# *TI Designs: TIDA-01496*
# *Single-Layer, 12-V, Programmable, 3-Phase BLDC Motor Driver With Speed Regulation Reference Design*

TEXAS INSTRUMENTS

## Description

This reference design is a sensorless BLDC motor sinusoidal drive using the DRV10983-Q1 motor driver and MSP430G2553 microcontroller (MCU). The MCU is only used for speed control while the DRV device is the main motor driver with integrated FETS which drives the motor. This design specifically targets small motor modules, particularly fans. This design allows access to proprietary sensorless control and the ability to tune motor parameters to optimize performance for the end application.

## Resources

| | |
|---|---|
| TIDA-01496 | Design Folder |
| DRV10983-Q1 | Product Folder |
| MSP430G2553-Q1 | Product Folder |
| DRV10xx Software | Software Download |

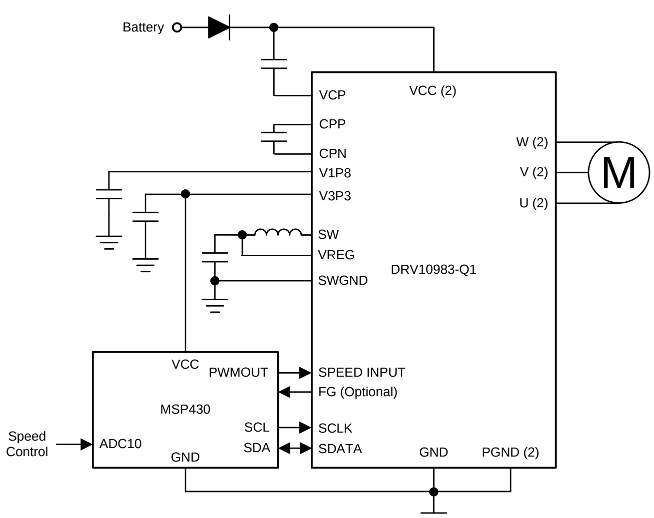
TI E2E™ Community — ASK Our E2E™ Experts

## Features

- 12-V Operational VBAT With up to 45-V Load Dump Support 3-Phase Brushless DC (BLDC) Motor Drive Solution Enabled by DRV10983-Q1
- Single-Layer Design Reduces Manufacturing Cost
- DRV10983-Q1 Uses Proprietary Sensorless Control Scheme to Provide Continuous Sinusoidal Drive, Significantly Reducing Pure Tone Acoustics
- MSP430G2553 MCU Processes Speed Input, Control Loop, and Programs Motor Parameters
- Uses DRV10983-Q1 Three-Phase Sensorless Motor Driver With Integrated Power MOSFETs to Provide Continuous Drive Current up to 2 A (3-A Peak)
- Output Slew Rate and Frequency Configurability Offered From DRV10983-Q1 Adds Flexibility to Tune for EMC Performance
- Integrated Buck Regulator Efficiently Steps Down Supply Voltage to Either 5.0 V or 3.3 V to Power Both Internal and External Circuits

## Applications

- Full Featured Seats
- Heated / Cooled Seats






An IMPORTANT NOTICE at the end of this TI reference design addresses authorized use, intellectual property matters and other important disclaimers and information.

# 1 System Description

This reference design is a cost-effective, small-form-factor (SFF), single-layer, three-phase sinusoidal motor drive for brushless DC (BLDC) motors. The board accepts 12 V at input and provides the three motor phase outputs to drive the BLDC motor sinusoidally. The board consists of the DRV10983-Q1 motor driver and MSP430G2553 MCU. The DRV10983-Q1 delivers 3.3 V to the MCU through its V3P3 LDO regulator.

The board also has a speed pin input, which is connected to the ADC10 of the MSP430™ MCU. The ADC10 takes an input voltage from 0 to 3.3 V and converts it to a digital 10-bit value. This value controls the percentage duty cycle for the output PWM signal. The MSP430 MCU creates a pulse width modulation (PWM) signal output, which connects to the DRV10983-Q1 SPEED input pin. The DRV10983-Q1 then uses this input to control the speed of the motor.

The SPEED pin in the DRV10983-Q1 accepts either an analog or PWM input. The DRV10983-Q1 device provides motor speed measurement information on the frequency generator (FG) output or $I^2C$, which can be used to implement the speed control loop. The $I^2C$ interface is also available as a header where an external graphical user interface (GUI) can be used for programming the DRV10983-Q1 device. For this reference design, the GUI can be used to configure and tune the motor parameters.

The MSP430G2553 MCU uses the speed measurement feedback to implement a speed control loop. The MSP430G2553 also programs the motor parameter registers into the DRV10983-Q1 without the need of a GUI. The DRV10983-Q1 device is specifically designed for cost-sensitive, low-noise, low-external-component count, small-motor applications. This reference design shows the application of an automotive seat blower fan. The DRV10983-Q1 device uses a proprietary sensorless control scheme to provide continuous sinusoidal drive, which significantly reduces the pure tone acoustics that typically occur as a result of commutation.

## *1.1 Key System Specifications*

**Table 1. Key System Specifications**

| PARAMETER | SPECIFICATIONS |
|---|---|
| DC input voltage | 8 to 16 V |
| Current | 1.0 A continuous |
| Power level | Thermal design for 8- to 16-W operation |
| Control method | Integrated 180° sinusoidal control |
| Analog speed input | 0 to 3.3 V |
| Protection circuits | Overcurrent, lock detection, voltage surge protection, UVLO protection, thermal shutdown |
| Operating ambient | –40°C to +125°C |

# 2 System Overview

## 2.1 Block Diagram

Figure 1 shows the system block diagram for this reference design. The system is supplied with a motor supply voltage of 12 V, which goes to the DRV10983-Q1 device. The DRV10983-Q1 driver's P3V3 LDO regulator supplies 3.3 V to power the MSP430 MCU. Speed control voltage ranging from 0 to 3.3 V is input to the MSP430 MCU, which drives the PWM duty cycle into the DRV10983-Q1, which drives the speed of the motor. $I^2C$ connections are used to program the registers onto the DRV10983-Q1 as well as provide speed feedback data for the speed control loop.

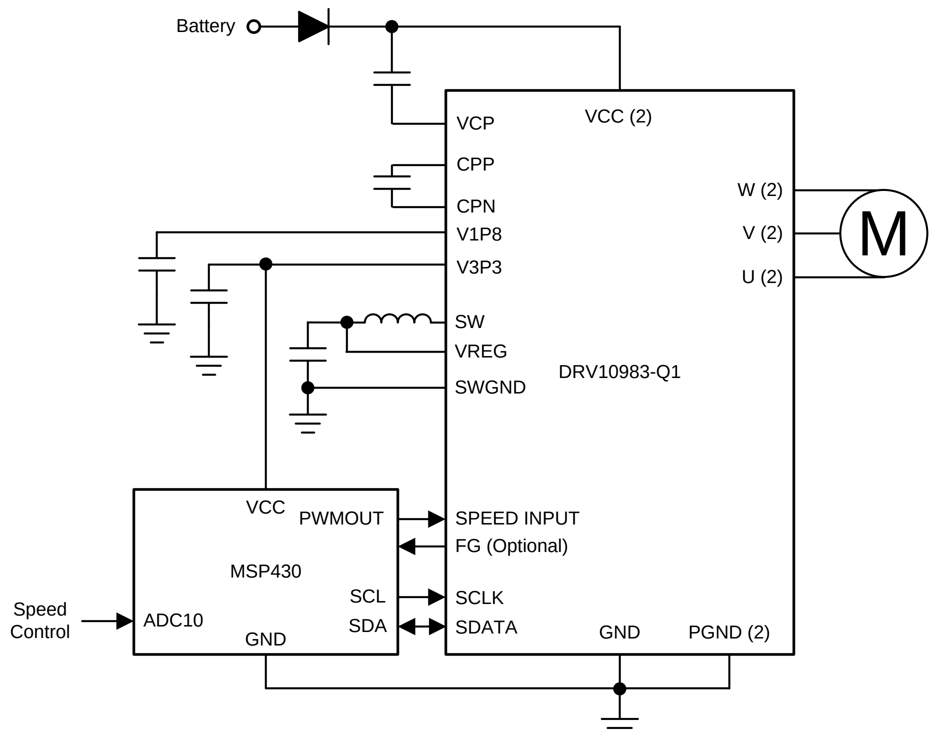



**Figure 1. System Block Diagram**

The DRV10983 driver can be configured through $I^2C$ using the external GUI with the header available onboard. The user can also implement the MSP430 MCU using a JTAG interface if additional features such as speed loop are required.

## 2.2 Highlighted Products

### 2.2.1 DRV10983-Q1

The DRV10983-Q1 device is a three-phase sensorless motor driver with integrated power MOSFETs, which can provide continuous drive current up to 2 A. The device is specifically designed for cost-sensitive, low-noise, low-external-component-count fan and pump applications.

The DRV10983-Q1 device preserves register setting down to 4.5 V and delivers current to the motor with supply voltage as low as 6.2 V. If the power supply voltage is higher than 28 V, the device stops driving the motor and protects the DRV10983-Q1 circuitry. This function is able to handle a load dump condition up to 45 V.

Table 2 highlights the different versions of the DRV10983-Q1: sleep version and standby version. The DRV10983-Q1 enters either sleep or standby to conserve energy. For more information, see *DRV10983-Q1 Automotive, Three-Phase, Sensorless BLDC Motor Driver*.

**Table 2. Device Options for DRV10983-Q1**

| DEVICE OPTION | DESCRIPTION | PACKAGE | BODY SIZE (NOM) |
|---|---|---|---|
| DRV10983Q | Sleep version | HTSSOP (24) | 7.80 × 6.40 mm |
| DRV10983SQ | Standby version | HTSSOP (24) | 7.80 × 6.40 mm |

The DRV10983-Q1 device uses a proprietary sensorless control scheme to provide continuous sinusoidal drive, which significantly reduces the pure tone acoustics that typically occur as a result of commutation. The interface to the device is designed to be simple and flexible. The motor can be controlled directly through PWM, analog, or $I^2C$ inputs. Motor speed feedback is available through both the FG pin and the $I^2C$ interface simultaneously.

The DRV10983-Q1 device features an integrated buck regulator to step down the supply voltage efficiently to 5 V for powering both internal and external circuits. The 3.3-V LDO also may be used to provide power for external circuits. The device is available in either a sleep mode or a standby mode version to conserve power when the motor is not running. The standby mode (8.5 mA) version (DRV10983SQ) leaves the regulator running and the sleep mode (48 µA) version (DRV10983Q) shuts off the regulator. Use the standby mode version in applications where the regulator is used to power an external microcontroller. Throughout this datasheet, the DRV10983-Q1 is used for both devices [DRV10983Q (sleep version) and DRV10983SQ (standby version)], except for specific discussions of sleep versus standby functionality.

