Advance Information

User's Guide

SLVUBA9–October 2017

BOOSTXL-DRV8304x EVM Sensorless Software User's Guide

This document is intended as a supplement to the *BOOSTXL-DRV8304H EVM User's Guide* to describe the functionality of the sensorless BLDC motor commutation firmware used to on the BOOSTXL-DRV8304x EVM. This user's guide outlines the different considerations for motor commutation as well as how to adjust the different code parameters provided in the sensorless firmware.

Contents

List of Figures

Trademarks

MSP430, Code Composer Studio are trademarks of Texas Instruments.
All other trademarks are the property of their respective owners.

1 Overview

Driving a BLDC motor involves electronically commutating the phases. The windings must be energized in a sequence which makes knowing the rotor position important. In a sensorless control solution, the rotor position is initially detected using the properties of the motor. Then, after start-up, the rotor position is detected using the back electromotive force (BEMF) generated by the running motor. The BOOSTXL-DRV8304x EVM sensorless software uses a BEMF integration technique for sensorless control.

This user's guide is designed to show how sensorless control works and to enable users to modify the application software for a specific system. This document has two major sections. The first section is an introduction to sensorless control. The second section is an explanation of how to customize the reference code.

2 Sensorless Control Background and Implementation

2.1 Initial Speed Control

One of the Important features of sensorless trapezoidal control is to capture the speed if the motor is already spinning at the startup. This method is referred as initial speed control (ISC) in the reference project. The ISC routine checks the available BEMF on the phases at the starting of motor. If the motor is spinning at speeds greater than the threshold speed where significant BEMF is available, the control calculates the duty cycle to be applied for the speed at which motor is spinning and switches directly to closed loop. If the motor speed is not sufficient or the direction of spinning is not correct control waits until the motor is stopped and then starts the motor again in required direction.

Figure 1. ISC Starting when Motor is Spinning in Correct Direction

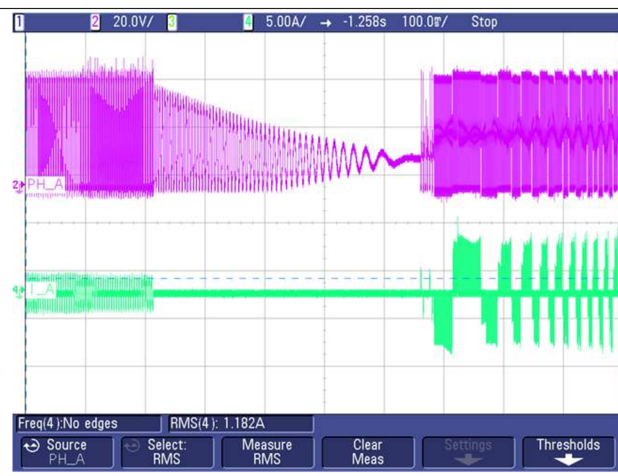

Figure 2. ISC Braking and Restarting Motor when Spinning in Reverse Direction

2.2 Initial Position Detection

To start up a motor, the rotor position must be known to enable the control to drive the correct phases without unwanted backspin. Traditionally using Hall Effect sensors or encoder, the rotor position can simply be read, setting the control sequence and spin up the motor. Removing the sensors from the control leads to the issue of not knowing where the rotor is. The remedy for this is initial position detection (IPD). IPD uses characteristics of the motor to detect where the rotor is before starting to spin.

The magnetic field created by the magnets on the rotor interacts with the electrical field generated by any current in the phases on the stator. Using this feature, measuring the time it takes for a current pulse to reach a specified level in all six drive states will indicate which phase has the lowest inductance, which is the position of the rotor. Figure 3 shows the six drive states. This method uses six pulses of current, one for each sequence, and measures the rise time of that current pulse.

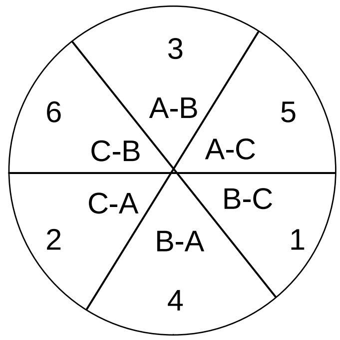

Figure 3. Six Pulses of Initial Position Detection

When the motor is energized with the current pulse, careful tuning is required to reduce any rotation of the motor. This tuning sets the current level to a value large enough to detect the change in rising times yet low enough to not begin to turn the rotor. Another step taken to reduce the movement of the rotor during IPD is to vary the pulse sequence to oppose any rotation. The numbers in Figure 3 specify the order of the IPD routine. Some audible noise may occur in the motor because of vibrations from the sudden current pulses.

As shown in Figure 3, each electrical rotation of the motor is split into six 60-degree drive states. The maximum torque is produced during drive if the excited phase is 90-degrees ahead of the rotor. Based on the minimum rise time, the rotor position is known to be in a 60-degree window. To reduce this window to 30 degrees, the two adjacent phases can be compared. For example, if sequence A-C was found to have the minimum rise time, comparing A-B and B-C rise times shows in which 30 degrees of A-C the rotor resides. From this, the drive state can be easily selected 90 degrees ahead of the rotor position.

To implement the IPD feature, phase-current sensing of the DRV8304 device and the ADC unit of the MSP430F5529 device is used. A current impulse is given to a drive state for a predefined amount of time. After the pulse is withdrawn, the current in the phase is measured by using ADC channels. Timer A1 is used to count for the specified amount of time and sets an interrupt to start the ADC conversion. This time also turns off the high-side FETs and leaves the low-side FET on.

Leaving the low side on recirculates the current on the low side through the FET and one body diode, which is the motor braking. After braking for an established amount of time, the low-side FET is turned off and causes the rest of the energy to return to the supply through the high-side body diode, which is coasting. When the coasting period is complete and the energy is expelled, the next pulse begins repeating this sequence. Finally, the rise times are compared, and the initial drive state is handed off to open-loop control.

