

# Digital Control Cost-Optimized 10-A Battery Formation and Test Reference Design



## Description

This reference design provides a cost-effective design for multichannel battery formation and test applications. The design uses the C2000™ real-time microcontroller (MCU) with 16-channel simultaneous sampling precision ADC, which can be scaled up to support 8-channel cell testing. The design utilizes the HRPWM module for precise duty cycle control, to achieve the performance of a precision digital-to-analog converter, which saves more than 30% in the bill of materials. The MCU integrates 2p2z control for both constant-current (CC) and constant-voltage (CV) control operation. Multichannel ADC optimizes the flexibility of current and voltage loops in software. This design allows multiple current and voltage levels output with one approach.

## Features

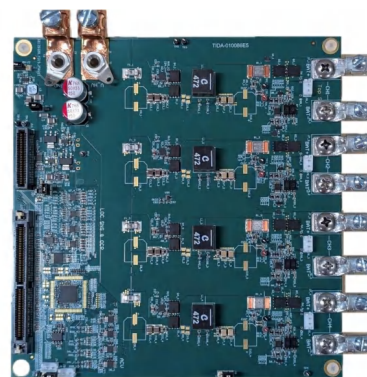
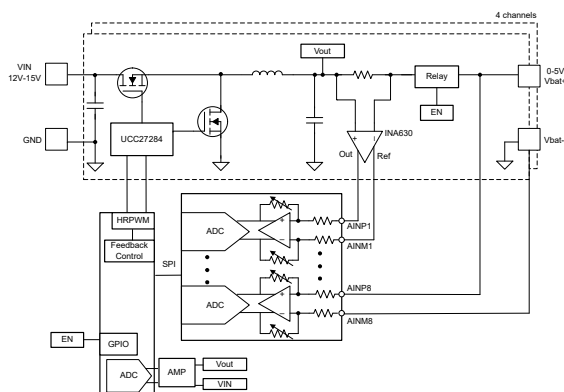
- Bidirectional current control up to 10 A with less than  $\pm 0.02\%$  accuracy
- 16-channel SAR ADC supporting  $\pm 5$  V true bipolar inputs and can be connected directly to battery cell
- Constant voltage mode is supported in both charging and discharging with regulation error  $< \pm 1$  mV
- Software Frequency Response Analyzer (SFRA) enables simple closed-loop transient response tuning

## Applications

- [Battery test](#)
- [Programmable DC power supply](#)

## Resources

<a href="#">TIDA-010086</a>	Design Folder
<a href="#">TMS320F28P65</a>	Product Folder
<a href="#">ADS9324</a> , <a href="#">INA630</a>	Product Folder
<a href="#">UCC27284</a> , <a href="#">REF50E</a>	Product Folder



# 1 System Description

## 1.1 Li-ion Cell Formation Equipment

The battery tester equipment includes a wide variety of equipment used to test single cells, battery modules, and high-voltage battery packs. Various tests need to be performed to validate the performance, capacity and safety of lithium-ion or other types of battery cells. Battery formation represents one of the critical steps in cell manufacturing. This process requires precision charge and discharge to a single cell to form solid electrolyte interface (SEI) layer. Cell grading and electrical testing evaluate the capacity and internal resistance of each cell. These tests require accurate current and voltage charge-discharge profiles and real-time data logging to capture detailed performance data.

Typical test equipment requires precision bidirectional power supplies and data acquisition systems to perform highly accurate current and voltage charge or discharge of a battery cell, often better than  $\pm 0.05\%$  of full scale. With growth in battery capacity and more integrated multifunction test systems, a multichannel power supply achieves a higher volume of test channels for battery cells.

Two approaches can be used to manage charge-discharge cycles and provide precise testing conditions. [Table 1-1](#) shows the difference between the two approaches:

**Table 1-1. Analog and Digital Control**

ASPECT	ANALOG CONTROL	DIGITAL CONTROL
<b>Control Logic</b>	Hardware-based feedback loops using analog components	Software-based algorithms running on DSP
<b>Flexibility</b>	Fixed configurability; changes require hardware modifications	Highly configurable through software; simple load compensation and supports multiple test profiles
<b>Complexity</b>	Simpler design for basic functions but grows complex for advanced features	More complex software development but simpler hardware for scalability
<b>Response Time</b>	Fast, continuous response due to analog feedback	Fast depends on microsecond latency, ADC and MCU speed
<b>Precision</b>	High precision and sensitive to component tolerances and drift	High precision with proper ADC resolution and calibration; less prone to drift
<b>Data Logging</b>	Limited; requires additional circuitry for data acquisition	Built-in data logging through MCU, enabling detailed analysis and traceability
<b>Cost</b>	Lower for simple systems; higher for complex, high-precision designs	Cost-effective for scalable, feature-rich systems

The TIDA-010086 reference design uses a digital control approach to create a multichannel, synchronous buck converter based on TMS320F28P650DK MCU and a 16-channel, integrated PGA, SAR ADC to achieve high accuracy, fast response, and high signal-chain density design.

## 1.2 Key System Specifications

**Table 1-2. Key System Specifications**

PARAMETER	SPECIFICATIONS
Low-voltage port – Battery port	50 mV to 5V
High-voltage port– Bus voltage	12 V to 15 V
Switching frequency	250kHz
Maximum DC current per channel, bidirectional	10 A
Current control accuracy	$\pm 0.02\%$ FSR
Voltage control accuracy	$\pm 0.02\%$ FSR
Current transient time	< 100 $\mu$ s
Power stage and mode of operation	Synchronous buck converter, CCM mode

## 2 System Overview

### 2.1 Block Diagram

Figure 2-1 shows a block diagram of the reference design. The TMS320F28P650DK MCU generates a high-resolution PWM for a synchronous buck power stage and performs current and voltage control functions. The INA630 current sense amplifier senses the battery current, and the ADC terminal connects to the battery directly for voltage sense. Current and voltage signals are converted to digital data by the external ADS9324 ADC. C2000™ on-chip window comparators implement overcurrent protection.

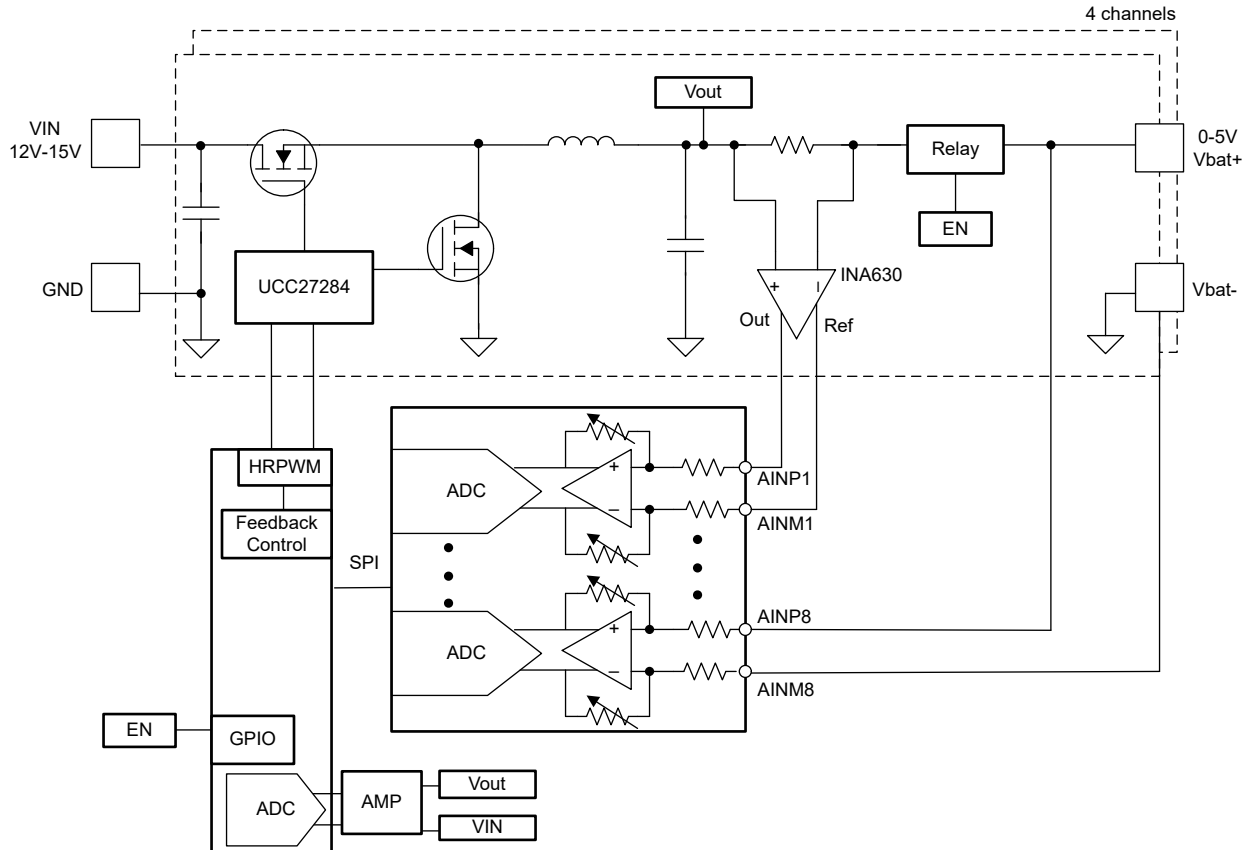


Figure 2-1. TIDA-010086 System Block Diagrams

## 2.2 System Design Theory

### 2.2.1 Feedback Controller

Figure 2-2 shows the software implementation of current and voltage control loops. Voltage loop is cascaded to the current to achieve both constant-current and constant-voltage in charging and discharging modes. When the battery voltage is far away from the constant-voltage setting (VSET), the voltage loop gets saturated to constant current setting (ISET). When battery voltage reaches close to VSET, the voltage loop is closed, and ISET is reduced to make sure the battery voltage does not exceed the VSET limit. The controller works in both charge and discharge modes. In the charge mode, VSET limits the maximum battery voltage, thus stops the charging. While in the discharge mode, VSET limits the minimum battery voltage which stops the discharging.

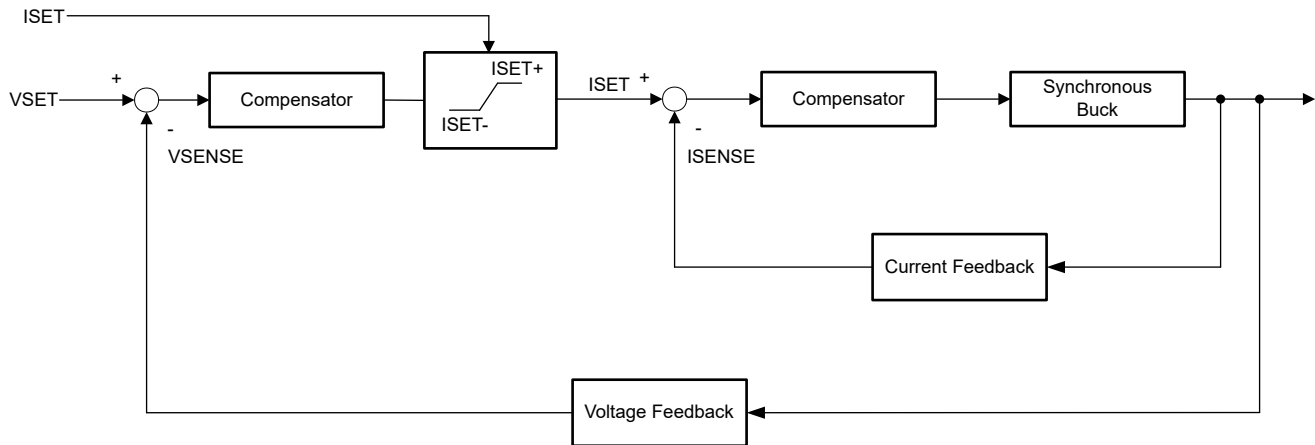


Figure 2-2. CCCV Feedback Controller

### 2.2.2 DC/DC Start-Up

A bidirectional power stage is used to charge and discharge batteries. In a normal start-up condition, buck converter output increases from 0V to the target voltage. If a battery load is connected while buck is ramping from 0V, this can result in large current overshoot. This problem can be avoided in two ways. The first method is starting the buck converter with output relay open, and setting the relay to closed position when buck converter reaches close to the battery voltage, as shown in the Figure 2-3. The second method is starting the buck converter in DCM mode with the low-side switch OFF during charging, or with the high-side OFF during discharging, as shown in Figure 2-4. A timer is required to switch from non-synchronous mode to synchronous mode.

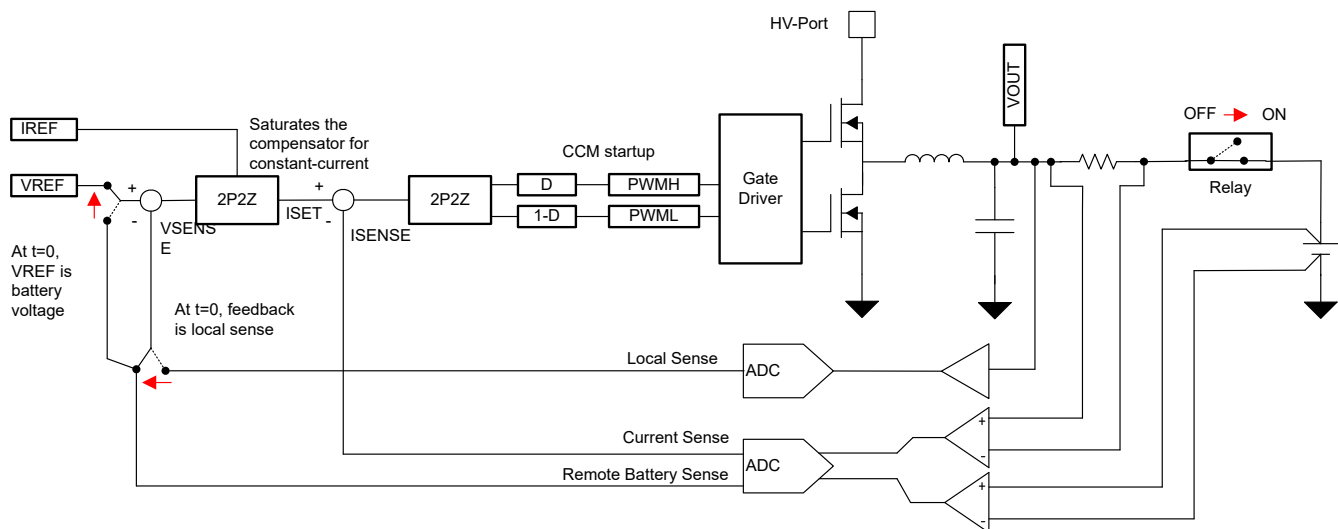


Figure 2-3. Synchronous Start-Up

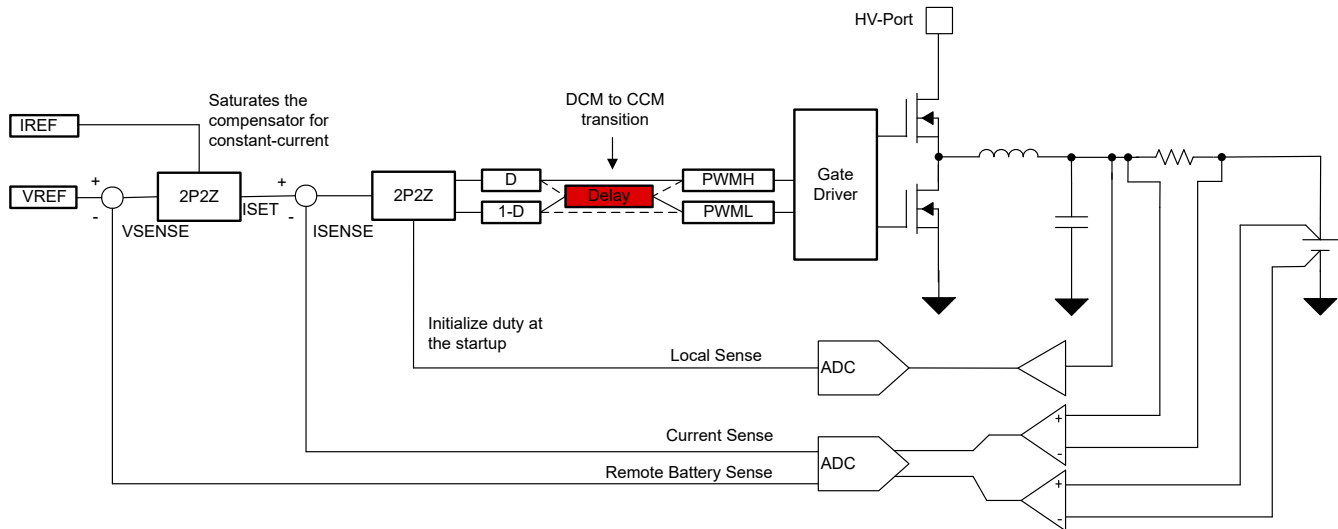


Figure 2-4. Nonsynchronous Start-Up

### 2.2.3 High-Resolution PWM Generation

High-resolution PWM achieves high-accuracy feedback control. The high-resolution counter provides 150ps time step capability. For 250kHz switching frequency and 200MHz EPWMCLK, this approach provides approximately 14.7 bits of resolution. Table 2-1 shows PWM resolution at different switching frequencies.