An $I^2C$ interface allows the user to reprogram specific motor parameters in registers and to program the EEPROM to help optimize the performance for a given application. The DRV10983-Q1 device is available in a thermally-efficient HTSSOP, 24-pin package with an exposed thermal pad. The operating ambient temperature is specified from –40°C to +125°C.

### 2.2.2 MSP430G2553

The Texas Instruments MSP430 family of ultra-low-power microcontrollers consists of several devices featuring different sets of peripherals targeted for various applications. The architecture, combined with five low-power modes, is optimized to achieve extended battery life in portable measurement applications. The device features a powerful 16-bit RISC CPU, 16-bit registers, and constant generators that contribute to maximum code efficiency. The digitally controlled oscillator (DCO) allows wake-up from low-power modes to active mode in less than 1 µs.

The MSP430G2x13 and MSP430G2x53 series are ultra-low-power mixed signal microcontrollers with built-in 16-bit timers, up to 24 I/O capacitive-touch enabled pins, a versatile analog comparator, and built-in communication capability using the universal serial communication interface. In addition, the MSP430G2x53 family members have a 10-bit analog-to-digital (A/D) converter.

Typical applications include low-cost sensor systems that capture analog signals, convert them to digital values, and then process the data for display or for transmission to a host system.

### 2.3 Design Considerations

Table 3 shows the recommended components for the DRV10983-Q1 device to function. In addition to these components, a Schottky diode is added at VCC for reverse voltage protection as well as an addition capacitor in parallel with $C_{VCC}$. JTAG pin headers are used for MSP430 programming and $I^2C$ headers are used for GUI connection and EEPROM register programming. For compact board layouts, it is sufficient to use a smaller inductor for the buck regulator to minimize board footprint. Additional layout guidelines are found in *DRV10983-Q1 Automotive, Three-Phase, Sensorless BLDC Motor Driver*.

**Table 3. External Components**

| COMPONENT | PIN 1 | PIN 2 | RECOMMENDED |
|---|---|---|---|
| C1 | VCC | GND | 10-µF ceramic capacitor rated for VCC |
| C2 | VCP | VCC | 0.1-µF ceramic capacitor rated for 10 V |
| C3 | CPP | CPN | 10-nF ceramic capacitor rated for VCC × 2 |
| C4 | VREG | GND | 10-µF ceramic capacitor rated for 10 V |
| C5 | V3P3 | GND | 1-µF ceramic capacitor rated for 5 V |
| C6 | V1P8 | GND | 1-µF ceramic capacitor rated for 5 V |
| C7 | DVCC | GND | 0.1-µF ceramic capacitor |
| C8 | RST | GND | 2200-pF ceramic capacitor |
| C10 | VCC | GND | (DNP) 10-µF ceramic capacitor rated for VCC |
| C11 | P1.1 | GND | 470-pF ceramic capacitor |
| C9 | VCC | GND | 0.1-µF ceramic capacitor rated for VCC |
| D1 | VSource | VCC | Schottky diode rated for 30 V, 2 A |
| L1 | SW | VREG | 47-µH ferrite rated for 1.15 A (inductive mode) |
| R1 | FG | V3P3 | 4.75-kΩ pullup to V3P3 |
| R2 | SDA | V3P3 | 4.75-kΩ pullup to V3P3 |
| R3 | SCL | V3P3 | 4.75-kΩ pullup to V3P3 |
| R4 | SPEED | GND | 4.75-kΩ pulldown |
| R5 | DIR | GND | 4.75-kΩ pulldown |
| R6 | DVCC | RST | 47 kΩ |
| R10 | FG | FG | Jumper |

## 2.4 System Design Theory

### 2.4.1 PI Speed Loop Implementation

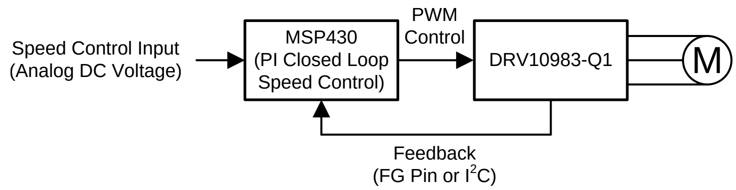



**Figure 2. High-Level System Block Diagram With Speed Loop**

The system design includes two main device components, the MSP430G2553 and DRV10983-Q1. The MSP430 MCU takes in a voltage input through its ADC10 and converts it to a 10-bit number. This number determines the percentage duty cycle of the PWM output to the DRV10983-Q1. The MSP430G2553 also programs the motor parameters into the DRV10983-Q1 EEPROM registers through $I^2C$. The DRV10983-Q1 drives the motor and at the same time send motor speed feedback information through $I^2C$ or FG pin. This feedback goes back to the MSP430 MCU, which processes the proportional integrator (PI) speed control loop.

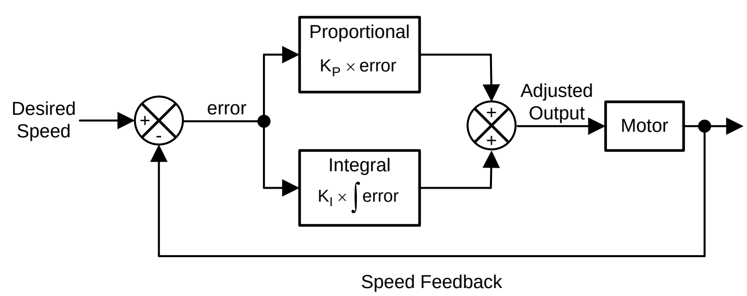



**Figure 3. PI Loop Implementation**

A PI controller is made up of two separate controller terms, the proportional term and the integral term. The system takes in the desired input and compares it to the current feedback from the output system. The difference between these two values is the error term. This calculated error term is the input to the controller. The proportional term is calculated by taking the proportion gain, $K_P$, and multiplying it by the magnitude of the error term. The integral of the error is also calculated and multiplied by the integral gain, $K_I$. These two values are then added to each other and output the new speed command to the motor. The motor takes this command and turn while at the same time providing feedback to the PI controller. The gains of the proportional and integral terms determine the overshoot and settling time.

# 3 Hardware, Software, Testing Requirements, and Test Results

## *3.1 Required Hardware and Software*

### 3.1.1 Hardware

This reference design is powered through the $V_{BAT}$ input and controlled using the $V_{SP}$ input. Specifically, $V_{SP}$ refers to the analog voltage used by the MCU that generates the PWM signals necessary to control the speed of the motor. The vias marked U, V, and W refer to the phase winding inputs of a three-phase BLDC motor. J1 is used for $I^2C$ communication with the motor driver and MCU while J2 is used for the JTAG interface with the MCU. While the DRV10983-Q1 can drive the motor without the use of an MCU, the design uses the MCU for implementing the speed loop functionality and providing the PWM speed signal for motor drive.

To set up the system, connect a power supply capable of providing 12 to 28 V and 3 A using $V_{BAT}$. Note to turn on the power supply only when all connections are finalized. Then, connect another voltage source capable of providing 3.3 V using $V_{SP}$. It is recommended to use a USB2ANY and the DRV10x Software GUI when communicating with the DRV10983-Q1 using $I^2C$. For pins 1 through 3 of the J1, see the SDA, SCL, and GND connections needed for $I^2C$ communication. See Figure 4 for more information when connecting to the system through $I^2C$.

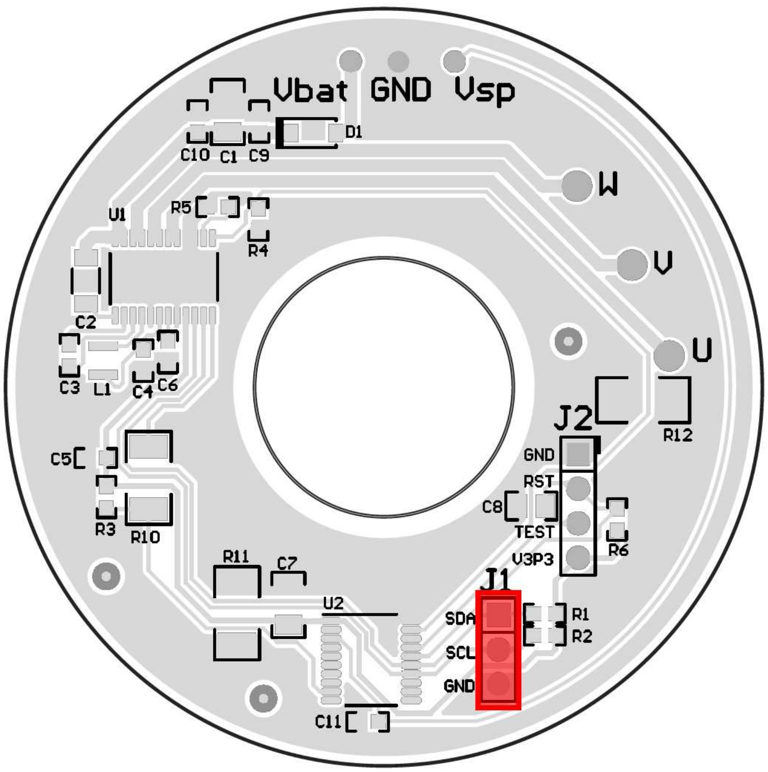



**Figure 4. Hardware Setup for $I^2C$ Communication**

It is also recommended to use a JTAG emulator like the one found through the MSP430 LaunchPad™ Value Line Development Kit (MSP-EXP430G2) to interact with the MSP430G2553. See Figure 5 for more information when connecting to the system using the JTAG interface. For pins 1 through 4 of J2, see the GND, RESET, TEST, and V3P3 required for the JTAG interface. For more information about I$^2$C and the JTAG interface, see Section 3.1.2.

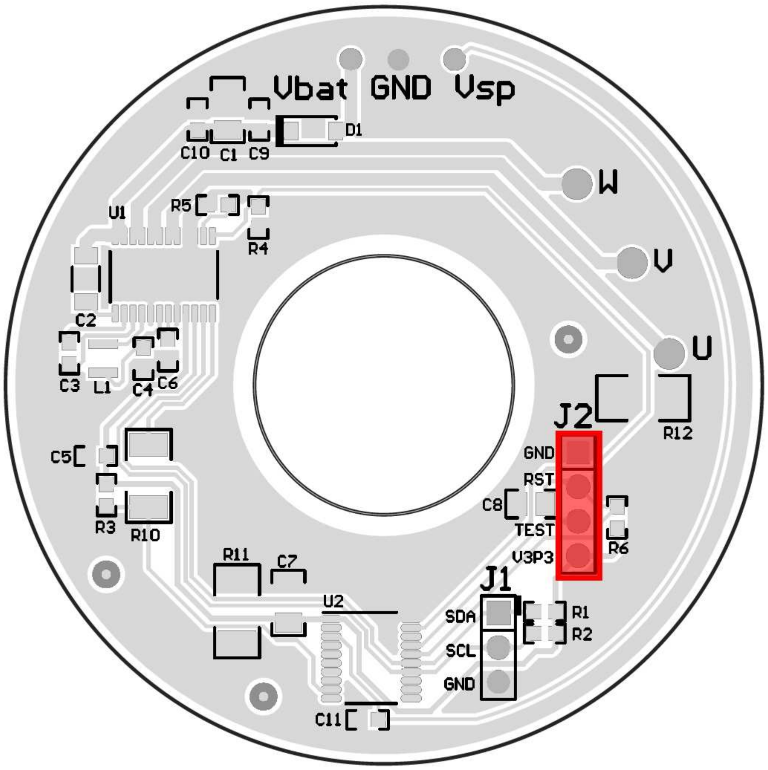



**Figure 5. Hardware Setup for JTAG Interface and Communication With MSP430 MCU**

Once the MCU has been programmed and the DRV10983-Q1's EEPROM settings have been tuned, the user only needs to power on the system and control the speed voltage to operate the design.