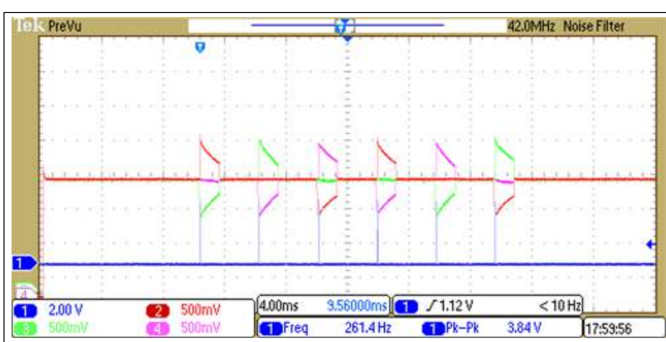

Figure 4. IPD 6 Current Pulses Across Phases

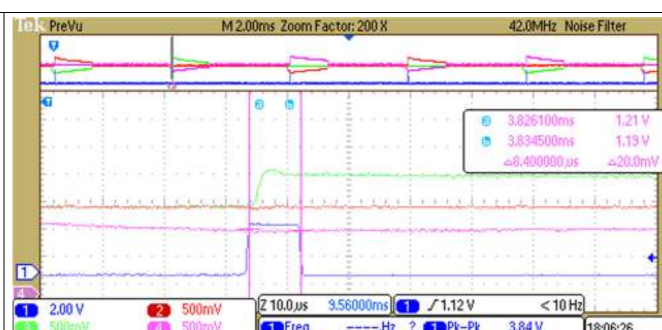

Figure 5. ADC Sampling Delay in Reading Phase Current

A slight delay occurs between turning off the phases and measuring the current values because of a delay in ADC sampling. This delay is negligible, and the amount of current drop can be ignored as shown in Figure 5. The phase current in the low-side switch represented by channel 4 (pink); the drop in the current value during sampling is observed to be negligible.

For some motors and some hardware setups, IPD is not a feasible start-up technique. In this case, a start-up method known as *align* is used. This start-up routine energizes two phases, one high and one low. This routine aligns the rotor to those phases. The phases are turned off after an established amount of time, therefore allowing any oscillations to settle. The rotor position is known at this time, and the drive state can move to a start-up.

2.3 Open-Loop Control

With the rotor position known, the next step is to begin driving the motor. However, the same problem persists; without sensors, the true position of the rotor is not known when it starts spinning. The IPD or align method handed off the initial drive state, and continuing to drive from there without feedback from the rotor position is open-loop control. When the motor reaches a certain speed, enough BEMF is available to use that measurement as feedback for the rotor position and move to closed-loop control.

Different methods of open-loop control are available, such as commutating a set number of times before handing off to a closed loop or increasing the established duty cycle to a fixed percentage before handing off. To make this system more robust, the method of open-loop control used in the reference code estimates the speed of the motor and from that, the distance traveled. Assuming the rotor follows the drive state, increasing the velocity at a fixed rate enables the control to estimate how far the rotor has traveled since the last commutation. When the distance calculation shows the rotor travels 60 degrees, the drive state is changed. When the speed reaches an established threshold, the control is switched to closed loop. To keep an accurate estimation of the distance and the speed, a timer is used to increase the velocity and calculate the distance. The same timer generates the PWM signal for the FETs.

In the open loop control , as the exact rotor position is not available , motor draws huge currents because of improper Voltage and speed combination. To have a better current profile , the refernce project takes the motor Rated voltage , Rated Speed , Motor under rating from the user to calculate the duty cycle to be applied during open loop operation. The control calculates the duty cycle based on the applied DC link voltage, speed at which motor is commutated and motor under rating value and is applied to the motor. By which duty cycle is increased linearly as the speed builds up and thus the current profile during open loop acceleration is improved. It is important to apply sufficient duty cycle at start to spin the motor from stall, for which we need to apply appropriate duty cycle based on the load connected. Motor under rating value is used to apply the additional duty cycle to be applied during open loop acceleration to compensate the load. By default this value is typically around 80% which implicates that amount of rated voltage to be applied to spin the motor at rated speeds at no load operation. This value can be increased if the applied load is higher. The motor under rating value, motor-rated speed, and motor-rated voltage can be updated using the GUI parameters in the Motor Parameter panel.

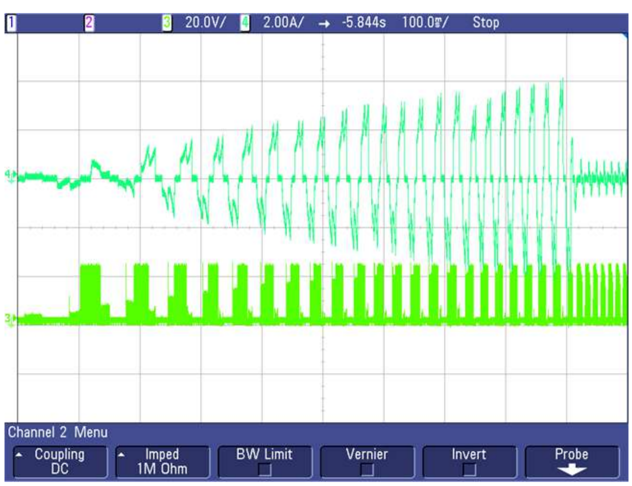

Figure 6. Open-Loop to Closed-Loop Threshold

2.4 Closed-Loop Control

When the motor spins fast enough, the BEMF can be read and used as position information. In this reference software, the BEMF integration method is used to control the motor. When the motor is driven, one high-side FET, and one low-side FET is on. One phase is floating with the high-side and low-side FET off. This phase is monitored with an ADC.

At the floating phases, the BEMF is either increasing or decreasing and eventually crosses the center tap value. To simplify the motor connections, the center tap is approximated as VCC / 2. BEMF integration begins to sample the BEMF after this crossing and integrates it. As the BEMF continues to increase or decrease, the integrated value continues to increase. When the integrated value reaches an established threshold, the drive state is switched.