Table 2-1. C2000™ MCU Resolution for PWM and HRPWM

PWM FREQUENCY (kHz)	REGULAR RESOLUTION (PWM)		HIGH-RESOLUTION PWM	
	100MHz EPWMCLK			
	BITS	%	BITS	%
20	12.3	0.02	18.1	0
50	11	0.05	16.8	0.001
100	10	0.1	15.8	0.002
150	9.5	0.15	15.2	0.003
200	9	0.2	14.8	0.004
250	8.6	0.25	14.4	0.005

### 2.2.4 Output Inductor and Capacitor Selection

The value of the output capacitor and the ESR of the capacitor determine the output voltage ripple and load transient performance. This capacitor is designed to achieve the best possible output voltage ripple and load transient performance.

Equation 1 calculates the duty cycle of the buck current.

$$D = \frac{V_{OUT(max)}}{V_{IN(min)}} \times \text{Efficiency} = \frac{6V}{12V} \times 90\% = 55.5\% \quad (1)$$

Inductor ripple current is typically defined between 0.2 to 0.3 times the output current. Since the control loop needs to be fast, the inductor can be picked at smaller value. So 0.2 is selected as the ripple current coefficient. Full scale of 12 A is used in the calculation for some margin, so the inductor ripple current is estimated to be 2.4 A.

The worst-case scenario for Continuous Conduction Mode (CCM) in a buck converter occurs when  $T_{off}$  is at the maximum as represented in Equation 2.

$$\Delta I_L = \frac{(V_{OUT(max)}) \times (1 - D)}{f_S \times L} \quad (2)$$

Solving the equation, the inductor value needs to be higher than 4.45  $\mu\text{H}$ . An inductor with smaller inductance has a larger saturation current. In this design, a 4.7  $\mu\text{H}$  inductor is selected. Equation 3 calculates the actual inductor current.

$$\Delta I_L = \frac{6 \text{ V} \times (1 - 0.555)}{250 \text{ kHz} \times 4.7 \mu\text{H}} \cong 2.27 \text{ A} \quad (3)$$

Equation 4 calculates the output capacitance without considering the ESR.

$$C_{\text{out}} = \frac{\Delta I_L}{8 \times f_{\text{sw}} \times \Delta V_{\text{Out\_Ripple}}} \quad (4)$$

The output voltage ripple is targeted to be 0.1% of the maximum output voltage, which is 6 mV. Plug in Equation 4, and the output capacitor is calculated at 192  $\mu\text{F}$ . In this design, four 47  $\mu\text{F}$  and two 1- $\mu\text{F}$  ceramic capacitors are placed in parallel to match the capacitance.

The ESR requirement of output capacitors is also critical to determine the total output voltage ripple. Higher ESR can lead to an overall higher output voltage ripple. To calculate the ESR requirements, use the following equation to calculate the voltage ripple due to the capacitor alone and due to ESR alone:

$$V_{\text{o\_pp}} = \sqrt{\left(\frac{\Delta I_L}{8 \times C_{\text{out}} \times f_{\text{sw}}}\right)^2 + (\Delta I_L \times R_{\text{ESR}})^2} \quad (5)$$

$$6 \text{ mV} = \sqrt{\left(\frac{2.4 \text{ A}}{8 \times 190 \mu\text{F} \times 250 \text{ kHz}}\right)^2 + (2.4 \text{ A} \times R_{\text{esr}})^2} \quad (6)$$

Solve for  $R_{\text{esr}} \cong 0.5 \text{ m}\Omega$ . This value can be achieved with paralleling ceramic capacitors; each capacitor has around 1.5  $\text{m}\Omega$  of ESR at 250 kHz switching frequency. Paralleling the connection of the capacitor can reduce the overall ESR to less than 1  $\text{m}\Omega$ .

### 2.2.5 Current and Voltage Feedback

The design targets for  $\pm 0.02\%$  FS current and voltage control and measurement accuracy over  $\pm 5^\circ\text{C}$  temperature variation. Current sense is done with an instrumentation amplifier to measure the voltage across the shunt and scale to ADC input voltage.

The offset and gain errors are not a big concern since they can be calibrated in the equipment. However, gain and offset drift is the critical parameters for accuracy  $\pm 0.02\%$  or  $\pm 200 \text{ ppm}$ . A smaller shunt resistor, helps reduce the thermal dissipation and total temperature drift when there is a large current going through the output stage. The reference design uses 2  $\text{m}\Omega$ , 25ppm/ $^\circ\text{C}$  current shunt for 10A full-scale current range. INA630 is used for current sense, the indirect current feedback (ICFB) topology of this instrumentation amplifier makes it a cost-effective approach because this device eliminates the process of laser trimming the internal precision trimmed resistor, and the gain is set through a discrete external resistor.

The INA630 has 0.5 $\mu\text{V}/^\circ\text{C}$  input offset drift and 3-ppm/ $^\circ\text{C}$  typical gain drift, for a 2- $\text{m}\Omega$  sense resistor and 10-A full-scale current rating, the signal chain drift is 25.18ppm/ $^\circ\text{C}$ . Since the gain is dictated by the matching of an external resistor divider, the gain drift can be reduced by even better temperature coefficients. For best performance, using matching resistor pairs can achieve lowest errors of gain drift.

The total ADC drift comes from the voltage reference, for with REF50E the temperature drift is 2.5 ppm/ $^\circ\text{C}$ . The total unadjusted error of the current path is  $\pm 5 \text{ }^\circ\text{C} \times \sqrt{(25^2 \times (2.5)^2)} \text{ ppm}/^\circ\text{C} = \pm 126.5 \text{ ppm}$ , which matches the design requirements.

The voltage sense path gives a better error margin over temperature change. The ADS9324 has 1M $\Omega$  input impedance and can support an input range of  $\pm 5 \text{ V}$  through an integrated programmable gain amplifier. For both positive and negative analog input, it can connect to 5V battery directly, eliminating the external difference amplifier and reducing BOM cost. It also has 100dB minimum CMRR to support high-accuracy voltage sense. For a swing of  $\pm 1 \text{ V}$  in a 5V cell, the error due to CMRR is  $\pm 5\mu\text{V}$  or  $\pm 10\text{ppm}$  of the full-scale voltage range. The offset drift is 0.5ppm/ $^\circ\text{C}$ , so the total error can be much simpler to reach below 50ppm margin.

## 2.3 Highlighted Products

### 2.3.1 TMS320F28P650DK

The TMS320F28P650DK C2000 device is used control the synchronous buck power stage. The device has 36 HRPWM channels that are sufficient to control 18 buck converters. See the [TMS320F28P65x Real-Time Microcontrollers](#) data sheet for more information.

### 2.3.2 ADS9324

The ADS9324 is a 16-channel, simultaneous sampling, 16-bit, simultaneous sampling (SAR), analog-to-digital converter (ADC). The device allows a maximum sample rate up to 750 ksp/s per channel with integrated LPF of 15 kHz and 30 kHz to reduce noise. True bipolar input range can accept  $\pm 5$  V, and  $\pm 6.25$  V as ADC full scale, so the device can eliminate precision voltage sense amplifier. When connected to a battery cell, the ADS9324 is sufficient for  $\pm 0.01\%$  accuracy and 1.5-kHz loop bandwidth. For more information see the [ADS9324](#).

### 2.3.3 INA630

The INA630 is a precision, cost-optimized indirect current feedback instrumentation amplifier with low-input offset ( $0.7 \mu\text{V}/\text{C}$ ) and typical gain drift ( $2.5 \text{ ppm}/\text{C}$ ) at gain of 100, which achieves  $\pm 0.02\%$  current control accuracy over  $\pm 5^\circ\text{C}$  temperature change. See also the [INA630 Precision, 126dB CMRR, Indirect Current Feedback Instrumentation Amplifier](#) data sheet.

### 2.3.4 UCC27284

The UCC27284 is a robust N-channel MOSFET driver with a maximum switch node (HS) voltage rating of 100 V. This device allows for two N-channel MOSFETs to be controlled in half-bridge or synchronous buck configuration-based topologies. It has 3A source and sink capability, with 16-ns typical propagation delay which minimizes the dead-time requirement and further improves efficiency

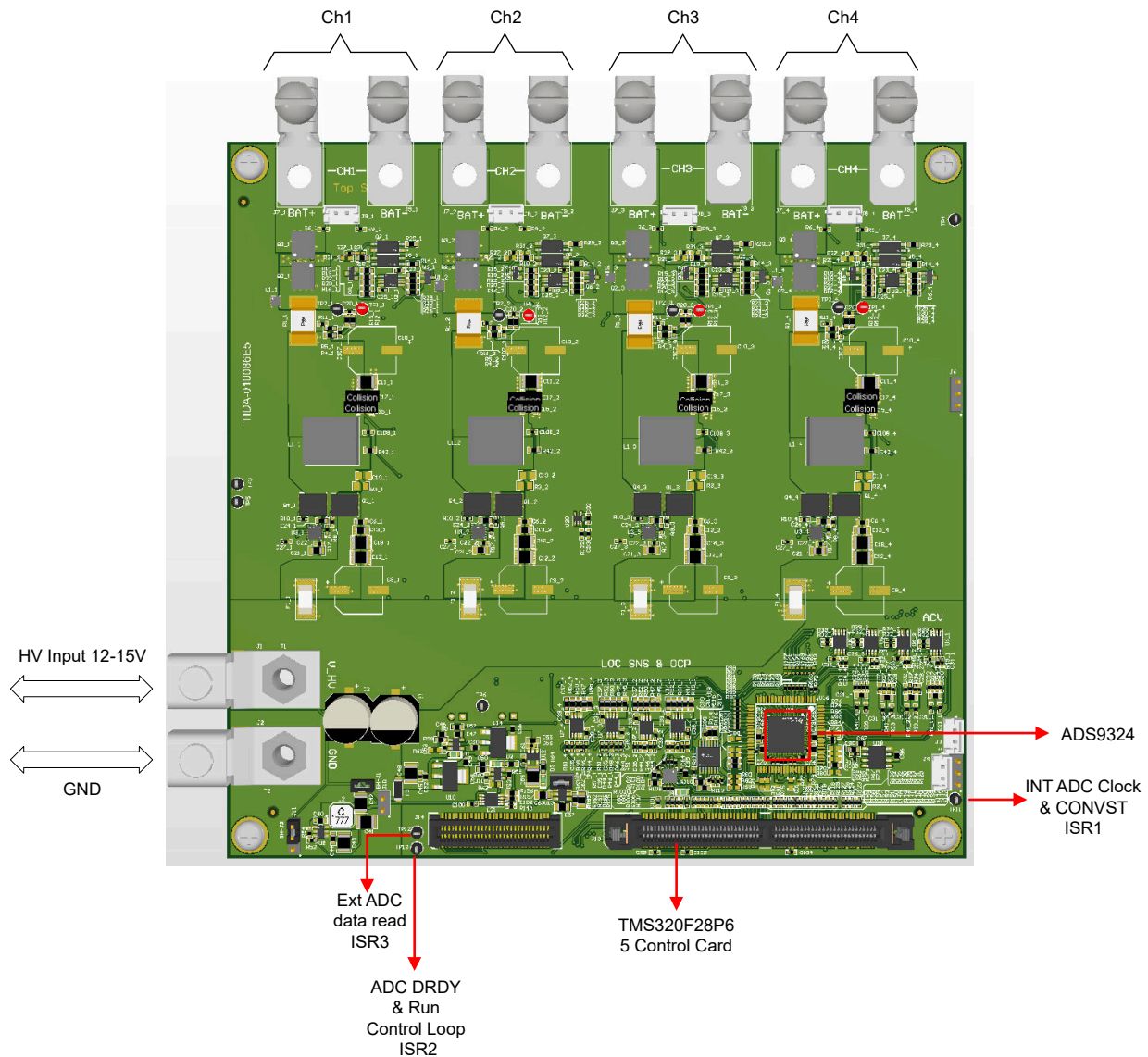
### 2.3.5 REF50E

The REF50xxE is a family of low-noise, low-drift, very high precision voltage references. It achieve excellent temperature drift ( $2.5 \text{ ppm}/\text{C}$ ) and high accuracy ( $0.025\%$ ) using proprietary design techniques. Combined with very low flicker noise ( $0.5 \mu\text{VPP}/\text{V}$ ), make the REF50xxE an excellent choice for use in high-precision data acquisition systems.

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

### 3.1 Hardware Requirements

Figure 3-1 shows various portions of TIDA-010086 hardware. The board requires a [F28P65 controlCARD](#) evaluation module to test the hardware and software performance.



**Figure 3-1. TIDA-010086 Hardware**

The following equipment is used to evaluate the reference design:

- DC system power supply: HP 6675A
- Lab power supply: Agilent E3634A
- DC Load: Chroma 63102A
- 6.5-digit resolution multimeter: Agilent 34401A
- Oscilloscope: Tektronix MDO34



## 3.2 Software

The software uses the [Code Composer Studio \(CCS\) Integrated Development Environment \(IDE\)](#) and is available in [C2000WARE-DIGITALPOWER-SDK](#) library.

### 3.2.1 Opening the Project Inside Code Composer Studio™

Use the following steps to start a project in Code Composer Studio (CCS):

1. Install Code Composer Studio from the [Code Composer Studio \(CCS\) integrated development environment \(IDE\)](#) tools folder. Version 12.4 or above is recommended.
2. Install [C2000WARE-DIGITALPOWER-SDK](#) in one of two ways:
  - a. Go to CCS and click on *View* → *Resource Explorer*. Under the TI Resource Explorer, go to C2000WARE-DIGITAL-POWER-SDK, and click on the install button.
  - b. Through the C2000Ware Digital Power SDK tools folder
3. Once installation completes, close CCS, and open a new workspace. CCS automatically detects powerSUITE. Sometimes CCS must be restarted for the change to take effect.

---

#### Note

By default, powerSUITE is installed with the SDK installation.

---

The firmware project can now be imported using one of the following methods:

- Using *Resource Explorer*
  1. In the *Resource Explorer*, under C2000WARE-DIGITAL-POWER-SDK, click on *powerSUITE* → *Solution Adapter Tool*.
  2. Select TIDA-010086 from the list of designs presented under DC-DC section.
  3. The development kit page is displayed. The icon to run the project appears in the top bar. Click *Run Project*.
  4. This action imports the project into the workspace environment, and a configuration page with a GUI similar to [Figure 3-2](#) appears.
  5. If this GUI page does not appear, see the FAQ section under powerSUITE in the C2000WAREDIGITAL-POWER-SDK resource explorer.
- Directly import from the solution folder:
  1. The user can also directly import the project by going inside CCS to click *Project* → *Import CCS Projects* and browsing to the solution folder located at /solutions/tida\_010086/f28p65x/ccs.
  2. Two project specifications appear: one of the projects is with powerSUITE, and the other without powerSUITE. Clicking on either creates a self-contained folder of the project with all the dependencies inside.
  3. All the following steps describe how to modify the relevant #defines in the settings.h and user\_settings.hfile, which are documented in this design guide.

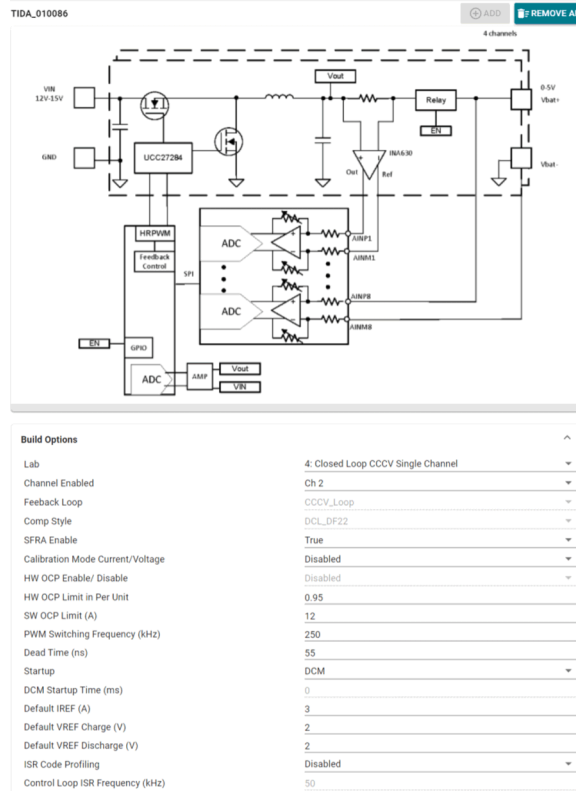


Figure 3-2. powerSUITE GUI for the Design

### 3.2.2 Project Structure

Figure 3-3 shows the general structure of the project. Once the project is imported, the Project Explorer appears inside CCS as shown in Figure 3-4.

#### Note

Figure 3-4 shows the project for F28p65x; however, if a different device is chosen from the page, the structure is similar.