### 3.1.2 Software

The firmware is written for the onboard MSP430 controller. The existing firmware serves two major functionalities:

- The firmware can configure the DRV10983-Q1 device. This firmware is usually a one-time configuration unless the user wants to modify parameters runtime of the configuration.
- The firmware processes the input speed using the ADC10 of the MSP430 MCU and implements a PI speed control system using feedback from the DRV10983-Q1 from either $I^2C$ or FG.

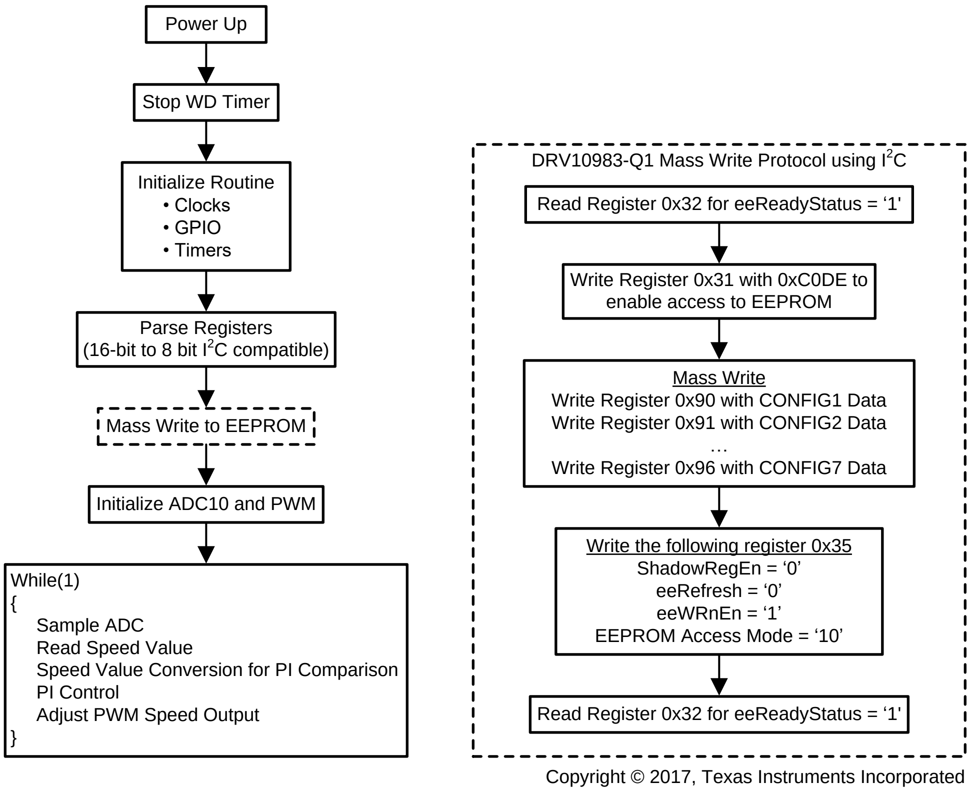



**Figure 6. Software System Block Diagram**

#### 3.1.2.1 Using Software to Configure DRV10983-Q1

Configuring the DRV10983-Q1 refers to programming the tuning parameters, which are written to the EEPROM. These settings contain intrinsic motor parameters unique to the motor, such as Kt and Phase Resistance, and tuning parameters just as the acceleration coefficients and open to closed loop threshold. Consult the *DRV10983-Q1 Tuning Guide* for more information.

Ideally, the motor parameters can be obtained by using $I^2C$ to communicate with the motor driver. Download the *DRV10xx Software GUI*. [(1)] to easily implement EEPROM settings and monitor motor faults during operation. A guide for installing the GUI software is found in "Appendix A GUI Installation and Overview" of the *DRV10983-Q1 Evaluation Module User's Guide*.

When interfacing with the MSP430G2553, download the latest version of *Code Composer Studio™ (CCS)* to easily implement the provided code. By simply importing the project or copying and pasting the code, the MCU can be programmed to implement EEPROM settings, the speed loop, and the PWM signals.

It is highly recommended to make all the connections first before turning on the voltage supply. Once the connections are made, the voltage supply can be turned on. A common error can occur during programming: "MSP430: Error initializing emulator: Could not find MSP-FET430UIF on specified COM port". Usually this error can be remedied by turning off the power supply, disconnecting the LaunchPad from the computer, reconnecting the LaunchPad first, and then turning on the power supply again.

Follow these instructions to program the MSP430 MCU:

1. Import the "TIDA-01496-ExampleSoftware" project using TI's CCS software.
2. Connect the LaunchPad programmer to the TIDA-01496 board, as shown in Figure 5.
3. Build the project by clicking the Build button (hammer icon), which if run successfully, appears as shown in Figure 7.
4. Click the Debug button (bug icon) as Figure 7 shows.

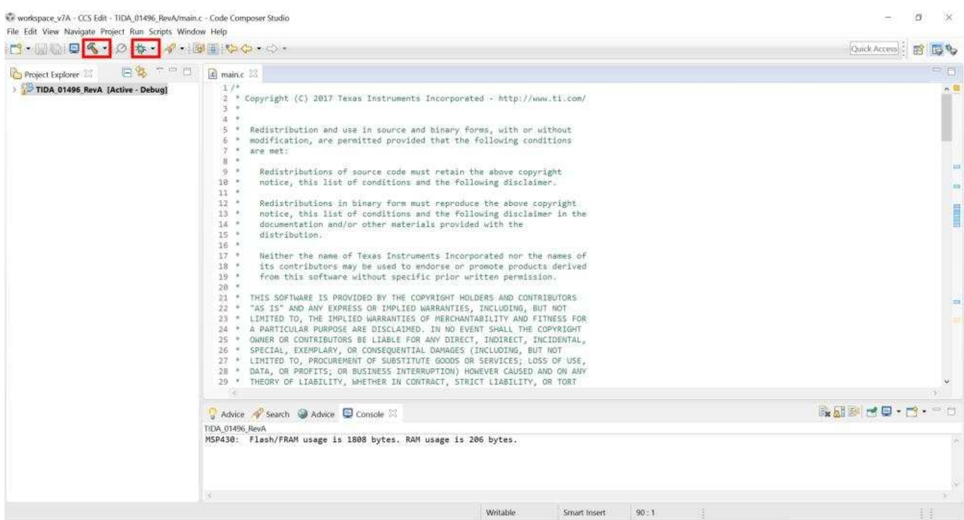


**Figure 7. Building and Debugging Project in CCS**

(1) This requires a TI.com login

Follow these instructions to program the motor parameter registers into the MSP430 software:

1. Find the optimum motor parameter settings using the GUI. For help tuning the motor, see the *DRV10983-Q1 Tuning Guide*.
2. Save the motor configurations using the GUI. This action saves the seven CONFIG registers as a .csv file as shown in Figure 8.


**Figure 8. How to Save a Motor Configuration File in DRV10x GUI**

3. Open the .csv file with Notepad. These registers are now implemented in the reference design CCS as shown in Figure 9.

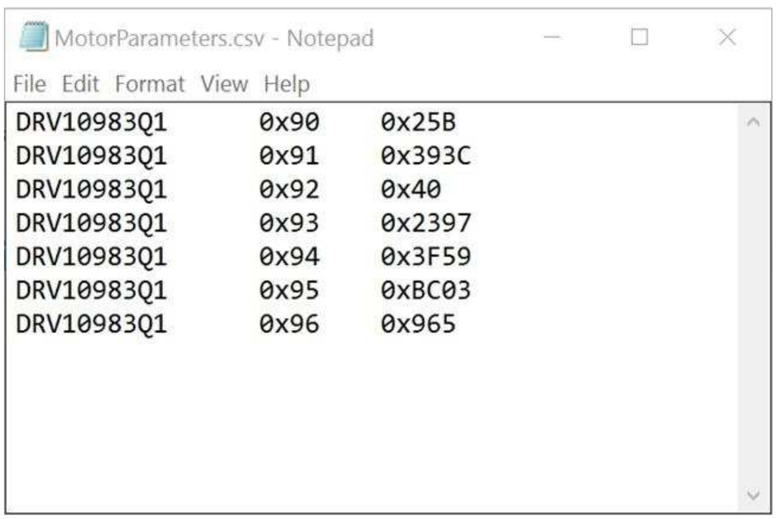



**Figure 9. Example Motor Configuration File in .csv Format**

4. Open the RegisterValues.h header file and place the CONFIG values in their corresponding registers as shown in Figure 10.

