Programming Guide for the DRV10983

ABSTRACT

This guide assists the user in programming the DRV10983 whether through the TI programming socket or a custom solution. The following procedures aid in creating a system for programming prototype or production devices. Figure 1 shows the general block diagram for the DRV10983. Standard I²C is used to communicate with the device to read and write values to the registers.

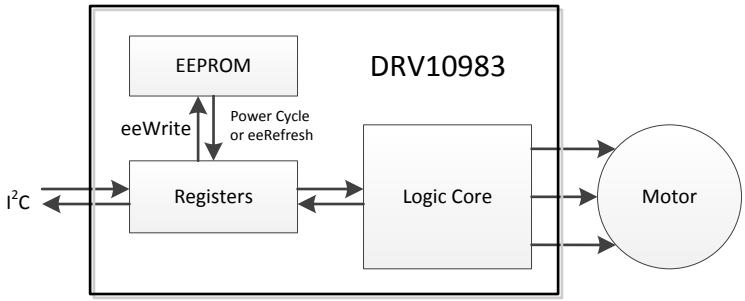

Figure 1. Basic Block Diagram of DRV10983

Contents

List of Figures

1 Introduction

To program this device, the user should first load the register values. When the registers have the desired values in them, setting the eeWrite bit writes them into EEPROM. For this process, use any controller or processor communicating with the DRV10983. This guide uses an external microcontroller (MCU), but the user could use a MCU already soldered onto the final product or a bulk programming tool connected to many ICs at the same time.

Figure 2 shows the general process for configuring these parts for their final application. It is important to have the correct register values to configure the DRV10983 to drive the specific motor in the application. If these values are not known, refer to the tuning guide (SLOU395) to find the optimized register settings.

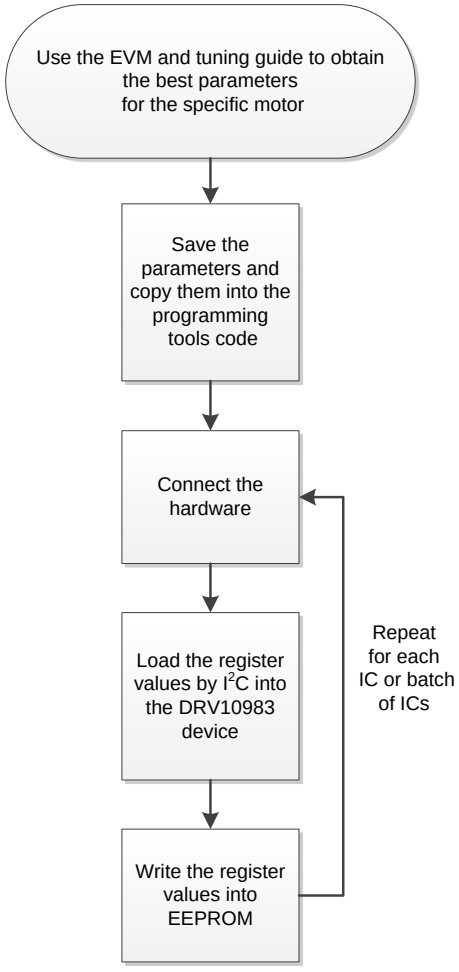

Figure 2. Flow Chart of the Programming Process

Code Composer Studio is a trademark of Texas Instruments.

Figure 3 shows the devices used when creating this guide. This setup is referenced throughout the document as an example. A PC is used to load the firmware onto a MSP430G2553 LaunchPad through USB. After the firmware is loaded, the PC is only used as a USB power supply. Any 5-V supply works to power the LaunchPad. The provided firmware configures the MSP430 to program a DRV10983 IC through I^2C communication. Each DRV10983 is easily connected to the necessary LaunchPad pins by the programming socket tool.

Find the programming socket tool, MSP430G2553 LaunchPad, DRV10983 ICs, and the Code Composer Studio™ (CCS) software project all on www.ti.com. Note that the programming socket tool does not come with any ICs. Order samples from www.ti.com for the DRV10983.

If using a MSP430 LaunchPad, ensure it is setup properly. For details, refer to the user's guide for the LaunchPad (SLAU318). Verify all jumpers are configured as shown in Figure 3. On J3, all jumpers are in place. On J5, only the jumper for P1.0 is in place. Remove the jumper for P1.6.