The integration threshold is established based on the velocity constant of the motor in units of volts per hertz (V/Hz). As the motor is spinning faster, the BEMF is larger, and when the motor is slow, the BEMF is smaller. Because the speed of the motor, or the speed at which the drive state changes, relates directly to the BEMF level, the established integration threshold is constant across motor speed. See Section 3.7.1 to calculate the BEMF threshold for a specific motor.

To vary the speed of the motor, the duty cycle is adjusted based on the user input. As the duty cycle changes, the time of the high-side FET being turned on changes. The ADC reading the BEMF must be sampled on the high-side PWM pulse because, at this point, the center tap is closest to VCC / 2. The exact sample point of the ADC is set before one quarter of the PWM so that the BEMF oscillations on the floating phase is settled. As when a motor is switched, depending on the motor, oscillation of the floating phase voltage occurs due to inductance. To avoid sampling any of the oscillations, a settling time must occur before sampling the ADC. This window, after the oscillations but before the falling PWM edge, results in a minimum duty cycle allowed by this drive method. However, if the motor has very small oscillations in the floating phase, the settling time can be short, and the minimum duty cycle can be decreased. See Equation 3 to calculate the time before which the BEMF is sampled before one fourth of PWM.

A feature of a switching motor is that only a short time passes after the drive state changes until the current in the floating phase must be depleted. This feature results in conduction through the body diode of one of the FETs. To avoiding sampling the ADC during this time, a blanking time must occur after commutation. The reference code takes advantage of this blanking time by reading VCC and the duty cycle input. See Section 3.7.4 for setting the number of PWM's to blank after every Commutation

The external DC supply of 0 to 3.3 V to ADC channel 6 is used as a duty cycle input. As previously mentioned, the duty cycle input is sampled during the commutation blanking time along with the Voltage supply being sampled for determining the center tap value. Because the ADC is on the MSP430™ microcontroller and settings must be changed before each sample, it takes advantage of the blanking time by using this time as a period to change the sampled ADC channel to the voltage supply and the duty input.

2.5 Auto BEMF Threshold Calculation

One of the Important features of sensorless trapezoidal control is to calculate the BEMF threshold automatically for a given motor. When the motor is properly tuned to spin in open-loop control, the motor can be calibrated to compute the BEMF threshold value automatically. This feature can be used by using the GUI widget to calculate the BEMF threshold value in the Motor Parameter panel. When the user tunes the motor to spin in open-loop control, the motor generates sufficient BEMF to switch to closed-loop control which is used to calculate the preliminary BEMF threshold. Using this preliminary BEMF threshold, the motor switches to the closed-loop control and spins the motor until 50% duty cycle is generated, which generates sufficient BEMF, and the motor is allowed to freewheel. Now the BEMF threshold is calculated again for forward direction and the motor is started in reverse direction and the previously described procedure is repeated. When the average BEMF threshold is obtained, it is updated in the GUI BEMF threshold parameter. This entire procedure typically happens for 5 s to 20 s based on the type of motor.

Figure 7. Auto BEMF Threshold Calculation

3 Customizing the Reference Code

The reference code is provided as a Code Composer Studio™ (CCS) project and an Evaluation GUI.

The BOOSTXL-DRV8304x GUI is a user interface (UI) to run and tune the motor on the BOOSTXL-DRV8304x EVM with the DRV8304_EVM_BLDC_FW software.

The user must install the BOOSTXL-DRV8304x GUI to run and modify the run time values of the parameters for the sensorless algorithm.

The user must download the Code Composer Studio™ (CCS) software 6.1.0 or above and install the DRV8304_EVM_BLDC_FW firmware.

The user must import DRV8304_MSP430F5529_Trapezoidal_Sensorless_BLDC project into CCS workspace. Active Build option from the *Build Configuration* should be set as *Debug*.

The following steps go through the process of modifying some parameters for sensorless control are as follows:

1. Open the CCS software.
2. Import the project, *DRV8304_MSP430F5529_Trapezoidal_Sensorless_BLDC*, from the folder where the demo software is located.
3. Select the file *TrapSensored_Parameters_Setup.h*. This folder contains most of the parameters used to run this application code. Some parameters require modifications to properly tune for different operating conditions. The sections that follow describe the parameters and the detail in which they can be modified. All of these parameters, except for the ISC parameters, can be varied using the GUI during run time for accurate tuning.

3.1 Customizing Idrive & Tdrive Parameters

The code for the Idrive & Tdrive Parameters is as follows:

```
//ISC User Parameters

/* Idrive & Tdrive Configuration Settings */
#define GATE_TO_DRAIN_CHARGE        8.7      // Set the gate to drain charge of the FET used on
the EVM in uC
#define RISE_TIME                   100     // Select the Maximum Rise time desired for the FET
in nano seconds
#define FALL_TIME                   50     // Select the Maximum Fall time desired for the FET in
nano Seconds
```

3.1.1 GATE_TO_DRAIN_CHARGE

The GATE_TO_DRAIN_CHARGE parameter defines the gate-to-drain charge or Q_{GD} in microcoulombs of the MOSFET used in the inverter. This value is used to compute the default IDRIVE and TDRIVE values as defined in the device data sheet.

3.1.2 RISE_TIME

This parameter defines the rise time of the MOSFET in nanoseconds used in the inverter. This value is used to compute the default IDRIVE and TDRIVE values as defined in the device data sheet.

3.1.3 FALL_TIME

This parameter defines the Fall time of the Mosfet in nanoseconds used in the inverter. This value is used to compute the default IDRIVE and TDRIVE values as defined in the device data sheet.

3.2 Customizing ISC User Parameters

The ISC user parameters are as follows:

```
//ISC User Parameters

#define ISC_THRESHOLD_MAX_SPEED     100
#define ISC_THRESHOLD_MIN_SPEED     50
#define ISC_MIN_LINE_BEMF        20
#define ISC_ZEROTH_PHASE_MATCH   12
#define ISC_FIRST_PHASE_MATCH    13
#define ISC_SECOND_PHASE_MATCH   14
```

3.2.1 ISC_THRESHOLD_MAX_SPEED

The ISC_THRESHOLD_MAX_SPEED parameter is used to set the maximum speed above which motor is allowed to coast at the start of the motor. During startup, if the motor is already spinning at high speeds, a high BEMF is detected along the phases, while braking, or while trying to apply the commutation state at such point could lead to huge current spikes. Therefore the motor is allowed to coast until the motor speed is less than the ISC_THRESHOLD_MAX_SPEED. When the speed is less than the defined speed by the ISC_THRESHOLD_MAX_SPEED parameter, ISC takes the control to either switch to closed loop (if it is spinning in correct direction) or coast the motor (if spinning in reverse direction) until it stops.