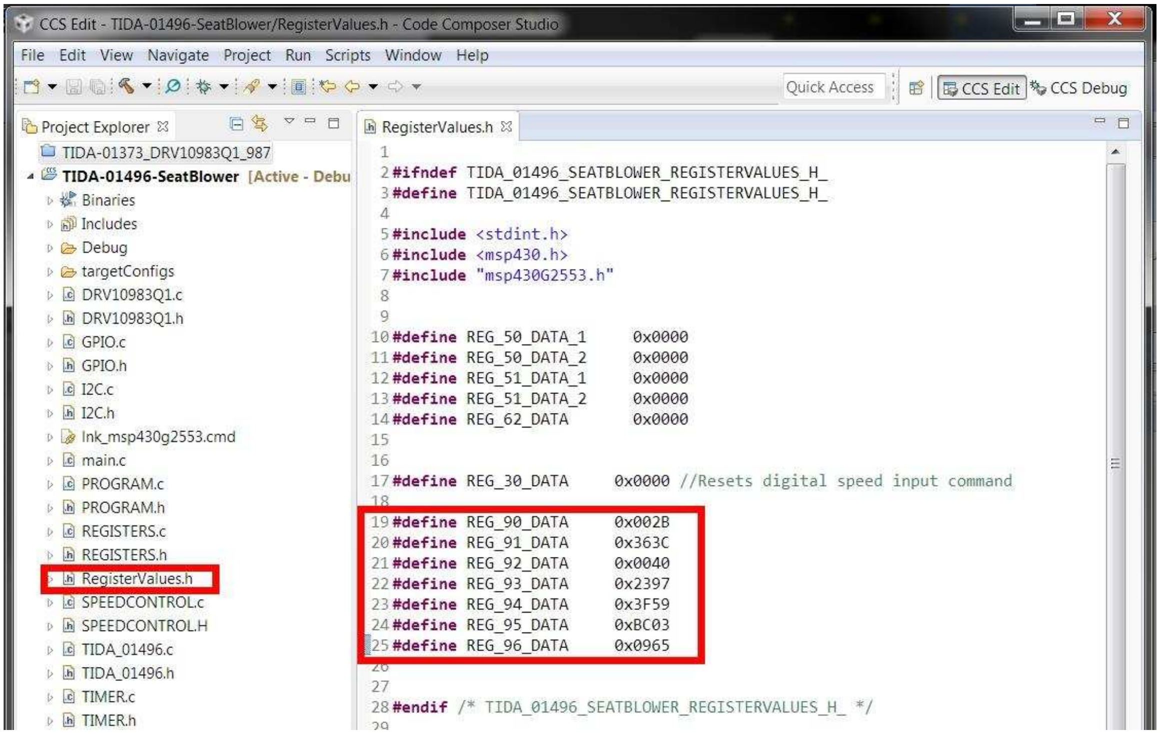


**Figure 10. Adding Motor Configuration Settings into RegisterValues.h**

5. Save the project and rebuild.
6. Optional: Program other registers the seven CONFIG registers if needed:
   a. Define the new register value in the RegisterValues.h header file.
   b. Define the number of total registers in the REGISTERS.h header file as shown in Figure 11.

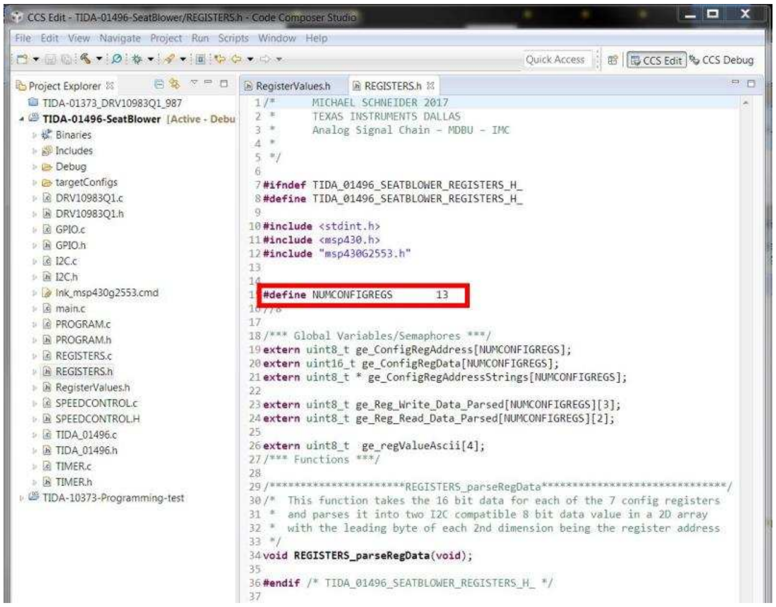


**Figure 11. How to Change Total Number of Registers Used in Project**

c. Define the register address in the REGISTERS.c source file in both corresponding arrays as shown in Figure 12.

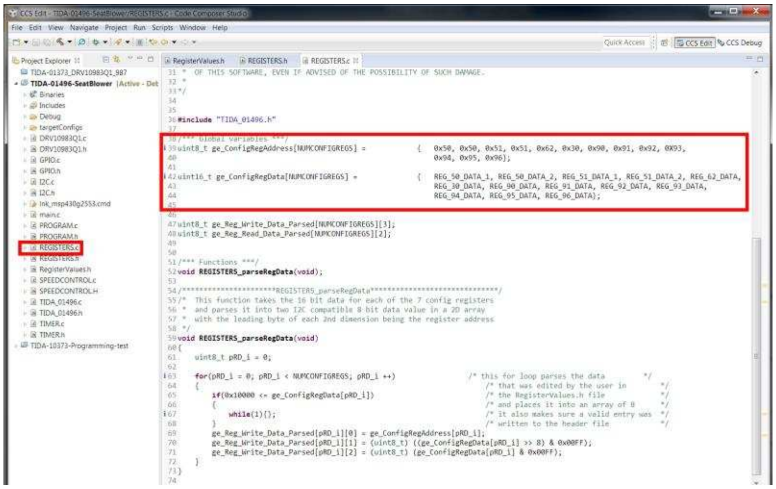


**Figure 12. How to Change Array of Registers Used in Project**

#### *3.1.2.2 Flow for Using ADC10 of MSP430 to Implement PI Speed Control System*

As shown in Figure 6, after initialization, the program stays in the while loop continually running the SPEEDCONTROL function. The SPEEDCONTROL function flows by the following process:

1. ADC samples voltage of MCU and translates into a bit value defined by *ADC_Value*.
2. Using this bit value, a speed level is assigned (that is, 0 to 5) defined by *Speed_Level*.
3. The speed level determines the target speed of the electrical cycle (Hz), defined by *SPEED_INPUT*, and scale value, defined by *Scale*.
4. The current speed of the electrical cycle speed is read through I$^2$C from the DRV10983-Q1 defined by *SPEED_MEASURED*.
5. The difference between the target speed and current speed is defined by *Error*.
6. The PI loop operation calculates the error and adjustments needed to reach the target speed defined by *OutPreSat*.
7. An if statement is used to bound the error and adjustments if the error is too large defined by *Out* (Hz).
8. The error for the next cycle used by the PI loop is determined by *SatError*.
9. The PWM signal uses the error and adjustments. This is defined by *CCR1* (Hz).

### *3.1.2.3 Tuning Speed Levels*

Each motor has its own specific speed curve and parameters. This requires tuning for the speed levels and PI loop. Any speed levels can be set for the device depending on the application of the system. The easiest method is to run the firmware with one simple modification. In the SPEEDCONTROL.c file, change:

`CCR1 = NEW_SPEED;` to `CCR1 = ADC_Value;`

This change is shown in Figure 13.

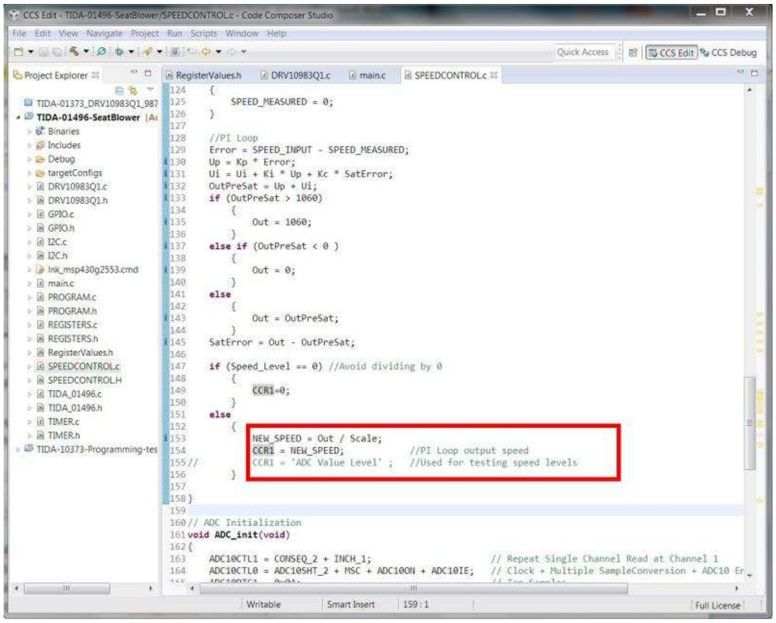


**Figure 13. Example of Changing SPEEDCONTROL.c File for Testing**

The ADC_Value is the value read by the ADC10 ranging from 0 to 1023. There needs to be a one-to-one relationship between the ADC_Value and the PWM output, meaning an ADC_Value of 1023 is 100% PWM and an ADC_Value of 511 is 50% PWM. Under ideal conditions, choose the required speed levels and convert them to an ADC_Value. See Table 4 for an example:

**Table 4. Translating Speed Levels, PWM Duty Cycles, and ADC_Value**

| SPEED LEVEL | PWM DUTY | ADC_Value (1023×DUTY) |
|---|---|---|
| 0 | 0% | 0 |
| 1 | 20% | 205 |
| 2 | 40% | 409 |
| 3 | 60% | 614 |
| 4 | 80% | 818 |
| 5 | 100% | 1023 |

Now set the ADC_Value ranges in the SPEEDCOMMAND.c file to correspond with these specified levels:

```
if (ADC_Value <= 50) {Speed_Level = 0;}
else if (ADC_Value > 50 && ADC_Value <=205)  {Speed_Level = 1;}
else if (ADC_Value >205 && ADC_Value <=409)  {Speed_Level = 2;}
else if (ADC_Value >409 && ADC_Value <=614)  {Speed_Level = 3;}
else if (ADC_Value >614 && ADC_Value <=818)  {Speed_Level = 4;}
else if (ADC_Value >818 && ADC_Value <=1023) {Speed_Level = 5;}
else {Speed_Level = 5;}
```

### 3.1.2.4 Finding Appropriate Scale for Each Speed Level

Usually, the ADC_Value input, and by extension the SPEED_INPUT, will be compared to the SPEED_MEASURED and put through the PI loop. Setting the CCR1 value will manually determine the PWM duty cycle and ignore the effects of the PI loop. This is important because unique motor parameters will cause different speeds at 100% duty cycle. Determining the max speed will determine the scale between the PWM duty cycle and the SPEED_INPUT.

To determine the scale, observe the PWM duty cycle and SPEED_MEASURED for each CCR1 value and determine the scale value by dividing the CCR1 value by the SPEED_MEASURED. This is implemented in Figure 14 and listed in Table 5.

To manually set CCR1, implement the following code:

```
CCR1 = 1060;
```

Each time a speed is set, run the motor and check the SPEED_MEASURED variable. To do this, run the debugger and add SPEED_MEASURED to the expressions table by right clicking the table and selecting "add global variable". Pause the debugger while the motor is running in closed loop to observe the speed value measured by the DRV10983-Q1 using $I^2C$.

