input mode is PWM, apply PWM signal to the speed pin with a given duty cycle with a low of 0 V to a high of 2.2 V (min). Once the speed command is issued, the device will start commutating and the motor should start spinning that is proportional to the duty cycle.
2. *Check if motor spins in closed loop at commanded speed.*

   In closed loop, the motor should spin at the commanded speed and there should not be any faults triggered.

   a. Enable the “Auto read fault status” toggle button towards the top right corner of the GUI. Then monitor the Fault Status in the side panel to the right.

   If no fault gets triggered, skip Section 3.6 and proceed to Section 4.
3. *Prepare device for fault handling if any fault gets triggered.*

   If the motor failed to spin successfully in closed loop, check the fault status.

   a. Set zero speed command by turning off the PWM, turning the potentiometer completely counterclockwise, or setting the I2C command percentage to 0%.
   b. Clear the fault status registers by clicking on the “Clear” button in the Fault Status side panel to the right as shown in Figure 3-1

   Check Section 3.6 for steps to debug faults.

This completes the essential controls section. At the end of this section, user should be able to spin the motor in closed loop. Save the register settings by clicking File->Save Registers in the GUI. If there were no faults, skip to Section 4.

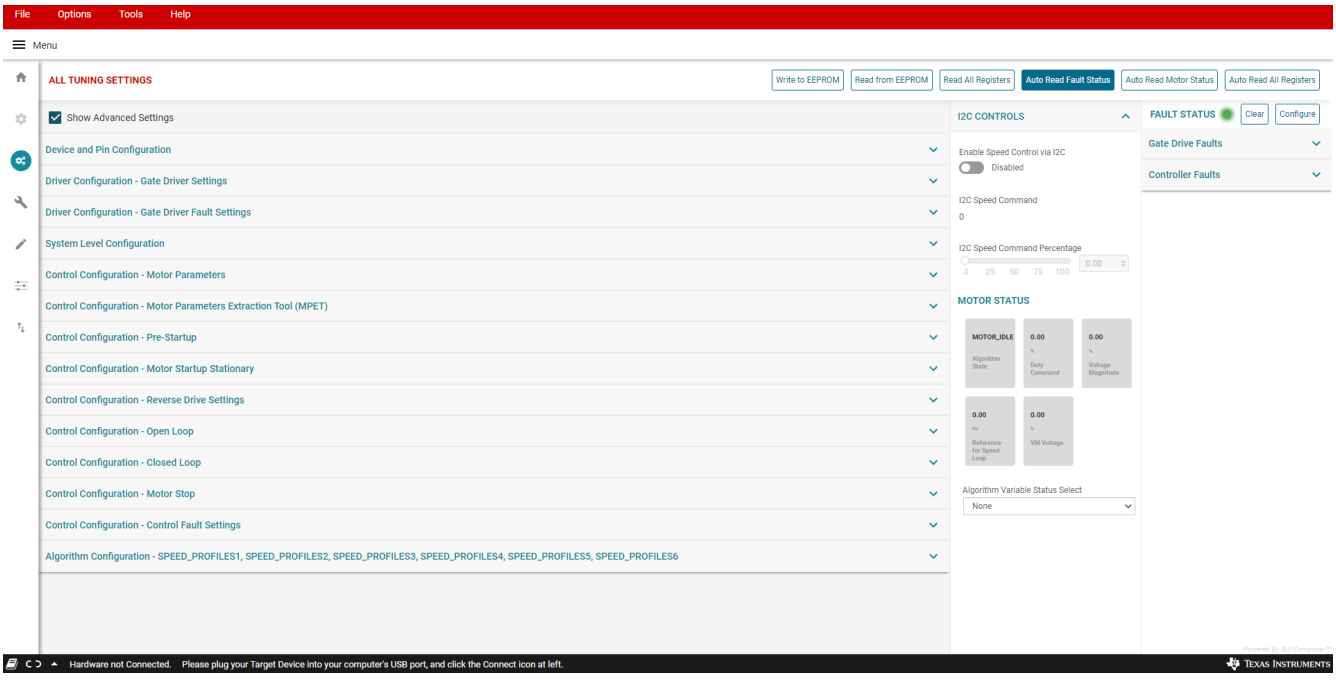

Figure 3-1. Fault Status

3.6 Fault handling

Below are the faults that can possibly get triggered based on the default register configuration specified in Section 3.1.

3.6.1 Abnormal Speed [ABN_SPEED]

MCT8316A monitors the speed continuously by calculating difference in time between subsequent back-EMF zero crossings. This fault gets triggered when motor speed exceeds abnormal speed threshold [LOCK_ABN_SPEED]. When the motor gets stuck in the first electrical cycle, device continues to commutate at frequencies above the configured abnormal speed threshold [LOCK_ABN_SPEED].

When this fault gets triggered, follow below recommendations.

1. Increase align time [ALIGN_TIME].
2. Decrease slow first cycle frequency [SLOW_FIRST_CYC_FREQ]
3. Decrease open loop acceleration A1 [OL_ACC_A1] and open loop acceleration A2 [OL_ACC_A2].
4. Decrease closed loop acceleration [CL_ACC]

3.6.2 Loss of Sync [LOSS_OF_SYNC]

This fault gets triggered when the motor loses synchronization due to improper selection of motor startup and open loop acceleration profiles.