3.2.2 ISC_THRESHOLD_MIN_SPEED

The ISC_THRESHOLD_MIN_SPEED parameter is used to set the minimum speed below which motor is allowed to coast until the motor stops. During startup, if the motor is already spinning at very low speeds, no sufficient BEMF is detected along the phases to switch to closed-loop control. Therefore the motor is allowed to coast until the motor stops and the motor is started again normally.

3.2.3 ISC_MIN_LINE_BEMF

The ISC_MIN_LINE_BEMF parameter is used to set the minimum BEMF in digital counts above which the motor is assumed to be spinning and generating significant BEMF.

3.2.4 ISC_ZEROTH_PHASE_MATCH

To calculate the speed of the motor when the motor is free wheeling, the time between two commutation points must be measured. For this free wheeling, repeatedly sampled BEMF on the phases and any matching of the BEMF voltages on two different phases are considered a commutation point. For better accuracy, the algorithm leaves few commutation cycles to start reading the phase BEMF. This parameter defines the number of commutations to leave for identifying the Zeroth phase match to start measuring the time interval before next phase match.

3.2.5 ISC_FIRST_PHASE_MATCH

The ISC_FIRST_PHASE_MATCH parameter defines the number of commutations to leave for the ISC first phase match. This value should be equal to ISC_ZEROTH_PHASE_MATCH + 1.

3.3 ISC_SECOND_PHASE_MATCH

The ISC_SECOND_PHASE_MATCH parameter defines the number of commutations to leave for the ISC second phase match. This value should be equal to ISC_FIRST_PHASE_MATCH + 1.

3.4 Customizing IPD User Parameters

The IPD user parameters are as follows:

```
//IPD User Parameters
#define IPD_ADD_BRAKE            30
#define IPD_PULSE_TIME           3000
#define IPD_DECAY_CONSTANT       3
```

3.4.1 IPD_ADD_BRAKE

After the control energizes two phases with a pulse during IPD, that energy must be exhausted. This control method uses two modes to deplete the energy: braking the motor or turning on the low-side FET, and coasting the motor or turning off all the FETs. The IPD_ADD_BRAKE parameter specifies what additional length of time to the rise time is used to brake the motor. This number can be tuned to optimize the IPD control for a specific motor.

$$\text{ISC BRAKE TIME} = \text{ISC\_BRAKE\_TIME} \times 40\ \mu s \quad (1)$$

3.4.2 IPD_PULSE_TIME

The IPD_PULSE_TIME parameter sets the number of clock pulses for which a phase is excited and creates a rise in the phase current. This phase current is used for the current measurements in the IPD six step.

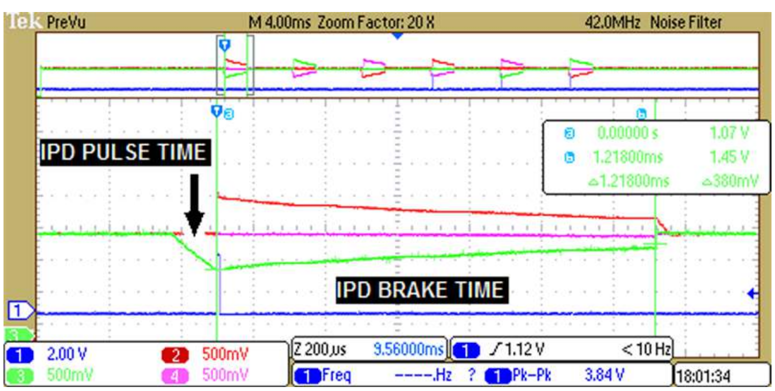

Figure 8. IPD_PULSE_TIME and IPD_ADD_BRAKE

3.4.3 IPD_DECAY_CONSTANT

After braking during IPD, the FETs are turned off; this is referred to as coasting. When the FETs are turned off, any remaining energy pumps back into the supply and the control begins the next IPD pulse. To allow some time for the remaining energy to escape before the next pulse, the IPD_DELAY_CONSTANT parameter sets the coast time to a multiple of the brake time.

$$\text{IPD Coast Time} = \text{IPD\_DECAY\_CONSTANT} \times \text{IPDBrakeTime} \quad (2)$$

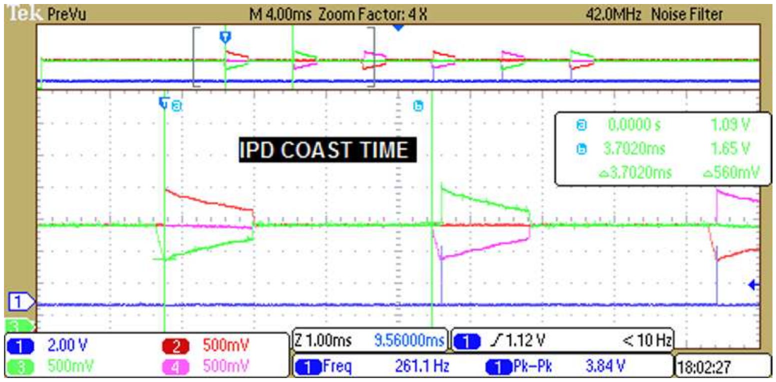

Figure 9. IPD Coast Time

3.5 Customizing ALIGN User Parameters

The ALIGN user parameters are as follows:

```
//Align User Parameters
#define ALIGN_SECTOR              1
#define ALIGN_WAIT_TIME           50
```

3.5.1 ALIGN_SECTOR

If the align function is used instead of the initial position detection, the control energizes the commutation sequence associated to this ALIGN_SECTOR parameter. The rotor starts the align sector position when the control moves to open-loop acceleration.