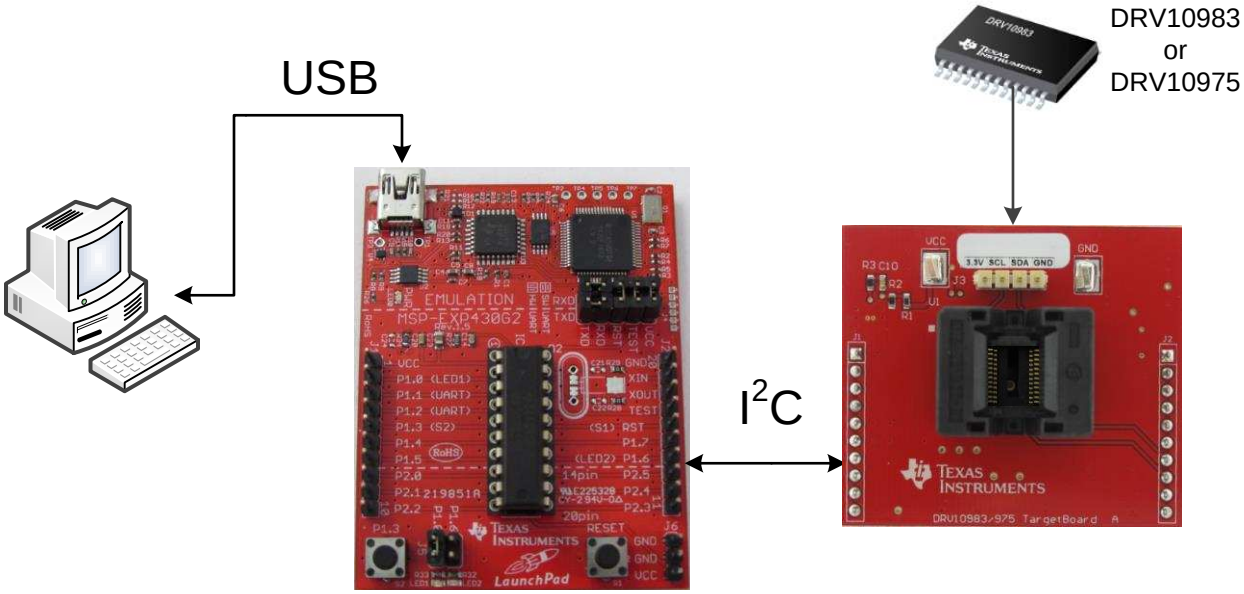

Figure 3. Communication Example With PC, MSP430, and Programming Socket

2 Using I²C to Communicate With the DRV10983

The DRV10983 uses the standard I²C protocol and is the slave device in the communicating pair. Its address is 1010 010. When a master addresses the part, it sends 8 bits, the address and one bit specifying a write (0) or read (1). The master sends 1010 0100 (0xA4) for write and 1010 0101 (0xA5) for read.

2.1 Writing to a Register

Send the following basic command chains (see Figure 4 and Figure 5) from the master to the slave to program the register settings. The first command enables the Sidata bit. If this bit is not enabled, the register values cannot change. Set the Sidata bit to '1' one time before setting all of the register values.

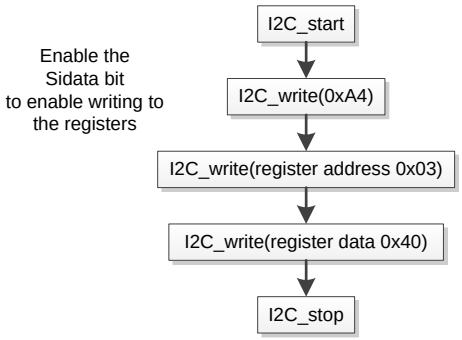

Figure 4. I²C Command Chain for Enabling Sidata Bit

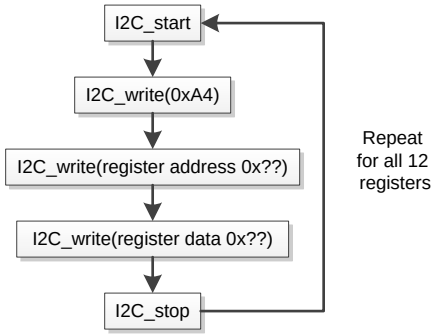

Figure 5. I²C Command Chain for Setting a Register

Figure 6. I²C Waveform of Writing to a Register

Figure 6 shows the waveforms for writing the value 0x39 to register 0x20.

Repeat those steps as many times as needed until all 12 registers (0x20 through 0x2B) are loaded with the correct values. For specific information on the registers, see the data sheet (SLVSCP6). The logic core runs the motor based on the register values so it is not necessary to write the register values to EEPROM. However, writing the values to EEPROM saves the parameters and auto loads them after a power cycle.

2.2 Writing the Registers to EEPROM

After the registers are loaded, the next step is to write them to EEPROM so the device can maintain those settings. Figure 7 describes the command chain for writing to EEPROM. First, enter the program key (0xB6) in the device control register (0x02), and then immediately after, set the eeWrite bit to 1 (0x50) in the EEPROM control register (0x03). If the eeWrite bit is not set directly after the program key is entered, the program key will be reset.

The programming time is about 24 ms, and when finished, the device clears the eeWrite bit. After the data is stored in EEPROM, the device can be powered down, and upon power-up, it auto loads the values into the registers. For the device to properly write to EEPROM, the V_{CC} must be at least 22 V and have at least a 24-ms delay before power cycling or refreshing the device.