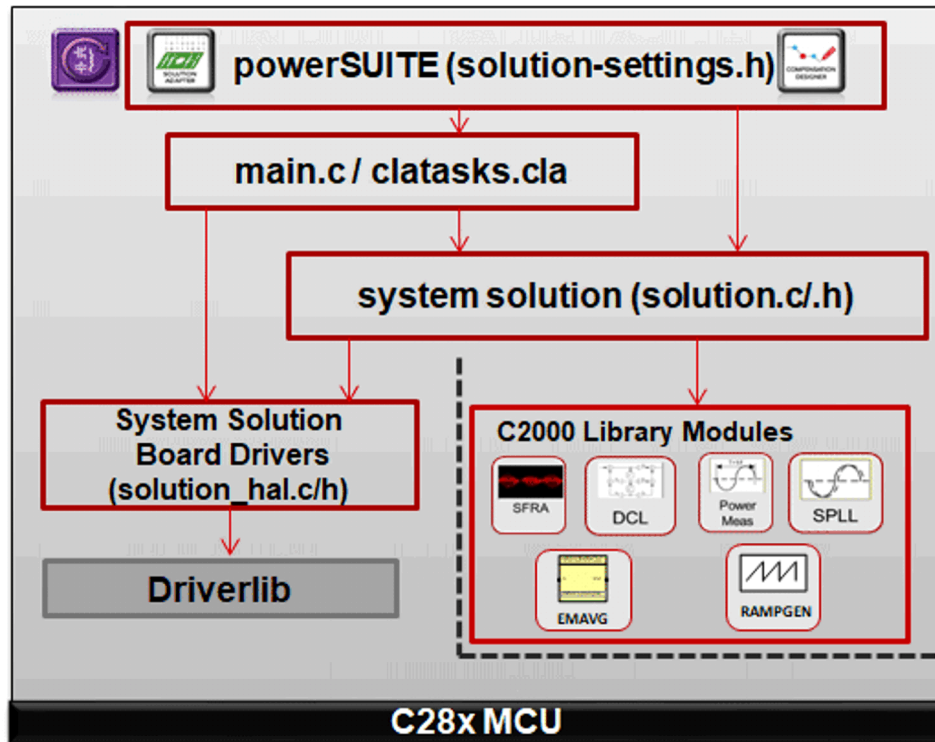


Figure 3-3. Project Structure Overview

Solution-specific and device-independent files that consist of the core algorithmic code are in **.c/h**.

Board-specific and device-specific files are in **\_hal.c/h**. This file consists of device-specific drivers to run the solution. If the user wants to use a different modulation scheme or a different device, the user is required only to make changes to these files, besides changing the device support files in the project.

The **-main.c** file consists of the main framework of the project. This file consists of calls to the board and solution file that help in creating the system framework, along with the interrupt service routines (ISRs) and slow background tasks.

For this design, the solution is **bt4ch**.

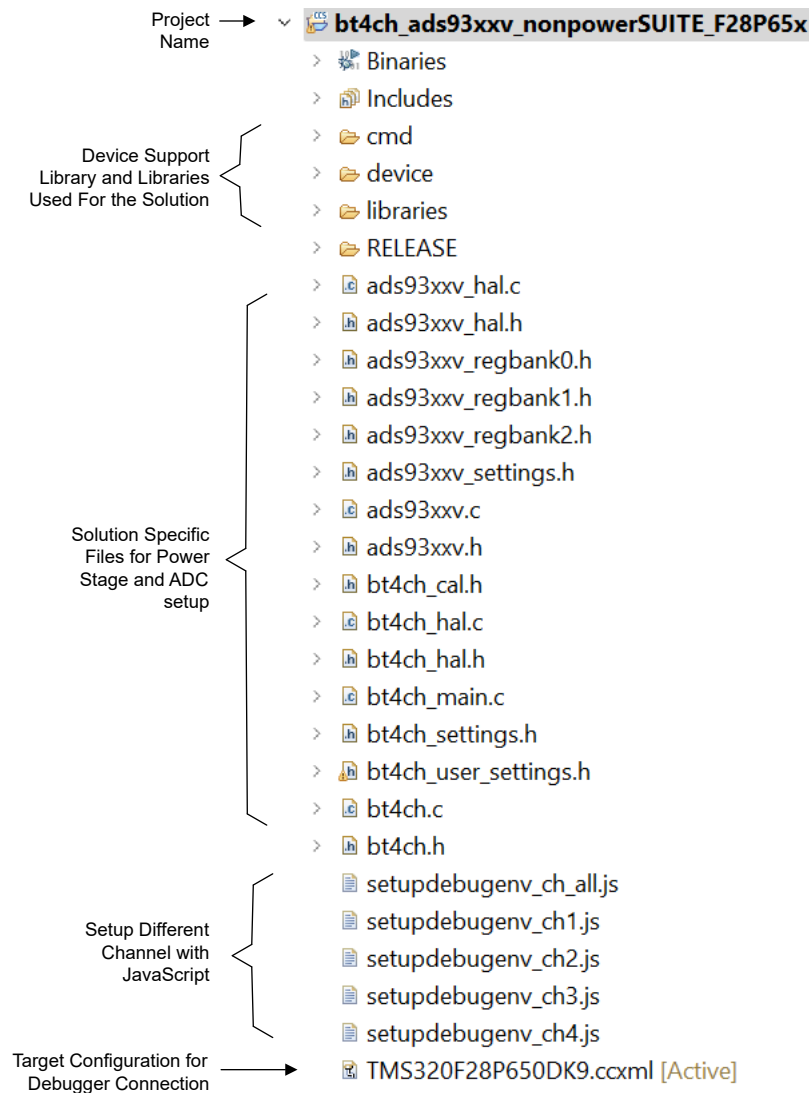
The powerSUITE page can be opened by clicking on the main.syscfg file, listed under the Project Explorer. The page generates the **\_settings.h** file. This file is the only C language-based file used in the compile of the project that is generated by the powerSUITE page. The user must not modify this file manually, because the changes are overwritten by powerSUITE each time the project is saved. The **\_settings.h** file includes function of the operation mode selection and the compensation settings for current and voltage control loop. **\_user\_settings.h** is included by the **\_settings.h** and can be used to keep any settings that are outside the scope of powerSUITE tools such as #defines for ADC mapping, GPIOs, and so forth.

The **\_cal.h** file consists of gain and offset values for current and voltage measurements.

The `_.ccxml` settings up which debugger to use for the reference design. In this design the connection of XDS110 USB debug probe is used and the target device is TMS320F28P65DK9.

The `Kit.json` and `solution.js` files are used internally by powerSUITE, and must not be modified by the user. Any changes to these files results in the project not functioning properly.

The solution name is also used as the module name for all the variables and defines used in the solution. Hence, all variables and function calls are prepended by the **BT4CH** header (for example, `BT4CH_userParam_chX`). This naming convention lets the user combine different solutions while avoiding naming conflicts.



**Figure 3-4. Project Explorer View of the BT4PH Project**

The `bt4ch` project consists of three ISRs (ISR1, ISR2, and ISR3).

- *ISR1* is used to sense the input supply voltage and output capacitor voltage of the buck converters. *ISR1* is triggered by ADCC conversion complete and runs at same frequency as the external ADC CONVST. ADCC senses input voltage and output voltage of the converters, and the output is used to implement the soft-start of the DC/DC.
- *ISR2* is triggered by the BUSY signal of the ADS9324. The external ADC is programmed for a 400 kSPS sample rate (CONVST), with oversampling rate (OSR) set to 8 which sets the ISR frequency to 50 kHz.
- *ISR3* is triggered by SPI receive FIFO interrupt. The ISR is used to read the external ADC data from FIFO registers, and run the control loop functions.

Figure 3-6 shows the time taken by ISR1, ISR2, and ISR3 when all four channels are ON. The total time taken three ISRs is less than 6  $\mu\text{s}$  which is less 30% of CPU usage for 50 kSPS control loop sample rate. Figure 3-5 and Figure 3-7 show ISR time for when only one channel is ON and all channels are OFF.

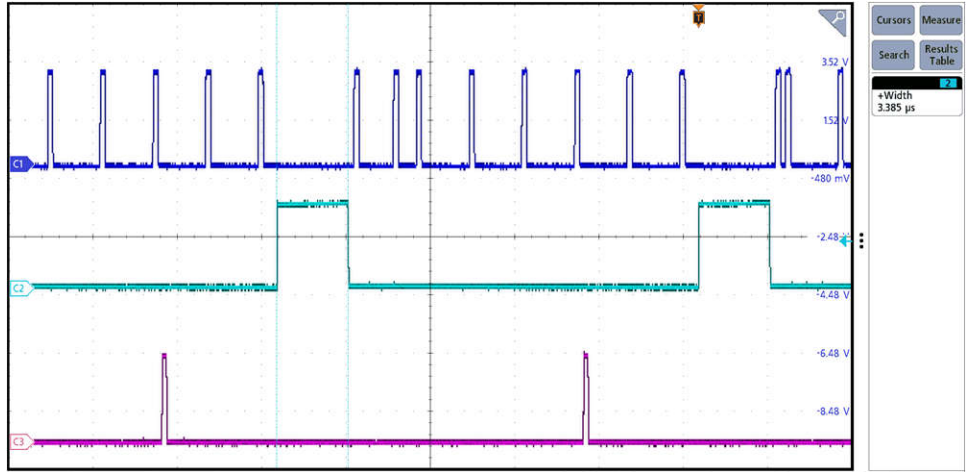


Figure 3-5. ISR Execution Time for One Channel

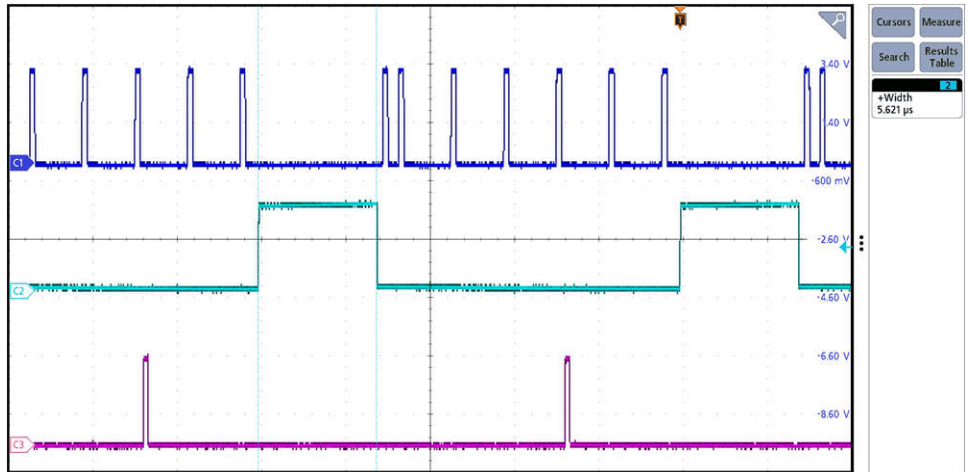


Figure 3-6. ISR Execution Time for Four Channels

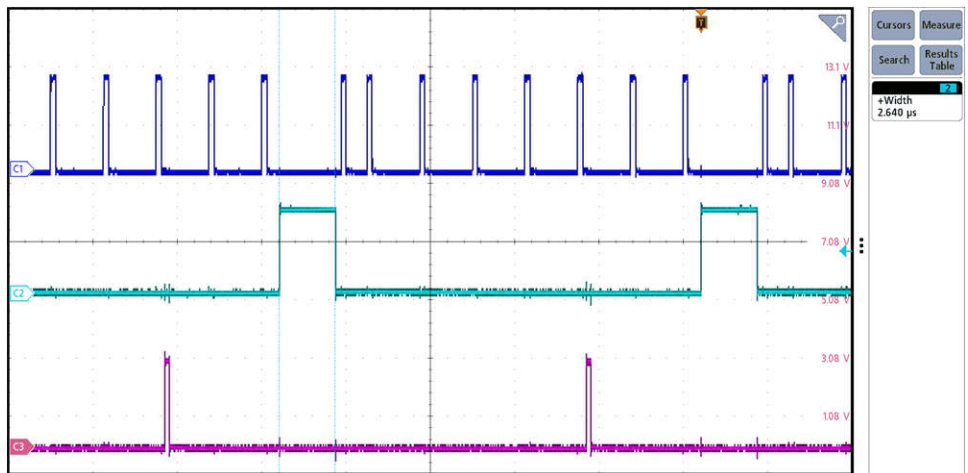


Figure 3-7. ISR Execution Time When All Channels are OFF

### 3.2.3 Software Flow Diagram

Figure 3-8 shows the software flow diagram.

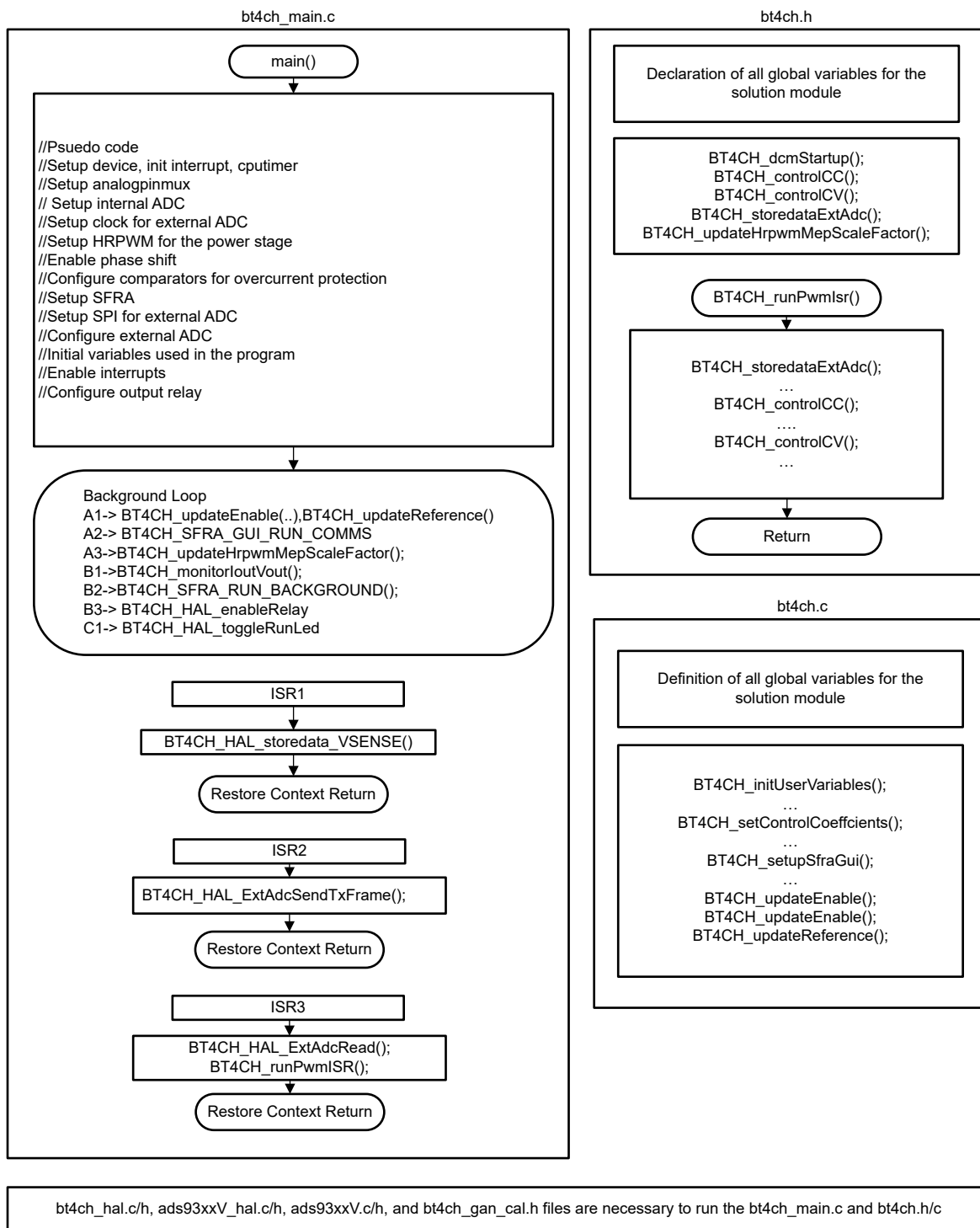


Figure 3-8. Software Flow Diagram

### 3.3 Test Setup

#### 3.3.1 Hardware Setup to Test Bidirectional Power Flow

Figure 3-9 shows the setup to test the bidirectional power. The input DC power supply operates at 12 V to 15 V as input bus voltage. The electronic load (e-load) provides discharging mode when the input power source cannot support bidirectional current flow. On the battery side, the system connects to a lab power supply and an e-load to simulate charging or discharging a battery cell.