3.5.2 ALIGN_WAIT_TIME

The ALIGN_WAIT_TIME parameter is how long the control turns on the phases and holds it there for the settling time. For a motor with larger inertia, this number must be larger to allow the rotor to settle in the specified location. Motors with smaller inertia that move to one position quickly and not oscillate back and forth can use a smaller number here. If the ALIGN_WAIT_TIME parameter is set too low, the start-up can become unreliable because the rotor position is not correctly aligned.

3.6 Customizing OPEN LOOP ACCELERATION User Parameters

The OPEN LOOP ACCELERATION user parameters are as follows:

```
//Open Loop Acceleration User Parameters
#define ACCEL_RATE                      10
#define ACCEL_STOP                      30
#define ACCEL_VELOCITY_INIT             10
#define MOTOR_UNDER_RATING             120
```

3.6.1 ACCEL_RATE

The ACCEL_RATE parameter defines how fast the motor accelerates during open-loop control. Specifically, this number is used to increase the velocity and distance calculation during open-loop control. The unit associated with this parameter is hertz per second (Hz/s) where hertz is the electrical speed of the motor. For a motor with greater inertia or that requires a longer time to accelerate, set this number to a small value such as 1. Motors that can ramp up quickly can use a larger value for ACCEL_RATE to decrease the start-up time.

3.6.2 ACCEL_STOP

The ACCEL_STOP parameter defines the velocity when the control transitions from open-loop control to closed-loop control. In open-loop control, each time the motor switches the commutation state, the control compares the calculated velocity with this number. The value is specified in electrical hertz (Hz) which is the electrical speed of the motor. For motors that produce a larger BEMF at low speeds, this number can be set low to decrease the start-up time. Conversely, if a motor must spin faster to produce sufficient BEMF for control, this number must be set higher.

3.6.3 ACCEL_VELOCITY_INIT

The ACCEL_VELOCITY_INIT parameter is used to set the initial speed during open-loop start-up. For some motors, the initial speed can be set high because the motor can spin up fast enough and is not required to start from 1. However, for motors with high inertia or that are difficult to start up, this number can be set to 1.

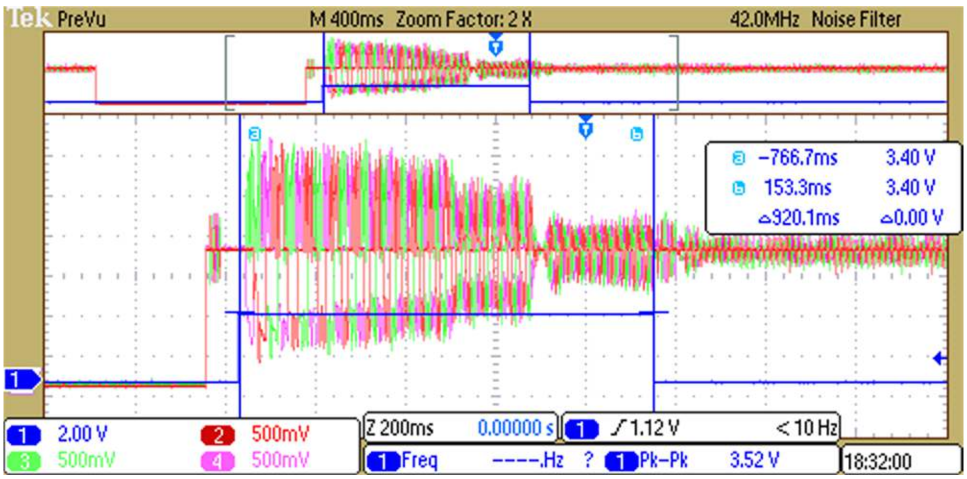

Figure 10. Open-Loop Acceleration

3.6.4 1.1.5. MOTOR UNDER RATING

The MOTOR UNDER RATING parameter defines the percentage of rated voltage at which motor spins at rated speed at no load. If a higher load is connected to the motor at startup, this value can be increased to compensate the load by applying higher duty cycles at start up. When the under rating value is increased, then the speed at which the motor can be switched to closed-loop control should be decreased so that the motor does not switch at very-high duty cycles.

3.7 Customizing CLOSED LOOP User Parameters

The CLOSED LOOP user parameters are as follows:

```
//Closed Loop User Parameters
#define BEMF_THRESHOLD            1260
#define RAMP_RATE_DELAY           20
#define RAMP_RATE                    1
#define COMMUTATION_BLANK_TIME       5
#define PWM_BLANK_COUNTS          5

#define MAX_DUTY_CYCLE            1000
#define MIN_OFF_DUTY              240
#define MIN_ON_DUTY                  250
#define PWM_FACTOR                0
```

3.7.1 BEMF_THRESHOLD

The motor is connected to the BOOSTXL-DRV8304x EVM and the phase voltage pins on the board are observed as shown in Figure 11. When the BLDC motor is run by the trapezoidal algorithm, the phase waveforms can be similar to the waveforms in Figure 11. The BEMF of the floating phase is either increasing or decreasing and is available for sampling by the ADC. This BEMF of the floating phase is periodically sampled and added (a digital form of integration) until it reaches the commutation point. Because this commutation point is not known, it is reverse-identified by using the BEMF threshold limit.

