Design Guide: TIDA-010090

4-Channel, 50A, Digital Control Battery Cell Tester Reference Design

Description

The reference design illustrates a method to precisely control the current and voltage of a bidirectional buck converter power stage using a C2000™ real-time microcontroller (MCU) and a precision ADC ADS8588S. The design achieves less than ±10mA current regulation error, and ±1mV voltage regulation error by utilizing a high-resolution pulse-width modulation (PWM) generation peripheral of the C2000 MCU.

Resources

TIDA-010090	Design Folder
LMG3100R017, INA241	Product Folder
TMS320F28P650DK, REF54	Product Folder
ADS8588S, TPSI3050, TVS0500	Product Folder
TPS7A20, TLV1117, LM2664, TPS736	Product Folder
TMDSCNCD28P65X	Tool Folder
C2000WARE-DIGITALPOWER-SDK	Tool Folder

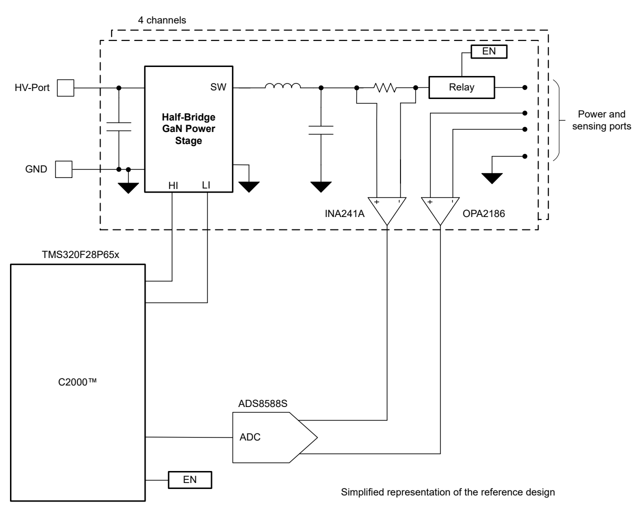
Ask our TI E2E™ support experts

Features

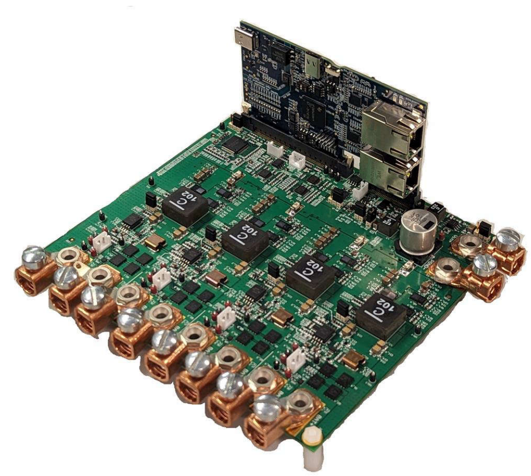
- Four channels, bidirectional buck power stage
- 400kHz switching frequency
- External 16-bit SAR ADC for closed-loop control
- Constant current charging and discharging with regulation error < ±10mA
- Constant voltage mode supported in both charging and discharging with regulation error < ±1mV
- Software Frequency Response Analyzer (SFRA) and compensation designer for ease of tuning of control loops
- powerSUITE support for easy adaptation of design for user requirements

Applications

- Battery cell formation and test equipment
- Programmable DC power supply



Simplified representation of the reference design



1 System Description

The battery tester equipment includes a wide variety of equipment used to test single cells, battery modules, and high-voltage battery packs. The test equipment contains precision power supplies and data acquisition systems, and is used for charging and discharging of batteries, and measures various parameters of the cells.

Figure 1-1 shows a simplified Li-Ion battery manufacturing process. The Final stage, End-of-Line Conditioning, includes cell formation and testing. Formation is a critical step when manufacturing Li-ion cells. During formation, the cells go through a process of initial charge and discharge, which results in the formation of the solid electrolyte interface (SEI) layer. The quality of the SEI layer impacts the capacity and reliability of the battery cell. To control the formation process, precise programmable power supplies are used for charging and discharging of cells. These power supplies are called battery formation systems or battery testers. The accuracy required in battery testers for voltage and current is typically between ±0.02% and ±0.05% of full-scale.

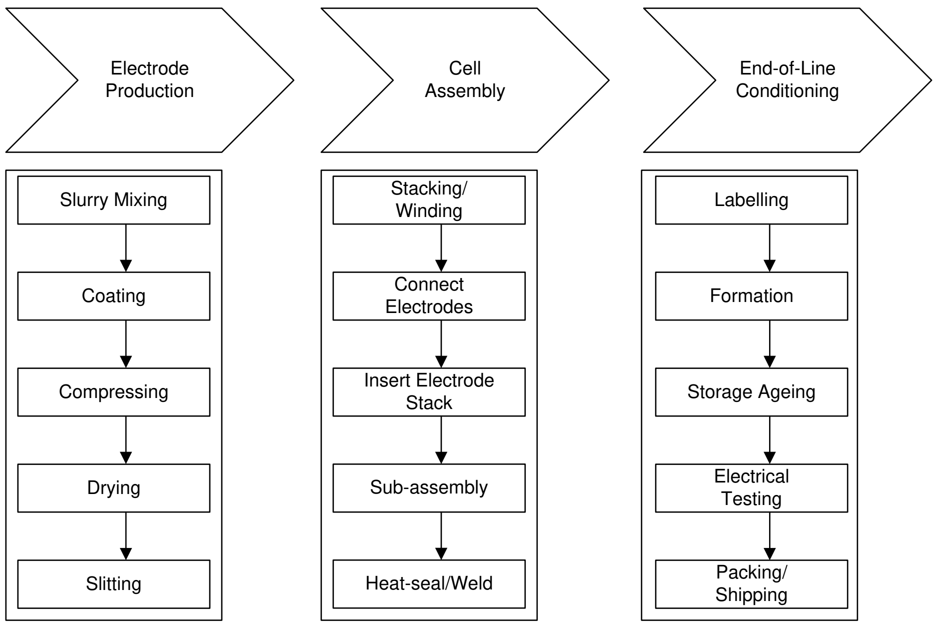


Figure 1-1. Simplified Li-Ion Battery Manufacturing Process

1.1 Key System Specifications

PARAMETER	SPECIFICATION
LV port – Battery port	50mV to 6V
HV port–Bus voltage	12V to 15V
Maximum Output Current (*Without heat sink*)	±20A
Maximum Output Current (*With heat sink*)	±50A
Number of Channels	4
Switching Frequency	400kHz
Control Loop Sample Rate	50kSPS
Current Regulation Error	< ±10mA (0.02% FS)
Voltage Regulation Error	< ±1mV (0.02% FS)

2 System Overview

2.1 Block Diagram

Figure 2-1 is 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 INA241 current sense amplifier senses the battery current and the OPA2186 operational amplifier senses the battery voltage. Current and voltage signals are converted to digital data by the external ADS8588S ADC. C2000 on-chip window comparators are used to implement overcurrent protection.

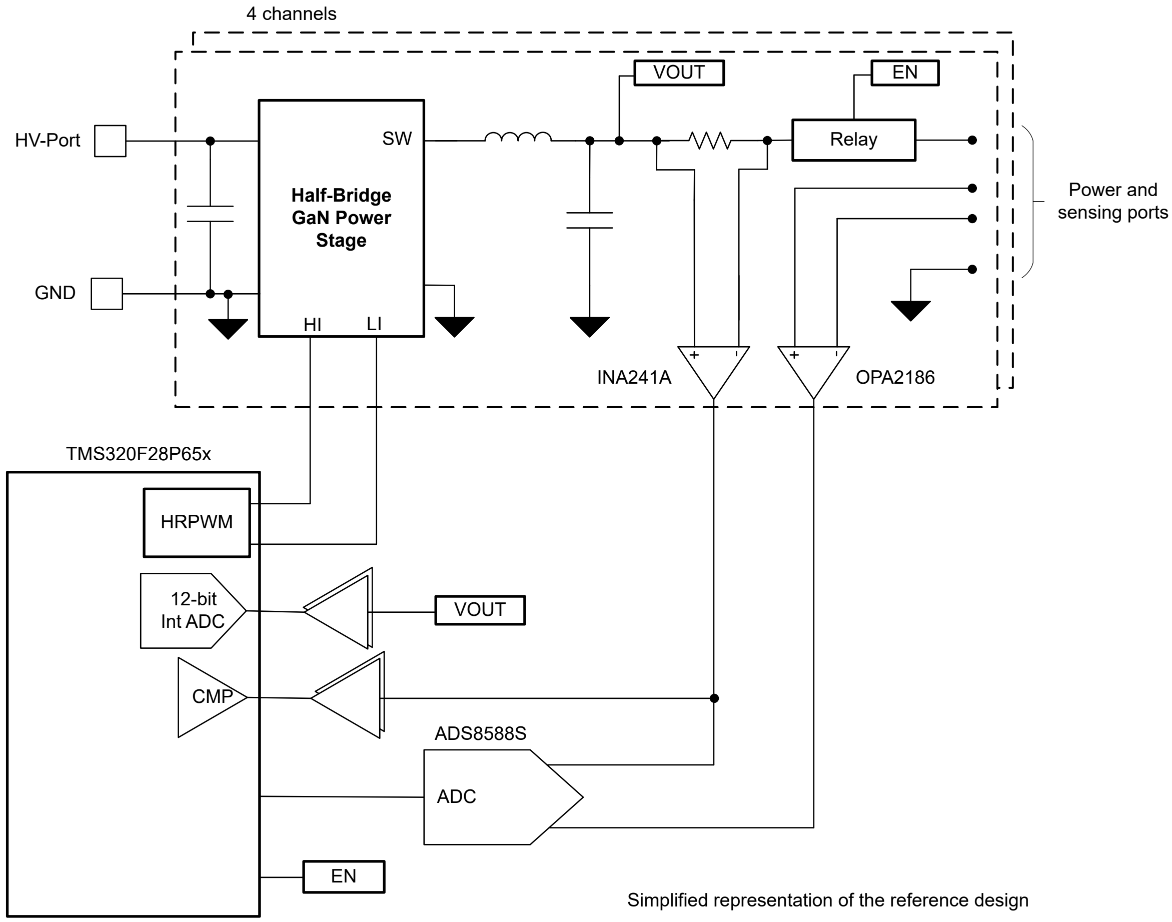


Figure 2-1. TIDA-010090 Block Diagram

2.2 Design Considerations

2.2.1 Current and Voltage 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.

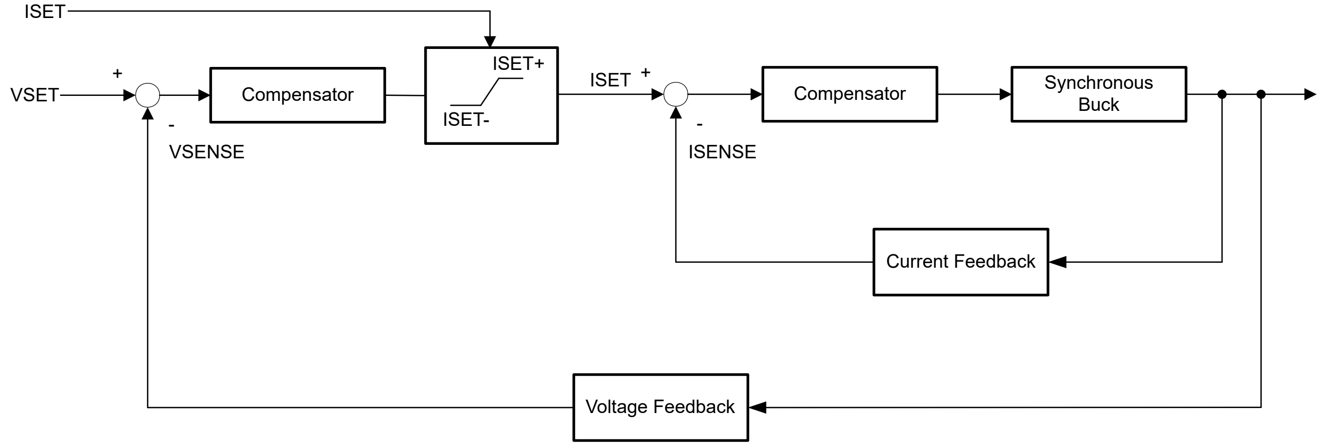


Figure 2-2. Current and Voltage Controller

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.