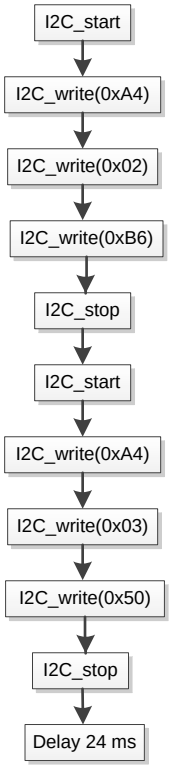

Figure 7. I²C Command Chain for Writing to EEPROM

2.3 Reading the Registers

Reading the registers is a way to verify the values saved in EEPROM or to obtain feedback from the DRV10983. Because the EEPROM cannot be read directly, the settings saved in EEPROM can be loaded into the registers by two methods. Power cycling the device auto loads the registers from EEPROM upon startup. Alternatively, the registers can be refreshed with the EEPROM values while still powered.

The device has a bit, 'eeRefresh', which loads the registers with the values in EEPROM. Setting 'eeRefresh' has the same effect as power cycling the device. After the bit is set to 1, the registers are loaded with the values stored in EEPROM, then the device clears the bit. Figure 8 shows the commands for setting eeRefresh.

To read data from the registers, the user must make a few changes from the writing process. Most importantly, the master must tell the slave to send information (see the command chain shown in Figure 9). Note that the user does not need to set the eeRefresh bit to read the registers; it is only for setting the registers back to the EEPROM values.