When motor loses synchronization, we might notice the motor either vibrate or stall.

When the motor vibrates, follow the below recommendations.

1. Increase align time [ALIGN_TIME].
2. Increase open loop acceleration A1 [OL_ACC_A1] and open loop acceleration A2 [OL_ACC_A2].

When the motor stalls, follow the below recommendation.

Decrease open loop acceleration A1 [OL_ACC_A1] and open loop acceleration A2 [OL_ACC_A2].

3.6.3 No Motor Fault [NO_MTR]

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

Step 1: Make sure the motor phases are connected to J8.

Step 2: If the fault persists, set the No-Motor lock current threshold [NO_MTR_THR] to 0.005 V.

Step 3: For low inductance motors, increase PWM switching frequency [PWM_FREQ_OUT].

Step 4: Increase Min duty cycle [MIN_DUTY] during closed loop operation.

Please note that device might trigger Loss of sync [LOSS_OF_SYNC] when motor phases are disconnected while the motor is spinning.

3.6.4 Cycle by cycle current limit [CBC_ILIMIT]

Cycle by cycle current limit [CBC_ILIMIT] gets triggered when the motor startup time is too low and the open loop acceleration is too high.

If this fault gets triggered, it is recommended to increase Cycle by cycle current limit (ILIMIT).

4 Basic Controls

The goal of this section is to provide customers tuning guidance to conform to use-case and explore various features in the device.

Note, the user is expected to skip the subsection use-cases and scenarios that do not apply to the system or end equipment.

4.1 Device and pin configuration

4.1.1 Real time variable tracking

MCT8316A has two 12-bit DAC which outputs analog voltage equivalent of digital variables on DACOUT1 and DACOUT2 pins with resolution of 12 bits and max voltage of 3V. Signals available on DACOUT pins can be used for monitoring motor speed, input duty cycle etc.

The address for variables for DACOUT1 and DACOUT2 is configured using register bits DACOUT1_VAR_ADDR and DACOUT2_VAR_ADDR. This is useful in applications which require tracking algorithm variables in real time without having any delay from the communication bus. Pin 37 and 38 should be configured as DACOUT1 and DACOUT2.

For example, if the user wants to read the motor speed from pin 37, configure pin 37 as DACOUT1 and program the motor speed register address (0x5AA) in Hex in [DACOUT1_VAR_ADDR].

Pin 38 [DAC_SOX_CONFIG] can be configured to either output DACOUT2 or CSA outputs SOA, SOB or SOC.

4.1.2 Power saver or sleep mode for battery operated applications

MCT8316A can be configured to be in Sleep mode or Standby mode. For applications that operate using batteries, it is important to save power while the device is not spinning the motor.

MCT8316A enters sleep mode when speed input is held continuously below the speed threshold for SLEEP_TIME.

In sleep mode, all motor outputs are disabled, the charge pump is disabled, the AVDD regulator is disabled, and the serial communication bus is disabled. MCT8316A can be configured to be in sleep mode by configuring register bits DEV_MODE. Please note that in Sleep mode, GUI will lose connectivity as there will be no I2C communication.

4.1.3 Direction and Brake pin override

Direction and Brake pin override will be helpful in applications where it is not intended for the users to affect the direction and brake pins in the device. In such cases, it is recommended to override both pins and configure them via EEPROM.

Direction pin in MCT8316A can be overwritten by configuring DIR_INPUT. Configuring DIR_INPUT to 01b will overwrite Hardware pin to clockwise rotation if motor phases A, B and C are connected to OUTA, OUTB and OUTC respectively. Commutation sequence in clockwise direction will be OUTA-OUTB-OUTC. Configuring DIR_INPUT to 10b will overwrite Hardware pin to anti-clockwise rotation if motor phases A, B and C are connected to OUTA, OUTB and OUTC respectively. Commutation sequence in anti-clockwise direction will be OUTA-OUTC-OUTB.

BRAKE pin in MCT8316A can be overwritten by configuring BRAKE_INPUT. Configuring BRAKE_INPUT to 01b will overwrite Hardware pin to active braking. Configuring BRAKE_INPUT to 10b will overwrite Hardware pin and disable brake functionality.

4.2 System level configuration

4.2.1 Tracking motor speed feedback in real time

MCT8316A device provides information about the motor speed through the frequency generator (FG) pin, which is also known as a TACH (tachometer) out. In MCT8316A, the FG pin behavior is configured through FG_CONFIG. Configure FG_SEL to output FG signal only in closed loop, both open loop and closed loop, or

only in open loop for the first try. Configure FG_DIV to number of motor poles so that the FG output matches the motor mechanical speed in Hz.

When FG_CONFIG is configured to 0 (FG active as long as motor is driven), the FG output is active as long as MCT8316A is driving the motor. FG will not be active during a motor stop and coasting condition. In this mode, FG is released high when MCT8316A enters sleep or standby mode. This mode is useful in applications that require real time motor speed information as long as MCT8316A is driving the motor.

When FG_CONFIG is configured to 1 (FG active till BEMF drops below FG_BEMF_THR), MCT8316A provides FG output until BEMF falls below FG_BEMF_THR. The FG output will continue to indicate motor speed even if the motor is not being actively driven. This mode is useful in applications that require motor speed information above a certain speed, and rotor motion information during the coast and braking conditions.