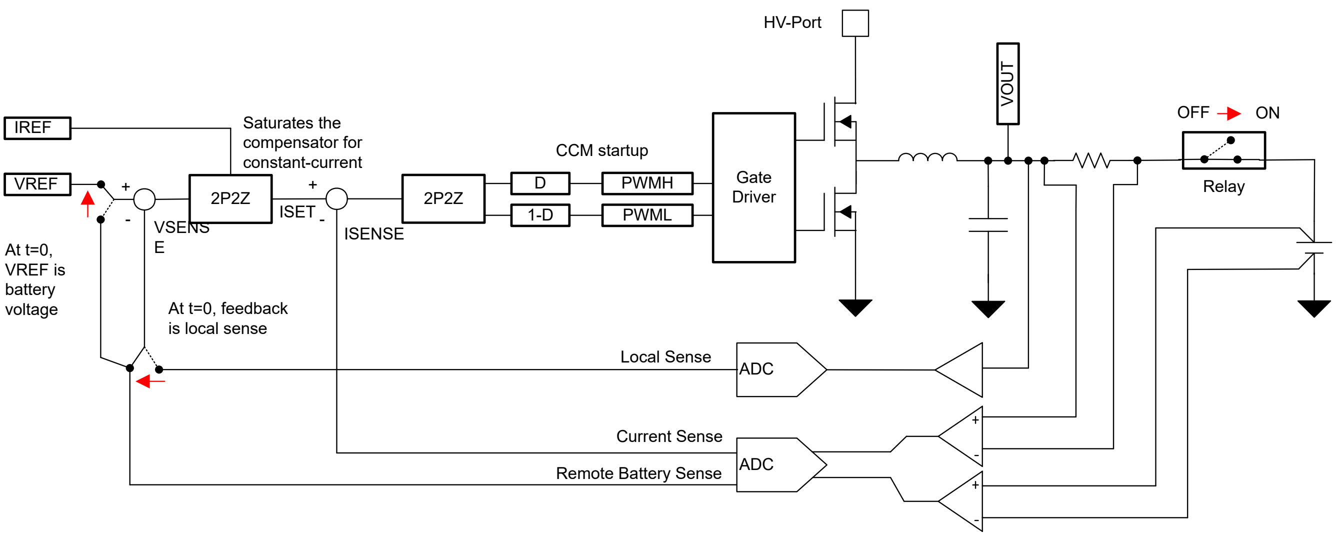


Figure 2-3. Synchronous Start-Up

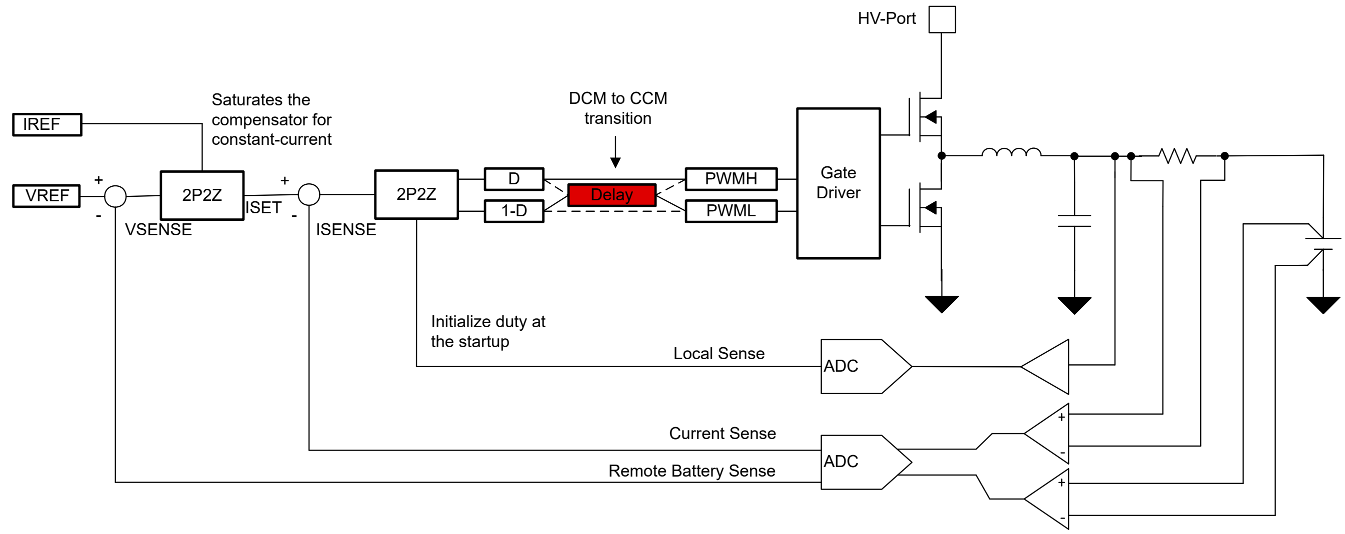


Figure 2-4. Nonsynchronous Start-Up

2.2.3 High-Resolution PWM Generation

To generate high resolution, a C2000 with high-resolution PWM output capability is used. The high-resolution counter is capable of 150ps time step, which is equivalent to 15-bit resolution at 192kHz PWM frequency for a 100MHz CPU clock. Table 2-1 shows PWM resolution at different switching frequencies.

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

PWM FREQUENCY	REGULAR RESOLUTION (PWM)		HIGH RESOLUTION PWM	
	100MHz EPWMCLK			
(kHz)	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

2.3 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 battery test channels or buck converters. See also the *TMS320F28P65x Real-Time Microcontrollers* data sheet.

2.3.2 ADS8588S

The ADS8588S is an eight-channel, simultaneously sampling, 16-bit, SAR, analog-to-digital converter (ADC) that allows maximum sample rates up to 200kSPS and is sufficient for ± 0.01% accuracy and 1kHz loop bandwidth. See also the *ADS8588S 16-Bit, High-Speed, 8-Channel, Simultaneous-Sampling ADC with Bipolar Inputs on a Single Supply* data sheet.

3 Hardware, Software, Testing Requirements, and Test Results

3.1 Hardware Requirements

Figure 3-1 shows the TIDA-010090 hardware. The TIDA-010090 board requires a TSM320F28P65X controlCARD Evaluation Module to operate.

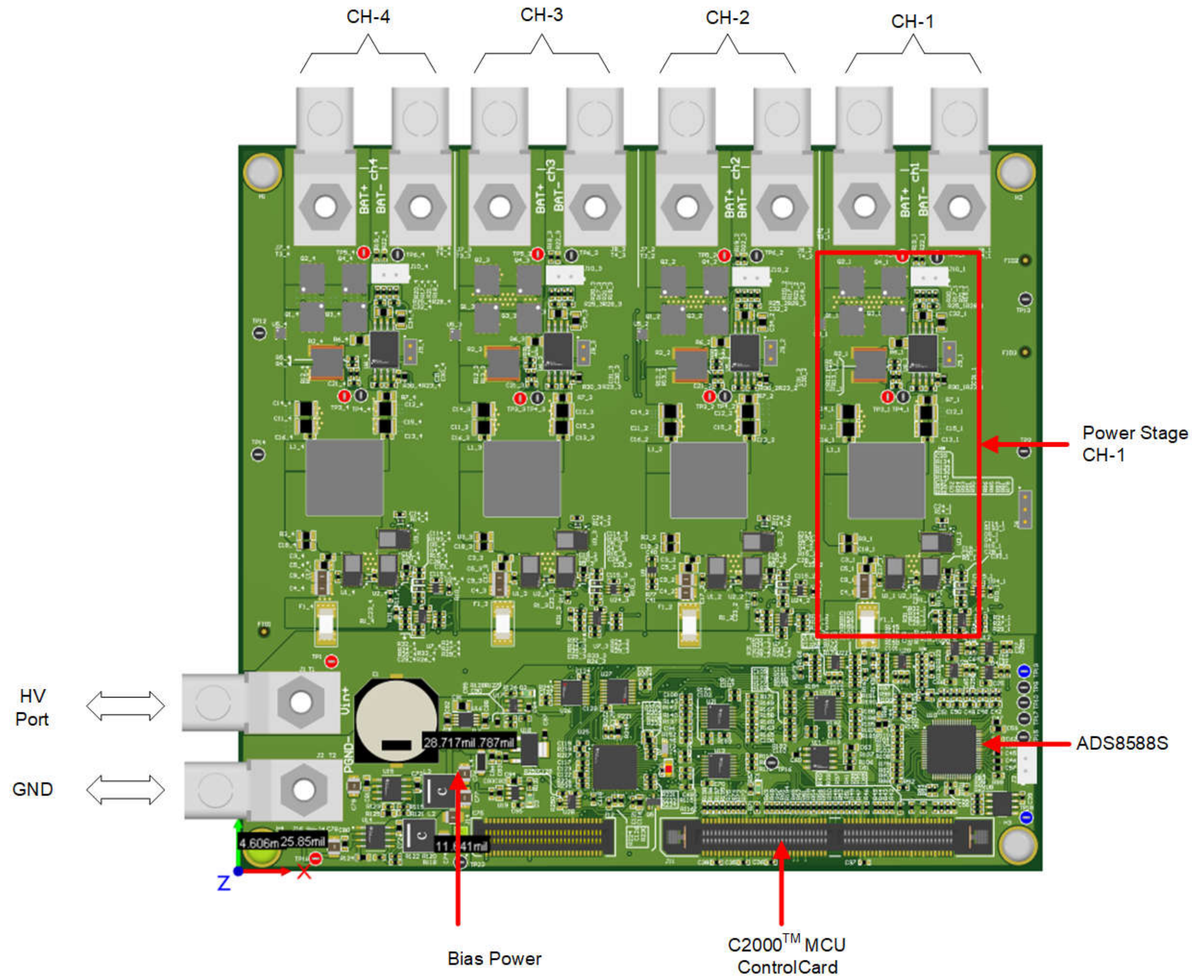


Figure 3-1. Board Overview

3.2 Software Requirements

The design software is available in the DigitalPower Software Development Kit (SDK) for C2000 MCUs (C2000WARE-DIGITALPOWER-SDK), and is supported inside the powerSUITE framework.