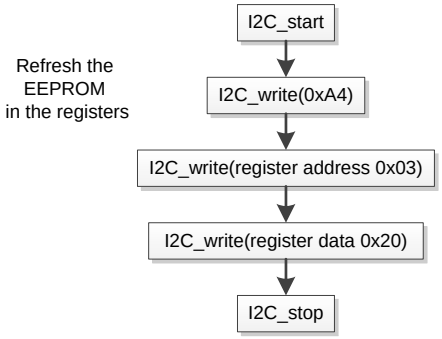

Figure 8. I²C Command Chain for Refreshing the Registers

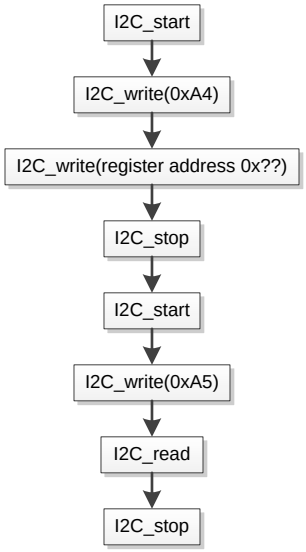

Figure 9. I²C Command Chain for Reading a Register

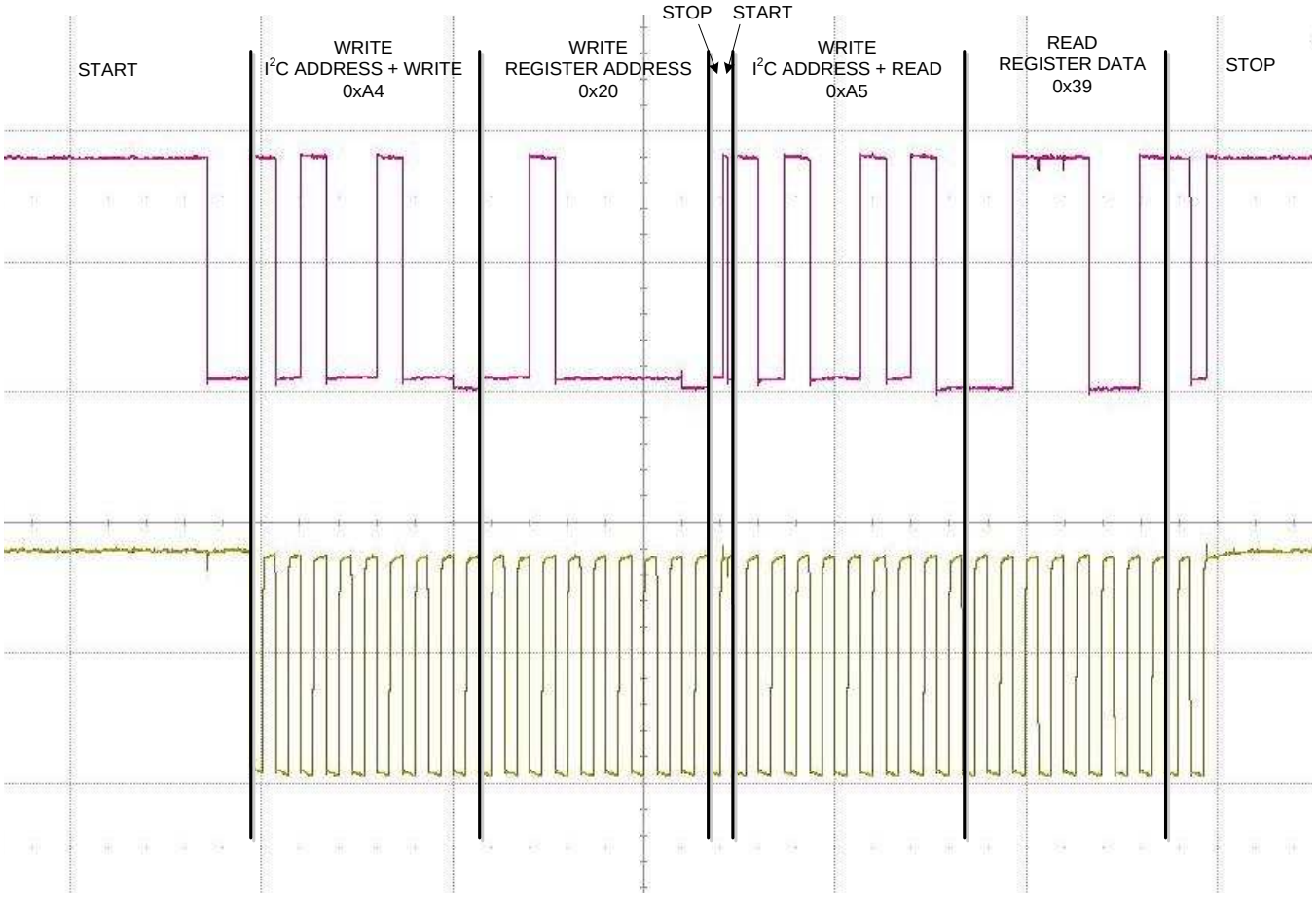

Figure 10. I²C Waveform of Reading a Register

Figure 10 shows the SDA and SCL lines during a read command. The register 0x20 is read and shows a value of 0x39.

3 Using the Hardware and Software

Using the program socket tool sold by Texas Instruments is a great way to program a small quantity of devices. This section gives the details needed to operate the program tool, but also provides details about programming the DRV10983 when using a different method for configuring the parts. Note that the socket board does not include any DRV10983 ICs. These ICs must be ordered separately.

3.1 Download the Software for the LaunchPad

To run the provided programming example (SLOC316), the user's PC must have CCS, IAR, or an IDE that works with the MSP430 LaunchPad. The project developed for download was used in CCS, but also tested in IAR.

The code used to program the DRV10983 with the MSP430G2553 LaunchPad and programming socket board can be downloaded from the product page for the DRV10983 on www.ti.com. After the download completes, extract the files.

If using a custom solution to program the devices, it is still useful to download the project. After the project is open, it is easy to see the process of programming the device by reading through the .c file of the project. Note that the code provided uses a software version of the I^2C interface. This simplifies the code and allows for easy modifications across multiple platforms. Most MCUs also include a hardware I^2C peripheral, which is another way to program and communicate with the DRV10983.

3.2 Hooking Up the Hardware

The DRV10983 devices are programmed by I^2C. If using the programming socket board with the LaunchPad, the socket board conveniently fits on the LaunchPad and connects the needed pins. The DRV10983 IC should be inserted into the socket tool while the setup is not powered. If using a custom solution, ensure that there is an I^2C connection between the DRV10983 and the MCU used to communicate as the master. The hardware files for the programming socket (SLAR101) can be downloaded from www.ti.com.

On the example setup in Figure 11, the last item to connect is a USB cord from the PC to the LaunchPad. For a custom hardware configuration, connect the MCU to the PC or testing equipment used to load the programming process and register values into the MCU. The USB connection is only for the initial programming of the MSP430 LaunchPad. Afterward, the USB is only used for power.

For each device (DRV10983), the V_{CC} needs to be at least 22 V for successful programming. The programming socket board uses a boost converter to boost the MSP430s from 3.3 to 23 V. Verify that the solution used can provide a V_{CC} of at least 22 V for the programming to work.

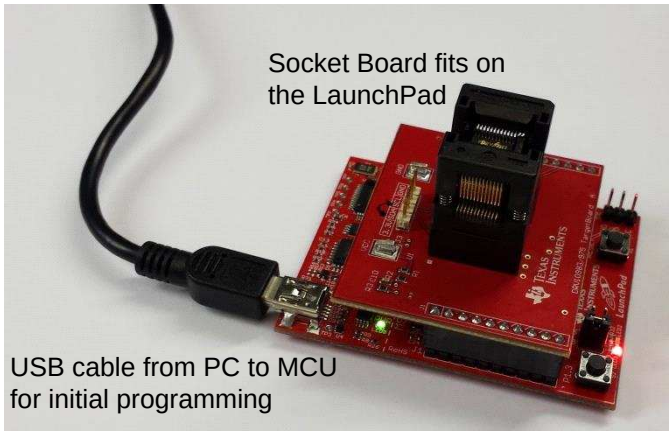

Figure 11. Hardware Setup of the MSP430 and Programming Socket

3.3 Modifying the Software for Custom Register Values

Import the project downloaded into CCS through the project dropdown menu or the resource explorer welcome page.

The software was written with register values for a specific motor that may or may not work for the end application. With that in mind, it is best to obtain the proper settings for the application and load those into the programming software. This process takes three steps.

1. Find the proper settings. The best way to get these settings is to follow the user's guide (SLOU393) and tuning guide (SLOU395) for the DRV10983.
2. After the system is tuned, save the parameters using the save feature of the GUI. When saving the file, it does not matter what the name is, but remember where it is saved. Locate the parameters file (that was just saved) and open it. The file should contain 12 rows with register data for the device. Figure 12 shows the file saved from the GUI on the left.