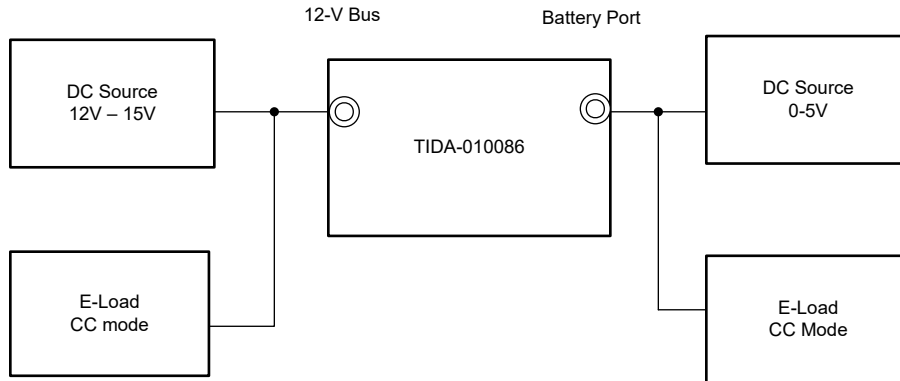


Figure 3-9. Hardware Setup to Test Bidirectional Power Flow

#### 3.3.2 Hardware Setup to Tune the Current and Voltage Loop

The control loop requires tuning during single-channel operation only. Engineers tune the current and voltage loops using this method. The battery load appears as a small impedance cable, ranging from milliohms to hundreds of milliohms. Without external instruments, impedance matching cable provides a simple approach to tune the control loop compared to a real battery cell.

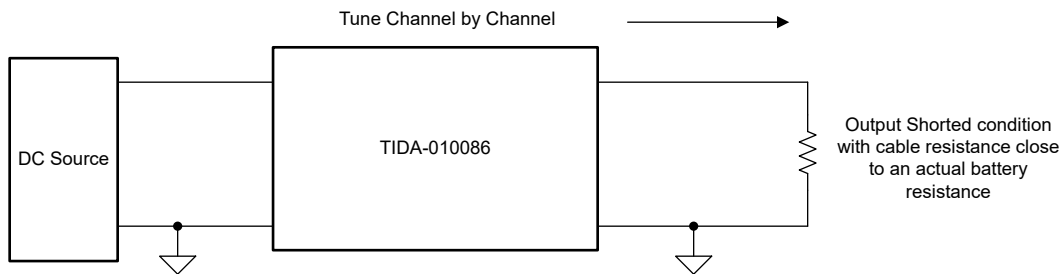
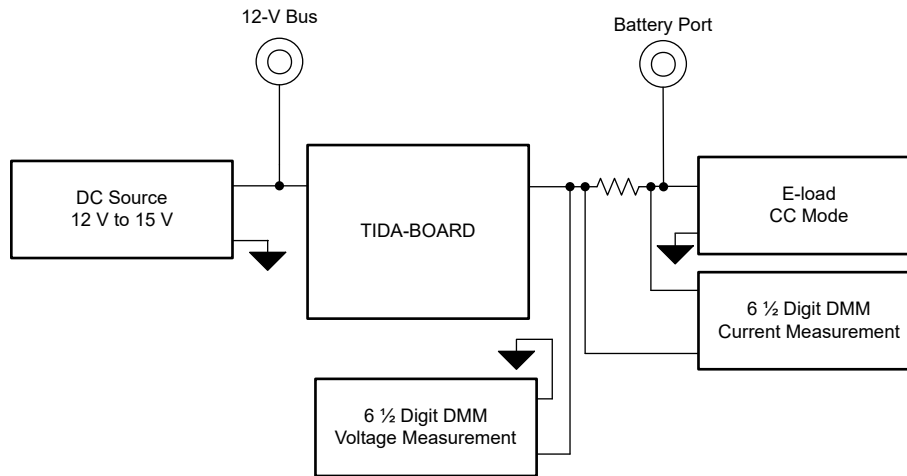


Figure 3-10. Hardware Setup to Tune the Current and Voltage Loop

#### 3.3.3 Hardware Setup for Current and Voltage Calibration

To calibrate the current and voltage, a 6.5-digit multimeter measures the output terminal voltage and voltage drop across the current shunt. Current calibration uses electronic load constant current mode or short-circuit conditions. Voltage calibration uses open circuit conditions.


**Figure 3-11. Hardware Setup for Current and Voltage Calibration**

### 3.3.4 Lab Variables Definitions

The BT4CH\_userParam\_ch1,2,3,4 variable is used to control the power stage in different Labs. See [Table 3-1](#) for the parameter definitions.

**Table 3-1. BT4CH\_userParam\_chX Definition**

BT4CH_userParam_chX	DATA TYPE	COMMENTS
Iref_A	float	Set current for both charging and discharging mode [0, 10]
VrefCharge_V	float	Set voltage in charge mode [0, 5]
VrefDischarge_V	float	Set voltage in discharge mode [0, 5]
Dir_bool	unsigned int	Set this parameter to 1 for charging mode
		Set this parameter to 0 for discharging mode
Relay_ON	unsigned int	Set this parameter to 1 to enable relay
		Set this parameter to 0 to disable the relay
En_bool	unsigned int	Set this parameter to 1 to enable the channel
		Set this parameter to 0 to disable the channel
remote_sense	unsigned int	Set this parameter to 1 for to use remote sense for closed-loop control
		Set this parameter to 0 to use local sense for closed-loop control
DutyRef_pu	float	Reference duty cycle for open loop mode. Range = 0 to 1.0
IbatCal_pu	float	Use this parameter to set output current in calibration mode. Range = 0 to 1.0
VbatCal_pu	float	Use this parameter to set output voltage in calibration mode. Range = 0 to 1.0
IoutGain_pu	float	The variable stores current gain calibration data
IoutOffset_pu	float	The variable stores battery current offset calibration data
IoutGain_A	float	The variable stores battery current gain calibration data
IoutOffset_A	float	The variable stores battery current offset calibration data
VoutGain_pu	float	The variable stores battery buck converter output voltage gain calibration data
VoutOffset_pu	float	The variable stores buck converter output voltage offset calibration data
VoutGain_V	float	The variable stores buck converter output voltage offset calibration data
VoutOffset_V	float	The variable stores buck converter output voltage offset calibration data
VbatGain_pu	float	The variable stores battery voltage calibration data
VbatOffset_pu	float	The variable stores battery voltage offset calibration data
VbatGain_V	float	The variable stores battery voltage gain calibration data
VbatOffset_V	float	The variable stores battery voltage offset calibration data



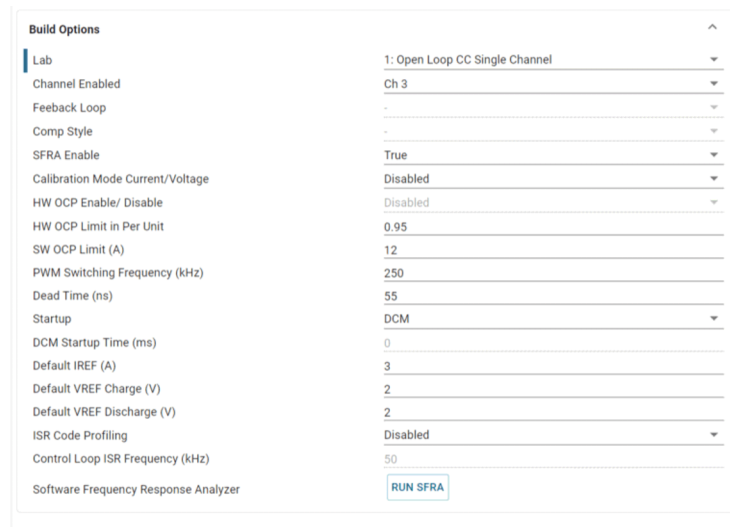
### 3.3.5 Test Procedure

#### 3.3.5.1 Lab 1. Open-Loop Current Control Single Phase

##### 3.3.5.1.1 Setting Software Options for Lab 1

1. Open the CCS project as outlined in [Section 3.2.1](#). If using the powerSUITE, go to [Step 2](#); otherwise jump to [Step 3](#).
2. Open the SYSCONFIG page and select under the *Build Options* section:
  - Select *Lab 1: Open Loop CC Single Channel* for the Lab
  - Select any of the four channels
  - Enable the SFRA
  - Save the page
3. When using the non-powerSUITE version of the project, the above settings are directly modified in the `solution_settings.h` file.


```
#define LAB_NUMBER (1)
#define CHANNEL_NUMBER (1)
#define SFRA_ENABLED (true)
```



**Figure 3-12. Build Options for Lab 1**


##### 3.3.5.1.2 Building and Loading the Project and Setting up Debug Environment

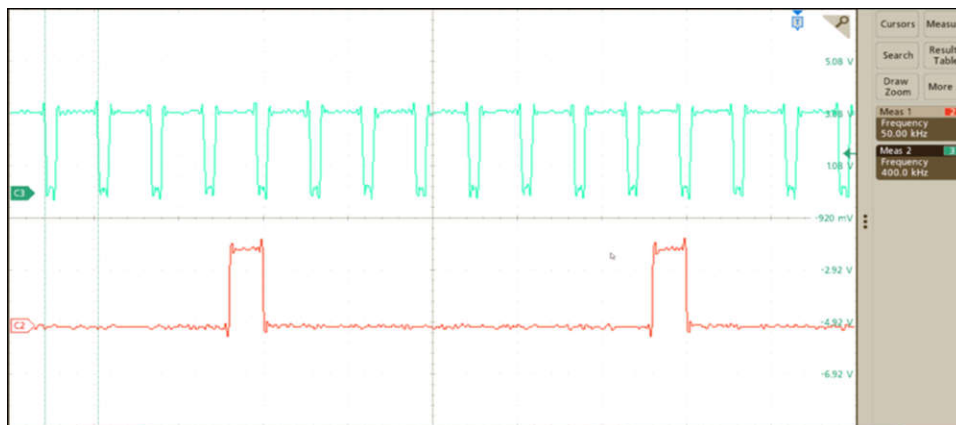
Use the following steps to build and load the project and to set up the debug environment.

1. Right-click on the project name and click **Rebuild Project**.
2. The project builds successfully.
3. In the Project Explorer, make sure the correct target configuration file is set as Active under targetConfigs.
4. Click **Run** → **Debug** to launch a debugging session.
5. The project then loads on the device and the CCS debug view becomes active. The code halts at the start of the main routine.
6. To add the variables in the watch/expressions window, click **View** → **Scripting Console** to open the scripting console dialog box. On the upper right corner of this console, click on *Open* and then browse to **setupdebugenv\_chX.js** the script file located inside the project folder. This populates the watch window with the appropriate variables needed to debug the system.
7. Click on the **Continuous Refresh** button  on the watch window to enable continuous update of values from the controller.

### 3.3.5.1.3 Running the Code

Use the following steps to run the code for Lab 1:

1. Use the test setup shown in [Section 3.3.2](#).
2. Run the project by clicking  from the menu bar.
3. In the watch view, check if the BT4H\_InputVoltageSense\_V is from 12V to 15V in the *Expression Window*.
4. Check the *DRDY* signal of the external ADC if the frequency is 50kHz using an oscilloscope. [Figure 3-13](#) shows the ADS9324 *DRDY* and *CONVST* signals when the MCU is running.
5. Set the following parameters from the *Expression Window*:
  - BT4CH\_userParam\_chX->dutyRef\_pu = 0.02
  - Set the BT4CH\_userParam\_chX->en\_bool = 1
  - Set the BT4CH\_userParam\_chX->Relay\_ON to 1 to enable the output relay
  - See [Figure 3-14](#) for the *Expression Window* settings
6. The BT4CH\_measure\_V\_I\_chX variable shows output current and voltage of the DC/DC converter. Adjust the BT4CH\_userParam\_chX->DutyRef\_pu to make sure the current is approximately 4.5A.
7. [Figure 3-15](#) shows the SFRA setup to extract the plant model for Open-Loop Current Control. Click on the *Run SFRA* icon from the SYSCONFIG page. The SFRA GUI pops up.
8. Select the options for the device on the SFRA GUI; for example, for F28P65x, select *Floating Point*. Click on the *Setup Connection* button. In the pop-up window, uncheck the boot-on-connect option and select an appropriate COM port. Click the *OK* button. Return to the SFRA GUI and click the *Connect* button.
9. The SFRA GUI connects to the device. An SFRA sweep can now be started by clicking *Start Sweep*. The complete SFRA sweep takes a few minutes to finish. Once complete, a graph with the measurement appears, as shown in [Figure 3-16](#).
10. The Frequency Response Data is saved in the project folder, under an SFRA Data folder, and is time-stamped with the time of the SFRA run.
11. After finishing the lab, set the following parameters from the *Expression Window* to stop the code:
  - BT4CH\_userParam\_chX->dutyRef\_pu = 0
  - Set the BT4CH\_userParam\_chX->en\_bool = 0
  - Set the BT4CH\_userParam\_chX->Relay\_ON to 0 to disable the output relay
  - Terminate the program



**Figure 3-13. ADS9324 CONVST and DRDY Signal**

Expression	Type	Value
BT4CH_lab	enum <unname...	Lab1_singleChannelOpenLoopCC
BT4CH_sfraStatus	enum <unname...	SFRA_Disabled
BT4CH_calibrationStatus	enum <unname...	Calibration_Disabled
BT4CH_calibrationMode	enum <unname...	Calibration_CC
BT4CH_startupMode	enum <unname...	DCM_Startup
BT4CH_HAL_InputVoltageSense_V	float	13.3130159
BT4CH_ISR2_Loading	float	0.126220882
BT4CH_ISR2_LoadingMax	float	0.12963891
BT4CH_userParam_ch3	struct <unname...	{Iref_A=5.0,VrefCharge_V=3.0,VrefDi...
Iref_A	float	5.0
VrefCharge_V	float	3.0
VrefDischarge_V	float	3.0
dir_bool	unsigned int	1
Relay_ON	unsigned int	1
Start_bool	unsigned int	1
remote_sense	unsigned int	0
DutyRef_pu	float	0.0199999996
IbatCal_pu	float	0.200000003
VbatCal_pu	float	0.400000006
IoutGain_pu	float	0.0805866644
IoutOffset_pu	float	0.000886499882
IoutGain_A	float	12.4090014
IoutOffset_A	float	-0.0110005783
VoutGain_pu	float	0.24248305
VoutOffset_pu	float	-0.000581979752
VoutGain_V	float	4.1239996
VoutOffset_V	float	0.00240008417
VbatGain_pu	float	0.200000018
VbatOffset_pu	float	-0.000600039959
VbatGain_V	float	4.99999952
VbatOffset_V	float	0.00300019956
BT4CH_measureVI_ch3	struct <unname...	{Ibat_A=4.24398136,Vout_V=0.0790...
Ibat_A	float	4.24188375

Figure 3-14. Lab 1 Expression Window, Open Loop

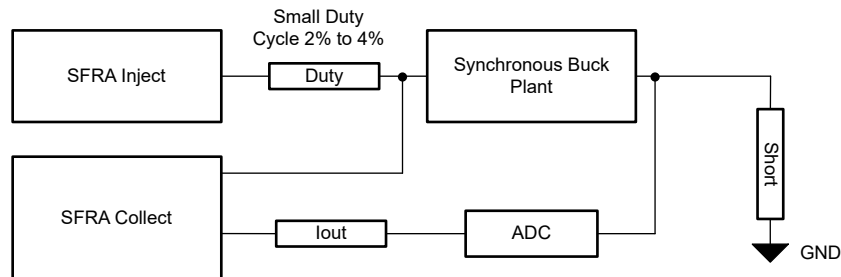


Figure 3-15. SFRA Setup for Open-Loop Current Control

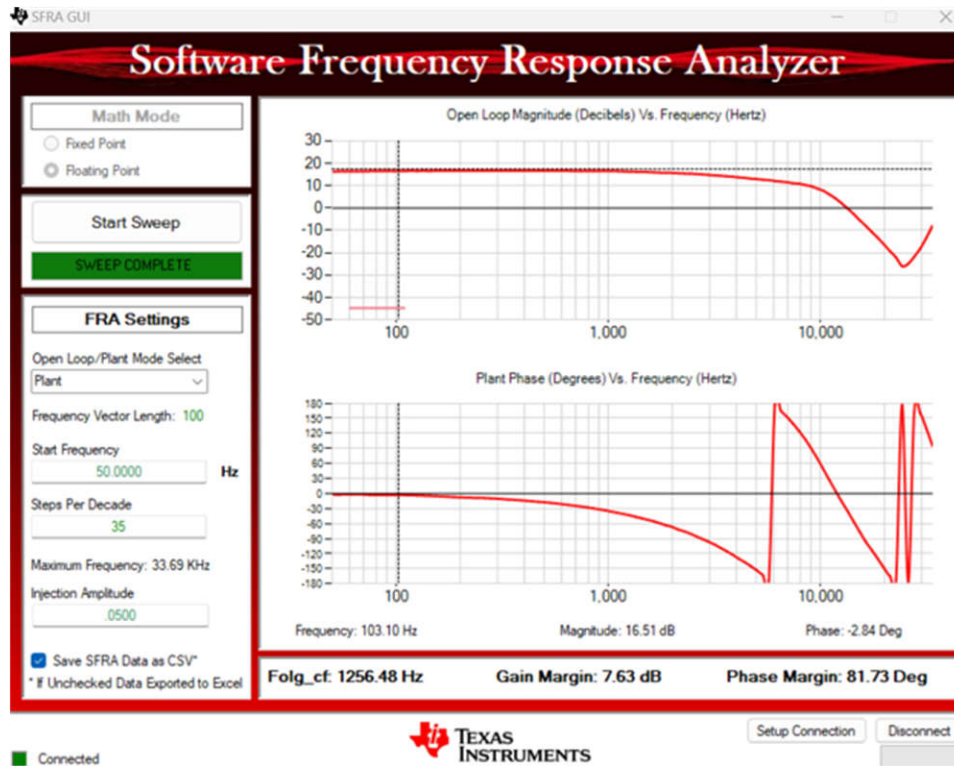



Figure 3-16. Current Control Open-Loop Frequency Response

### 3.3.5.2 Lab 2. Closed Loop Current Control Single Channel

#### 3.3.5.2.1 Setting Software Options for Lab 2

1. To run this lab, make sure the hardware is set up as outlined in the previous section, [Section 3.3.2](#)
2. Open the CCS project as outlined in [Section 3.2.1](#). If using the powerSUITE, go to [Step 3](#), otherwise, jump to [Step 4](#).
3. Open the SYSCONFIG page and select under the *Build Options* section:
  - Select *Lab 2: Closed Loop CC Single Channel* for the Lab
  - Select the Channel
  - Enable the SFRA
  - Open the Compensation Designer  by clicking the *Run Compensation Design* button.
  - The compensation designer then launches and prompts the user to select a valid SFRA data file. Import the SFRA data from the run in Lab 1 into the compensation designer to design a two-pole, two-zero compensator. Keep more margins during this iteration of the design to make sure that when the loop is closed, the system is stable.
4. The design uses 2p2z control, which can also be modeled as a Type II compensator with OTA. To tune the control loop, this design uses the following steps:
  - First, select the desired crossover frequency  $f_c$  and phase margin  $p_0$ . Take note of the open-loop plant gain  $G_{f_c}$  and plant phase,  $p_1$ , at the crossover [frequency response analyzer](#).
  - The phase *boost* is the compensated effort to achieve  $p_0$ , the desired crossover frequency  $f_c$ , can be calculated with [Equation 7](#).
  - To calculate the compensation pole frequency  $f_p$ , use [Equation 8](#).
  - As the phase peaks at the geometric mean between the pole and zero, calculate the compensation for zero  $f_z$  with [Equation 9](#).
  - For Type II compensator with OTA, the transfer function at the crossover frequency can be represented as [Unable to auto-generate link text](#).
  - Applying the pole and zero from [Equation 8](#) and  $f_z$ ,  $K_{dc}$  can be calculated as [Equation 11](#).
  - The second zero placement can be placed at much higher than the first zero.