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 installation of SDK.

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-010090 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.
- Direct 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_010090/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. The non-powerSUITE project is provided for customers who find the powerSUITE GUI limiting or want to remove powerSUITE for production code.
  4. This document guides the user through the powerSUITE project, but all the steps can be repeated with the non-powerSUITE project with modification to the relevant `#defines` in the powerSUITE `settings.h` file, which are documented in this design guide.

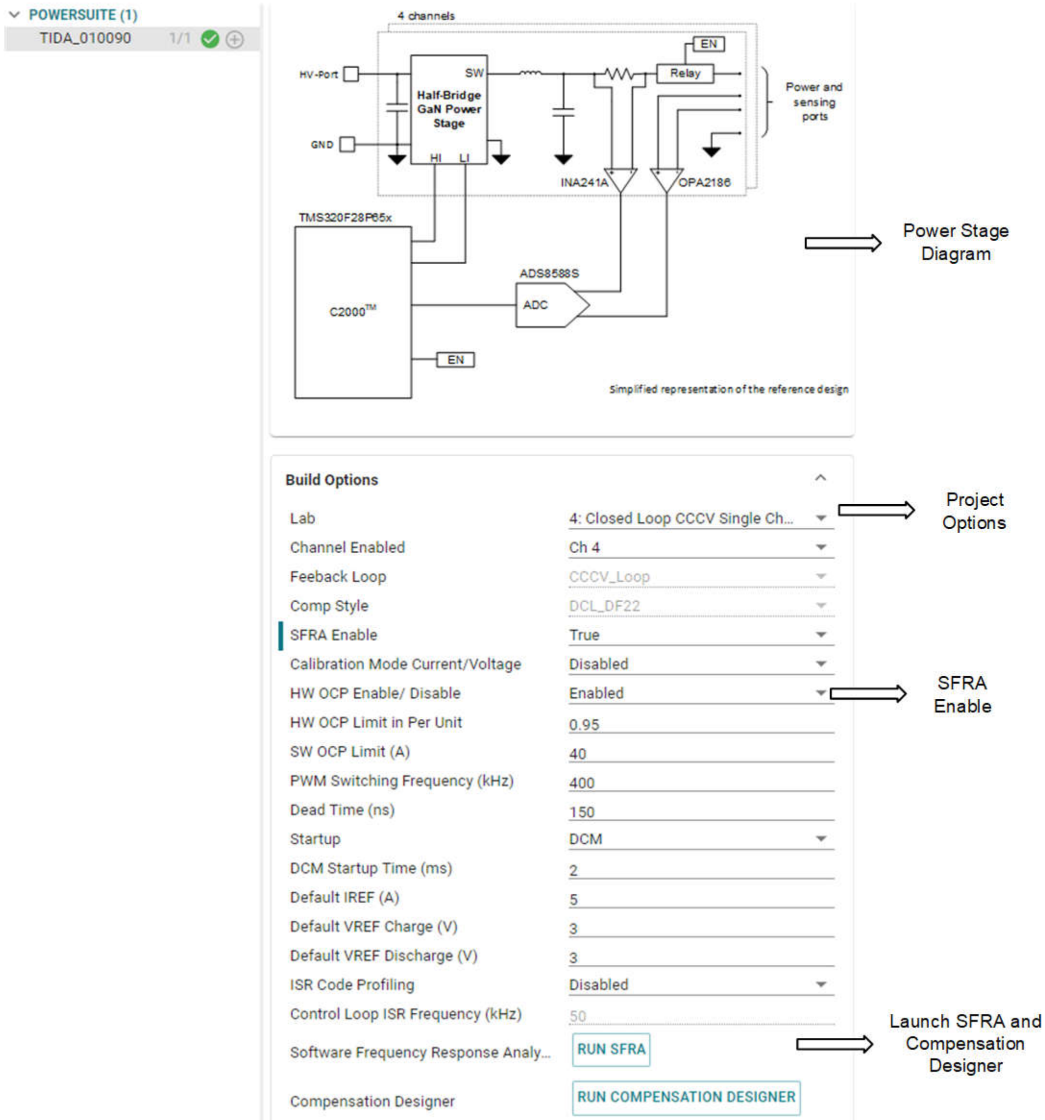


Figure 3-2. powerSUITE Page for the Design

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.

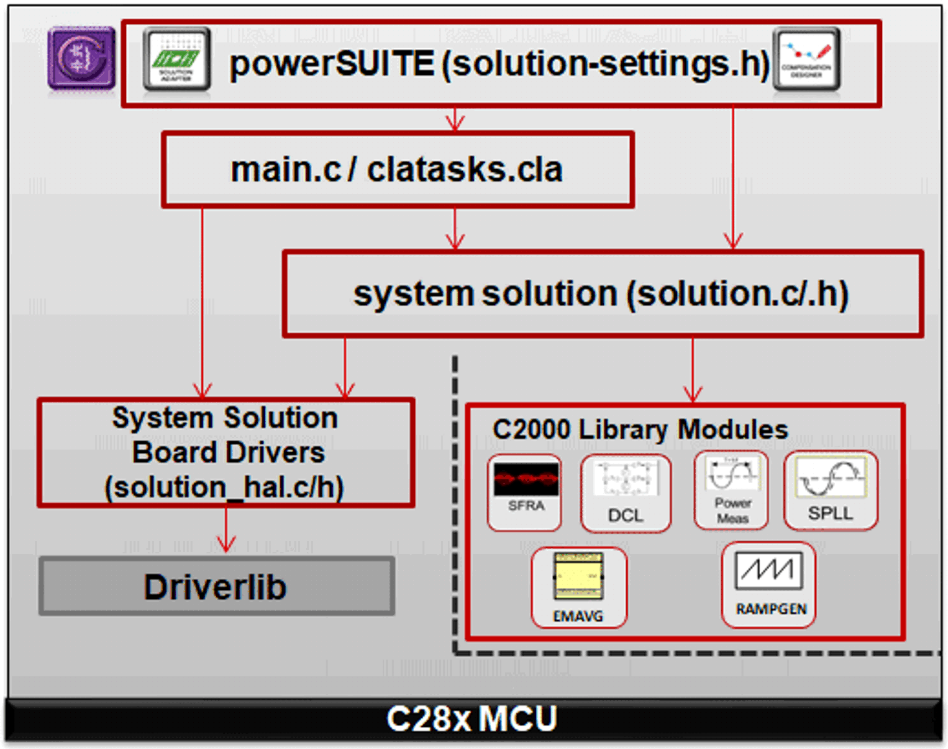


Figure 3-3. Project Structure Overview

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_gan`.

The powerSUITE page can be opened by clicking on the `main.syscfg` file, listed under the Project Explorer. The powerSUITE 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. `_user_settings.h` is included by the `_settings.h` file 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 `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* name (for example, *BT4CH_userParam_chX*). This naming convention lets the user combine different solutions while avoiding naming conflicts.

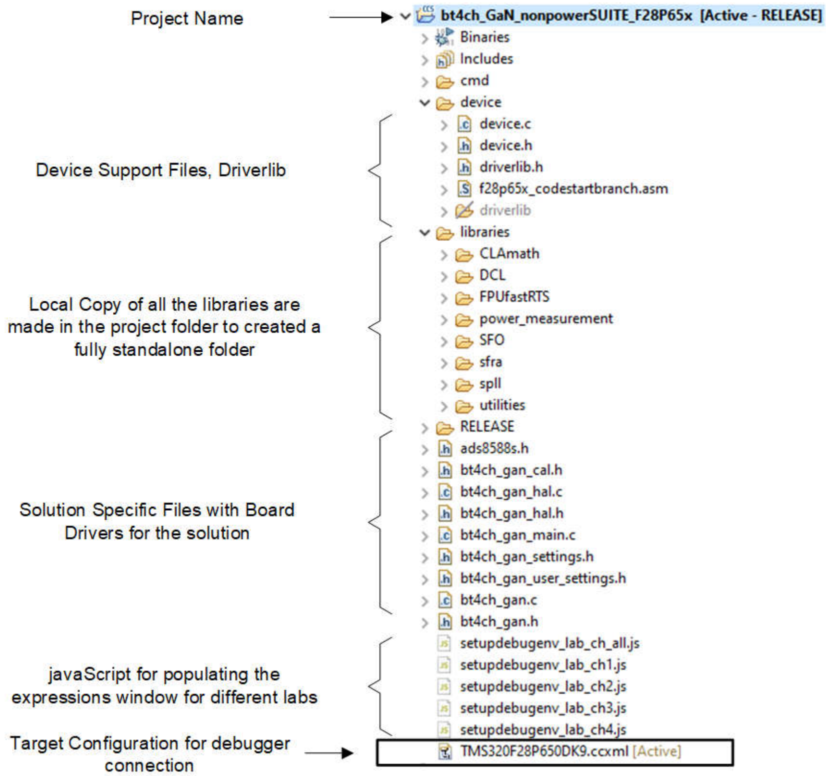


Figure 3-4. Project Explorer View of the BT4PH Project

The bt4ch_gan 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. ADCC senses input voltage and output voltage of the converters, and the output is used to implement soft-start of the DC/DC.

ISR2 is triggered by the BUSY signal of the ADS8588S. The external ADC is programmed for a 50kSPS sample rate, which sets the ISR frequency.