**Table 5. Example of Tuning SPEED_INPUT(1) Using Scale Functionality**

| SPEED LEVEL | EXAMPLE PWM DUTY | EXAMPLE CCR1 Value | EXAMPLE MEASURED (SPEED_MEASURED) | EXAMPLE SCALE (CCR1/SPEED_MEASURED) |
|---|---|---|---|---|
| 0 | 0% | 0 | 0 | — |
| 1 | 20% | 205 | 31 | 6.61 |
| 2 | 40% | 409 | 63 | 6.49 |
| 3 | 60% | 614 | 94 | 6.53 |
| 4 | 80% | 818 | 126 | 6.49 |
| 5 | 100% | 1060 | 163 | 6.5 |

(1) SPEED_INPUT is motor specific

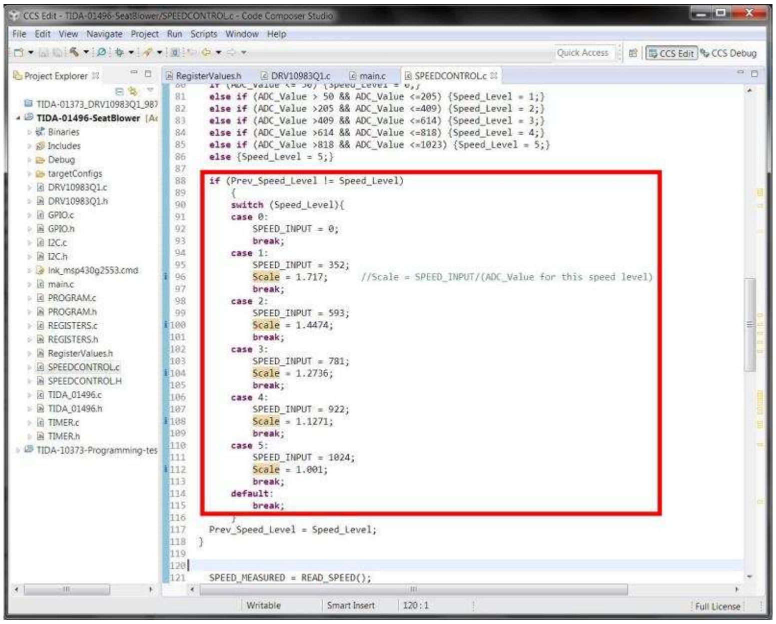


**Figure 14. Example Implementation of Scaled SPEED_INPUT in CCS Project**

These speed levels may or may not be linear. This case is for a non-linear relationship between the input command and output speed due to the motor's inherent load properties. For a more linear motor, a constant scale can be set for any input without using speed levels and these calculations can be simplified.

### *3.1.2.5 Tuning the PI Loop*

Tuning the PI loop consists of simply modifying the proportional and integral constants in the SPEEDCONTROL.c file.

The PI controller is implemented within a while loop, which runs continuously. First, the error is calculated between the desired speed and the feedback speed. The error is multiplied by the proportional gain and the output of the proportional term is stored in the $U_p$term. The integral output is calculated by adding together the previous integral output, the product of the integral gain and proportional output, and the product of the integral gain factor and the saturated error. The pre-saturated value is the sum of the integral and proportional outputs, $U_i$ and $U_p$. The saturated error is calculated between the output after saturation and the pre-saturation output and applied to the integral output. A PI loop is an essential controller method when closing the speed loop within a motor. The loop allows motors to work across different voltages and different loads while maintaining the same desired speed input. To learn more about the PI loop theory, see Section 2.4.1.

Tuning a PI loop is required to make the system stable and responsive while minimizing overshoot. The last two objectives often conflict with each other, and a compromise must be met to achieve the best results. Tuning a PI loop for a system with a fast response such as for this motor application must have a low P-gain. Set the P-gain low and the I-gain to near zero and run the motor to see if oscillations occur. Once oscillations occur, halve the P-gain value. Next, set a very small integral value. Slowly increase and run the motor until oscillations occur more than half the integral value. Figure 15 demonstrates the effects of increasing the proportional and integral terms on overshoot and setting time.

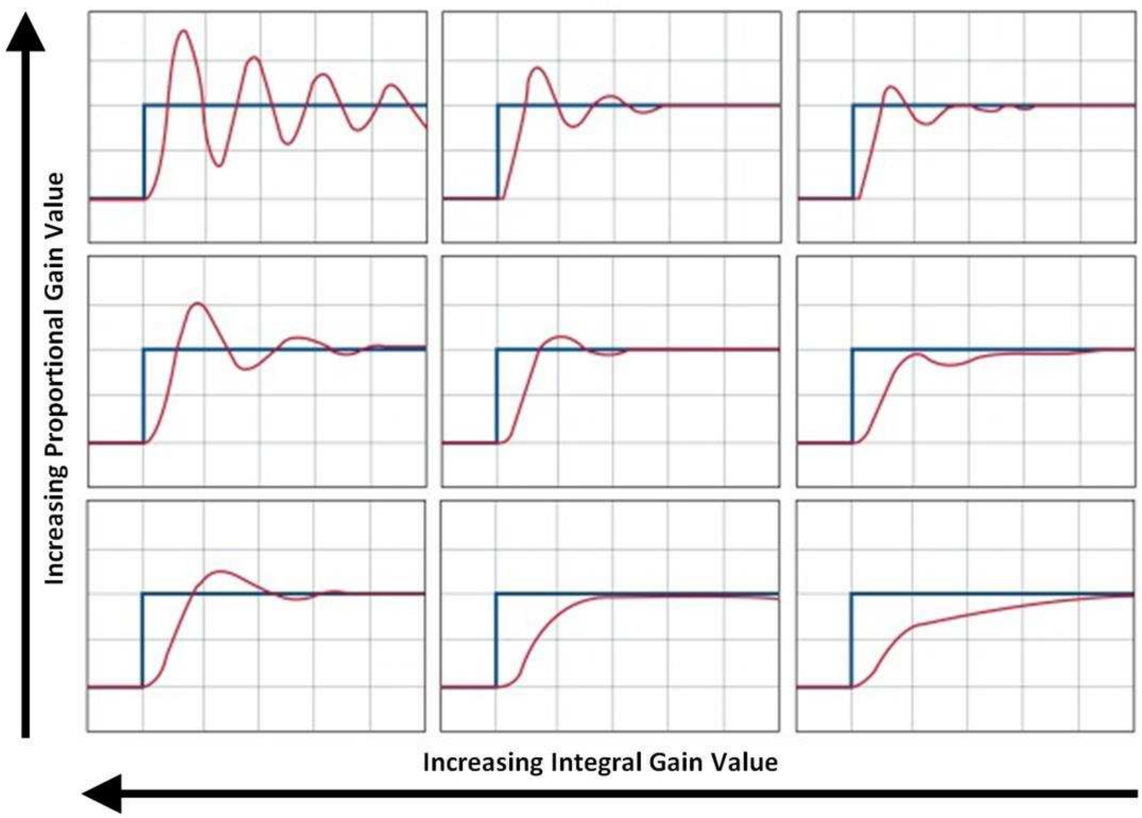



**Figure 15. Effects of Tuning PI Controller Gains**

## *3.2 Testing and Results*

### 3.2.1 Test Setup

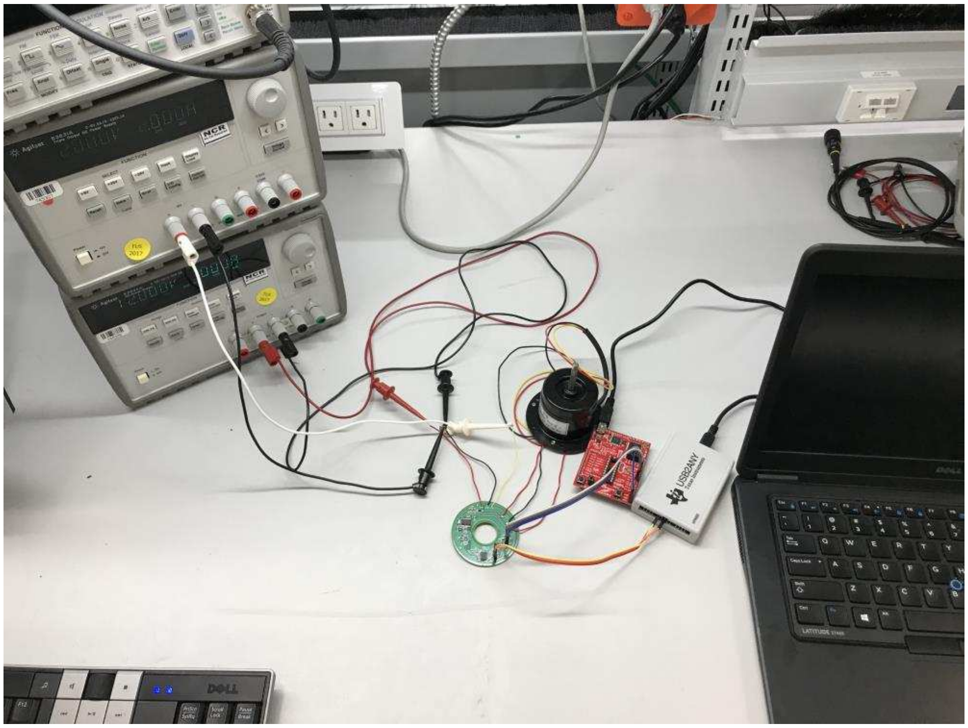


**Figure 16. Example Bench Setup**

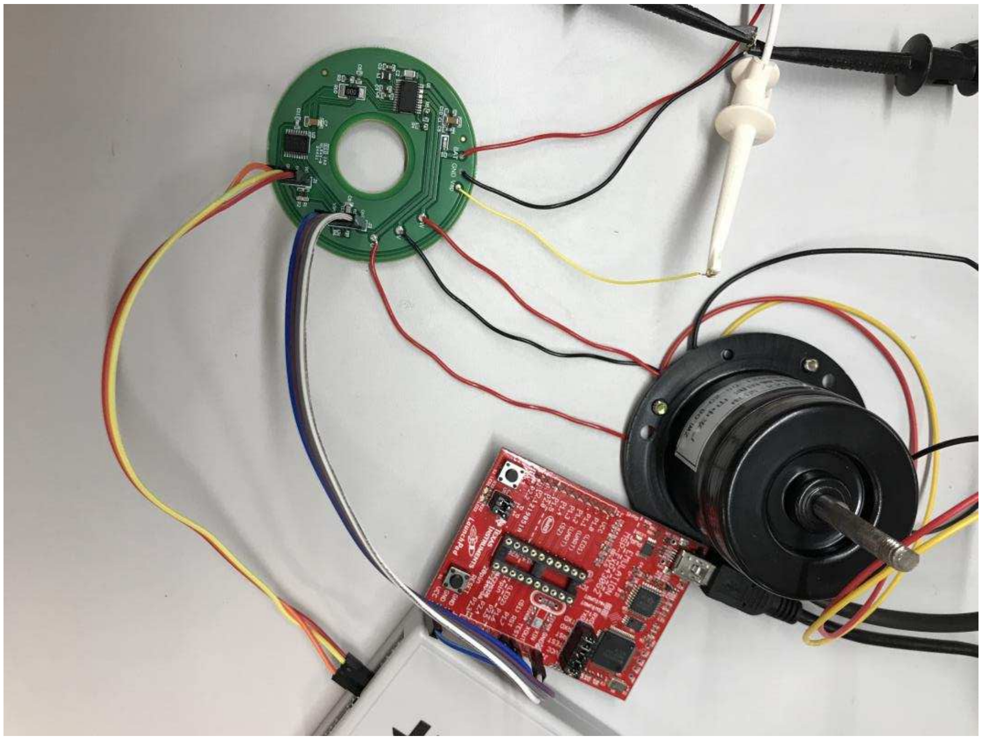


**Figure 17. Closeup of Connections**

### 3.2.2 Functional Test

The functional test refers to basic programming and start up using both the DRV10xx Software GUI through I²C and the MSP430 MCU using CCS and JTAG. The MSP430 MCU specifically is tested with a low- and high-voltage start-up of 8 V and 16 V, respectively.