- Applying the pole and zero from Equation 8,  $f_z$ , and  $K_{dc}$  into the compensation designer, the 2p2z compensation can be calculated in the powerSUITE GUI.
  - Figure 3-18 shows compensation parameters for the *Current Loop*.
  - Click on the *Save Comp* button to save the compensation. Close the *Compensation Designer* tool.
  - Save the SYSCONFIG page.
5. When using the non-powerSUITE version of the project, *Build Settings* are directly modified in solution\_settings.h file. *Compensation Designer* is found at *C2000Ware\_DigitalPower\_Install\_Location\powerSUITE\source\utils*.

```
#define LAB_NUMBER (2)
#define CHANNEL_NUMBER (3)
#define SFRA_ENABLED (true)
```

The following equations are referenced in the previous list:

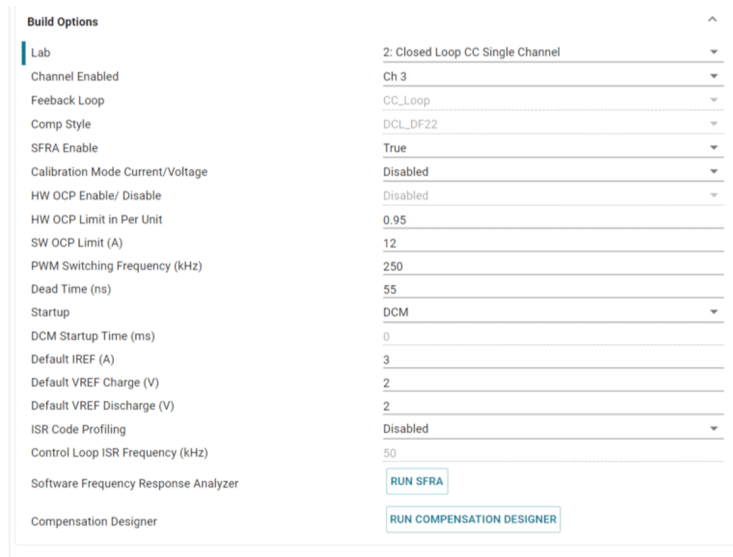
$$\text{boost} = - (p_1 - p_0 + 90) \tag{7}$$

$$f_p = \left( \tan\left(\text{boost} \times \frac{\pi}{180}\right) + \sqrt{\tan^2\left(\text{boost} \times \frac{\pi}{180}\right) + 1} \right) \times f_c \tag{8}$$

$$f_z = \frac{f_c^2}{f_p} \tag{9}$$

$$|G(f_c)| = K_{dc} \frac{\sqrt{1 + \left(\frac{f_z}{f_c}\right)^2}}{\sqrt{1 + \left(\frac{f_c}{f_p}\right)^2}} \tag{10}$$

$$K_{dc} = 2\pi f_c \frac{\sqrt{1 + \left(\frac{f_c}{f_p}\right)^2}}{\sqrt{1 + \left(\frac{f_z}{f_c}\right)^2}} \tag{11}$$



**Figure 3-17. Build Options for Lab 2**

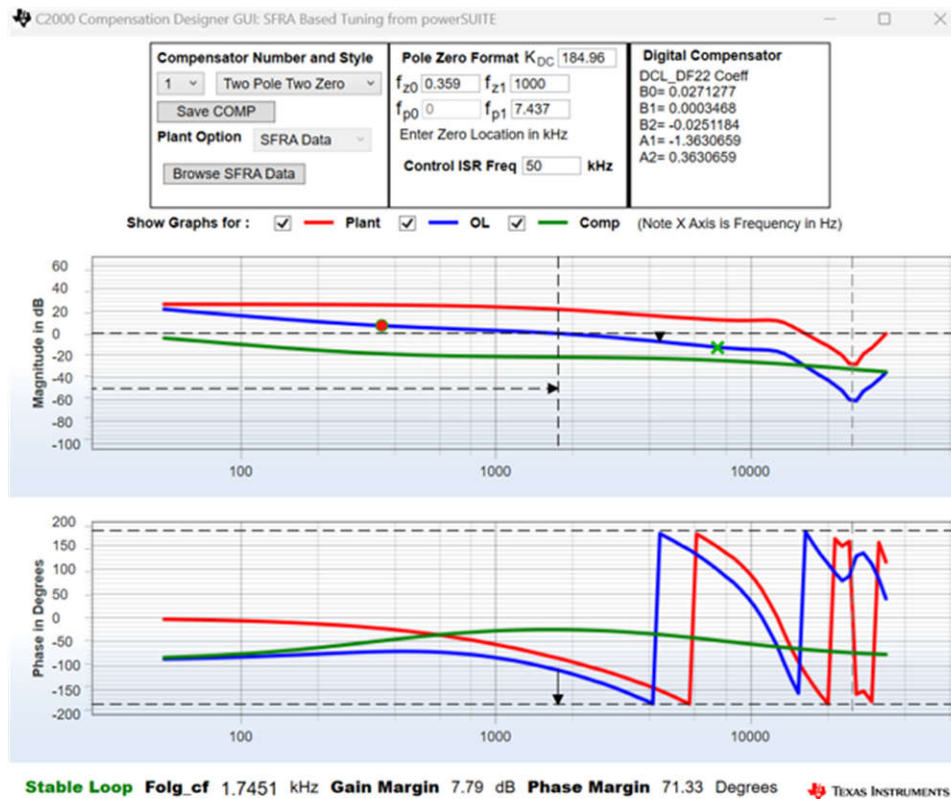




Figure 3-18. Tuning Current Loop Using Compensation Designer

### 3.3.5.2.2 Building and Loading the Project and Setting up Debug Environment

1. Right-click on the project name and click **Rebuild Project**.
2. The project builds successfully.
3. In the Project Explorer, make sure the correct target configuration file is set as Active under targetConfigs.
4. Click **Run** → **Debug** to launch a debugging session.
5. The project then loads on the device and the CCS debug view becomes active. The code halts at the start of the main routine.
6. To add the variables in the watch/expressions window, click **View** → **Scripting Console** to open the scripting console dialog box. On the upper right corner of this console, click on *Open* and then browse to the **setupdebugenv\_chX.js** script file located inside the project folder. This populates the watch window with the appropriate variables needed to debug the system.
7. Click on the **Continuous Refresh** button  on the watch window to enable continuous update of values from the controller.

### 3.3.5.2.3 Run the Code

Use the following steps to run the code for Lab 2:

1. To run this lab, make sure the hardware is set up as outlined in [Section 3.3.1](#).
2. Run the project by clicking  from the menu bar.
3. In the watch view, check if the BT4CH\_InputVoltageSense\_V is from 12V to 15V in the *Expression Window*.
4. Set the following parameters from the *Expression Window*:
  - Set the BT4CH\_userParam\_chX->Relay\_ON to 1 to enable the output relay
  - BT4CH\_userParam\_chX->iref\_A = 7.0
  - Set the BT4CH\_userParam\_chX->en\_bool = 1
  - See [Figure 3-19](#) for the *Expression Window* settings
5. The BT4CH\_measureVI\_chX variable shows output current and voltage of the DC/DC converter. Isense1\_A display value is close to iref\_A setting with  $\pm 1\text{mA}$  error.

6. Figure 3-20 shows the SFRA setup to test the loop stability. Click on the *Run SFRA* icon from the SYSCONFIG page. The SFRA GUI pops up.
7. Select the options for the device on the SFRA GUI; for example, for F28P65x, select *Floating Point*. Click on the *Setup Connection* button. In the pop-up window, uncheck the boot-on-connect option and select an appropriate COM port. Click the *OK* button. Return to the SFRA GUI and click the *Connect* button.
8. The SFRA GUI connects to the device. A SFRA sweep can now be started by clicking *Start Sweep*. The complete SFRA sweep takes a few minutes to finish. Once complete, a graph with the measurement appears, as shown in Figure 3-21.
9. The Frequency Response Data is saved in the project folder, under an SFRA Data folder, and is time-stamped with the time of the SFRA run.
10. After finishing the lab, set the following parameters from the *Expression Window* to stop the code:
  - Set the BT4CH\_userParam\_chX->en\_bool = 0
  - Set the BT4CH\_userParam\_chX->Relay\_ON to 0 to disable the output relay
  - Terminate the program

Expression	Type	Value
BT4CH_lab	enum <unname...	Lab2_singleChannelClosedLoopCC
BT4CH_sfraStatus	enum <unname...	SFRA_Enabled
BT4CH_calibrationStatus	enum <unname...	Calibration_Disabled
BT4CH_calibrationMode	enum <unname...	Disabled
BT4CH_startupMode	enum <unname...	DCM_Startup
BT4CH_HAL_InputVoltageSense_V	float	12.4000731
BT4CH_ISR2_Loading	float	0.149474204
BT4CH_ISR2_LoadingMax	float	0.17483741
BT4CH_userParam_ch3	struct <unname...	{Iref_A=7.0,VrefCharge_V=3.0,VrefDi...
Iref_A	float	7.0
VrefCharge_V	float	3.0
VrefDischarge_V	float	3.0
dir_bool	unsigned int	1
Relay_ON	unsigned int	1
Start_bool	unsigned int	1
remote_sense	unsigned int	0
DutyRef_pu	float	0.0
IbatCal_pu	float	0.200000003
VbatCal_pu	float	0.400000006
IoutGain_pu	float	0.0730994046
IoutOffset_pu	float	0.00584799051
IoutGain_A	float	13.6800022
IoutOffset_A	float	-0.0800005198
VoutGain_pu	float	0.24248305
VoutOffset_pu	float	-0.000581979752
VoutGain_V	float	4.1239996
VoutOffset_V	float	0.00240008417
VbatGain_pu	float	0.200000018
VbatOffset_pu	float	-0.000600039959
VbatGain_V	float	4.99999952
VbatOffset_V	float	0.00300019956
BT4CH_measureV1_ch3	struct <unname...	{Ibat_A=6.99993467,Vout_V=0.2138...
Ibat_A	float	6.99981737
Vout_V	float	0.2136783
Vbat_V	float	0.196476862

**Figure 3-19. Lab 2 Expression Window, Closed Loop**

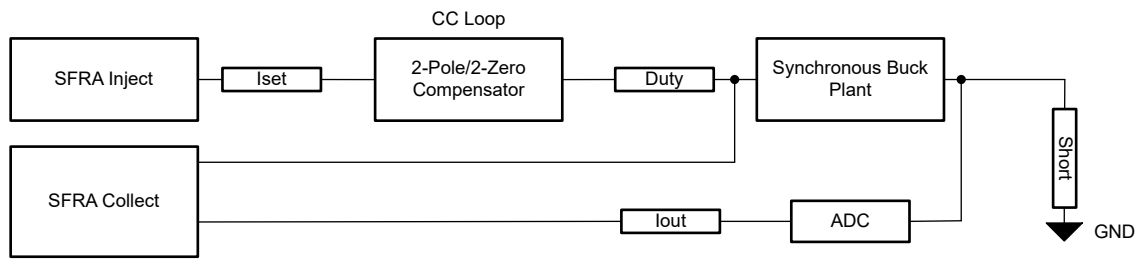


Figure 3-20. SFRA Setup for Closed-Loop Current Control

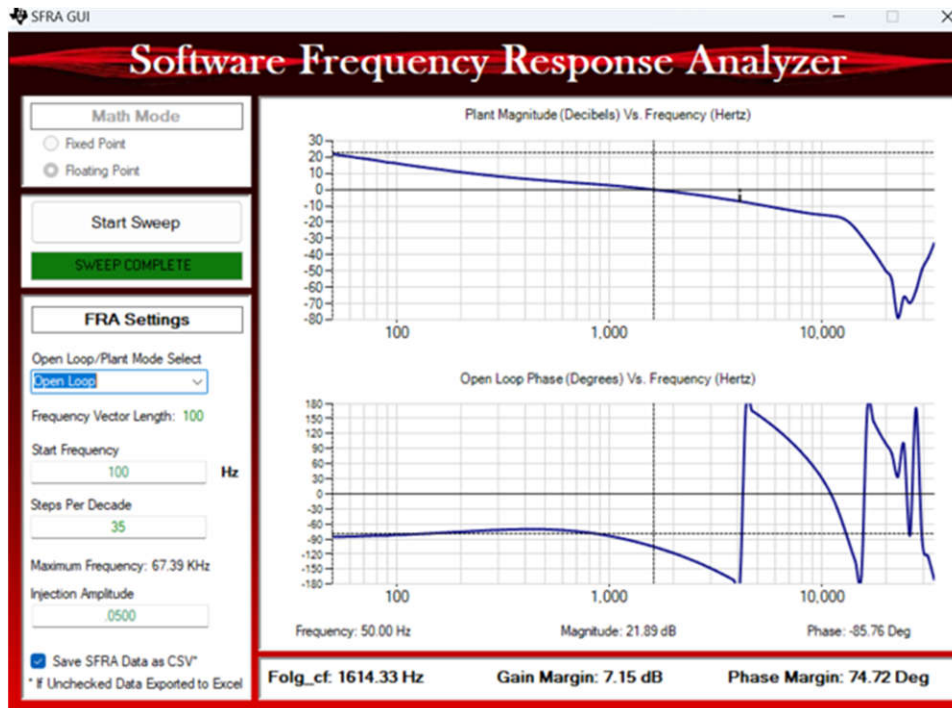


Figure 3-21. Current Control Closed-Loop Frequency Response

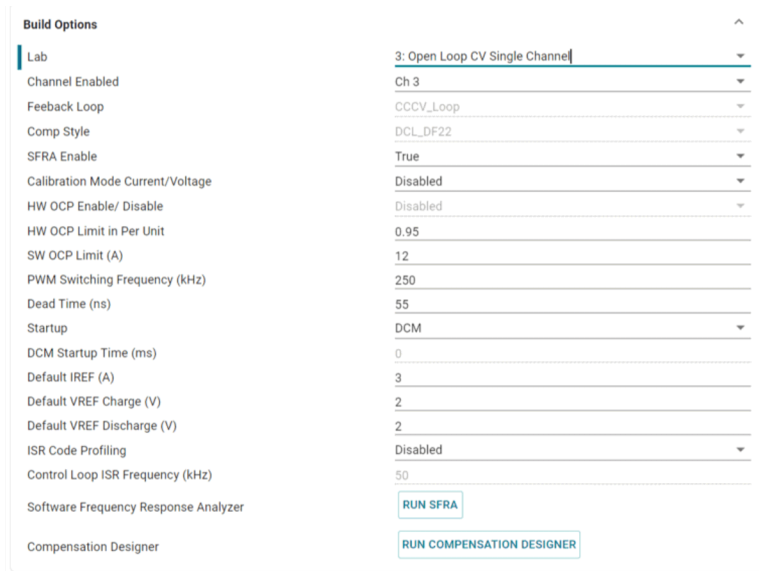
### 3.3.5.3 Lab 3. Open Loop Voltage Control Single Channel

#### 3.3.5.3.1 Setting Software Options for Lab 3

1. Open the CCS project as outlined in Section 3.2.1. If using the powerSUITE, go to Step 2; otherwise jump to 3
2. Open the SYSCONFIG page and perform the following steps under the *Build Options* section:
  - Select *Lab 3: Single Channel Open-Loop CV Control* for the lab
  - Select any of the four channels
  - Enable the SFRA
  - Save the page
3. When using the non-powerSuite version of the project, the above settings are directly modified in the `solution_settings.h` file.

```
#define LAB_NUMBER (3)
#define CHANNEL_NUMBER (3)
#define SFRA_ENABLED (true)
```






**Figure 3-22. Build Option for Lab 3**


### 3.3.5.3.2 Building and Loading the Project and Setting up Debug Environment

Use the following steps to build and load the project and to set up the debug environment.