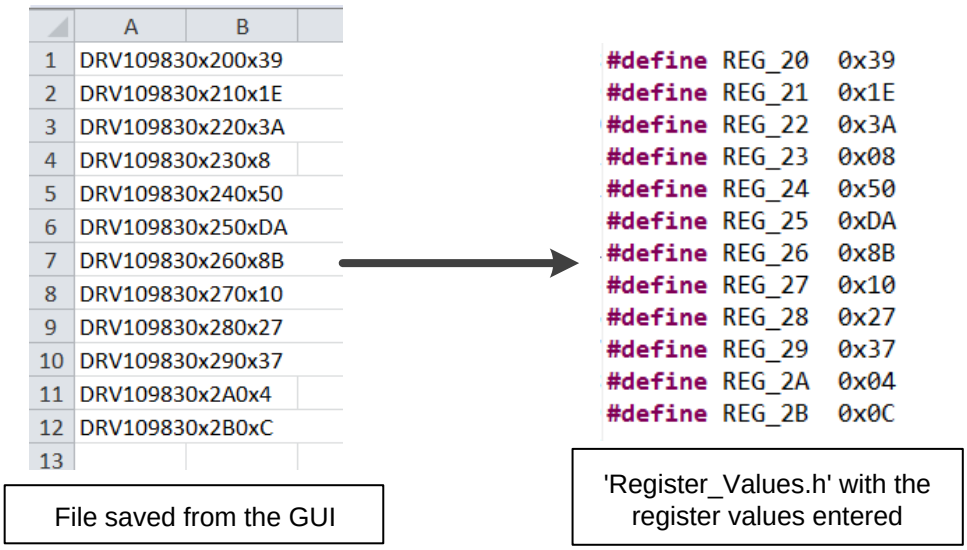

Figure 12. Example Register Settings Entered into the Programming Tools Code

3. Define the register values in the programming code to match the tuned values from the GUI. The downloaded project has the registers defined in 'Register_Values.h', but these need to be changed to the tuned values from the GUI. As shown in Figure 12, the defined register values in the code (right side) need to match the registers in the file saved from the GUI (left side). To open 'Register_Values.h', expand the project in the project explorer window and double click 'Register_Values.h'. Manually enter each register value from the GUI file into the program code. Make sure to save 'Register_Values.h' after the new register values are entered.

3.4 Running the Project

With the register values entered into the code, the next step is to load it into the MSP430 LaunchPad. After the downloaded project is imported into the workspace in CCS, the project should reference all of the necessary header files from the path variables. Build, load, and run the code. For help running a project in CCS, refer to the web resources link in the welcome menu on the resource explorer page. Find this by clicking on the TI Resource Explorer under the View dropdown menu. If not using CCS, make sure all the necessary files are in the workspace for the project to compile correctly.

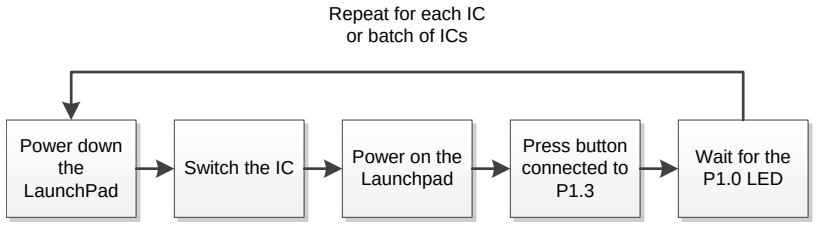

Figure 13. Flowchart Specific to Programming ICs With the LaunchPad and Socket

In the example code when the LaunchPad receives power, it waits to configure the DRV10983 until first the V_{CC} is up to at least 22 V, then the button is pushed. It is important to disconnect the power before removing the IC from the socket and wait until the new IC is in place before powering on the LaunchPad. One option is to use a switched USB power port to avoid needing to unplug and plug the USB cord repeatedly.

4 Different Methods of Programming

The example this guide refers to uses an external MCU to program the DRV10983 parts one at a time. Each DRV10983 is configured before it is soldered down. This setup was designed to be an easy way to create prototypes or to program a small amount of ICs. It is not practical to use this method for programming production parts. This section describes alternative methods for easily programming parts in production.

The IC can be easily configured during two portions of the manufacturing process. The first opportunity is before the DRV10983 is soldered down (pre-soldering). Alternatively, the part can be configured after it has been fixed on the board (post-soldering). Regardless of the configuration method used, it is important that the V_{CC} is at least 22 V to program the EEPROM.

4.1 Pre-Soldering

Programming the parts before soldering them down on the final board follows a process similar to the one described previously in this document. The IC is placed into a socket and programmed through an external connection. Some developers may choose to design their own programming solution based on the example provided or may work with a third party to develop a mass programming solution.

4.2 Post-Soldering

Figure 14 outlines two methods for programming the device after it is soldered down. The first method has a controller or processor in the circuit, which is already connected to the DRV10983. The pins used to configure that MCU or MPU for normal operation are the same pins that can be used for loading the configuration settings. This method is useful if the circuit has a controller.