C1 shows $V_{BAT}$, C2 shows the Frequency Generate (FG) pin on the DRV10983-Q1, and C3 shows the current of the U phase of the motor.

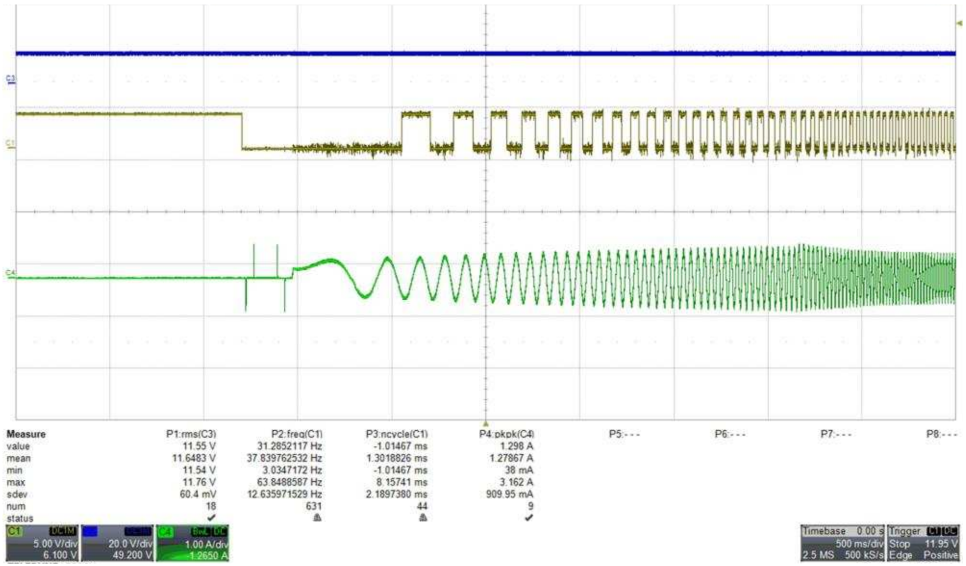


**Figure 18. Startup Profile, GUI Control, 12 V, 50% Speed**

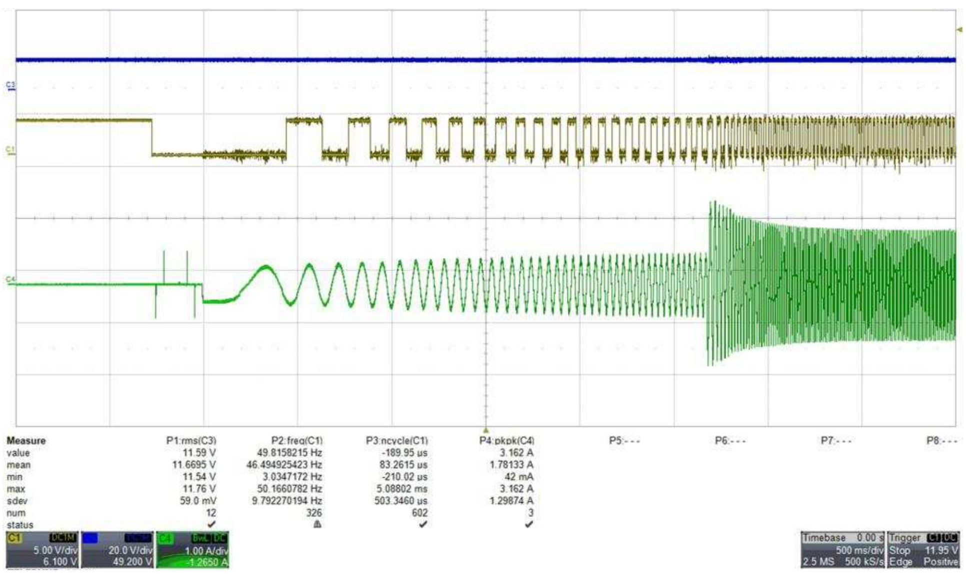


**Figure 19. Startup Profile, GUI Control, 12 V, 100% Speed**

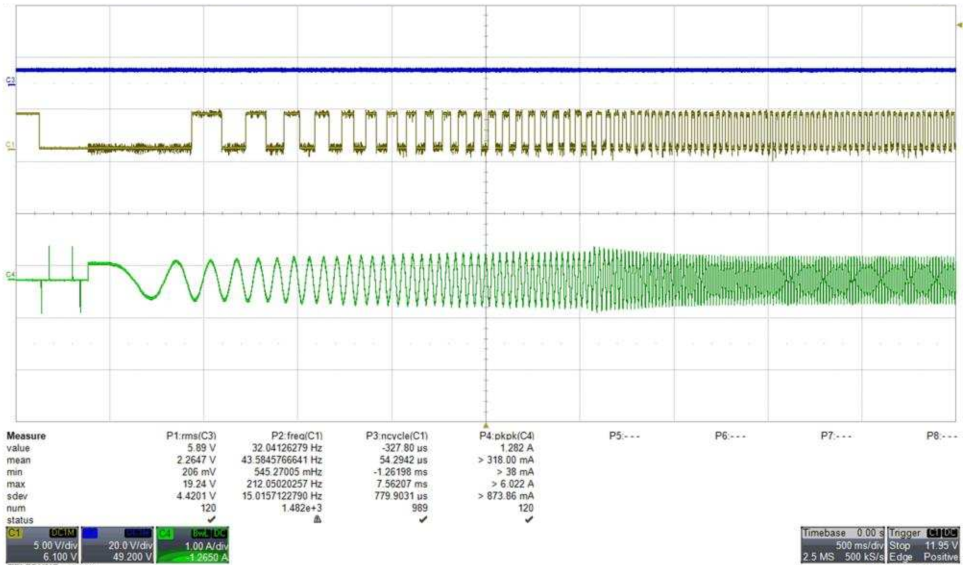



**Figure 20. Startup Profile, MSP430 Speed Control, 8 V, 100% Speed**

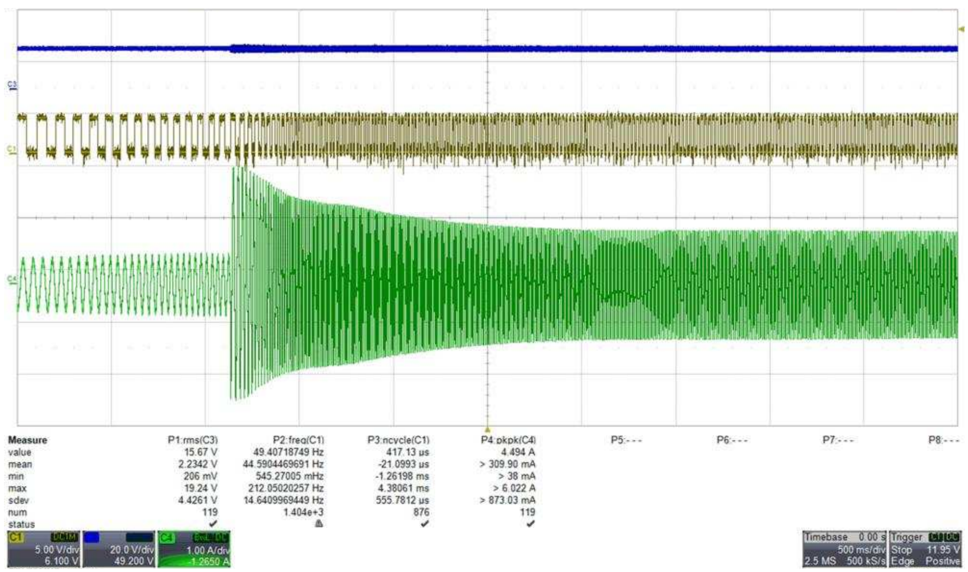



**Figure 21. Startup Profile, MSP430 Control, 16 V, 100% Speed**

### 3.2.3 Speed Loop Regulation Test

Five motor specific speed levels are programmed. Each speed level is tested for voltage ranging from 8 to 16 V in 2-V increments. Measurements are taken in Hz but converted to RPM. For more information, see Table 6 and Figure 22.

**Table 6. Output Speeds versus Input Speed Commands and Input Voltage Ranges**

| VOLTAGE INPUT | SPEED INPUT = 1000 RPM | SPEED INPUT = 1692 RPM | SPEED INPUT = 2236 RPM | SPEED INPUT = 2636 RPM | SPEED INPUT = 2956 RPM |
|---:|---:|---:|---:|---:|---:|
| 8 V | 1001 | 1691 | 2041 | 2210 | 2314 |
| 10 V | 1000 | 1691 | 2236 | 2559 | 2686 |
| 12 V | 1000 | 1691 | 2236 | 2636 | 2956 |
| 14 V | 1000 | 1691 | 2236 | 2640 | 2956 |
| 16 V | 1001 | 1691 | 2236 | 2641 | 2956 |

Figure 22 compares the output speeds versus input speed commands and input voltage ranges.