For example, if the motor Ke is 5 mV/Hz and the application require measuring motor speed above 4 Hz, then the user can configure FG_BEMF_THR to 20 mV. Once when the motor speed reaches 4 Hz, the device will output FG as the BEMF voltage will be 20 mV at 4 Hz.

4.2.2 Monitoring power supply voltage fluctuations for normal motor operation

In applications where the power supply may fluctuate, it is recommended to specify the minimum and maximum power supply voltage range. During an undervoltage condition, the motor might operate at lower speeds. During an overvoltage condition, the MOSFETs, MCT8316A and motor may be stressed with continued operation in high voltage.

Step 1: Keep decreasing the supply voltage until we see a drop in the speed. Measure the bus voltage at which the speed drops and set MIN_VM_MOTOR to that value. The range of minimum bus voltage that can be configured is between 4.5V and 12.5V.

Step 2: Keep increasing the bus voltage to a point where the motor phase voltage reaches the maximum rated voltage of the motor. MAX_VM_MOTOR will be 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 20V and 35V.

Note

MCT8316A provides an undervoltage recovery mode [MIN_VM_MODE] and 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.

4.3 Control configurations

4.3.1 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, or 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 wrong direction command is issued. Motors with higher inertia coast for a longer period of time. It is recommended to have ISD enabled in applications that require frequent motor starts and stops and use higher inertia motors.

Step 1: Enable ISD [ISD_EN]

Step 2: Enable Motor ISD Resynchronize [RESYNC_EN]

Note

If the motor fails to startup, increase the Motor Stationary BEMF Threshold [STAT_DETECT_THR].

4.3.2 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. MCT8316A 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 4-1.

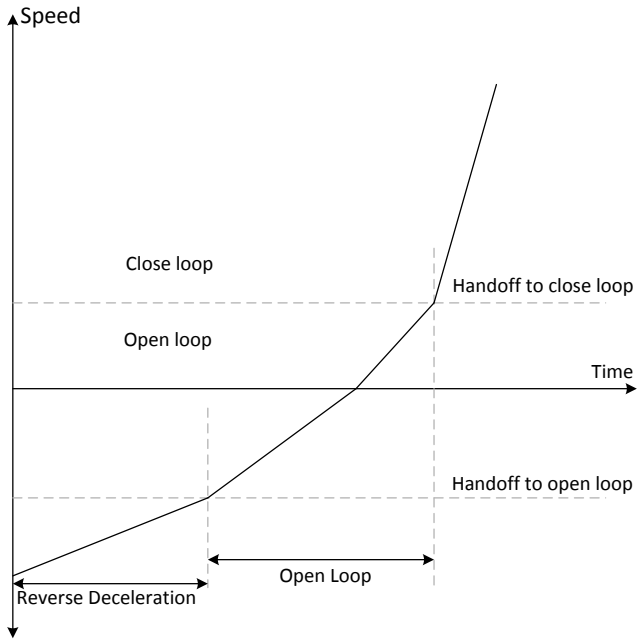

Figure 4-1. Reverse Drive Function

MCT8316A 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 specific direction for desired results. For such applications, it is recommended to follow the below recommendations.

Step 1: Enable ISD [ISD_EN]

Step 2: Enable Motor ISD Reverse drive [RVS_DR_EN]

Step 3: Enable reverse resynchronization [RESYNC_EN]

4.3.3 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 initialize the motor. With the proper IPD setting, the motor startup is also faster. 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 generated by IPD is not acceptable during startup, it is recommended to 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 “Control Configuration – Motor Startup Stationary” 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. It is recommended to start with 50% of the rated current of the motor. If the motor startup is unsuccessful, then we recommend increasing the threshold till the motor starts up successfully. Note that the IPD current threshold should not be higher than the rated current of the motor. Use Equation 3 to choose the correct IPD_CURR_THR.

$$[IPD\_CURR\_THR](V) = IPD\ current\ threshold\ (A) \times \left[CSA\_GAIN\right]\left(\frac{V}{A}\right) \quad (3)$$

Step 3: Select IPD clock value [IPD_CLK_FREQ]. The 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 last longer, so we suggest setting the clock as fast as possible as long as IPD current is able to settle down completely.

Looking at Figure 8, the current does not settle completely, which means the clock is too fast for this motor. This will result in IPD not being able to reliably identify the initial position of the motor.

Step 4: Select IPD Advance angle [IPD_ADV_ANGLE]. This decides how much angle to be added to IPD vector. To start with, choose smaller value to get smoother spin-up. Highest start up torque is achieved with 90-degree advance input.

Note

Device triggers IPD timeout faults [IPD_T1_FAULT] and [IPD_T2_FAULT] for motors with very high inductance, or if the motor is not connected. If this fault gets triggered, it is recommended to check if motor is connected to the device. If the fault still persists, it is recommended to set the IPD release mode [IPD_RLS_MODE] to Tri-state if any overshoot in DC bus voltage is acceptable.

Device triggers IPD Frequency fault [IPD_FREQ_FAULT] if the IPD clock frequency is set too high. If this fault gets triggered, it is recommended to decrease the IPD Clock value [IPD_CLK_FREQ].