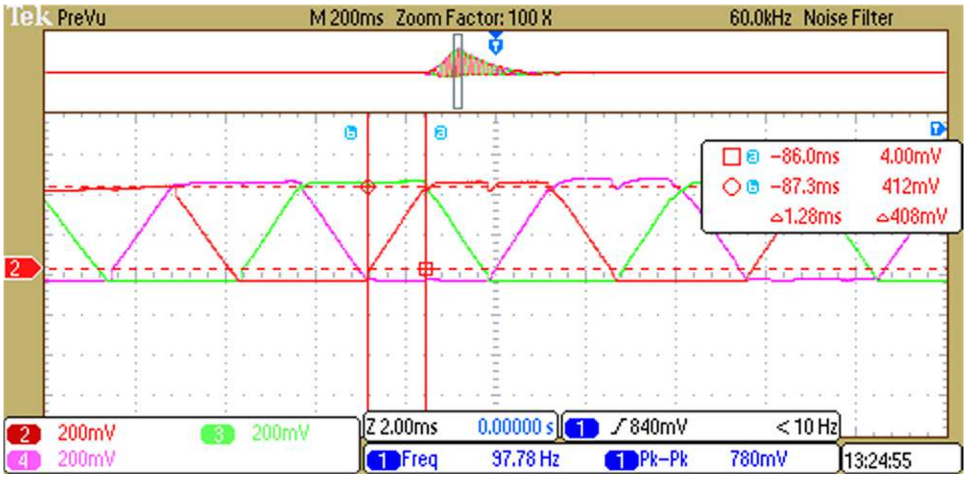

Figure 11. Observation of per Phase BEMF Waveform of the Motor When Motor is in Generating Mode

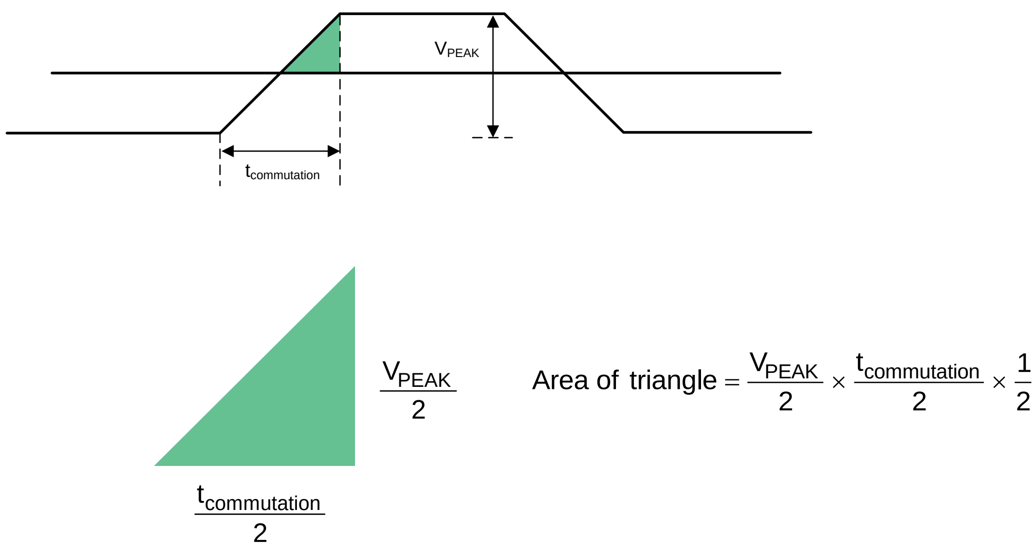

Figure 12. Calculation of BEMF Threshold Limit Using Trapezoidal BEMF Integration Method

Whenever the BEMF is greater than VCC / 2, the BEMF is sampled and summed up, which is equivalent to the area of the triangle shown in Figure 12. Therefore, whenever the summed up value crosses the area of the triangle (BEMF threshold limit), commutation of phases occurs.

From Figure 11, V_{PEAK} = 408 mV in digital counts 3.3 V, which is approximately 4096 counts; therefore 0.408 V = 506 counts with $t_{commutation}$ = 1.28 ms. As the ADC is sampled every 25 kHz or 41 µs (approximately), $t_{commutation}$ = 1.28 ms / 41 µs ≈ 31 samples. According to the CLOSED LOOP user parameters (see Section 3.7), the area of the triangle, or the BEMF threshold, equals (506 × 31) / 8 ≈ 1960.

3.7.2 RAMP_RATE_DELAY

The RAMP_RATE_DELAY parameter sets how many PWM_PERIOD interrupts must occur before adjusting the duty cycle. Changing this value changes how fast the duty cycle is adjusted. For example, if the PWM_PERIOD is 1024 or 40.96 µs and the RAMP_RATE_DELAY is 24, the duty cycle is adjusted every 983 µs. This parameter controls the acceleration and deceleration of the motor.

3.7.3 RAMP_RATE

The RAMP_RATE parameter indicates the amount of increase or decrease in the duty cycle for every PWM_PERIOD interrupt. If the system is either ramping up or down, and the acceleration count is reached, the duty cycle is increased or decreased by the RAMP_RATE parameter.

3.7.4 COMMUTATION_BLANK_TIME

When the closed-loop control switches the active phases of the motor, a short time must occur when the previous phase must deplete the energy built up in it. This depletion occurs as conduction through one of the body diodes of the FETs. To avoid this conduction being sampled in the control, a blanking time is available where the control does not monitor the BEMF. The COMMUTATION_BLANK_TIME parameter specifies the number of PWMs the control skips monitoring the BEMF. Figure 13 shows the brief period where the BEMF is blanked, indicated by the blue line going low. The blue trace is zero for a short period after every commutation.