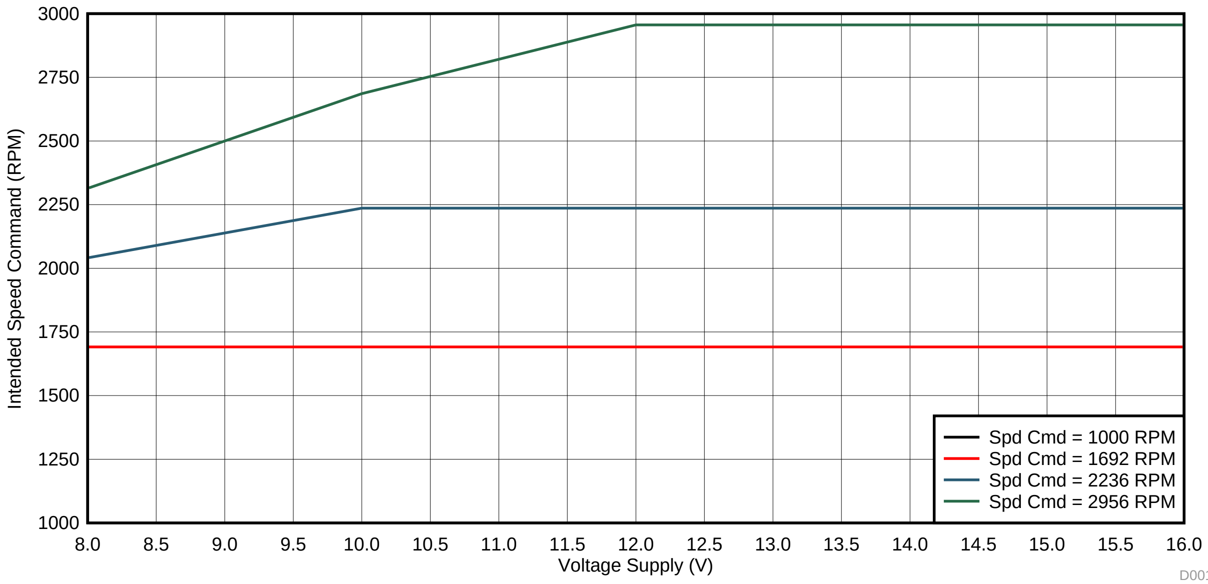



**Figure 22. Speed Regulation: Speed (RPM) versus Voltage Input**

### 3.2.4 Load Dump Test

The recommended operation voltage of the DRV10983-Q1 device is from 6.2 V to 28 V. The device is able to drive the motor within this $V_{CC}$ range. In the load dump condition, $V_{CC}$ can rise up to 45 V. Once the DRV10983-Q1 device detects that $V_{CC}$ is higher than VOV_R3 , it stops driving the motor and protects its own circuitry. Find more information in Table 7.

Load dump protection is effectively enabled by the device when the 45-V pulse is injected from the supply. The device shuts down to protect its circuitry and then turns back on again.

C1 shows the voltage on the Frequency Generator (FG) pin of the DRV10983-Q1, C2 shows the voltage on the U phase of the motor, C3 shows the current on the U phase of the motor, and C4 shows the voltage of $V_{BAT}$.

**Table 7. Load Dump Protection Specifications**

| LOAD DUMP PROTECTION | DESCRIPTION | MIN | TYP | MAX | UNIT |
|---|---|---|---|---|---|
| $V_{OV\_R}$ | Load dump protection mode entry on rising $V_{CC}$ threshold | 28.5 | 29.2 | 30 | V |
| $V_{OV\_F}$ | Load dump protection mode entry on falling $V_{CC}$ threshold | 27.7 | 28.2 | 28.8 | V |
| $V_{OV\_HYS}$ | Load dump protection mode hysteresis | 0.73 | 1 | 1.1 | V |

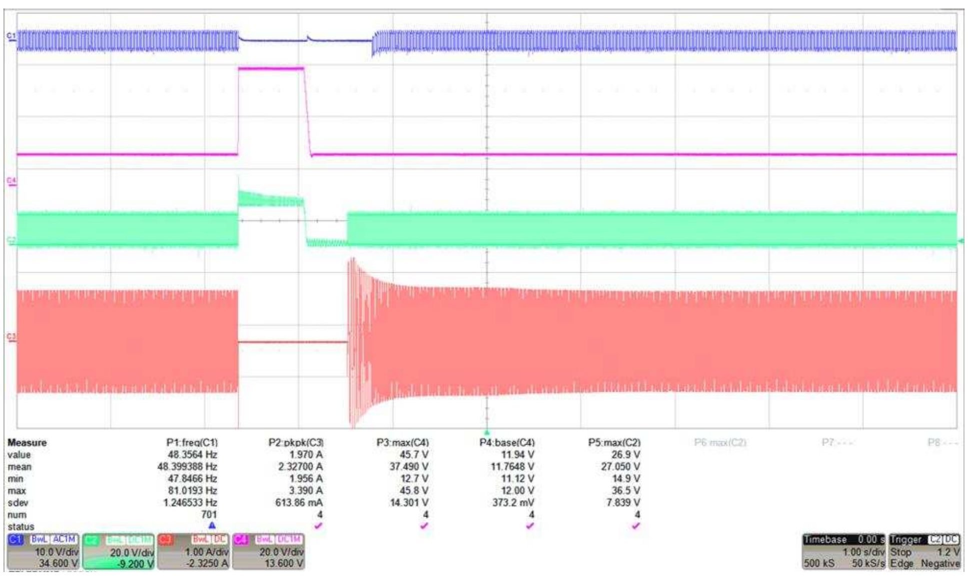



**Figure 23. Load Dump Protection Waveforms After Injecting 45-V Pulses**

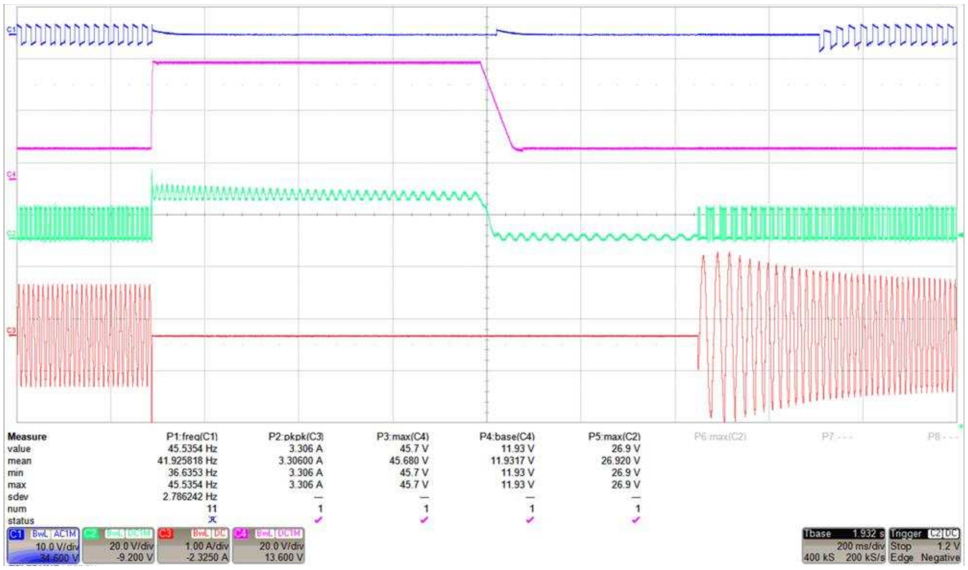



**Figure 24. Load Dump Protection Zoomed Waveforms After Injecting 45-V Pulses**

The load dump voltage signal is created using the sequence outlined in Table 8.

**Table 8. Sequence for Creating Load Dump Signal**

| SEQUENCE | DURATION (s) | SLEW RATE (V/ms) | VOLTAGE (V) |
|---|---|---|---|
| 1 | 5.5 | 0.001 | 12 |
| 2 | 0.3 | 10 | 45 |
| 3 | 0.4 | 0.001 | 45 |
| 4 | 0.3 | 10 | 12 |
| 5 | 5.5 | 0.001 | 12 |

### 3.2.5 Thermal Test

Thermal testing is conducted for a range of temperatures. The device operated and spun the motor at all temperatures. The device operated on average 20°C higher than the ambient temperature, which remained consistent in all the results.