Step 5: Select IPD Advance Angle [IPD_ADV_ANGLE]. It is recommended to Start with 90º to get maximum startup torque. If there is sudden jerk observed during startup, then it is recommended to reduce the angle to 60º or 30º for a smoother startup.

Option 2: Slow first cycle

Follow below 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 current threshold [ALIGN_CURR_THR]. Lower current threshold 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, it is recommended to increase the current reference. In applications where, sustained oscillations or sudden jerks are observed, it is recommended to decrease the current threshold.

Step 3: Select Align current ramp rate [ALIGN_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. It is recommended to start with setting up the ramp time to 0.5 sec to ramp to rated current of the motor. In applications where the startup torque is high, the motor might lose synchronization. In such applications, it is recommended to increase the current ramp rate. In applications where, sustained oscillations or sudden jerks are observed, it is recommended to decrease the current ramp rate.

Step 4: Select Frequency of first cycle [SLOW_FIRST_CYC_FREQ]. Lower frequency may give a jerky motion at startup. Higher frequency may not be able to synchronize the motor. It is recommended to 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, it is recommended to decrease the frequency. In applications where jerky motions are observed, it is recommended to increase the frequency.

4.3.4 Faster startup timing

Startup time is the time taken for the motor to reach closed loop from zero speed. For applications that require quick startup time, we recommend choosing either Initial Position Detection (IPD) or Align 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. Use equation 4 to choose the correct IPD_CURR_THR.

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 3.4.4 (Step 3) for more details.

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

Step 5: Configure IPD release mode [IPD_RLS_MODE] to Tri state.

Step 6: Select Open loop current limit [OL_ILIMIT] to be the same as cycle by cycle current limit [ILIMIT].

If the device triggers cycle by cycle current limit [CBC_ILIMIT], it is recommended to increase [ILIMIT] upto the stall current of the motor. Configuring this to a value higher than motor stall current might overheat or damage the motor.

Step 7: Increase Open loop acceleration coefficient A1 [OL_ACC_A1] and Open loop acceleration coefficient A2 [OL_ACC_A2].

Note

A1 and A2 can be increased until open loop current reaches Lock detection current threshold [LOCK_ILIMIT]. Open loop current can be measured using oscilloscope.

Increasing Open loop acceleration coefficient A1 [OL_ACC_A1] and Open loop acceleration coefficient A2 [OL_ACC_A2] might trigger LOCK_LIMIT or CBC_ILIMIT. If this happens, reduce A1 and A2 until LOCK_LIMIT no longer triggers.

Step 8: For ultra-fast startup time (less than 100 ms) it is recommended to follow below steps.

- Disable auto-handoff [AUTO_HANDOFF].
- Configure Open loop handoff cycles [OL_HANDOFF_CYCLES] to 3.
- Configure INTEG_ZC_METHOD to Integration.
- Enable Dynamic degauss [DYN_DEGAUSS_EN]
- Configure open to closed loop handoff threshold [OPN_CL_HANDOFF_THR] to a value lesser than or equal to 20 Hz.

For startup times above 100ms, it is recommended to follow below steps.

- Enable auto-handoff [AUTO_HANDOFF].
- Configure Open loop handoff cycles [OL_HANDOFF_CYCLES] to 6.
- Enable Dynamic degauss [DYN_DEGAUSS_EN]

Note

If Abnormal speed fault [ABN_SPEED] gets triggered, it is recommended to decrease open loop acceleration constants [OL_ACC_A1] and [OL_ACC_A2] and also retune IPD by increasing the IPD current threshold [IPD_CURR_THR] and IPD repeat times [IPD_REPEAT].

Step 9: Increase Closed loop acceleration rate [CL_ACC]

Closed loop acceleration rate [CL_ACC] can be increased until closed loop current reaches Lock detection current threshold [LOCK_ILIMIT]. Closed loop current can be measured using oscilloscope. Increasing closed

loop acceleration rate [CL_ACC] might trigger LOCK_LIMIT. If this happens, reduce closed loop acceleration rate [CL_ACC] until no longer triggers.

Option 2: Align

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

Step 2: Configure align time [ALIGN_TIME] to 5 ms.

Step 3: Follow Step 6 to Step 9 in Option 1.

Figure 4-2 shows FG, phase current and motor electrical speed waveform. Motor takes 50 ms to reach target speed from zero speed.