1. Right-click on the project name and click **Rebuild Project**.
2. The project builds successfully.
3. In the Project Explorer, make sure the correct target configuration file is set as Active under targetConfigs.
4. Click **Run** → **Debug** to launch a debugging session.
5. The project then loads on the device and the CCS debug view becomes active. The code halts at the start of the main routine.
6. To add the variables in the watch/expressions window, click **View** → **Scripting Console** to open the scripting console dialog box. On the upper right corner of this console, click on *Open* and then browse to **setupdebugenv\_chX.js** the script file located inside the project folder. This populates the watch window with the appropriate variables needed to debug the system.
7. Click on the **Continuous Refresh** button  on the watch window to enable continuous update of values from the controller.

### 3.3.5.3.3 Running the Code

Use the following steps to run the code for Lab 3:

1. Use the test setup shown in [Section 3.3.1](#).
2. Run the project by clicking  from the menu bar.
3. In the watch view, check if BT4CH\_InputVoltageSense\_V is from 12V to 15V in the *Expression Window*.
4. Set the following parameters from the *Expression Window*:
  - Set BT4CH\_userParam\_chX->Relay\_ON to 1 to enable the output relay
  - BT4CH\_userParam\_V\_I\_chm->iref\_A = 15.0
  - Set the BT4CH\_userParam\_chX->en\_bool = 1
  - See [Figure 3-23](#) for the *Expression Window* settings
5. The BT4CH\_measureVI\_chX variable shows output current and voltage of the DC/DC converter. IbatSense\_A display value is close to iref\_A with ±1mA error.
6. [Figure 3-24](#) shows the SFRA setup to measure Open-Loop Voltage Control Frequency Response.
7. Click on the *Run SFRA* icon from the SYSCONFIG page. The SFRA GUI pops up.
8. Select the options for the device on the SFRA GUI; for example, for F28P65x, select *Floating Point*. Click on the *Setup Connection* button. In the pop-up window, uncheck the boot-on-connect option and select an appropriate COM port. Click the *OK* button. Return to the SFRA GUI and click the *Connect* button.

9. The SFRA GUI connects to the device. A SFRA sweep can now be started by clicking the *Start Sweep* button. The complete SFRA sweep takes a few minutes to finish. Once complete, a graph with the measurement appears, as shown in [Figure 3-25](#).
10. The Frequency Response Data is saved in the project folder, under an SFRA Data Folder, and is time-stamped with the time of the SFRA run.
11. After finishing the lab, set the following parameters from the *Expression Window* to stop the code:
  - BT4CH\_userParam\_V\_I\_chm->iref\_A = 0
  - Set the BT4CH\_userParam\_chX->en\_bool = 0
  - Set the BT4CH\_userParam\_chX->Relay\_ON to 0 to disable the output relay
  - Terminate the program

BT4CH_calibrationMode	enum <unname...	Disabled
BT4CH_startupMode	enum <unname...	DCM_Startup
BT4CH_HAL_InputVoltageSense_V	float	12.4089355
BT4CH_ISR2_Loading	float	0.149675161
BT4CH_ISR2_LoadingMax	float	0.174261391
BT4CH_userParam_ch3	struct <unname...	{Iref_A=7.0,VrefCharge_V=1.0,VrefDi...
iref_A	float	7.0
VrefCharge_V	float	1.0
VrefDischarge_V	float	3.0
dir_bool	unsigned int	1
Relay_ON	unsigned int	1
Start_bool	unsigned int	1
remote_sense	unsigned int	0
DutyRef_pu	float	0.0
IbatCal_pu	float	0.200000003
VbatCal_pu	float	0.400000006
IoutGain_pu	float	0.0730994046
IoutOffset_pu	float	0.00584799051
IoutGain_A	float	13.6800022
IoutOffset_A	float	-0.0800005198
VoutGain_pu	float	0.24248305
VoutOffset_pu	float	-0.000581979752
VoutGain_V	float	4.1239996
VoutOffset_V	float	0.00240008417
VbatGain_pu	float	0.200000018
VbatOffset_pu	float	-0.000600039959
VbatGain_V	float	4.99999952
VbatOffset_V	float	0.00300019956
BT4CH_measureVI_ch3	struct <unname...	{Ibat_A=7.00024748,Vout_V=0.1169..
Ibat_A	float	7.00016308
Vout_V	float	0.116786078
Vbat_V	float	0.0993618295

Figure 3-23. Lab 3 Expression Window, Closed Loop

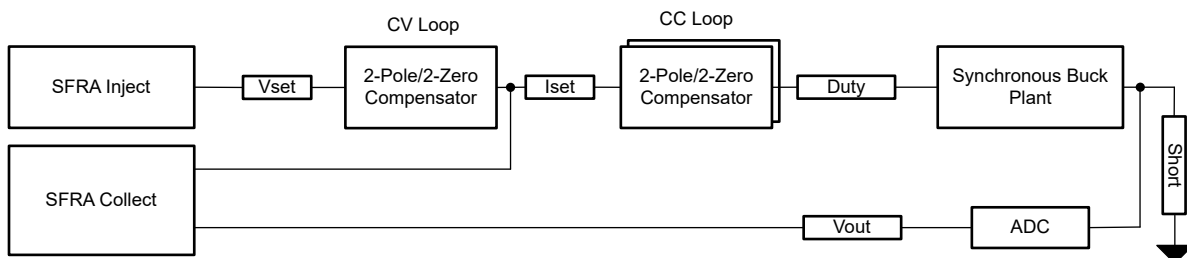


Figure 3-24. SFRA Setup for Open-Loop Voltage Control

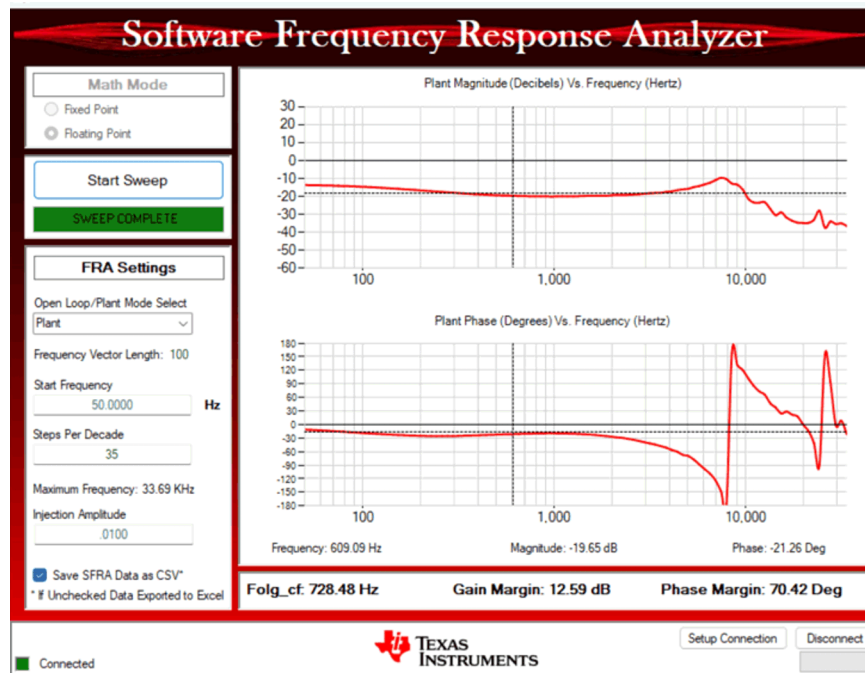


Figure 3-25. Voltage Control Open-Loop Frequency Response

### 3.3.5.4 Lab 4. Closed Loop Current and Voltage Control Single Channel

#### 3.3.5.4.1 Setting Software Options for Lab 4

1. To run this lab, make sure the hardware is set up as outlined in the previous section, [Section 3.3.2](#)
2. Open the CCS project as outlined in [Section 3.2.1](#). If using the powerSUITE, go to [Step 3](#), otherwise, jump to [Step 4](#).
3. Open the SYSCONFIG page and select under the *Build Options* section:
  - Select *Lab 4: Closed Loop CCCV Single Channel*
  - Select the Channel
  - Enable the SFRA
  - Open the Compensation Designer
4. When using non-powerSuite version of the project, *Build Settings* are directly modified in solution\_settings.h file. Compensation Designer is found at C2000Ware\_DigitalPower\_Install\_Location\powerSUITE\source\utils.

```

#define LAB_NUMBER (4)
#define CHANNEL_NUMBER (3)
#define SFRA_ENABLED (true)
  
```

Build Options	
Lab	4: Closed Loop CCCV Single Channel
Channel Enabled	Ch 3
Feedback Loop	CCCV_Loop
Comp Style	DCL_DF22
SFRA Enable	True
Calibration Mode Current/Voltage	Disabled
HW OCP Enable/ Disable	Disabled
HW OCP Limit in Per Unit	0.95
SW OCP Limit (A)	12
PWM Switching Frequency (kHz)	250
Dead Time (ns)	55
Startup	DCM
DCM Startup Time (ms)	0
Default IREF (A)	3
Default VREF Charge (V)	2
Default VREF Discharge (V)	2
ISR Code Profiling	Disabled
Control Loop ISR Frequency (kHz)	50
Software Frequency Response Analyzer	<input type="button" value="RUN SFRA"/>
Compensation Designer	<input type="button" value="RUN COMPENSATION DESIGNER"/>

Figure 3-26. Build Options for Lab 4

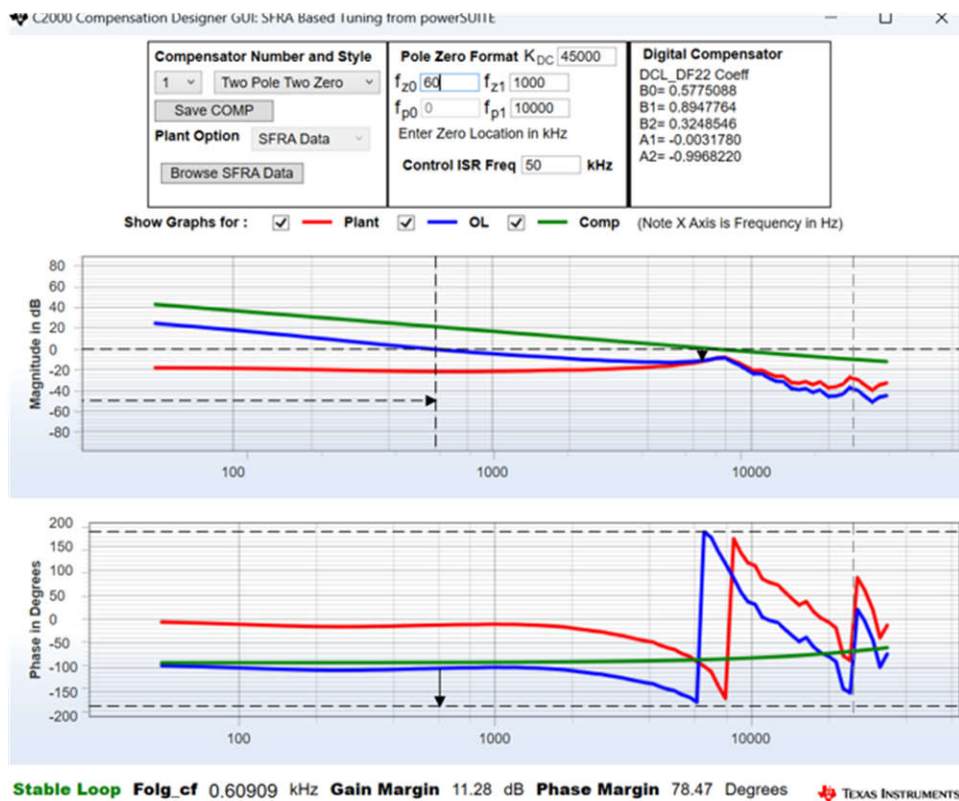



Figure 3-27. Tuning the Voltage Loop


### 3.3.5.4.2 Building and Loading the Project and Setting up Debug Environment

1. Right-click on the project name and click **Rebuild Project**.
2. The project builds successfully.
3. In the Project Explorer, make sure the correct target configuration file is set as *Active* under targetConfigs.
4. Then, click **Run** → **Debug** to launch a debugging session.
5. The project then loads on the device and the CCS debug view becomes active. The code halts at the start of the main routine.

6. To add the variables in the watch/expressions window, click **View** → **Scripting Console** to open the scripting console dialog box. On the upper right corner of this console, click on *open* and then browse to the **setupdebugenv\_chX.js** script file located inside the project folder. This populates the watch window with the appropriate variables needed to debug the system.
7. Click on the **Continuous Refresh** button  on the watch window to enable continuous update of values from the controller.

### 3.3.5.4.3 Running the Code

Use the following steps to run the code for Lab 4:

1. Use the test setup shown in [Section 3.3.1](#).
2. Run the project by clicking  from the menu bar.
3. In the watch view, check if the BT4CH\_InputVoltageSense\_V is from 12V to 15V in the *Expression Window*.
4. Set the following parameters from the *Expression Window*:
  - Set the BT4CH\_userParam\_chX->Relay\_ON to 1 to enable the output relay
  - BT4CH\_userParam\_chX->iref\_A = 8.5
  - BT4CH\_userParam\_chX->vrefCharge\_V = 0.075
  - Set the BT4CH\_userParam\_chX->en\_bool = 1
  - See [Figure 3-28](#) for the *Expression Window* settings
5. The BT4CH\_measureVI\_chX variable shows the output current and voltage of the DC/DC converter. Vbatsense\_V display value is close to vrefCharge\_V with  $\pm 0.5\text{mV}$  error.
6. [Figure 3-29](#) shows the SFRA setup to measure Closed-Loop Voltage Control Frequency Response.
7. Click on the *Run SFRA* icon from the SYSCONFIG page. The SFRA GUI pops up
8. Select the options for the device on the SFRA GUI; for example, for F28P65x, select *Floating Point*. Click the *Setup Connection* button. In the pop-up window, uncheck the boot-on-connect option and select an appropriate COM port. Click the OK button. Return to the SFRA GUI and select the *Connect* button.
9. The SFRA GUI connects to the device. An SFRA sweep can now be started by clicking the *Start Sweep* button. The complete SFRA sweep takes a few minutes to finish. Once complete, a graph with the measurement appears, as shown in [Figure 3-30](#).
10. The Frequency Response Data is saved in the project folder, under an SFRA Data Folder, and is time-stamped with the time of the SFRA run.
11. After finishing the lab, set the following parameters from the *Expression Window* to stop the code:
  - BT4CH\_userParam\_chX->iref\_A = 0
  - BT4CH\_userParam\_chX->vrefCharge\_V = 0
  - Set the BT4CH\_userParam\_chX->en\_bool = 0
  - Set the BT4CH\_userParam\_chX->Relay\_ON to 0 to disable the output relay
  - Terminate the program

Expression	Type	Value
BT4CH_calibrationMode	enum <unname...	Disabled
BT4CH_startupMode	enum <unname...	DCM_Startup
BT4CH_HAL_InputVoltageSense_V	float	12.4133673
BT4CH_ISR2_Loading	float	0.165582791
BT4CH_ISR2_LoadingMax	float	0.179018527
BT4CH_userParam_ch3	struct <unname...	{Iref_A=8.5,VrefCharge_V=0.075000...
Iref_A	float	8.5
VrefCharge_V	float	0.075000003
VrefDischarge_V	float	3.0
dir_bool	unsigned int	1
Relay_ON	unsigned int	1
Start_bool	unsigned int	1
remote_sense	unsigned int	1
DutyRef_pu	float	0.0
IbatCal_pu	float	0.200000003
VbatCal_pu	float	0.400000006
IoutGain_pu	float	0.0730994046
IoutOffset_pu	float	0.00584799051
IoutGain_A	float	13.6800022
IoutOffset_A	float	-0.0800005198
VoutGain_pu	float	0.24248305
VoutOffset_pu	float	-0.000581979752
VoutGain_V	float	4.1239996
VoutOffset_V	float	0.00240008417
VbatGain_pu	float	0.200000018
VbatOffset_pu	float	-0.000600039959
VbatGain_V	float	4.99999952
VbatOffset_V	float	0.00300019956
BT4CH_measureVI_ch3	struct <unname...	{Ibat_A=5.27749491,Vout_V=0.0879...
Ibat_A	float	5.27365923
Vout_V	float	0.0880440623
Vbat_V	float	0.0750049874