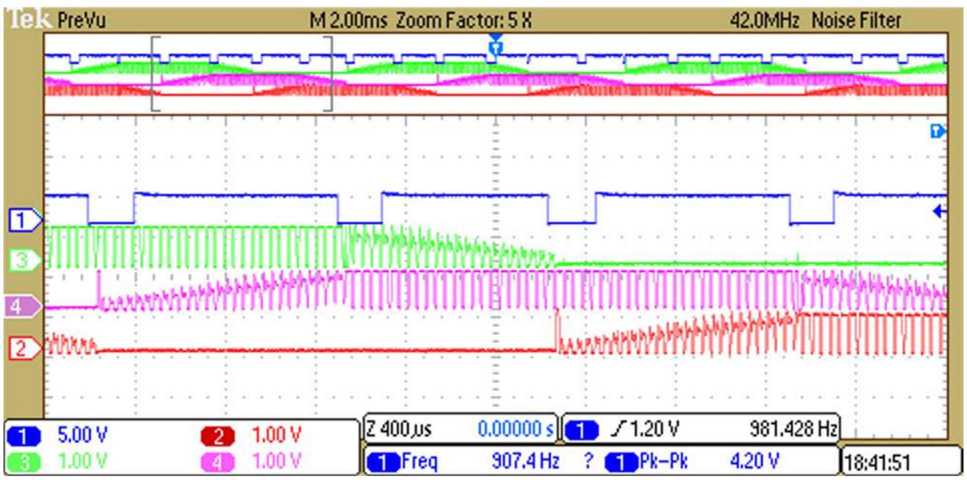

Figure 13. Commutation Blank Time

3.7.5 PWM_BLANK_COUNTS

The PWM_BLANK_COUNTS parameter sets a number of clock cycles before the one-fourth of the PWM width that the ADC for BEMF is sampled. The BEMF of some motors with high inductance require a longer time to settle to the final value. This number is a trade-off between the minimum duty cycle allowed and the settling time of the BEMF. Because this number specifies clock cycles, use Equation 3 to calculate the time.

$$\text{Time before PWM edge} = \frac{\text{PWM\_BLANK\_COUNTS}}{25\ \text{MHz}} \tag{3}$$

In Figure 14 the PWM_BLANK_COUNT is 80 which is equivalent to 3.2 µs.

As shown in Figure 14, when the duty cycle is around 15 µs / 40 µs = 35%, until the BEMF value starts falling.

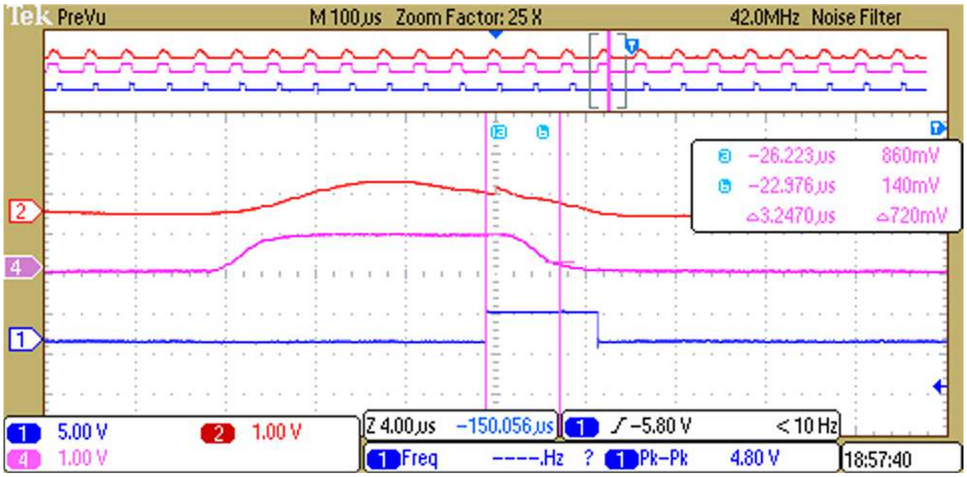

Figure 14. PWM BLANK COUNTS

3.7.6 MAX_DUTY_CYCLE

The MAX_DUTY_CYCLE parameter sets the maximum threshold that the input duty-cycle command can reach. Every time the input is read, the duty-cycle input command is compared to the MAX_DUTY_CYCLE parameter. If the duty-cycle command exceeds this parameter, the target duty cycle is set to the MAX_DUTY_CYCLE value.

$$\text{Maximum Duty Cycle (\%)} = \frac{\text{MAX\_DUTY\_CYCLE}}{\text{PWM\_PERIOD}} \tag{4}$$

3.7.7 MIN_OFF_DUTY_CYCLE

The MIN_OFF_DUTY_CYCLE parameter sets the minimum threshold that the input duty-cycle command is allowed to reach after already starting. Every time the input is read, the duty-cycle input command is compared to the MIN_OFF_DUTY_CYCLE parameter. If the duty-cycle input command is below this parameter, the target duty cycle is set to 0.

$$\text{Minimum Off Duty Cycle (\%)} = \frac{\text{MIN\_OFF\_DUTY\_CYCLE}}{\text{PWM\_PERIOD}} \tag{5}$$

3.7.8 MIN_ON_DUTY_CYCLE

The MIN_ON_DUTY_CYCLE parameter sets the threshold that the input duty-cycle command must reach before starting the control. After initialization, the input duty-cycle command is read and compared to the MIN_ON_DUTY_CYCLE parameter. The control waits to continue until the input is greater than the value of MIN_ON_DUTY_CYCLE.

$$\text{Minimum On Duty Cycle (\%)} = \frac{\text{MIN\_ON\_DUTY\_CYCLE}}{\text{PWM\_PERIOD}} \tag{6}$$

3.8 Customizing FAULT HANDLING User Parameters

The FAULT HANDLING user parameters are as follows:

```
/* Fault handling setup */
#define UNDER_VOLTAGE_LIMIT (8)
#define OVER_VOLTAGE_LIMIT    (20)
#define FULL_SCALE_VOLTAGE  (40)
#define MIN_POWER_SUPPLY    (5)
#define STALLDETECT_REV_THRESHOLD    (1)
#define STALLDETECT_TIMER_THRESHOLD (200)
#define MOTOR_PHASE_CURRENT_LIMIT (868)
#define AUTO_FAULT_RECOVERY_TIME (6000)
```

3.8.1 UNDER_VOLTAGE_LIMIT

Voltage monitoring measures the VCC applied through the internal ADC and compares the ADC measurement with the specified limits. If the voltage is less than the specified UNDER_VOLTAGE_LIMIT value, the code shuts off the predrivers and the device goes into the FAULT state.

$$\text{UNDER\_VOLTAGE\_LIMIT} = \frac{\text{MinVCC (V)}}{V_{FS}} \times 4096 \tag{7}$$

3.8.2 OVER_VOLTAGE_LIMIT

Coupled with the UNDER_VOLTAGE_LIMIT parameter, if the voltage is found to be above the specified OVER_VOLTAGE_LIMIT value, the code shuts off the predrivers and goes into the FAULT state.

$$\text{OVER\_VOLTAGE\_LIMIT} = \frac{\text{MinVCC (V)}}{V_{FS}} \times 4096 \tag{8}$$

3.8.2.1 FULL_SCALE_VOLTAGE

The FULL_SCALE_VOLTAGE parameter defines the maximum voltage that can be applied to the booster pack and can be sensed across the VSENSE resistance using an ADC (3.3 V maximum).