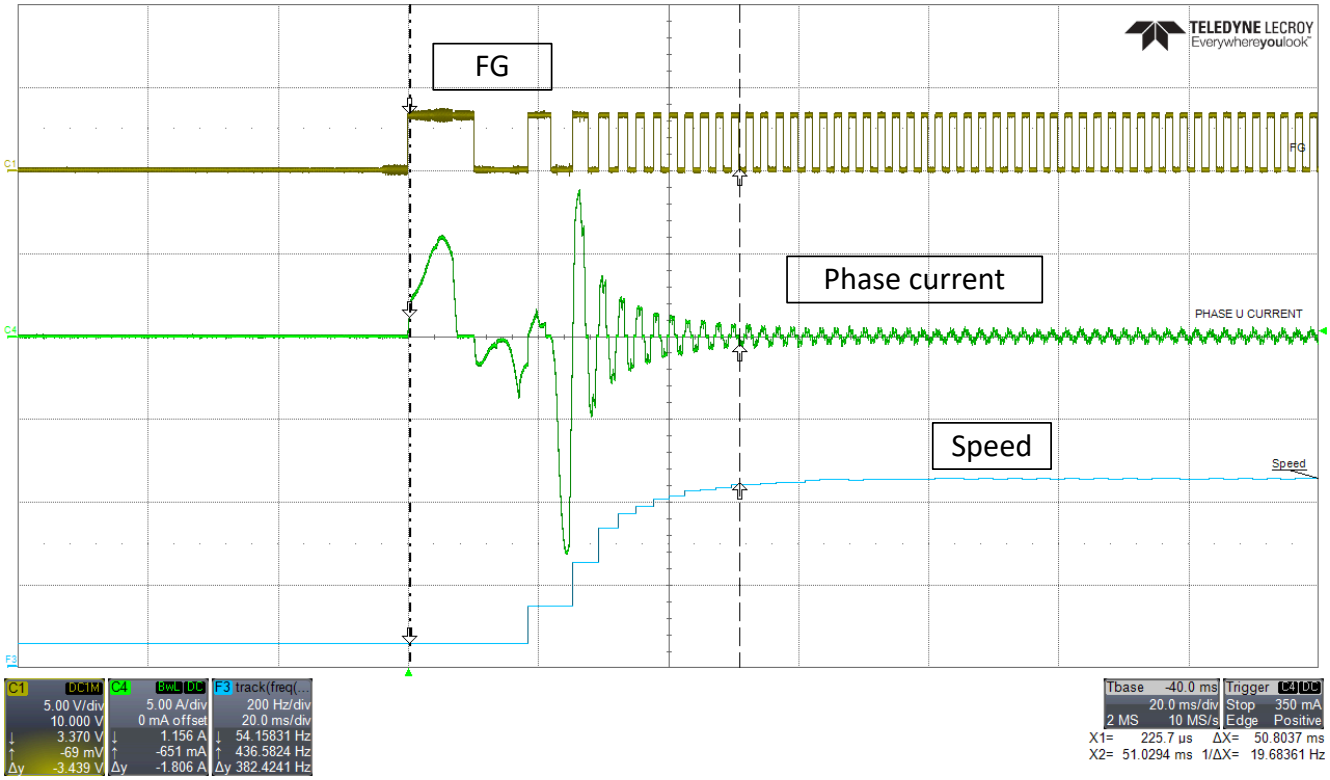

Figure 4-2. Phase current, FG and motor speed - Faster startup time

Note

If Abnormal speed fault [ABN_SPEED] or Loss of sync [LOSS_OF_SYNC] fault gets triggered, it is recommended to follow below debug steps.

1. Select Double align as the motor startup method in [MTR_STARTUP].
2. Increase align time [ALIGN_TIME].
3. Configure align current threshold [ALIGN_CURR_THR] to 50% of cycle by cycle current limit [ILIMIT].
4. Configure First cycle frequency select [FIRST_CYCLE_FREQ_SEL] to 0.

4.3.5 Improving speed regulation

For applications that require better speed regulation, it is recommended to tune Speed loop PI controllers [SPD_LOOP_KP] and [SPD_LOOP_KI]. Kp coefficient of speed loop [SPD_LOOP_KP] controls the settling time and speed overshoots. Ki coefficient of Speed loop [SPD_LOOP_KI] controls speed overshoot and ensures regulation of speed at set value and drives the error to zero.

Step 1: Enable speed loop [SPEED_LOOP_DIS]

Step 2: Tuning speed loop Kp [SPD_LOOP_KP] and Ki [SPD_LOOP_KI] is experimental. It is recommended to manually tune Speed loop Kp and Ki till the desired results are achieved.

Refer to Section 4.1.1 to read back motor speed using DACOUTs.

4.3.6 Stopping motor quickly

MCT8316A provides two options for applications that require stopping the motor quickly.

Option 1: Stopping motor quickly while spinning in closed loop.

Motor can be stopped by issuing 0% input speed duty cycle.

Step 1: Configure closed loop deceleration [CL_DEC] to a value that is same as closed loop acceleration [CL_ACC].

Step 2: Configure Motor stop option [MTR_STOP] to either High side or low side braking.

Step 3: Set the active spin brake threshold [ACT_SPIN_BRK_THR] to a value such that Overcurrent protection fault does not get triggered. Check GATE_DRIVER_FAULT_STATUS (register address: 0x000000E0) bit 28 for Overcurrent protection fault status.

Step 4: Configure Recirculation Brake time [RECIR_BRK_TIME] to a value such that motor is stopped completely. Recirculation brake time might be higher for motors with higher inertia.

Step 5: Enable Fast deceleration [FAST_DECEL_EN].

Note: Fast deceleration will be active till the input speed duty cycle decelerates to active spin brake threshold [ACT_SPIN_BRK_THR]. Device will engage High side or low side braking when the input speed duty cycle is below the active spin brake threshold [ACT_SPIN_BRK_THR].

Option 2: Stopping motor quickly during all stages of motor operation.

Step 1: Configure closed loop deceleration [CL_DEC] to a value that is same as closed loop acceleration [CL_ACC].