Figure 3-28. Lab 4 Expression Window, Closed Loop

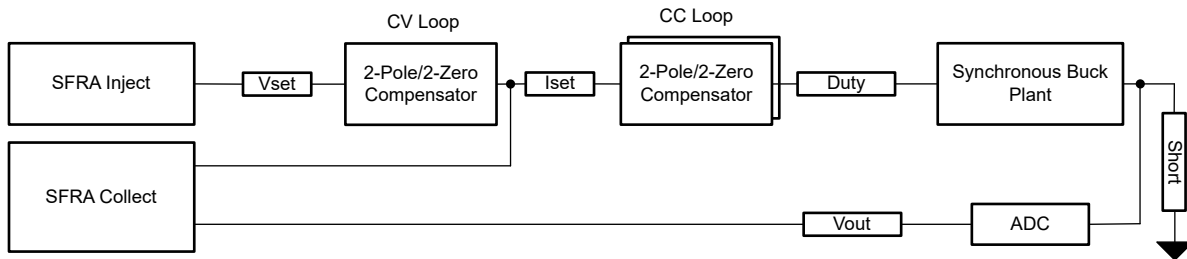
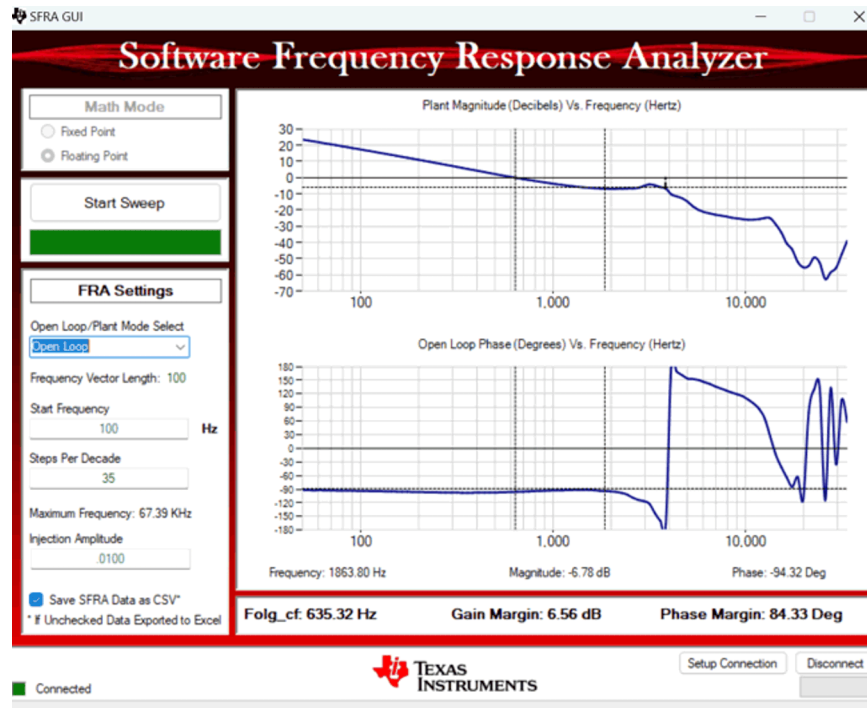


Figure 3-29. SFRA Setup for Closed-Loop Voltage Control



**Figure 3-30. Voltage Control Closed-Loop Frequency Response**

### 3.3.5.5 Lab 5. Closed Loop Current and Voltage Control Four Channels

#### 3.3.5.5.1 Setting Software Options for Lab 5

1. To run this lab, make sure the hardware is set up as outlined in the previous section, [Section 3.3.2](#).
2. Open the CCS project as outlined in [Section 3.2.1](#). If using the powerSUITE, go to [Step 3](#), otherwise, jump to [Step 4](#).
3. Open the SYSCONFIG page and select under the *Build Options* section:
  - Select *Lab 5: Closed-Loop CCCV All Channels for the Lab*
  - Select all Channels
  - Disable the SFRA option
  - Open the Compensation Designer
4. When using non-powerSuite version of the project, *Build Settings* are directly modified in `solution_settings.h` file.

```


#define LAB_NUMBER (5)
#define CHANNEL_NUMBER (5)
#define SFRA_ENABLED (false)
  
```

Build Options	
Lab	5: Closed Loop CCCV All Channels
Channel Enabled	Ch All
Feedback Loop	-
Comp Style	-
SFRA Enable	False
Calibration Mode Current/Voltage	Disabled
HW OCP Enable/ Disable	Disabled
HW OCP Limit in Per Unit	0.95
SW OCP Limit (A)	12
PWM Switching Frequency (kHz)	250
Dead Time (ns)	55
Startup	DCM
DCM Startup Time (ms)	0
Default IREF (A)	3
Default VREF Charge (V)	2
Default VREF Discharge (V)	2
ISR Code Profiling	Disabled
Control Loop ISR Frequency (kHz)	50

**Figure 3-31. Build Options for Lab 5**


### 3.3.5.5.2 Building and Loading the Project and Setting up Debug Environment

Use the following steps to build and load the project and to set up the debug environment.

1. Right-click on the project name and click **Rebuild Project**.
2. The project builds successfully.
3. In the Project Explorer, make sure the correct target configuration file is set as Active under targetConfigs.
4. Click **Run** → **Debug** to launch a debugging session.
5. The project then loads on the device and the CCS debug view becomes active. The code halts at the start of the main routine.
6. To add the variables in the watch/expressions window, click **View** → **Scripting Console** to open the scripting console dialog box. On the upper right corner of this console, click on **Open** and then browse to **setupdebugenv\_chX.js** the script file located inside the project folder. This populates the watch window with the appropriate variables needed to debug the system.
7. Click on the **Continuous Refresh** button  on the watch window to enable continuous update of values from the controller.

### 3.3.5.5.3 Running the Code

Use the following steps to run the code for Lab 5:

1. Use the test setup shown in [Section 3.3.1](#).
2. Run the project by clicking  from the menu bar.
3. In the watch view, check if the BT4CH\_InputVoltageSense\_V is from 12V to 15V in the *Expression Window*.
4. Set the following parameters from the *Expression Window*:
  - Set the BT4CH\_userParam\_chX->Relay\_ON to 1 to enable the output relay
  - BT4CH\_userParam\_chX->iref\_A = 5.0
  - BT4CH\_userParam\_chX->vrefCharge\_V = 3
  - Set the BT4CH\_userParam\_chX->en\_bool = 1
  - See [Figure 3-32](#) for the *Expression Window* settings
5. The BT4CH\_measureVI\_chX variable shows output current and voltage of the DC/DC converter.
6. Change the Iref and Vref to see the transition between constant current and constant voltage modes.
7. To change the current direction, toggle BT4CH\_userParam\_chX->Dir\_bool.
8. After finishing the lab, set the following parameters from the *Expression Window* to stop the code:
  - BT4CH\_userParam\_chX->iref\_A = 0
  - BT4CH\_userParam\_chX->vrefCharge\_V = 0
  - Set the BT4CH\_userParam\_chX->en\_bool = 0
  - Set the BT4CH\_userParam\_chX->Relay\_ON to 0 to disable the output relay
  - Terminate the program



- Repeat on all the channels until every channel is shut down

Expression	Type	Value
BT4CH_lab	enum <unname...	Lab5_AllChannelClosedLoopCV
BT4CH_sfraStatus	enum <unname...	SFRA_Disabled
BT4CH_calibrationStatus	enum <unname...	Calibration_Disabled
BT4CH_calibrationMode	enum <unname...	Disabled
BT4CH_startupMode	enum <unname...	DCM_Startup
BT4CH_HAL_InputVoltageSense_V	float	13.2465382
BT4CH_ISR2_Loading	float	0.280758649
BT4CH_ISR2_LoadingMax	float	0.292939395
BT4CH_userParam_ch1	struct <unname...	(Iref_A=5.0,VrefCharge_V=2.0,VrefDi...
Iref_A	float	5.0
VrefCharge_V	float	2.0
VrefDischarge_V	float	3.0
dir_bool	unsigned int	1
Relay_ON	unsigned int	1
Start_bool	unsigned int	1
remote_sense	unsigned int	0
DutyRef_pu	float	0.0
IbatCal_pu	float	0.200000003
VbatCal_pu	float	0.400000006
IoutGain_pu	float	0.0789889395
IoutOffset_pu	float	0.00829383731
IoutGain_A	float	12.6600008
IoutOffset_A	float	-0.104999982
VoutGain_pu	float	0.24248305
VoutOffset_pu	float	-0.000581979752
VoutGain_V	float	4.1239996
VoutOffset_V	float	0.00240008417
VbatGain_pu	float	0.200000018
VbatOffset_pu	float	-0.000600039959
VbatGain_V	float	4.99999952
VbatOffset_V	float	0.00300019956
BT4CH_measureVI_ch1	struct <unname...	(Ibat_A=-0.0642971396,Vout_V=1.6...
Ibat_A	float	-0.0635485798

Figure 3-32. Lab 5 Expression Windows

### 3.3.5.6 Calibration

- To run this lab, make sure the hardware is set up as shown in [Section 3.3.3](#). The two-point calibration method is used to calibrate gain and offset errors.
- There are three ways to measure current:
  - Use an external precision resistor and a 6.5-digit DMM in voltage mode measure the voltage drop across the resistor, then the current can be calculated.
  - Measure the voltage across the sense resistors on the TIDA-010086 boards
  - Probe Tp1 and Tp2 on the board to measure the voltage drop across the shunt
  - Use e-load readings, but this approach requires high-accuracy electronic load or source measure unit (SMU) equipment.
- To measure voltage, use a DMM across the buck converter output voltage and remote sense Terminal J8
- Open the SYSCONFIG page, select Lab 5, and set *Calibration Mode* to *Current Calibration*. [Figure 3-33](#) shows the SYSCONFIG page setting for current calibration.
  - Save the SYSCONFIG page, and run the code.
  - Open the *Expression Window*.
  - The output current is updated using BT4PH\_userParam\_V\_I\_chX->ibatCal\_pu parameter.
  - Set the BT4CH\_userParam\_chX->Relay\_ON to 1 to enable the output relay.
  - Set the BT4CH\_userParam\_chX->en\_bool = 1.
  - Set the BT4CH\_userParam\_chX->ibatCal\_pu to "0.3" and "0.5", and note the output current readings.

- Update the actual output current readings in bt4ch\_gan\_cal.h file.

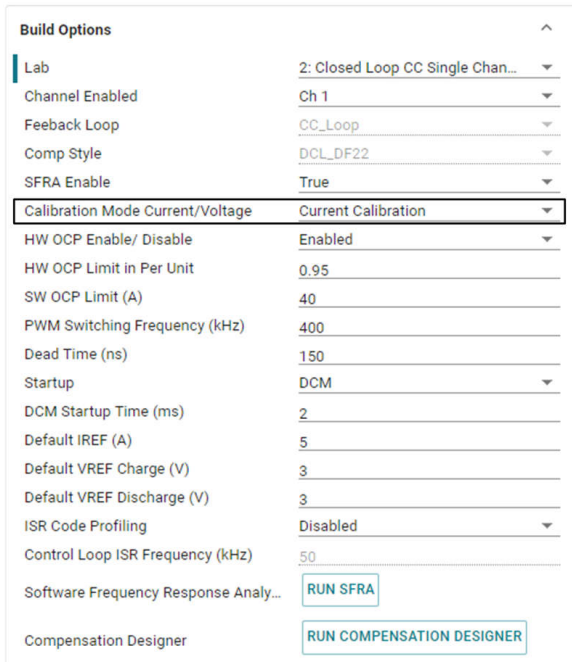
```
#define BT4CH_IBAT_ACTUAL_CH1_P1_A ((float32_t)3.59)
#define BT4CH_IBAT_ACTUAL_CH1_P2_A ((float32_t)6.02)
```

- Repeat the steps for channel 2, 3 and 4.
- Open the SYSCONFIG page, select Lab 5, and set *Calibration Mode* to *Voltage Calibration*. [Figure 3-34](#) shows the SYSCONFIG page setting for voltage calibration.
    - Save the SYSCONFIG page, and run the code.
    - Open the *Expression Window*.
    - The output current is updated using BT4PH\_userParam\_V\_I\_chX->vbatCal\_pu parameter.
    - Set the *BT4CH\_userParam\_chX->Relay\_ON* to 1 to enable the output relay.
    - Set the *BT4CH\_userParam\_chX->en\_bool* = 1.
    - Set the *BT4CH\_userParam\_chX->vbatCal\_pu* to "0.2" and "0.6", and note the output current readings. Update the actual output current readings in bt4ch\_cal.h file.

```
#define BT4CH_VBAT_ACTUAL_CH1_P1_V ((float32_t)0.9976)
#define BT4CH_VBAT_ACTUAL_CH1_P2_V ((float32_t)2.998)
```

- Repeat the steps for channel 2, 3 and 4.
- After calibration finishes, stop the project and open the SYSCONFIG page, disable the calibration mode.
  - When using non-powerSuite version of the project, *Build Settings* are directly modified in the solution\_settings.h file. Set CALIBRATION\_MODE to (1) for current calibration, and (2) for voltage calibration.

```
#define LAB_NUMBER (5)
#define CHANNEL_NUMBER (5)
#define CALIBRATION_ENABLED (true)
#define CALIBRATION_MODE (1)
```



Build Options

Lab: 2: Closed Loop CC Single Chan...

Channel Enabled: Ch 1

Feedback Loop: CC\_Loop

Comp Style: DCL\_DF22

SFRA Enable: True

Calibration Mode Current/Voltage: Current Calibration

HW OCP Enable/ Disable: Enabled

HW OCP Limit in Per Unit: 0.95

SW OCP Limit (A): 40

PWM Switching Frequency (kHz): 400

Dead Time (ns): 150

Startup: DCM

DCM Startup Time (ms): 2

Default IREF (A): 5

Default VREF Charge (V): 3

Default VREF Discharge (V): 3

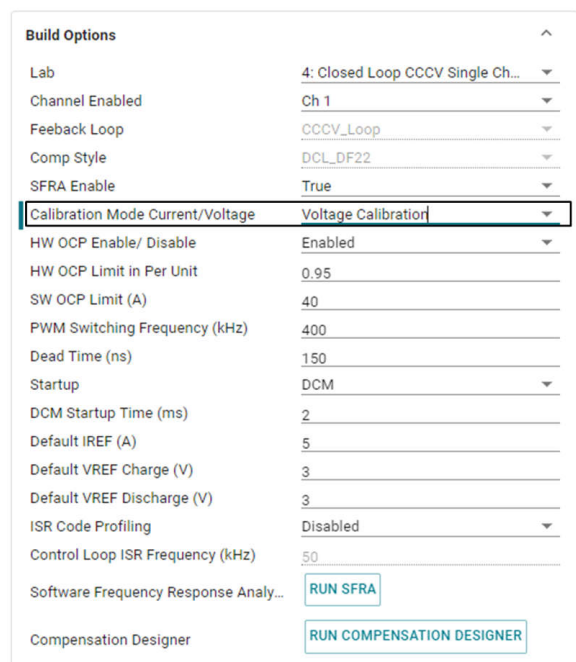
ISR Code Profiling: Disabled

Control Loop ISR Frequency (kHz): 50

Software Frequency Response Analy...: RUN SFRA

Compensation Designer: RUN COMPENSATION DESIGNER

Figure 3-33. Build Options for Current Calibration



Build Options

Lab: 4: Closed Loop CCCV Single Ch...

Channel Enabled: Ch 1

Feedback Loop: CCCV\_Loop

Comp Style: DCL\_DF22

SFRA Enable: True

Calibration Mode Current/Voltage: Voltage Calibration

HW OCP Enable/ Disable: Enabled

HW OCP Limit in Per Unit: 0.95

SW OCP Limit (A): 40

PWM Switching Frequency (kHz): 400

Dead Time (ns): 150

Startup: DCM

DCM Startup Time (ms): 2

Default IREF (A): 5

Default VREF Charge (V): 3

Default VREF Discharge (V): 3

ISR Code Profiling: Disabled

Control Loop ISR Frequency (kHz): 50

Software Frequency Response Analy...: RUN SFRA

Compensation Designer: RUN COMPENSATION DESIGNER

Figure 3-34. Build Options for Voltage Calibration

### 3.4 Test Results

#### 3.4.1 Current Load Regulation

Use the setup in [Section 3.3.3](#) to test the current load regulation.