If the application will not allow V_{CC} to reach 22 V to program the EEPROM, an alternate method can be used to configure the DRV10983. Every time the device is powered on, the registers can be setup for the application by the MCU. As long as the values in the registers are correct for the application, the device will run the motor regardless of the EEPROM settings. However, every time the device is powered down then powered up, the registers load from the EEPROM settings; this occurrence is the reason that in this configuration method, the registers must be reconfigured after every startup.

If the DRV10983 is a standalone device without a MCU in the circuit, the process is only slightly different. The user needs an external device to send the I^2C commands to the part. An example of this is using the EVM with the GUI. The USB2ANY is the external device that communicates through I^2C with the DRV10983.

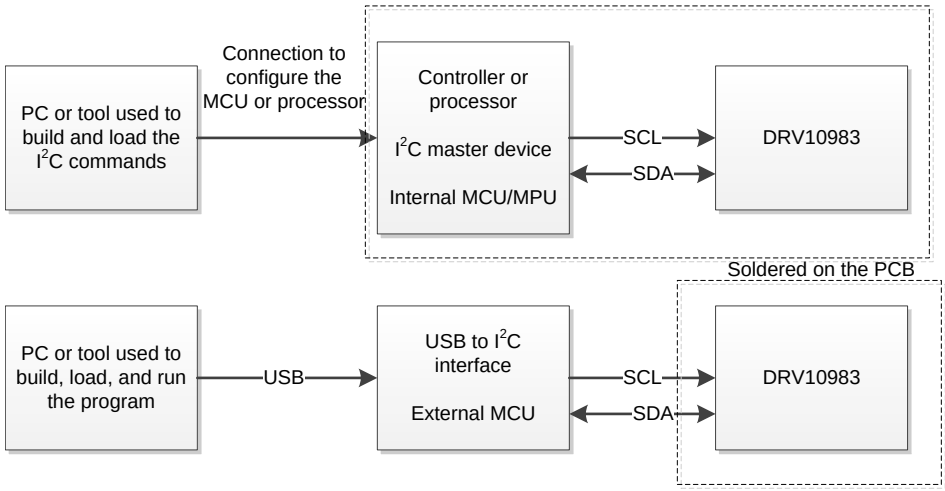

Figure 14. General Configuration Methods

The DRV10983 can be programmed as part of the PCB test even when a MCU is not used. To do this, program an in-circuit tester or bed-of-nails tester to send the I^2C commands to the IC. For this to work, it is essential that the circuit board is designed to provide 22 V and that the tester can access the SDA and SCL lines. Figure 15 is an example of a simple circuit that shows the SDA and SCL connection brought out and exposed so the pins on the testing equipment can make contact at those points. It is important to design the contact points to fit the tester that is used.

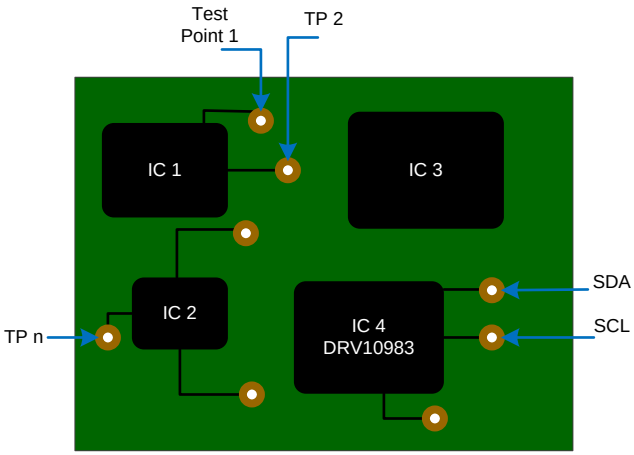

Figure 15. Example Circuit for Bed of Nails Tester

The testing tool should be able to provide the 22-V supply for writing to the EEPROM of the DRV10983. After V_{CC} is at least 22 V and the testing tool has connected with the SDA and SCL contact points, send the I^2C commands shown in *Section 2* to configure the device.

5 Summary

To setup the DRV10983 for its end application, write specific values to registers 20 through 2B that describe the motor and how to best operate it, then save those in EEPROM. This setup can be done using multiple methods, pre-soldering and post-soldering. Use the procedure outlined in this document as an easy way to understand how the configuration process is done and to program small quantities of devices. This guide also described methods to configure the ICs for production.