Step 2: Configure brake input [BRAKE_INPUT] to “Hardware pin brake” to apply brake using the BRAKE pin. Use Brake switch S1 in MCx8316AEVM to pull the BRAKE pin HIGH or LOW. MCT8316A applies brake when BRAKE pin is pulled HIGH. Configure brake input [BRAKE_INPUT] to “Overwrite Hardware pin with Active Brake” to apply brake using I**2**C.

Step 3: Set the brake duty threshold [BRAKE_DUTY_THRESHOLD] to a value such that Overcurrent protection fault does not get triggered. Check GATE_DRIVER_FAULT_STATUS (register address: 0x000000E0) bit 28 for Overcurrent protection fault status.

Step 4: Enable Fast deceleration [FAST_DECEL_EN].

Note: Fast deceleration will be active till the input speed duty cycle decelerates to brake duty threshold [BRAKE_DUTY_THRESHOLD]. Device will engage High side or low side braking when the input speed duty cycle is below the brake duty threshold [BRAKE_DUTY_THRESHOLD].

4.3.7 Faster deceleration

Follow below steps to decelerate the motor quickly.

Step 1: Configure closed loop deceleration [CL_DEC] to a value that is same as closed loop acceleration [CL_ACC].

Step 2: Enable Fast deceleration [FAST_DECEL_EN].

Step 3: Configure Fast deceleration current limit [FAST_DECEL_CURR_LIM]. Fast deceleration current limit will be higher for motors with higher inertia.

Step 4: Enable AVS [AVS_EN] to protect the power supply from voltage overshoots during motor deceleration. If AVS is enabled, the time taken to decelerate the motor will increase.

Disable AVS, if the power supply can withstand voltage overshoots.

Step 5: Configure Fast Decel Brake Threshold [FAST_BRK_DELTA] if the fast deceleration should be disabled before the actual motor speed reaches the target speed. For example, if the Fast Decel Brake Threshold [FAST_BRK_DELTA] is configured to 0.5% and target duty cycle is configured to 5%, device will stop decelerating the motor at 5.5%. This can be helpful in applications where the motor is decelerated to critically low speeds. While the motor decelerates quickly to critically low speeds, there are possibilities that the motor might stop. In such applications, it is required to stop decelerating the motor at slightly higher speeds to avoid the device from completely stopping the motor.

Figure 4-3 shows phase current and motor electrical speed waveform when the motor decelerates from 100% duty cycle to 10% duty cycle. Time taken for the motor to decelerate from 100% duty cycle to 10% duty cycle when fast deleration is disabled is around 10 seconds. Figure 4-4 shows phase current and motor electrical speed waveform when the motor decelerates from 100% duty cycle to 10% duty cycle. Time taken for the motor to decelerate from 100% duty cycle to 10% duty cycle when fast deleration is enabled is around 1.5 seconds. Please note that when fast deceleration is enabled and anti-voltage surge (AVS) is disabled, there might be voltage spikes seen in supply voltage. Enable AVS to protect the power supply from voltage overshoots during motor deceleration.