**Table 3-2. Current Load Regulation**

FSR (A)	10							
OUTPUT MODE	CHARGING				DISCHARGING			
ISET (A)	0.1	1	5	10	0.1	1	5	10
E-LOAD CV MODE	TERMINAL VOLTAGE READING							
VSET(V)	I <sub>actual</sub> (A)	I <sub>actual</sub> (A)	I <sub>actual</sub> (A)	I <sub>actual</sub> (A)	I <sub>actual</sub> (A)	I <sub>actual</sub> (A)	I <sub>actual</sub> (A)	I <sub>actual</sub> (A)
1	0.0996	0.9998	4.99993	10.0005	-0.1005	-1.0001	-5.001	-10.001
2	0.0995	0.9996	4.99995	10.0008	-0.1002	-1.00025	-5.0011	-10.001
3	0.09985	0.9995	4.99995	10.0008	-0.1005	-1.0002	-5.001	-10.0008
4	0.09925	0.9995	4.99995	10.0008	-0.1005	-1.0002	-5.001	-10.0011
Error (mA)	0.7500	0.5	0.05	-0.8	-0.5	-0.25	-1.1	-1.1
Error (%FSR)	0.0075	0.0050	0.0005	0.0080	0.0050	0.0025	0.0110	0.0110

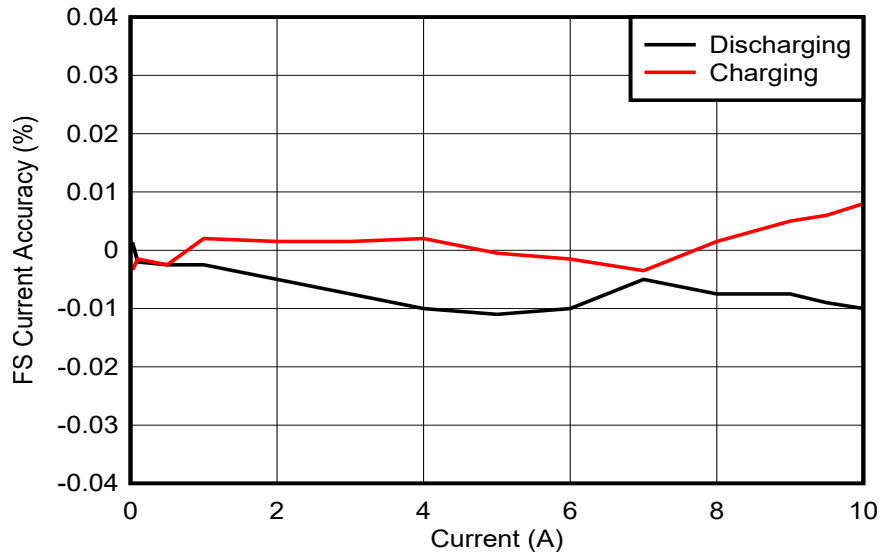
#### 3.4.2 Voltage Load Regulation

**Table 3-3. Voltage Load Regulation**

FSR (V)	5					
VSET(V)	0.2	1	2	3	4	5
E-LOAD CC MODE	CURRENT READING					
ISET(A)	V <sub>actual</sub> (V)	V <sub>actual</sub> (V)	V <sub>actual</sub> (V)	V <sub>actual</sub> (V)	V <sub>actual</sub> (V)	V <sub>actual</sub> (V)
No load	0.20003	1.00001	2.00001	2.99997	3.99994	4.99991
1	0.20002	0.99998	1.99998	2.99998	3.99994	5.0001
4	0.20002	1.0002	2.00004	3.00006	3.99997	5.00006
8	0.20002	1.00005	2.00002	3.00002	3.99993	5.00002
10	0.20002	1.00004	2.00002	3.00004	3.99994	5.00002
Error (mV)	0.0300	0.0500	0.0400	0.0600	-0.0700	-0.0900
Error (%FSR)	0.0003	0.0005	0.0004	0.0006	0.0007	0.0009

### 3.4.3 Current Linearity Test

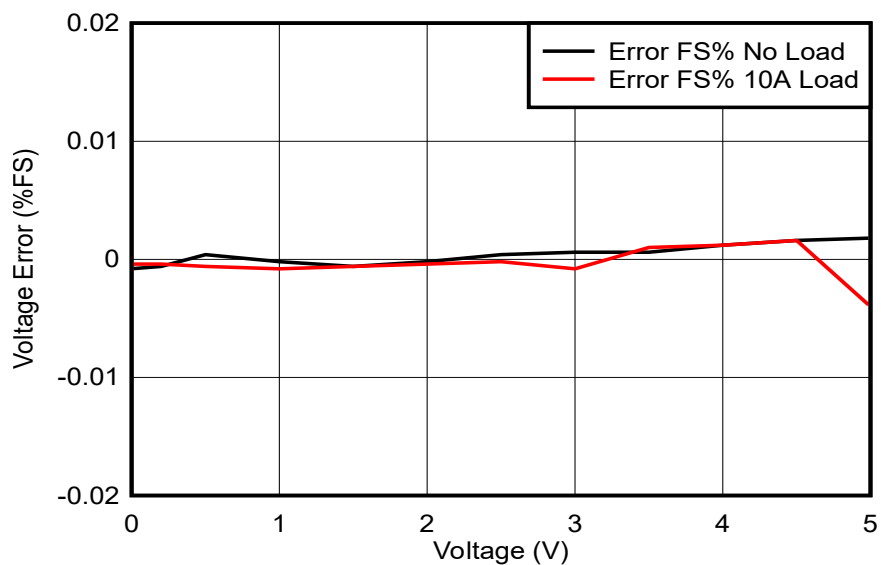
The current control accuracy depends on the current-sensing resistor, the gain, offset, and drift of the current sense amplifier. The test applies two-point calibration in charging mode. The overall current error remains below  $\pm 0.02\%$  after calibration. [Figure 3-35](#) shows the results.



**Figure 3-35. Current Control Accuracy Test**

### 3.4.4 Voltage Loop Linearity Test

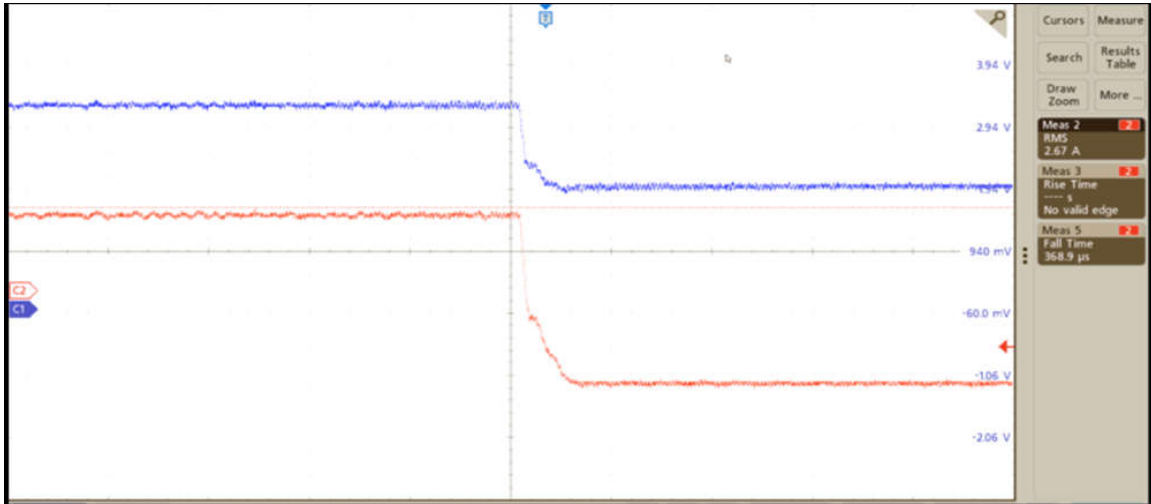
The voltage loop calibration occurs under no-load conditions. The black curve in [Figure 3-36](#) shows voltage control linearity under no-load conditions, and the red curve presents voltage accuracy under 10A load conditions. Testing proves that voltage loop regulation error remains below  $\pm 0.01\%$  under different output load conditions.



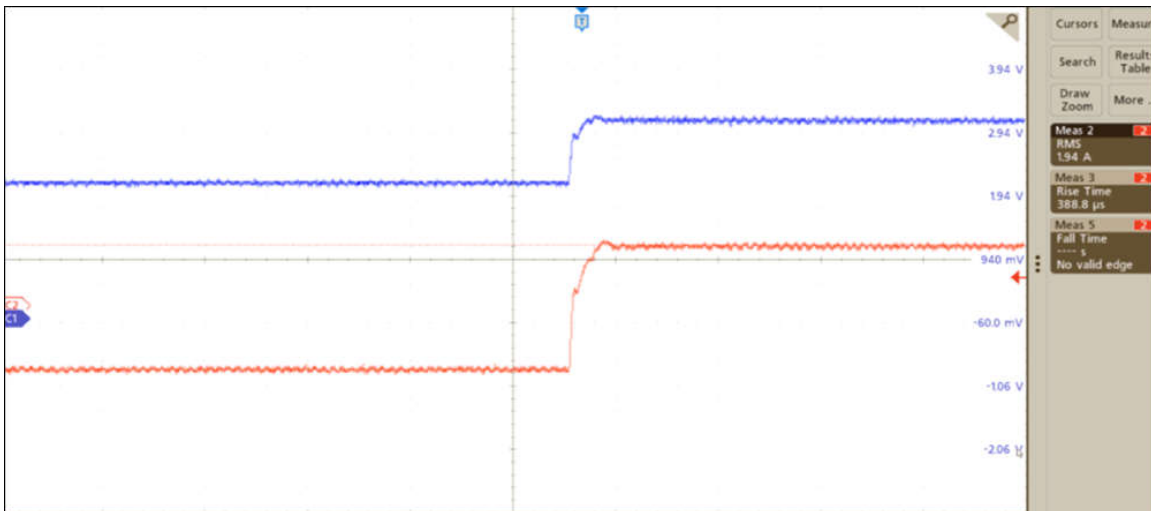
**Figure 3-36. Voltage Control Linearity Test**

### 3.4.5 Bidirectional Current Switching Time

For both charge and discharge transitions (see [Figure 3-37](#) and [Figure 3-38](#)), the current response time remains fast and smooth. With proper tuning, the transient response time for switching current operation can reach within 400µs.



**Figure 3-37. Current Transition, Charge to Discharge**



**Figure 3-38. Current Transition, Charge to Discharge**

### 3.4.6 Current Step Response

The control loop optimization occurs at 15mΩ output load, achieving within 100μs rise time and fall time. Different battery loads can lead to different results. Optimizing the loop with the battery load remains important to achieve the desired results. See [Figure 3-39](#) and [Figure 3-40](#) for the current step waveforms.

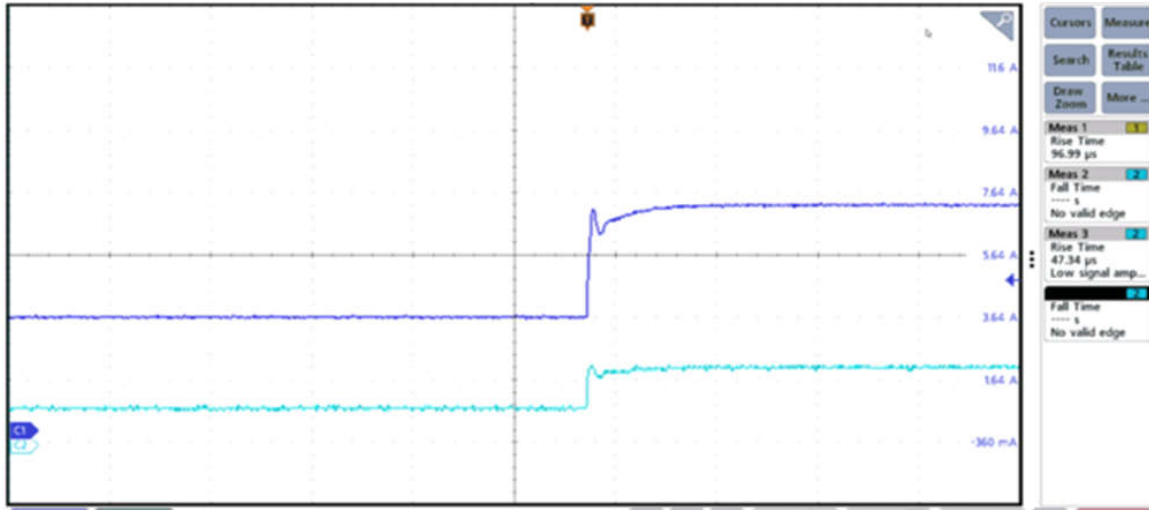


Figure 3-39. Current Step Rise Time

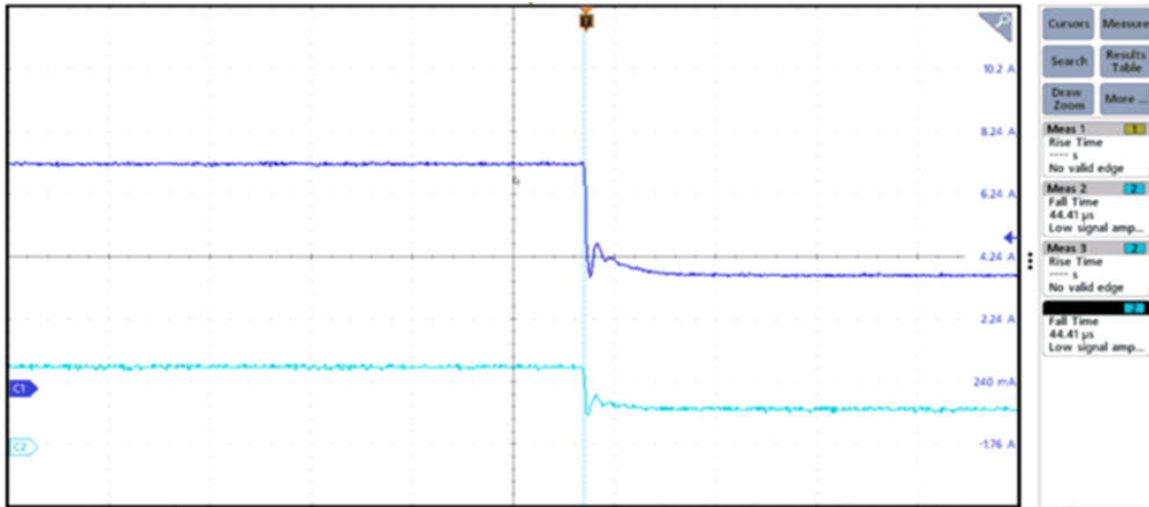


Figure 3-40. Current Step Fall Time

## 4 Design and Documentation Support

### 4.1 Design Files

#### 4.1.1 Schematics

To download the schematics, see the design files at [TIDA-010086](#).

#### 4.1.2 BOM

To download the bill of materials (BOM), see the design files at [TIDA-010086](#).

### 4.2 Tools and Software

#### Tools

[TMDSCNCD28P65X](#) F28P65 controlCARD Evaluation Module

#### Software

[CCSTUDIO](#) Code Composer Studio (CCS) Integrated Development Environment (IDE)

[C2000WARE-DIGITALPOWER-SDK](#) DigitalPower software development kit (SDK) for C2000™ MCUs

### 4.3 Documentation Support

1. Texas Instruments, [TMS320F28P65x Real-Time Microcontrollers Data Sheet](#)
2. Texas Instruments, [ADS9324 16-Channel, 16-Bit, 1MSPS, Simultaneous-Sampling SAR ADC With Integrated Analog Front-End Data Sheet](#)
3. Texas Instruments, [INA630 Precision, 126dB CMRR, Indirect Current Feedback Instrumentation Amplifier Data Sheet](#)

### 4.4 Support Resources

[TI E2E™ support forums](#) are an engineer's go-to source for fast, verified answers and design help — straight from the experts. Search existing answers or ask your own question to get the quick design help you need.

Linked content is provided "AS IS" by the respective contributors. They do not constitute TI specifications and do not necessarily reflect TI's views; see TI's [Terms of Use](#).

### 4.5 Trademarks

TI E2E™, and C2000™ Code Composer Studio™ are trademarks of Texas Instruments. All trademarks are the property of their respective owners.

## 5 About the Author

**ETHAN YU** is a systems engineer at Texas Instruments, where he is responsible for developing reference designs for Test and Measurement applications. Ethan earned his Bachelor of Science degree in electrical engineering from Texas A&M University.

The author thanks SHAURY ANAND, MEGHANA MANAVAZHI, and TIM PRICE for the support given with this reference design.

## 6 Revision History

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

<b>Changes from Revision * (November 2020) to Revision A (December 2025)</b>	<b>Page</b>
• Replaced Li-ion Battery Formation section with <a href="#">Li-ion Cell Formation Equipment</a> .....	2
• Updated <a href="#">Key System Specifications</a> table.....	2
• Updated <a href="#">Figure 2-1</a> .....	3
• Updated <a href="#">Section 2.2</a> .....	4
• Added <a href="#">Output Inductor and Capacitor Selection</a> section.....	5
• Updated <a href="#">Highlighted Products</a> with new devices.....	7
• Updated <a href="#">TIDA-010086 Hardware</a> image.....	8
• Updated <a href="#">Software</a> section.....	9
• Added the <a href="#">Test Procedure</a> section.....	17
• Updated <a href="#">Test Results</a> section.....	35



## IMPORTANT NOTICE AND DISCLAIMER

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

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

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

TI's products are provided subject to [TI's Terms of Sale](#), [TI's General Quality Guidelines](#), or other applicable terms available either on [ti.com](#) or provided in conjunction with such TI products. TI's provision of these resources does not expand or otherwise alter TI's applicable warranties or warranty disclaimers for TI products. Unless TI explicitly designates a product as custom or customer-specified, TI products are standard, catalog, general purpose devices.

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

Copyright © 2025, Texas Instruments Incorporated

Last updated 10/2025