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

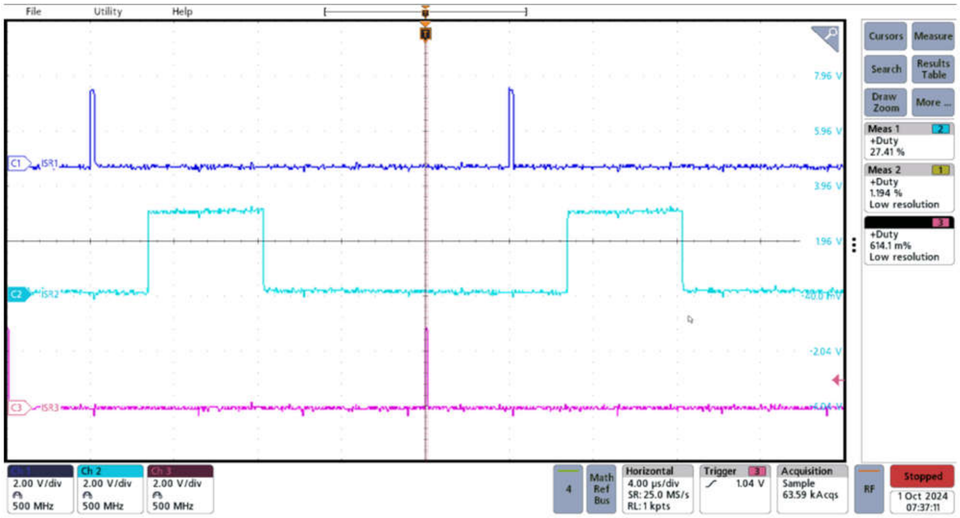


Figure 3-5. ISR Execution Time for Four Channels

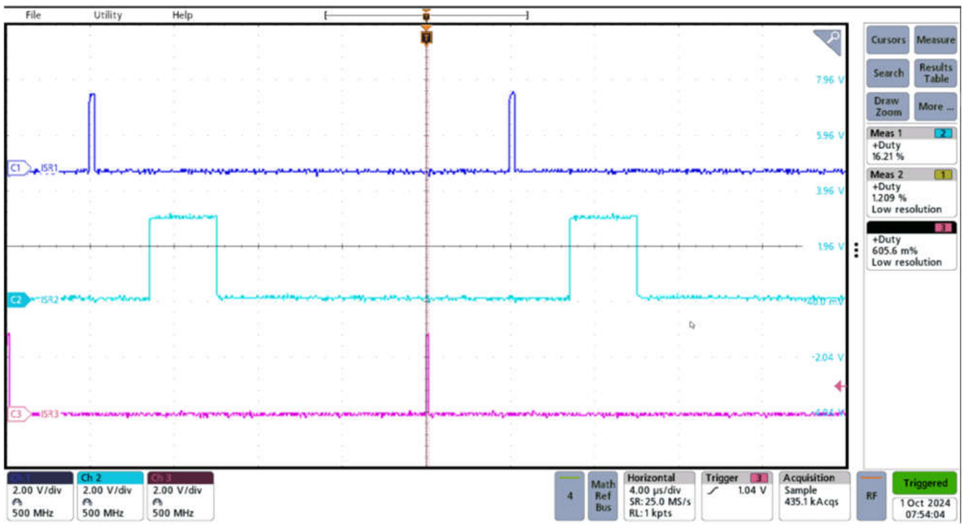


Figure 3-6. ISR Execution Time for One Channel

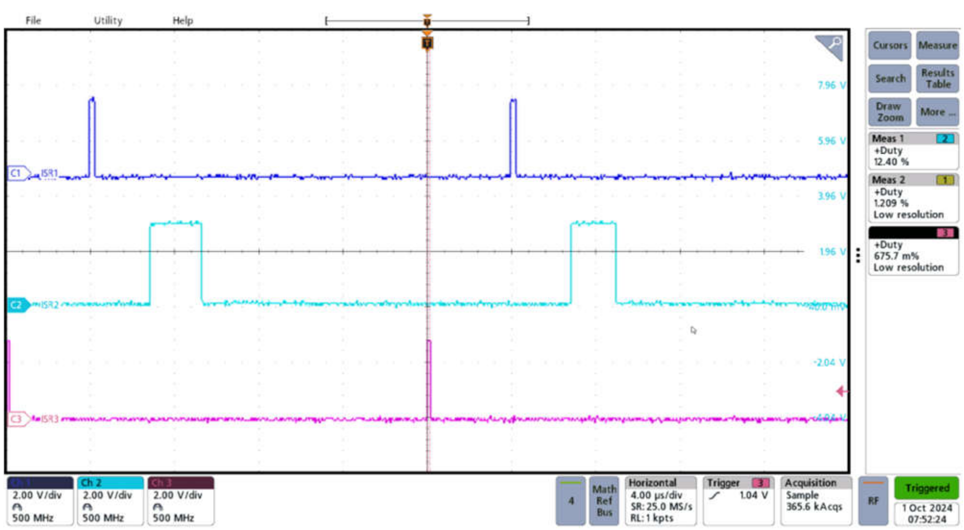


Figure 3-7. ISR Execution Time When All Channels are OFF

3.2.3 Software Flow Diagram

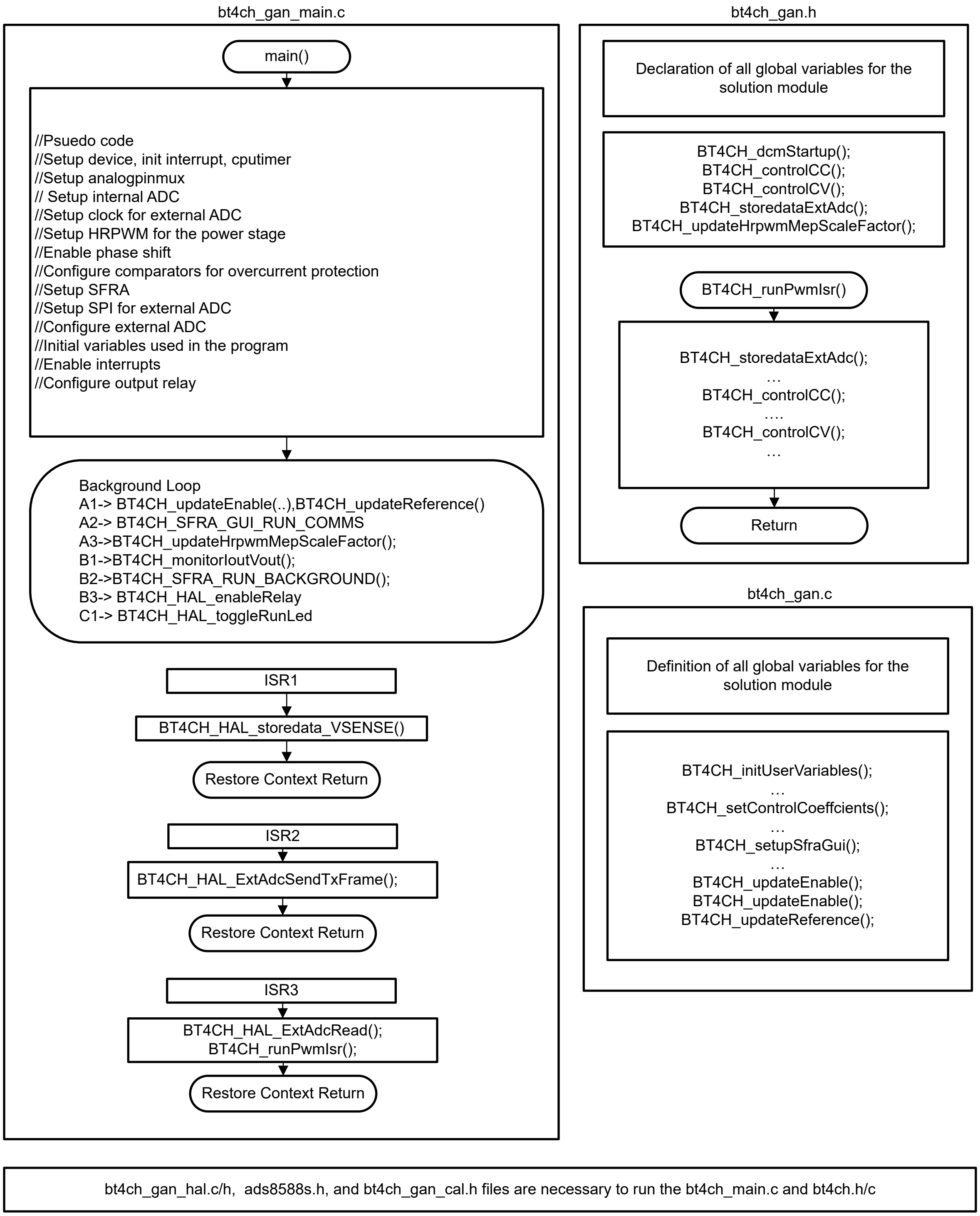


Figure 3-8. Software Flow Diagram

3.3 Test Setup

3.3.1 Hardware Setup to Tune the Current and Voltage Loop

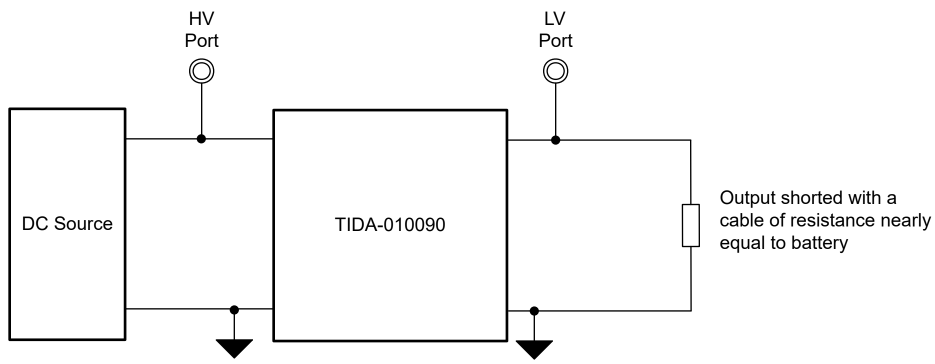


Figure 3-9. Hardware Setup to Tune the Current and Voltage Loops

3.3.2 Hardware Setup to Test Bidirectional Power Flow

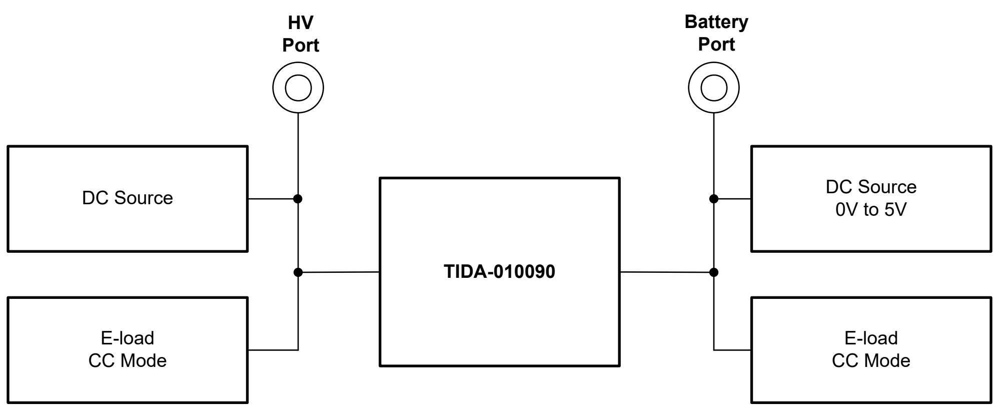


Figure 3-10. Hardware Setup to Test Bidirectional Power Flow

3.3.3 Hardware Setup for Current and Voltage Calibration

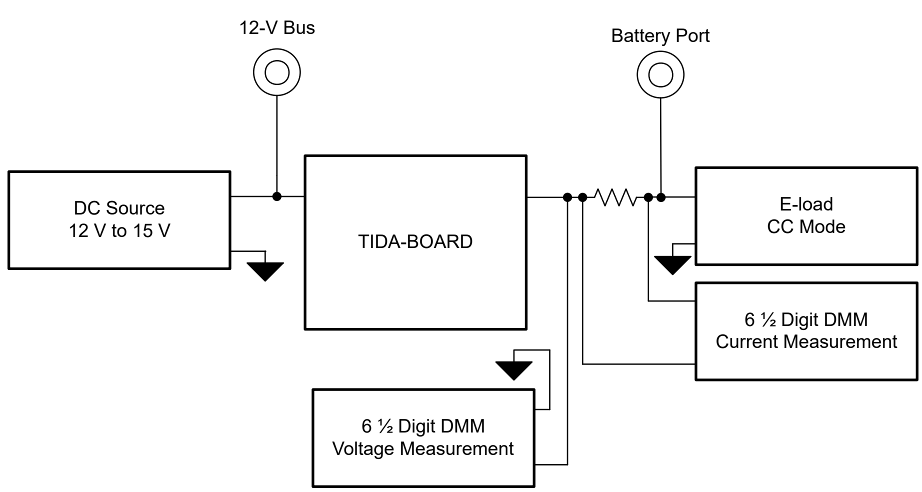


Figure 3-11. Hardware Setup for Current and Voltage Calibration

3.4 Test Procedure

3.4.1 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, 100]
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
En_bool	unsigned int	Set this parameter to 1 to enable the channel
remote_sense	unsigned int	Set this parameter to 1 for to use remote 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