6 Schematic

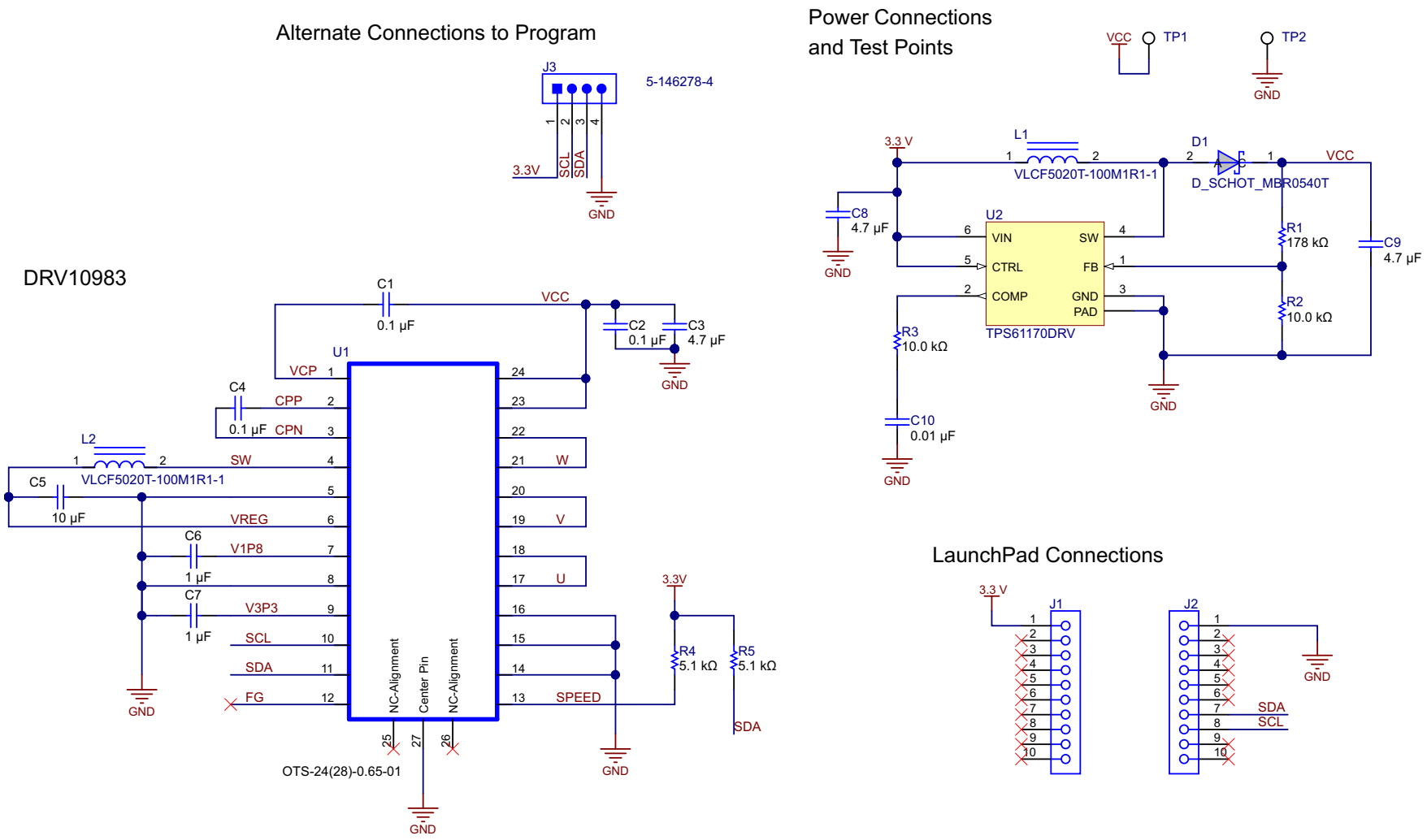

Figure 16. Schematic of the Programming Socket

7 Bill of Materials

Designator	Quantity	Value	Part Number	Manufacturer	Description	Package Reference
PCB	1		DRV10983/975 TargetBoard	Any	Printed Circuit Board	
C1, C2, C4	3	0.1uF	C1608X7R1H104K	TDK	CAP, CERM, 0.1uF, 50V, +/-10%, X7R, 0603	0603
C3	1	4.7uF	GRM32ER71H475KA88L	MuRata	CAP, CERM, 4.7uF, 50V, +/-10%, X7R, 1210	1210
C5	1	10uF	CGA5L3X5R1H106K160AB	TDK	CAP, CERM, 10uF, 50V, +/-10%, X5R, 1206_190	1206_190
C6, C7	2	1uF	GRM188R61H105KAALD	MuRata	CAP, CERM, 1uF, 50V, +/-10%, X5R, 0603	0603
C8	1	4.7uF	0603ZD475KAT2A	AVX	CAP, CERM, 4.7uF, 10V, +/-10%, X5R, 0603	0603
C9	1	4.7uF	C2012X5R1H475K125AB	TDK	CAP, CERM, 4.7uF, 50V, +/-10%, X5R, 0805	0805
C10	1	0.01uF	C1608X7R1H103K	TDK	CAP, CERM, 0.01uF, 50V, +/-10%, X7R, 0603	0603
D1	1	MBR0540T	MBR0540T	OnSemi	Diode, Schottky, 0.5A, 40V	SOD-123
J1, J2	2		CRD-081413-B-T	Major League Electronics	Connector, Receptacle, 100mil, 10x1, TH	10x1 Receptacle
J3	1		5-146278-4	TE Connectivity	Header, 100mil, 4x1, Tin, TH	Header, 4x1, 100mil, TH
L1, L2	2	10uH	VLCF5020T-100M1R1-1	TDK	Inductor, SMT, yyA, zzmilliohm	0.157 x 0.157 inch
R1	1	178k	RC0603FR-07178KL	Yageo America	RES, 178k ohm, 1%, 0.1W, 0603	0603
R2, R3	2	10.0k	CRCW060310K0FKEA	Vishay-Dale	RES, 10.0k ohm, 1%, 0.1W, 0603	0603
R4, R5	2	5.1k	CRCW06035K10JNEA	Vishay-Dale	RES, 5.1k ohm, 5%, 0.1W, 0603	0603
TP1, TP2	2	SMT	5016	Keystone	Test Point, Compact, SMT	Testpoint_Keystone_Compact
U1	1		OTS-24(28)-0.65-02	Enplas	24 pin socket with no center pin	HTTOP
U2	1		TPS61170DRV	Texas Instruments	1.2A High Voltage Boost Converter in 2x2mm QFN Package, DRV0006A	DRV0006A