**NOTE:** The numbers shown in the following images are in degrees Celsius.

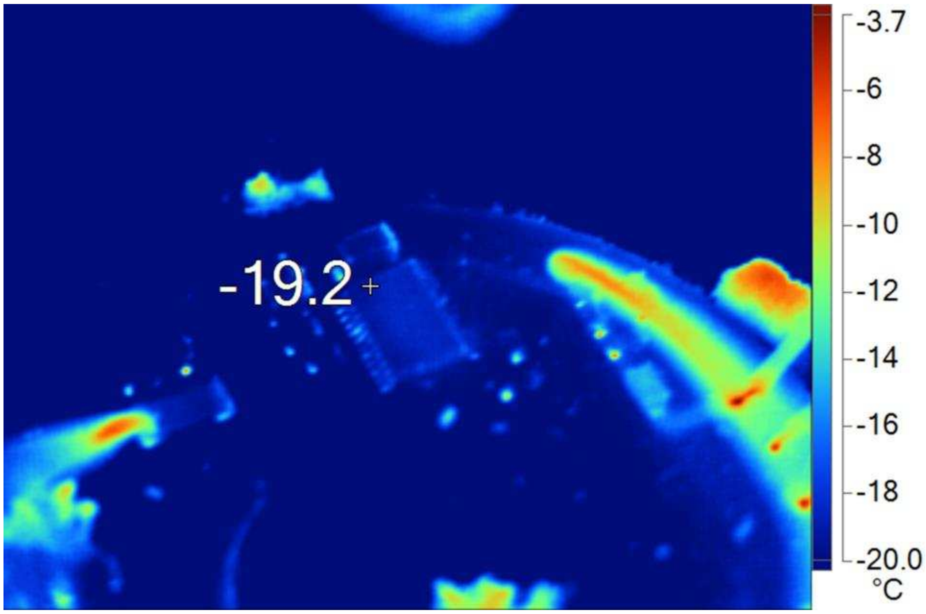



**Figure 25. Thermal Image After 3 Hours at 12 W, –40°C Ambient**

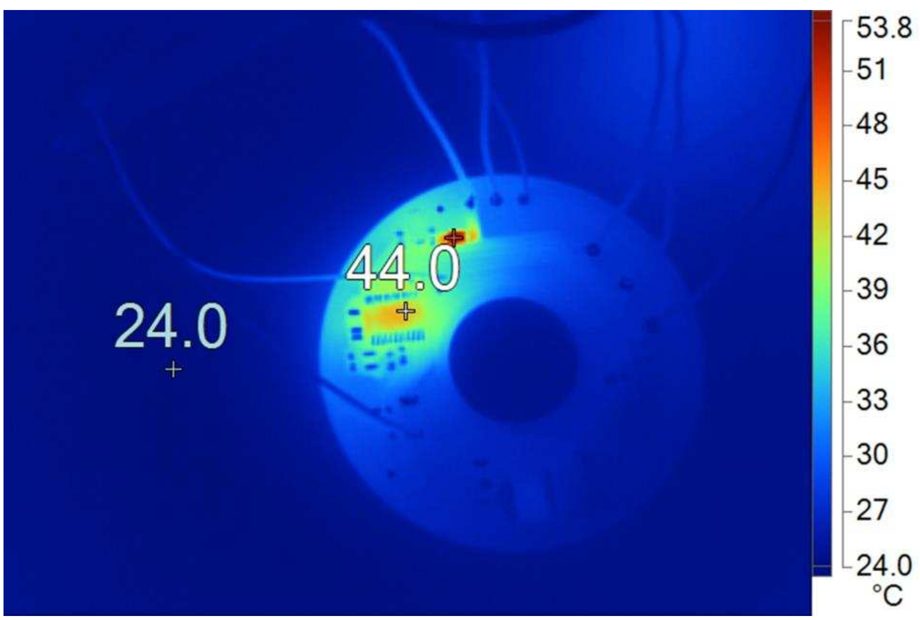



**Figure 26. Thermal Image After 15 Hours at 12 W, –40°C Ambient**

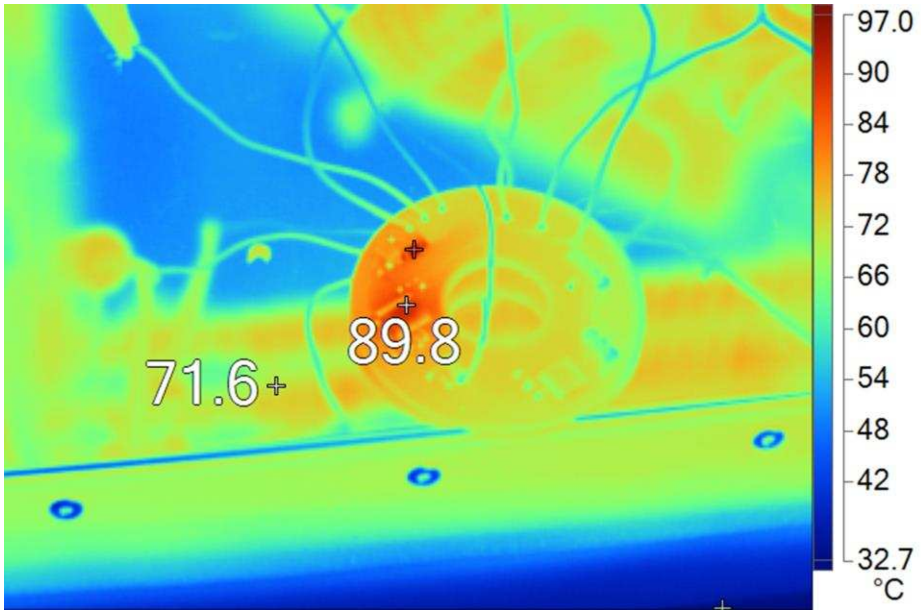



**Figure 27. Thermal Image After 15 Hours at 12 W, 70°C Ambient**

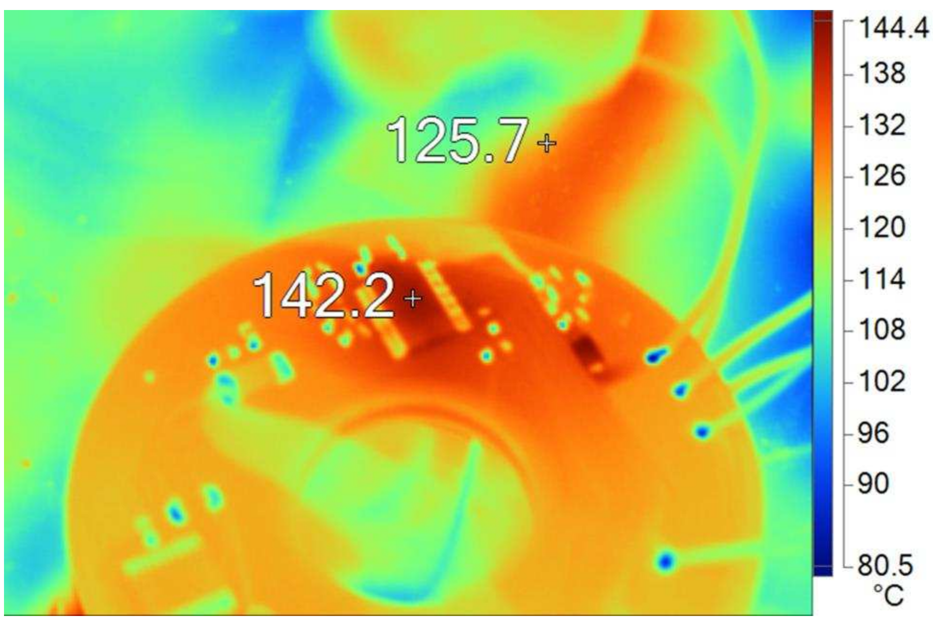



**Figure 28. Thermal Image After 15 Hours at 12 W, 125°C Ambient**

# 4 Design Files

## 4.1 Schematics

To download the schematics, see the design files at TIDA-01496.

## 4.2 Bill of Materials

To download the bill of materials (BOM), see the design files at TIDA-01496.

## 4.3 PCB Layout Recommendations

- Place the VCC, GND, U, V, and W pins with thick traces because high current passes through these traces.
- Place the 10-µF capacitor between VCC and GND, and as close to the VCC and GND pins as possible.
- Place the capacitor between CPP and CPN, and as close to the CPP and CPN pins as possible.
- Connect the GND, PGND, and SWGND under the thermal pad.
- Keep the thermal pad connection as large as possible, on both the bottom side and top sides. The connection must be one piece of copper without any gaps.
- Single-layer boards often suffer in optimal thermal and GND loop designs. Layout can be supplemented using 0-Ω resistors to connect isolated GND planes.

### 4.3.1 Layout Prints

To download the layer plots, see the design files at TIDA-01496.

## 4.4 Altium Project

To download the Altium project files, see the design files at TIDA-01496.

## 4.5 Gerber Files

To download the Gerber files, see the design files at TIDA-01496.

## 4.6 Assembly Drawings

To download the assembly drawings, see the design files at TIDA-01496.

# 5 Software Files

To download the software files, see the design files at TIDA-01496.

**NOTE:** The software files of this reference design require a TI login to access.

# 6 Related Documentation

1. Texas Instruments, *DRV10983-Q1 Tuning Guide*
2. Texas Instruments, *DRV10983-Q1 Evaluation Module User's Guide*

## 6.1 Trademarks

E2E, MSP430, LaunchPad, Code Composer Studio are trademarks of Texas Instruments.
All other trademarks are the property of their respective owners.

## 7 Terminology

**BLDC —** Brushless DC motor

**ESD—** Electrostatic discharge

**FETs, MOSETs—** Metal-oxide-semiconductor field-effect transistor

**IGBT—** Insulated gate bipolar transistor

**IPD—** Initial position detection

**OCP—** Overcurrent protection

**OTP—** Overtemperature protection

**PWM—** Pulse width modulation

**RPM—** Revolutions per minute

**RMS—** Root mean square

**UVLO—** Undervoltage lockout

## 8 About the Authors

**COLE MACIAS** is an applications engineer at Texas Instruments, where he is responsible for supporting customers and customer issues for the DRV10x family of products. These include 3-phase BLDC motor drive solutions in a wide range of applications. Cole obtained his BSEE at Arizona State University.

**MOSTAFA SHUBBAR** is a product marketing engineer at Texas Instruments currently working in the Integrated Motor Controller group. He completed his BSEE at the University of Texas at Dallas and is currently pursuing an MSEE at UTD.

# IMPORTANT NOTICE FOR TI DESIGN INFORMATION AND RESOURCES

Texas Instruments Incorporated ('TI") technical, application or other design advice, services or information, including, but not limited to, reference designs and materials relating to evaluation modules, (collectively, "TI Resources") are intended to assist designers who are developing applications that incorporate TI products; by downloading, accessing or using any particular TI Resource in any way, you (individually or, if you are acting on behalf of a company, your company) agree to use it solely for this purpose and subject to the terms of this Notice.

TI's provision of TI Resources does not expand or otherwise alter TI's applicable published warranties or warranty disclaimers for TI products, and no additional obligations or liabilities arise from TI providing such TI Resources. TI reserves the right to make corrections, enhancements, improvements and other changes to its TI Resources.

You understand and agree that you remain responsible for using your independent analysis, evaluation and judgment in designing your applications and that you have full and exclusive responsibility to assure the safety of your applications and compliance of your applications (and of all TI products used in or for your applications) with all applicable regulations, laws and other applicable requirements. You represent that, with respect to your applications, you have all the necessary expertise to create and implement safeguards that (1) anticipate dangerous consequences of failures, (2) monitor failures and their consequences, and (3) lessen the likelihood of failures that might cause harm and take appropriate actions. You agree that prior to using or distributing any applications that include TI products, you will thoroughly test such applications and the functionality of such TI products as used in such applications. TI has not conducted any testing other than that specifically described in the published documentation for a particular TI Resource.

You are authorized to use, copy and modify any individual TI Resource only in connection with the development of applications that include the TI product(s) identified in such TI Resource. NO OTHER LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE TO ANY OTHER TI INTELLECTUAL PROPERTY RIGHT, AND NO LICENSE TO ANY TECHNOLOGY OR INTELLECTUAL PROPERTY RIGHT OF TI OR ANY THIRD PARTY IS GRANTED HEREIN, including but not limited to any patent right, copyright, mask work right, or other intellectual property right relating to any combination, machine, or process in which TI products or services are used. Information regarding or referencing third-party products or services does not constitute a license to use such products or services, or a warranty or endorsement thereof. Use of TI Resources may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

TI RESOURCES ARE PROVIDED "AS IS" AND WITH ALL FAULTS. TI DISCLAIMS ALL OTHER WARRANTIES OR REPRESENTATIONS, EXPRESS OR IMPLIED, REGARDING TI RESOURCES OR USE THEREOF, INCLUDING BUT NOT LIMITED TO ACCURACY OR COMPLETENESS, TITLE, ANY EPIDEMIC FAILURE WARRANTY AND ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, AND NON-INFRINGEMENT OF ANY THIRD PARTY INTELLECTUAL PROPERTY RIGHTS.

TI SHALL NOT BE LIABLE FOR AND SHALL NOT DEFEND OR INDEMNIFY YOU AGAINST ANY CLAIM, INCLUDING BUT NOT LIMITED TO ANY INFRINGEMENT CLAIM THAT RELATES TO OR IS BASED ON ANY COMBINATION OF PRODUCTS EVEN IF DESCRIBED IN TI RESOURCES OR OTHERWISE. IN NO EVENT SHALL TI BE LIABLE FOR ANY ACTUAL, DIRECT, SPECIAL, COLLATERAL, INDIRECT, PUNITIVE, INCIDENTAL, CONSEQUENTIAL OR EXEMPLARY DAMAGES IN CONNECTION WITH OR ARISING OUT OF TI RESOURCES OR USE THEREOF, AND REGARDLESS OF WHETHER TI HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

You agree to fully indemnify TI and its representatives against any damages, costs, losses, and/or liabilities arising out of your non-compliance with the terms and provisions of this Notice.

This Notice applies to TI Resources. Additional terms apply to the use and purchase of certain types of materials, TI products and services. These include; without limitation, TI's standard terms for semiconductor products http://www.ti.com/sc/docs/stdterms.htm), evaluation modules, and samples (http://www.ti.com/sc/docs/sampterms.htm).