3.4.2 Lab 1. Open-Loop Current Control Single Phase

3.4.2.1 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)
```

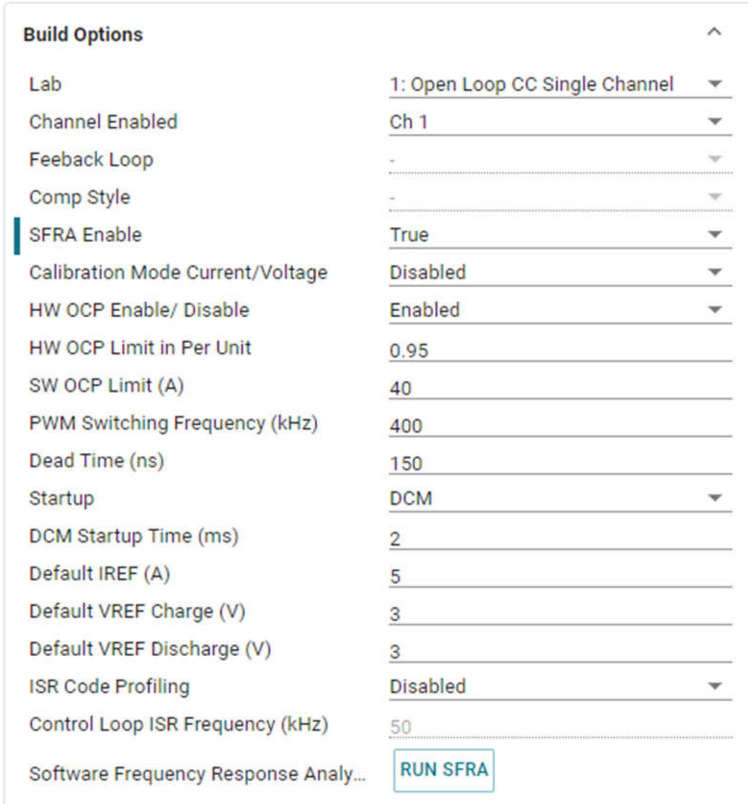


Figure 3-12. Build Options for Lab 1

3.4.2.2 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.

3.4.2.3 Running the Code

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

1. Use the test setup shown in Figure 3-9.
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 *BUSY* signal of the external ADC if the frequency is 50kHz using an oscilloscope. Figure 3-13 shows the ADS8588S BUSY and CONVST signals when the MCU is running.
5. Set the following parameters from the *Expression Window*:
   - `BT4CH_userParam_chX->dutyRef_pu` = 0.06
   - 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 15A.
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.

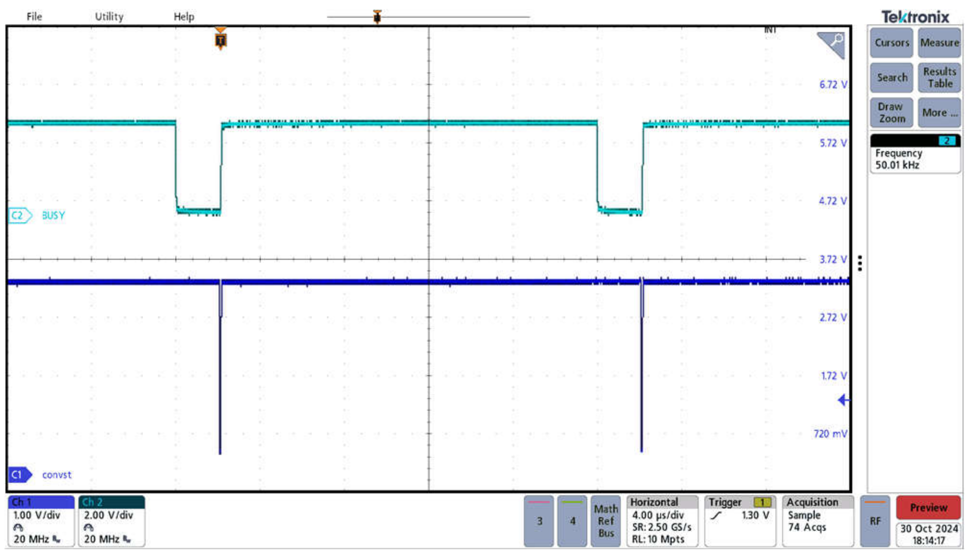
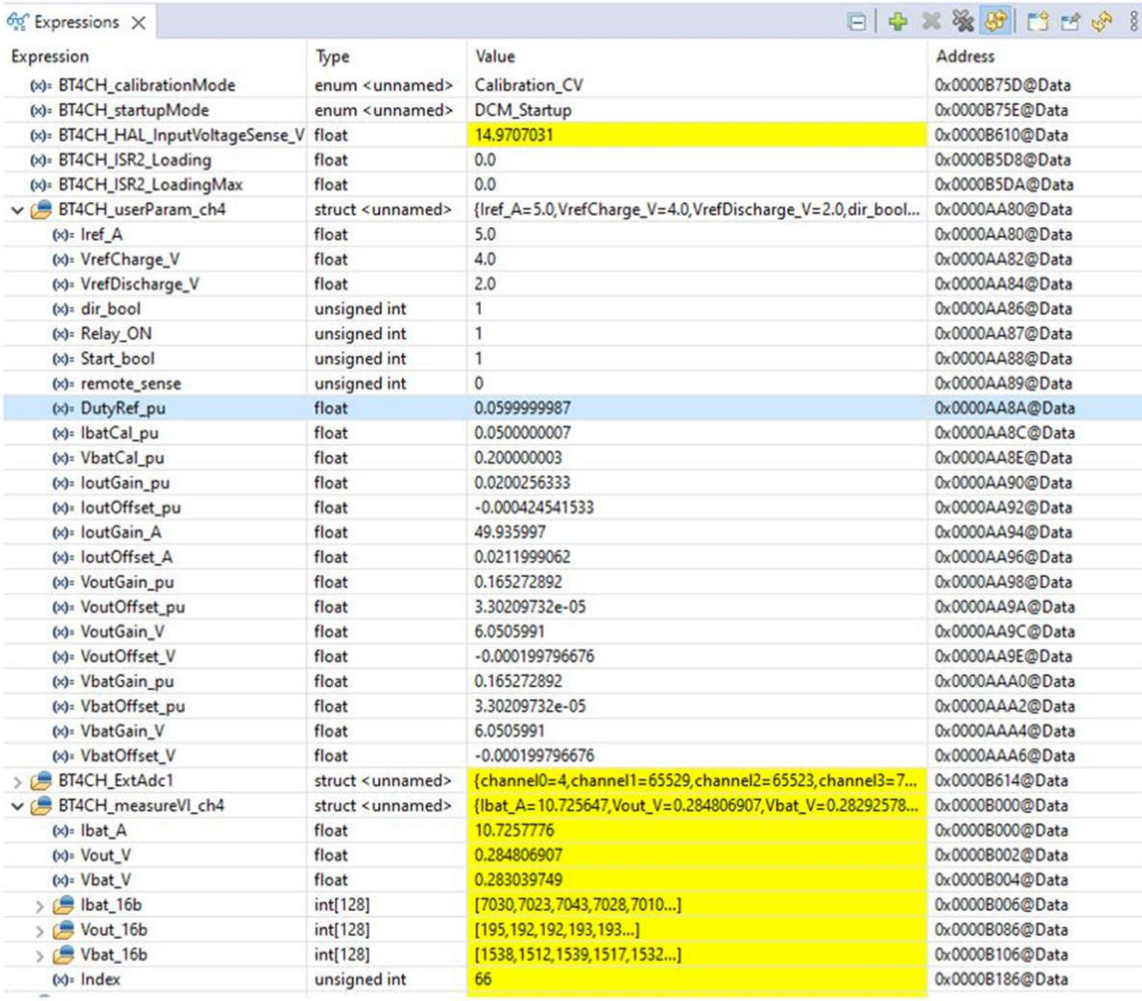


Figure 3-13. ADS8588S CONVST and BUSY Signal



Expressions

Expression	Type	Value	Address
BT4CH_calibrationMode	enum <unnamed>	Calibration_CV	0x0000B75D@Data
BT4CH_startupMode	enum <unnamed>	DCM_Startup	0x0000B75E@Data
BT4CH_HAL_InputVoltageSense_V	float	14.9707031	0x0000B610@Data
BT4CH_ISR2_Loading	float	0.0	0x0000B5D8@Data
BT4CH_ISR2_LoadingMax	float	0.0	0x0000B5DA@Data
BT4CH_userParam_ch4	struct <unnamed>	{Iref_A=5.0,VrefCharge_V=4.0,VrefDischarge_V=2.0,dir_bool...	0x0000AA80@Data
Iref_A	float	5.0	0x0000AA80@Data
VrefCharge_V	float	4.0	0x0000AA82@Data
VrefDischarge_V	float	2.0	0x0000AA84@Data
dir_bool	unsigned int	1	0x0000AA86@Data
Relay_ON	unsigned int	1	0x0000AA87@Data
Start_bool	unsigned int	1	0x0000AA88@Data
remote_sense	unsigned int	0	0x0000AA89@Data
DutyRef_pu	float	0.0599999987	0x0000AA8A@Data
IbatCal_pu	float	0.0500000007	0x0000AA8C@Data
VbatCal_pu	float	0.200000003	0x0000AA8E@Data