IMPORTANT NOTICE

Texas Instruments Incorporated and its subsidiaries (TI) reserve the right to make corrections, enhancements, improvements and other changes to its semiconductor products and services per JESD46, latest issue, and to discontinue any product or service per JESD48, latest issue. Buyers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. All semiconductor products (also referred to herein as “components”) are sold subject to TI’s terms and conditions of sale supplied at the time of order acknowledgment.

TI warrants performance of its components to the specifications applicable at the time of sale, in accordance with the warranty in TI’s terms and conditions of sale of semiconductor products. Testing and other quality control techniques are used to the extent TI deems necessary to support this warranty. Except where mandated by applicable law, testing of all parameters of each component is not necessarily performed.

TI assumes no liability for applications assistance or the design of Buyers’ products. Buyers are responsible for their products and applications using TI components. To minimize the risks associated with Buyers’ products and applications, Buyers should provide adequate design and operating safeguards.

TI does not warrant or represent that any license, either express or implied, is granted under any patent right, copyright, mask work right, or other intellectual property right relating to any combination, machine, or process in which TI components or services are used. Information published by TI regarding third-party products or services does not constitute a license to use such products or services or a warranty or endorsement thereof. Use of such information 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.

Reproduction of significant portions of TI information in TI data books or data sheets is permissible only if reproduction is without alteration and is accompanied by all associated warranties, conditions, limitations, and notices. TI is not responsible or liable for such altered documentation. Information of third parties may be subject to additional restrictions.

Resale of TI components or services with statements different from or beyond the parameters stated by TI for that component or service voids all express and any implied warranties for the associated TI component or service and is an unfair and deceptive business practice. TI is not responsible or liable for any such statements.

Buyer acknowledges and agrees that it is solely responsible for compliance with all legal, regulatory and safety-related requirements concerning its products, and any use of TI components in its applications, notwithstanding any applications-related information or support that may be provided by TI. Buyer represents and agrees that it has all the necessary expertise to create and implement safeguards which anticipate dangerous consequences of failures, monitor failures and their consequences, lessen the likelihood of failures that might cause harm and take appropriate remedial actions. Buyer will fully indemnify TI and its representatives against any damages arising out of the use of any TI components in safety-critical applications.

In some cases, TI components may be promoted specifically to facilitate safety-related applications. With such components, TI’s goal is to help enable customers to design and create their own end-product solutions that meet applicable functional safety standards and requirements. Nonetheless, such components are subject to these terms.

No TI components are authorized for use in FDA Class III (or similar life-critical medical equipment) unless authorized officers of the parties have executed a special agreement specifically governing such use.

Only those TI components which TI has specifically designated as military grade or “enhanced plastic” are designed and intended for use in military/aerospace applications or environments. Buyer acknowledges and agrees that any military or aerospace use of TI components which have ***not*** been so designated is solely at the Buyer's risk, and that Buyer is solely responsible for compliance with all legal and regulatory requirements in connection with such use.

TI has specifically designated certain components as meeting ISO/TS16949 requirements, mainly for automotive use. In any case of use of non-designated products, TI will not be responsible for any failure to meet ISO/TS16949.

Products		**Applications**	
Audio	www.ti.com/audio	Automotive and Transportation	www.ti.com/automotive
Amplifiers	amplifier.ti.com	Communications and Telecom	www.ti.com/communications
Data Converters	dataconverter.ti.com	Computers and Peripherals	www.ti.com/computers
DLP® Products	www.dlp.com	Consumer Electronics	www.ti.com/consumer-apps
DSP	dsp.ti.com	Energy and Lighting	www.ti.com/energy
Clocks and Timers	www.ti.com/clocks	Industrial	www.ti.com/industrial
Interface	interface.ti.com	Medical	www.ti.com/medical
Logic	logic.ti.com	Security	www.ti.com/security
Power Mgmt	power.ti.com	Space, Avionics and Defense	www.ti.com/space-avionics-defense
Microcontrollers	microcontroller.ti.com	Video and Imaging	www.ti.com/video
RFID	www.ti-rfid.com		
OMAP Applications Processors	www.ti.com/omap	**TI E2E Community**	e2e.ti.com
Wireless Connectivity	www.ti.com/wirelessconnectivity		

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