3.8.3 STALLDETECT_REV_THRESHOLD

In a certain amount of time, the motor should be spinning at least an established amount of revolutions. The number of revolutions is fixed by this parameter. In the set amount of time specified by the STALLDETECT_TIMER_THRESHOLD parameter, if the motor has not spun at least the count specified by this value, then the motor is assumed to have stalled.

3.8.4 STALLDETECT_TIMER_THRESHOLD

TimerB0 generates an interrupt service routine (ISR) every 1 ms and each ISR has a count that is increased. When the count reaches the STALLDETECT_TIMER_THRESHOLD value, if the current revolution count is less than the STALLDETECT_REV_THRESHOLD, the motor is stalled and the state machine goes into the FAULT state.

3.8.5 MOTOR_PHASE_CURRENT_LIMIT

The MOTOR_PHASE_CURRENT_LIMIT parameter defines the maximum allowed motor-phase peak current in amperes. The phase current of Motor A– is monitored every electrical cycle during commutation of phase A, and whenever the current limit is reached, an overcurrent (OC) fault is triggered. This value is scaled because it uses a 7-mΩ sense resistor on the DRV8304x EVM. The current-sense amplifier (CSA) gain 5 V/V is set using the firmware in the Initialization section for better sensitivity. This value can be modified accordingly when higher values of current sensing are required. With an ADC of 3.3-V full-scale value and 12-bit resolution, use Equation 9 to calculate the overcurrent limit in digital counts.

$$\text{MOTOR\_PHASE\_CURRENT\_LIMIT} = \frac{\text{Amperes} \times 0.007\ \Omega \times 5\ \frac{V}{V}}{3.3\ V} \times 4096 \tag{9}$$

3.8.6 AUTO_FAULT_RECOVERY_TIME

TimerB0 is used to recover after a fault has been detected. The FAULT_RECOVERY_TIME parameter is the value used for how many timer interrupts must occur before reinitializing the system. For example, if the TimerB0 interrupt is set to occur every 1 ms and the FAULT_RECOVERY_TIME is set to 3000, the system reinitializes 3 s after a fault was detected.

3.9 Customizing SPI REGISTER User Parameters

For the DRV8304 device set the SPI register settings accordingly from the *DRV8304 38-V Three-Phase Smart Gate Driver* data sheet. Modify the register settings using register page found in the GUI.

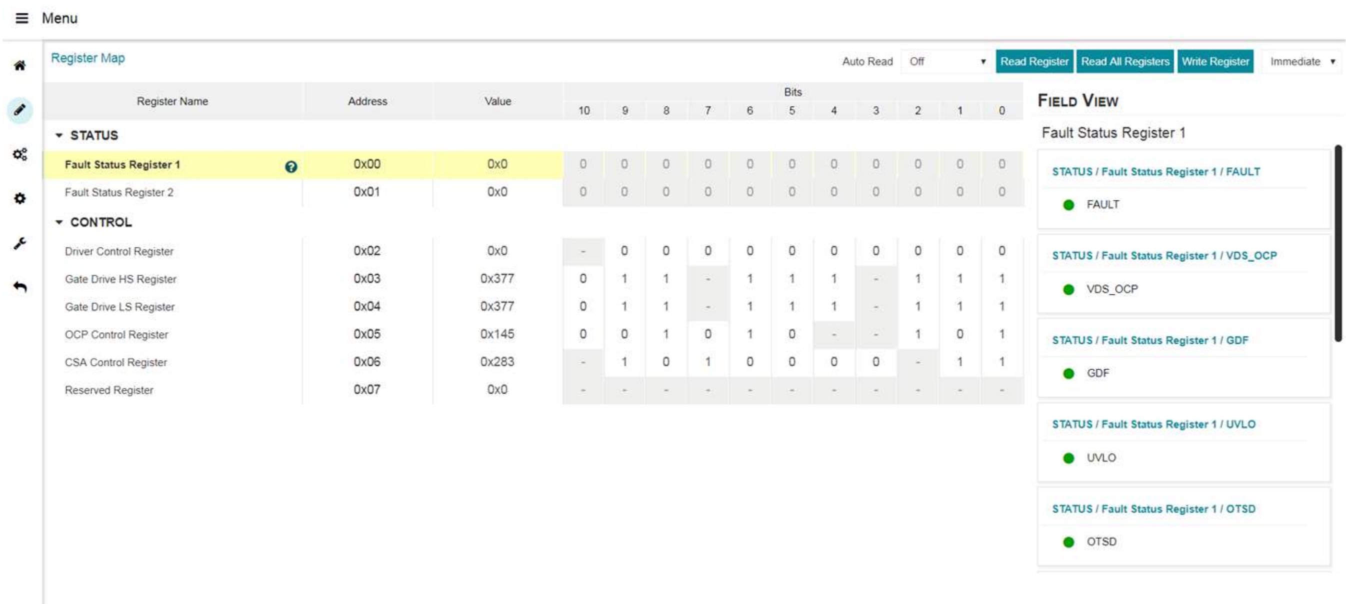

Figure 15. SPI REGISTER Setting Page in GUI

4 Running the Project in Code Composer Studio

To run the project in CCS, perform the steps that follow:

1. Install CCS software V6.1 or above.
2. Read through how to customize user parameters to tune the control for the specific motor.
3. Compile the modified project.
4. Connect the MSP430F5529 Launchpad to download and run the modified program.

The reference software was written for the Annaheim Motor. If a different motor is used and the reference code is unable to spin the motor, the motor was most likely improperly tuned. To properly tune the motor parameters, see Section 3.

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