IoutGain_pu	float	0.0200256333	0x0000AA90@Data
IoutOffset_pu	float	-0.000424541533	0x0000AA92@Data
IoutGain_A	float	49.935997	0x0000AA94@Data
IoutOffset_A	float	0.0211999062	0x0000AA96@Data
VoutGain_pu	float	0.165272892	0x0000AA98@Data
VoutOffset_pu	float	3.30209732e-05	0x0000AA9A@Data
VoutGain_V	float	6.0505991	0x0000AA9C@Data
VoutOffset_V	float	-0.000199796676	0x0000AA9E@Data
VbatGain_pu	float	0.165272892	0x0000AAA0@Data
VbatOffset_pu	float	3.30209732e-05	0x0000AAA2@Data
VbatGain_V	float	6.0505991	0x0000AAA4@Data
VbatOffset_V	float	-0.000199796676	0x0000AAA6@Data
BT4CH_ExtAdc1	struct <unnamed>	{channel0=4,channel1=65529,channel2=65523,channel3=7...	0x0000B614@Data
BT4CH_measureVI_ch4	struct <unnamed>	{Ibat_A=10.725647,Vout_V=0.284806907,Vbat_V=0.28292578...	0x0000B000@Data
Ibat_A	float	10.7257776	0x0000B000@Data
Vout_V	float	0.284806907	0x0000B002@Data
Vbat_V	float	0.283039749	0x0000B004@Data
Ibat_16b	int[128]	[7030,7023,7043,7028,7010...]	0x0000B006@Data
Vout_16b	int[128]	[195,192,192,193,193...]	0x0000B086@Data
Vbat_16b	int[128]	[1538,1512,1539,1517,1532...]	0x0000B106@Data
Index	unsigned int	66	0x0000B186@Data

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

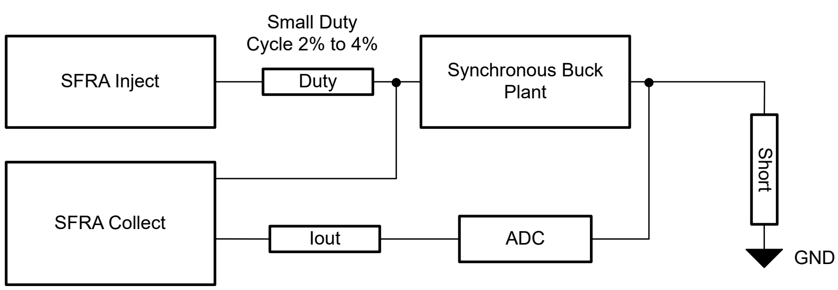


Figure 3-15. SFRA Setup for Open-Loop Current Control

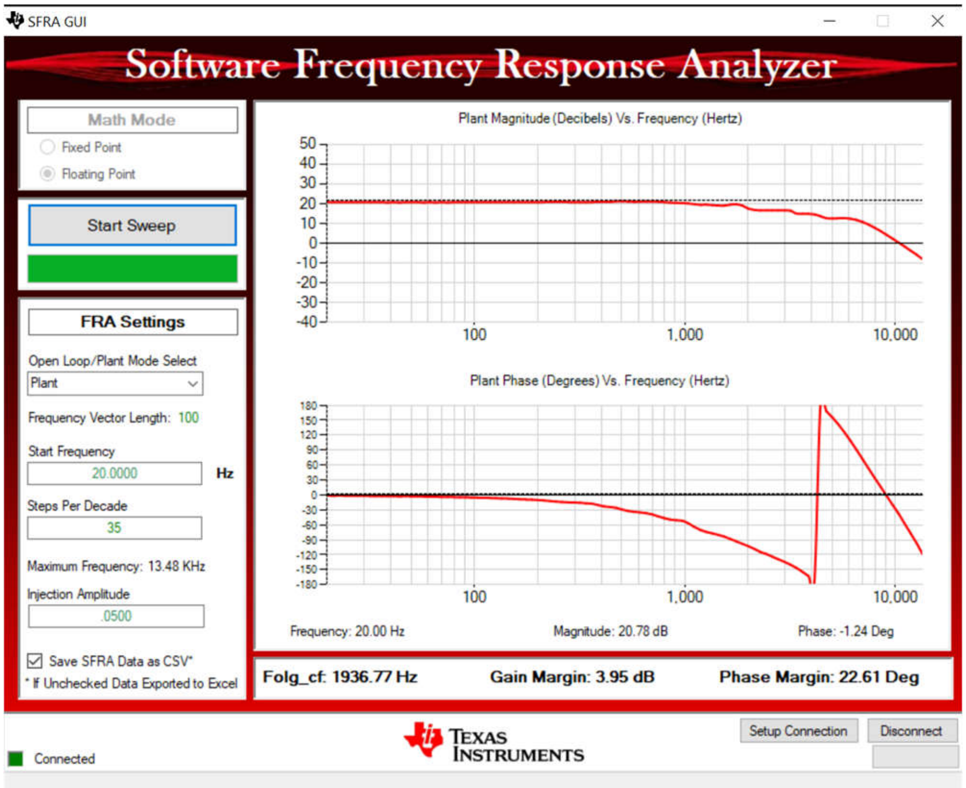


Figure 3-16. Current Control Open-Loop Frequency Response

3.4.3 Lab 2. Closed Loop Current Control Single Phase

3.4.3.1 Setting Software Options for Lab 2

1. To run this lab, make sure the hardware is set up as outlined in the previous section, Figure 3-9
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.
   - 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.
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 (2)
#define CHANNEL_NUMBER (1)
#define SFRA_ENABLED (true)
```