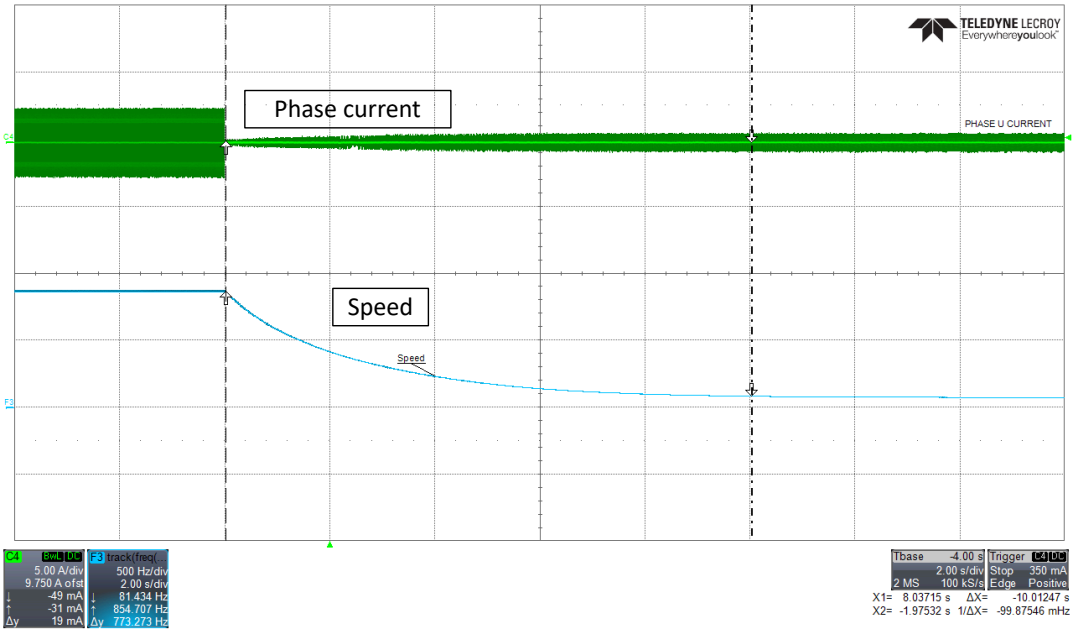

Figure 4-3. Phase current and motor speed - Faster deceleration disabled

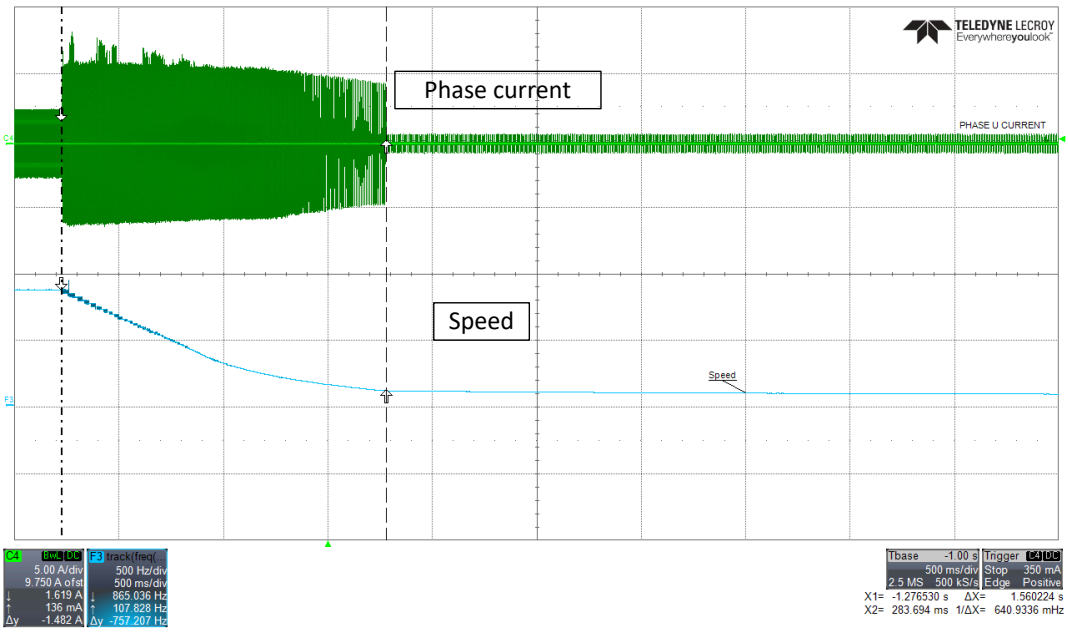

Figure 4-4. Phase current and motor speed -Faster deceleration enabled

MCT8316A provides a dynamic current limit option during fast deceleration to improve the stability of fast deceleration when braking to very low speeds. Using this feature, the current limit during fast deceleration can be reduced as the motor speed decreases.

If motor stalls at lower speeds, it is recommended to follow below steps.

Step 1: Enable dynamic decrease in current limit [DYNAMIC_BRK_CURR].

Step 2: Configure FAST_DEC_DUTY_THR. This sets the speed below which fast deceleration will be implemented. For example, if FAST_DEC_DUTY_THR is set to 70%, any deceleration from speeds above 70% will not use fast deceleration until the speed goes below 70%. If AVS is enabled, AVS will be active when motor decelerates from 100% to 70% duty cycle.

Step 3: Configure DYN_BRK_CURR_LOW_LIM. This sets the current limit at zero speed.

Step 4: Configure FAST_DEC_DUTY_WIN. This is used to set the minimum deceleration window (initial speed - target speed) below which fast deceleration will not be implemented. For example, if FAST_DEC_DUTY_WIN is set to 7.5%, FAST_DEC_DUTY_THR is set to 75% and 75% to 50% deceleration command is received, fast deceleration will be active below 67.5%. If FAST_DEC_DUTY_WIN is set to 15% and 50%->40% deceleration command is received, fast deceleration is not used to reduce the speed from 50% to 40% since the deceleration window (10%) is smaller than FAST_DEC_DUTY_WIN.

Note

Disable WCOMP_BLANK_EN if voltage spikes are seen on supply voltage.

4.3.8 Preventing supply voltage overshoot during motor stop and deceleration

For applications that require preventing supply voltage overshoots during motor stop or deceleration, it is recommended to follow below steps.

Preventing supply voltage overshoot during motor stop:

Step 1: Configure Motor stop options [MTR_STOP] to Recirculation Mode.

Step 2: Configure Recirculation Brake Time [RECIR_BRK_TIME] to “Auto configured”.

Preventing supply voltage overshoot during motor deceleration:

Enable Anti-voltage surge (AVS) [AVS_EN]

4.3.9 Protecting against rotor lock or stall condition

Follow below recommendations based on the type of fault triggered by MCT8316A.

Case 1: Follow below recommendations ff motor gets locked due to LOCK_LIMIT fault. Device triggers LOCK_LIMIT fault when the motor accelerates in closed loop or when the motor is overloaded.

Step 1: Increase lock detection current threshold [LOCK_ILIMIT]

Step 2: Increase the Lock detection current limit deglitch time [LOCK_ILIMIT_DEG] if the motor can withstand [LOCK_ILIMIT] for [LOCK_ILIMIT_DEG] time.

Step 3: Decrease CL_ACC.

Step 4: Enable dynamic degauss.