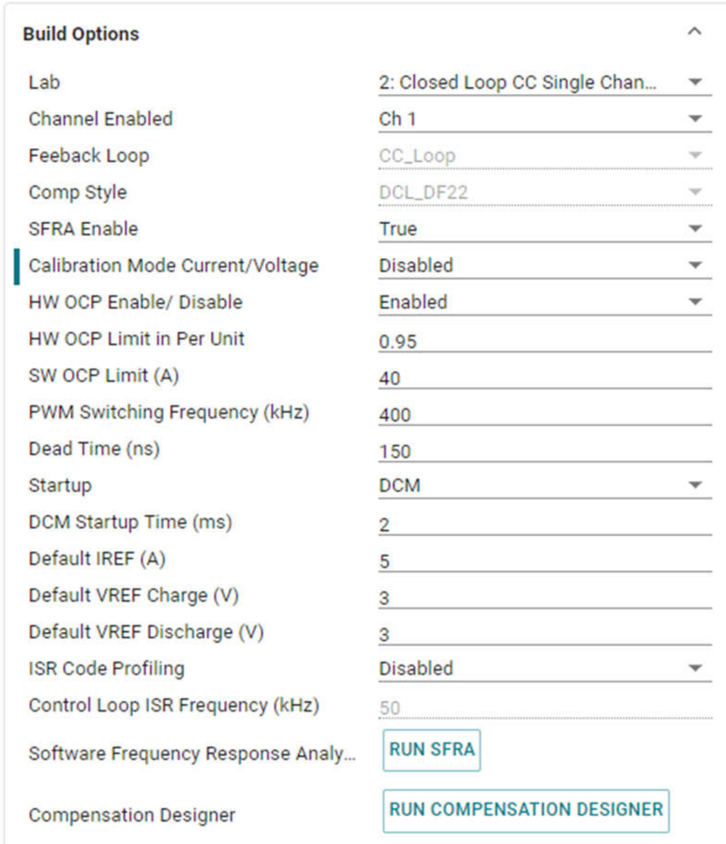


Figure 3-17. Build Options for Lab 2

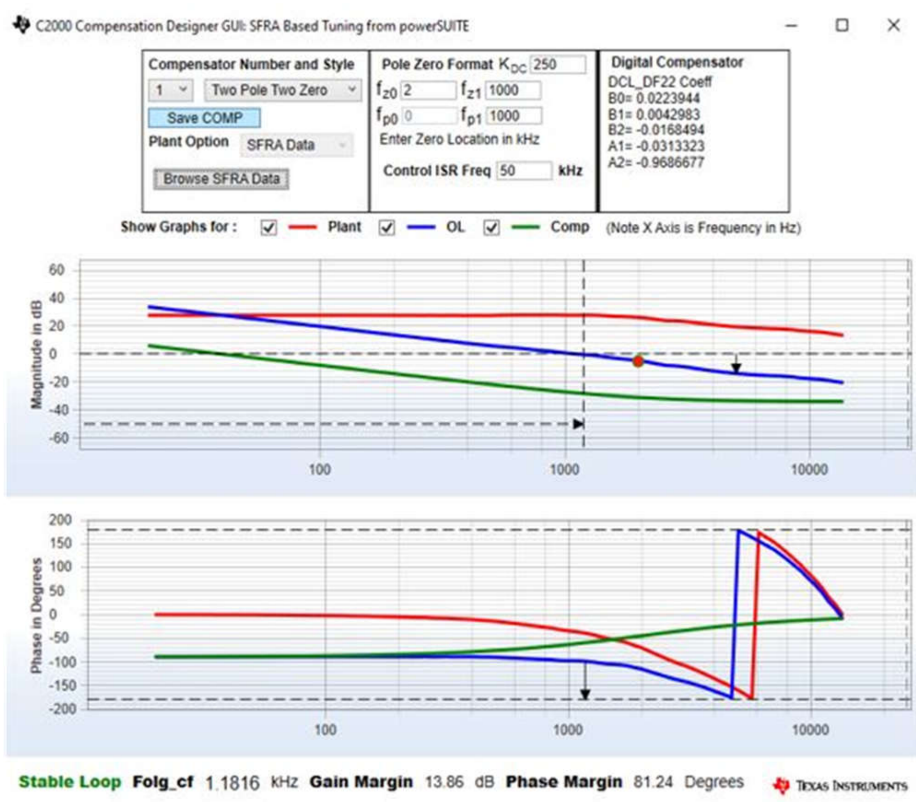


Figure 3-18. Tuning Current Loop Using Compensation Designer

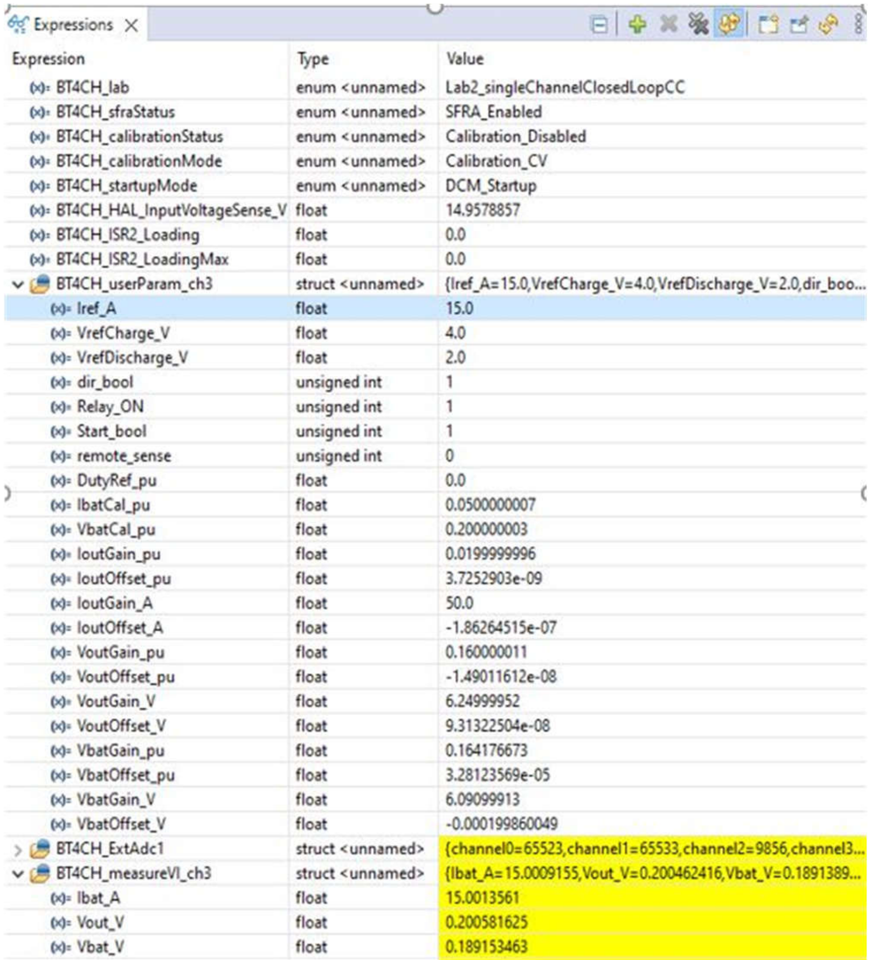
3.4.3.2 Building and Loading the Project and Setting up Debug Environment

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.

3.4.3.3 Running 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 3.3.2.
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` = 15.0.
   - 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 ±1mA 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.



Expression	Type	Value
BT4CH_lab	enum <unnamed>	Lab2_singleChannelClosedLoopCC
BT4CH_sfraStatus	enum <unnamed>	SFRA_Enabled
BT4CH_calibrationStatus	enum <unnamed>	Calibration_Disabled
BT4CH_calibrationMode	enum <unnamed>	Calibration_CV
BT4CH_startupMode	enum <unnamed>	DCM_Startup
BT4CH_HAL_InputVoltageSense_V	float	14.9578857
BT4CH_ISR2_Loading	float	0.0
BT4CH_ISR2_LoadingMax	float	0.0
BT4CH_userParam_ch3	struct <unnamed>	{Iref_A=15.0,VrefCharge_V=4.0,VrefDischarge_V=2.0,dir_boo...
Iref_A	float	15.0
VrefCharge_V	float	4.0
VrefDischarge_V	float	2.0
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	0.0500000007
VbatCal_pu	float	0.200000003
IoutGain_pu	float	0.0199999996
IoutOffset_pu	float	3.7252903e-09
IoutGain_A	float	50.0
IoutOffset_A	float	-1.86264515e-07
VoutGain_pu	float	0.160000011
VoutOffset_pu	float	-1.49011612e-08
VoutGain_V	float	6.24999952
VoutOffset_V	float	9.31322504e-08
VbatGain_pu	float	0.164176673
VbatOffset_pu	float	3.28123569e-05
VbatGain_V	float	6.09099913
VbatOffset_V	float	-0.000199860049
BT4CH_ExtAdc1	struct <unnamed>	{channel0=65523,channel1=65533,channel2=9856,channel3...
BT4CH_measureVI_ch3	struct <unnamed>	{Ibat_A=15.0009155,Vout_V=0.200462416,Vbat_V=0.1891389...
Ibat_A	float	15.0013561
Vout_V	float	0.200581625
Vbat_V	float	0.189153463

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

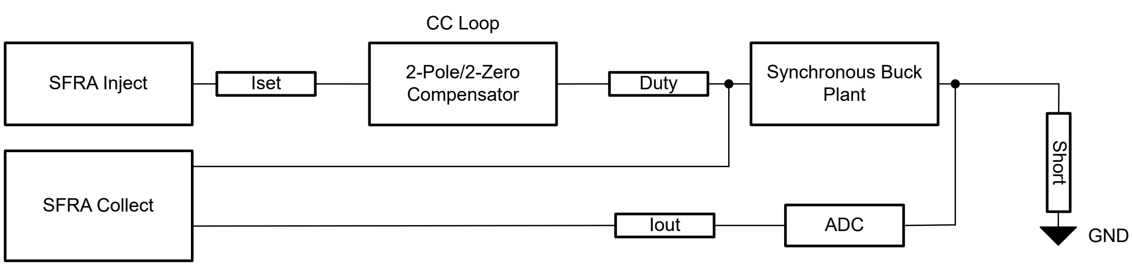


Figure 3-20. SFRA Setup for Closed-Loop Current Control

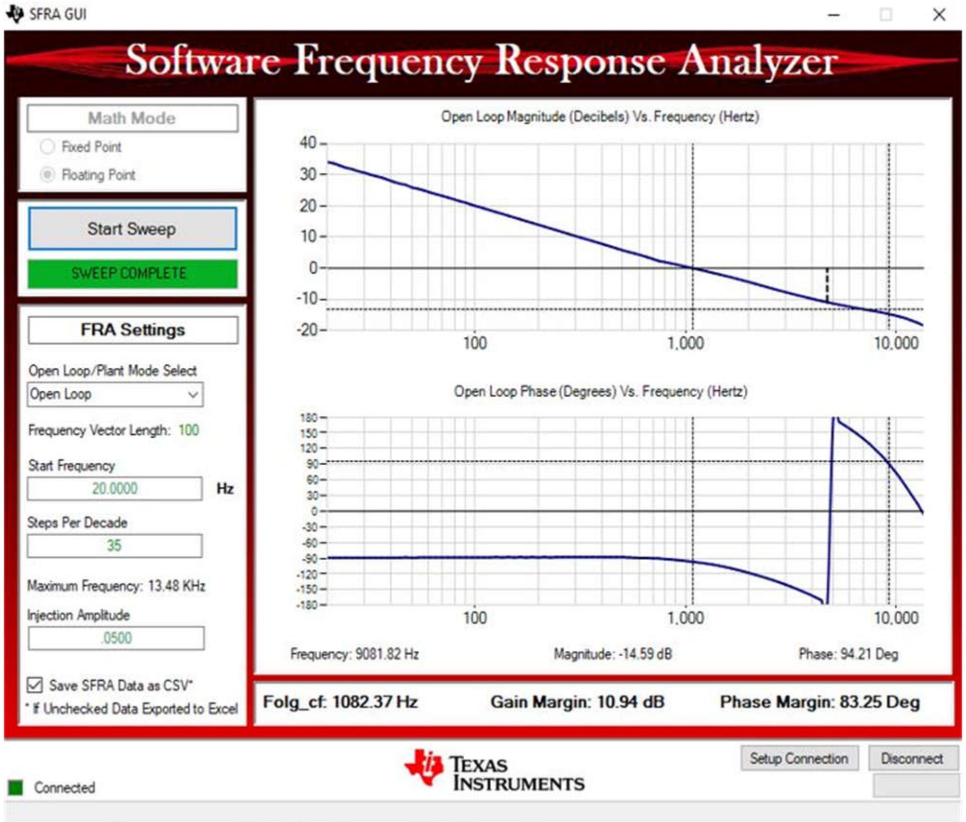


Figure 3-21. Current Control Closed-Loop Frequency Response

3.4.4 Lab 3. Open Loop Voltage Control Single Channel

3.4.4.1 Setting Software Options for Lab 3

1. Use the test setup shown in Figure 3-9.
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 3: Single Channel Open-Loop CV Control* for the Lab
   - Select the channel
   - Enable the SFRA
   - Save the page
4. When using non-powerSuite version of the project, above settings are directly modified in `solution_settings.h` file.

```
#define LAB_NUMBER (3)
#define CHANNEL_NUMBER (1)
#define SFRA_ENABLED (true)
```