Case 2: Follow below recommendations if the device triggers CBC_ILIMIT, MTR_LCK, LOSS_OF_SYNC or ABN_SPEED.

Step 1: Increase CBC current limit

Step 2: Decrease CL_ACC.

Step 3: Enable dynamic degauss.

Case 3: If the motor vibrates and device fails to detect any fault, set the abnormal speed threshold to a value slightly above maximum speed. For example, if the maximum speed is 1000 Hz, set the abnormal speed threshold to 1100 Hz.

Case 4: If the motor generates high frequency noise and device fails to detect any fault, set the maximum degauss window [DEGAUSS_MAX_WIN] to either 15°or 18°. Default value is 22.5°.

Case 5: If the motor is loaded at lower speeds, MTR_LCK fault might get triggered when Min duty cycle is set too low. We recommend to se the Min duty cycle after loading the motor at lower speeds.

Note

MCT8316A provides options to either latch LOCK_LIMIT fault or auto retry for [AUTO_RETRY_TIMES] after every [LCK_RETRY] seconds. This can be configured in LOCK_ILIMIT_MODE. We have defaulted the auto retry time and lock retry.

4.3.10 Maximizing thermal efficiency and increasing thermal performance

Thermal performance can be improved by minimizing power dissipation. Power dissipation across $R_{DS(on)}$, Switching losses and operating supply current dissipation are the major sources of Power dissipation in MCT8316A. Power dissipation across $R_{DS(on)}$ is fixed as MOSFETs are integrated in MCT8316A. Please follow below recommendations to minimize switching losses and operating supply current dissipation.

Step 1: Increase Gate driver Slew rate [SLEW_RATE]

Please note that increasing Slew rate will increase EMI noise.

Step 2: Decrease PWM output frequency [PWM_FREQ_OUT]

Please note that decreasing PWM output frequency may lead to discontinuous phase current for very low inductance motors.

Step 3: Configure PWM modulation [PWM_MODUL] to Mixed modulation. In this modulation scheme, the switching losses among high and low side MOSFETs are evenly distributed.

Step 4: Enable ASR [EN_ASR] and AAR [EN_AAR] and configure PWM Mode [PWM_MODE] to Single ended mode.

Please note that enabling ASR when AAR is disabled might results in loss of Sync.

4.3.11 Mitigating Electromagnetic Interference (EMI)

Please follow the below recommendations to mitigate EMI in your system.

Step 1: Decrease gate driver Slew rate [SLEW_RATE]

Please note that decreasing Slew rate will increase power dissipation due to switching losses.

Step 2: Enable Spread spectrum Modulation [SSM_CONFIG]

4.3.12 Improving Motor efficiency

Please follow below recommendations to improve the motor efficiency.

Step 1: Configure PWM Mode [PWM_MODE] to Single ended mode. In this mode, switching losses are lower compared to Complementary mode.

Step 2: In closed loop, set the motor speed to the desired operating load-speed region. Set the lead angle polarity [LD_ANGLE_POLARITY] and tune the lead Angle [LD_ANGLE] in both directions till minimum motor phase current is achieved.

Note: The optimal lead angle value changes with speed and torque. The same lead angle need not be applicable for best efficiency in the entire range of speed-load curve operation.

4.3.13 Limiting and regulating supply power

MCT8316A provides options to limit and regulate supply power. This feature can be utilized in battery powered motor driver applications such as cordless vacuum cleaners, power tools etc.

Follow below steps to limit supply power. In this mode, supply power is only limited to reference power and not actively regulated.

Step 1: Configure CONST_POWER_MODE to "Power limit control".

Step 2: Configure MAX_POWER. This sets the maximum power that MCT8316A can draw from the DC input supply at 100% duty command.

Step 3: Configure CONST_POWER_LIMIT_HYST. This sets the hysteresis band for the power reference. For example, if MAX_POWER is set to 25W and device draws 25W at 100% duty command and CONST_POWER_LIMIT_HYST is set to 5%, input power will still be limited to 25W at 95% speed command.

CONST_POWER_LIMIT_HYST can be configured in applications where there are oscillations in speed around MAX_POWER. If speed oscillation is observed, it is recommended to increase the CONST_POWER_LIMIT_HYST.

Step 4: MCT8316A uses the same PI controller parameters as in the speed loop mode. Kp and Ki coefficients are configured through SPD_POWER_KP and SPD_POWER_KI. Tuning SPD_POWER_KP and SPD_POWER_KI is experimental. It is recommended to manually tune both parameters till the desired results are achieved.

Follow below steps to regulate supply power. In this mode, supply power is always actively regulated.

Step 1: Configure CLOSED_LOOP_MODE to "Power loop".

Step 2: Configure CONST_POWER_MODE to "Closed loop power control".

Step 3: Configure MAX_POWER. This sets the maximum power that MCT8316A can draw from the DC input supply at 100% duty command.

For example, if MAX_POWER is configured to 25W, MCT8316A will draw 12.5W from power supply at 50% duty command.

Step 4: MCT8316A uses the same PI controller parameters as in the speed loop mode. Kp and Ki coefficients are configured through SPD_POWER_KP and SPD_POWER_KI. Tuning SPD_POWER_KP and SPD_POWER_KI is experimental. It is recommended to manually tune both parameters till the desired results are achieved.

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