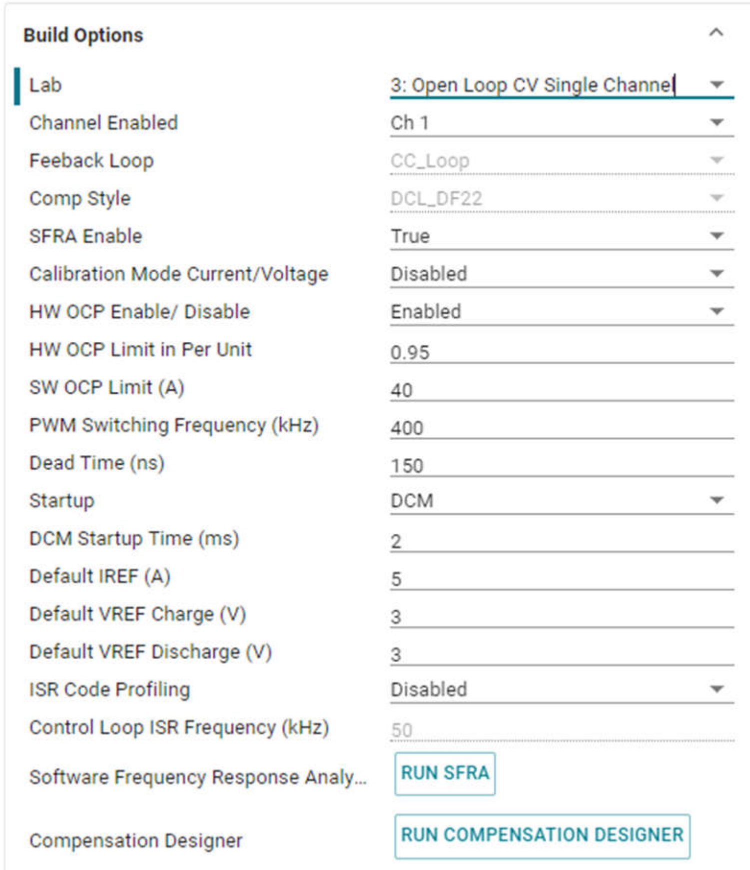


Figure 3-22. Build Options for Lab 3

3.4.4.2 Building and Loading the Project and Setting up Debug Environment

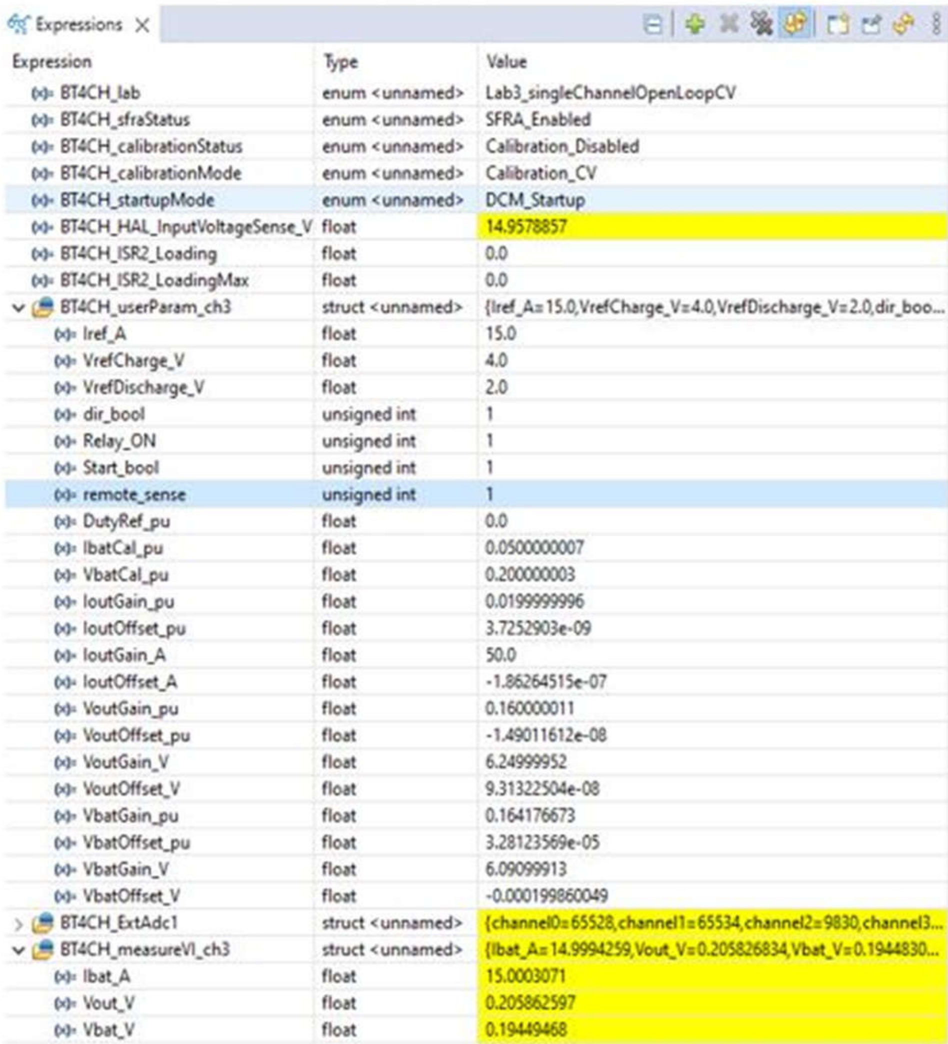
1. Right-click on the project name and click ***Rebuild Project***.
2. The project builds successfully.
3. In the Project Explorer, make sure the correct target configuration file is set as Active under targetConfigs.
4. Then, click ***Run*** → ***Debug*** to launch a debugging session.
5. The project 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 the *Open* button 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.

3.4.4.3 Running the Code

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

1. Use the test setup shown in Section 3.3.2.
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.


Expressions

Expression	Type	Value
BT4CH_lab	enum <unnamed>	Lab3_singleChannelOpenLoopCV
BT4CH_sfraStatus	enum <unnamed>	SFRA_Enabled
BT4CH_calibrationStatus	enum <unnamed>	Calibration_Disabled
BT4CH_calibrationMode	enum <unnamed>	Calibration_CV
BT4CH_startupMode	enum <unnamed>	DCM_Startup
BT4CH_HAL_InputVoltageSense_V	float	14.9578857
BT4CH_ISR2_Loading	float	0.0
BT4CH_ISR2_LoadingMax	float	0.0
BT4CH_userParam_ch3	struct <unnamed>	{Iref_A=15.0,VrefCharge_V=4.0,VrefDischarge_V=2.0,dir_boo...
Iref_A	float	15.0
VrefCharge_V	float	4.0
VrefDischarge_V	float	2.0
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	0.0500000007
VbatCal_pu	float	0.200000003
IoutGain_pu	float	0.0199999996
IoutOffset_pu	float	3.7252903e-09
IoutGain_A	float	50.0
IoutOffset_A	float	-1.86264515e-07
VoutGain_pu	float	0.160000011
VoutOffset_pu	float	-1.49011612e-08
VoutGain_V	float	6.24999952
VoutOffset_V	float	9.31322504e-08
VbatGain_pu	float	0.164176673
VbatOffset_pu	float	3.28123569e-05
VbatGain_V	float	6.09099913
VbatOffset_V	float	-0.000199860049
BT4CH_ExtAdc1	struct <unnamed>	{channel0=65528,channel1=65534,channel2=9830,channel3...
BT4CH_measureVI_ch3	struct <unnamed>	{Ibat_A=14.9994259,Vout_V=0.205826834,Vbat_V=0.1944830...
Ibat_A	float	15.0003071
Vout_V	float	0.205862597
Vbat_V	float	0.19449468

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

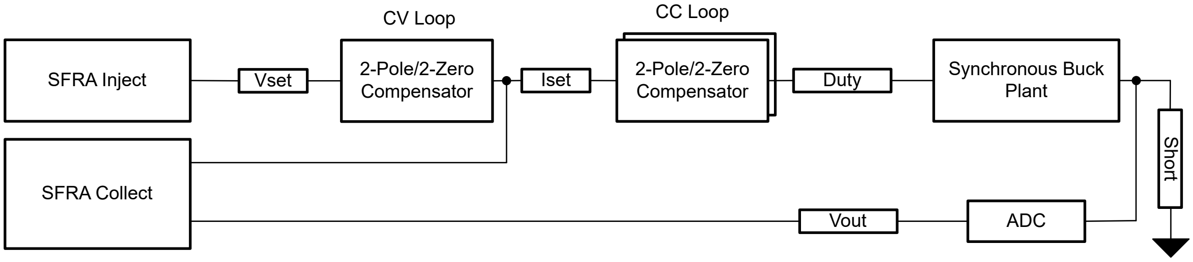


Figure 3-24. SFRA Setup for Open-Loop Voltage Control

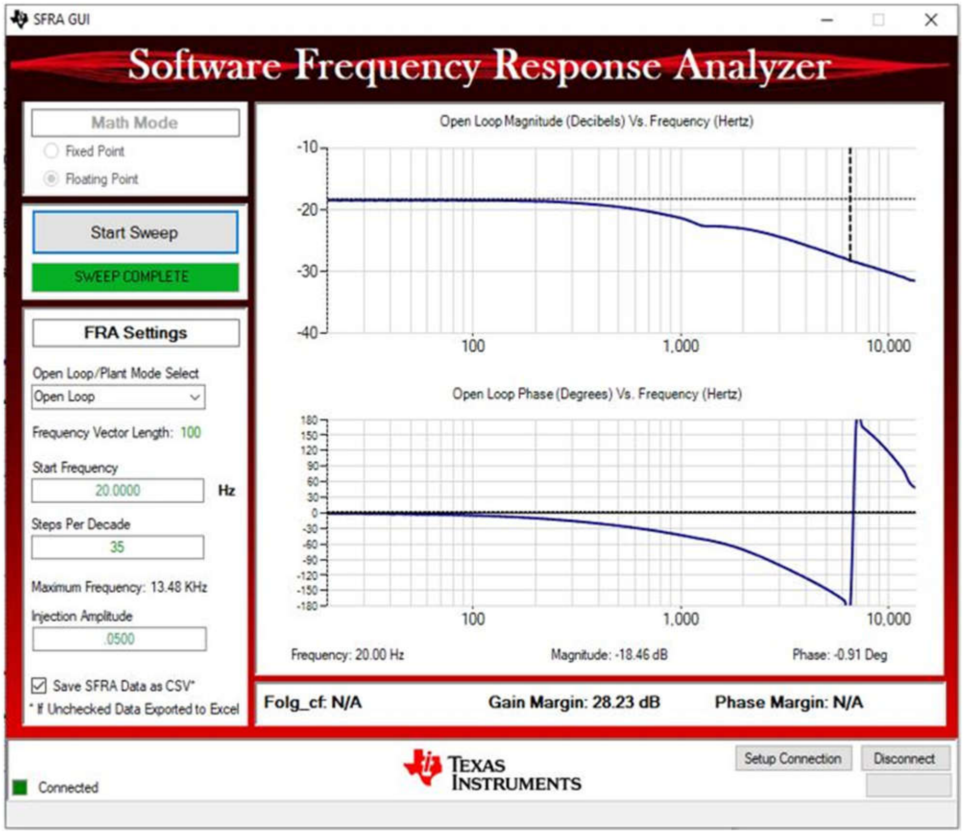


Figure 3-25. Voltage Control Open-Loop Frequency Response

3.4.5 Lab 4. Closed Loop Current and Voltage Control Single Channel

3.4.5.1 Setting Software Options for Lab 4

1. Use the test setup shown in Figure 3-9.
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: Single Channel Closed-Loop CV* for the Lab
   - Select Channel
   - Enable the SFRA
   - Save the page
4. When using non-powerSuite version of the project, the above settings are directly modified in the `solution_settings.h` file.

```
#define LAB_NUMBER (4)
#define CHANNEL_NUMBER (1)
#define SFRA_ENABLED (true)
```

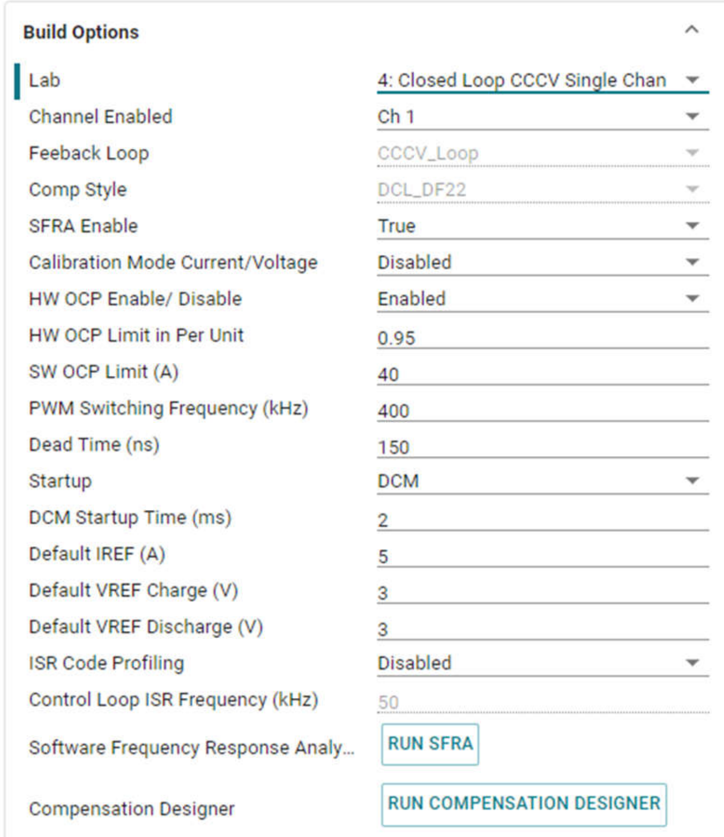


Figure 3-26. Build Options for Lab 4

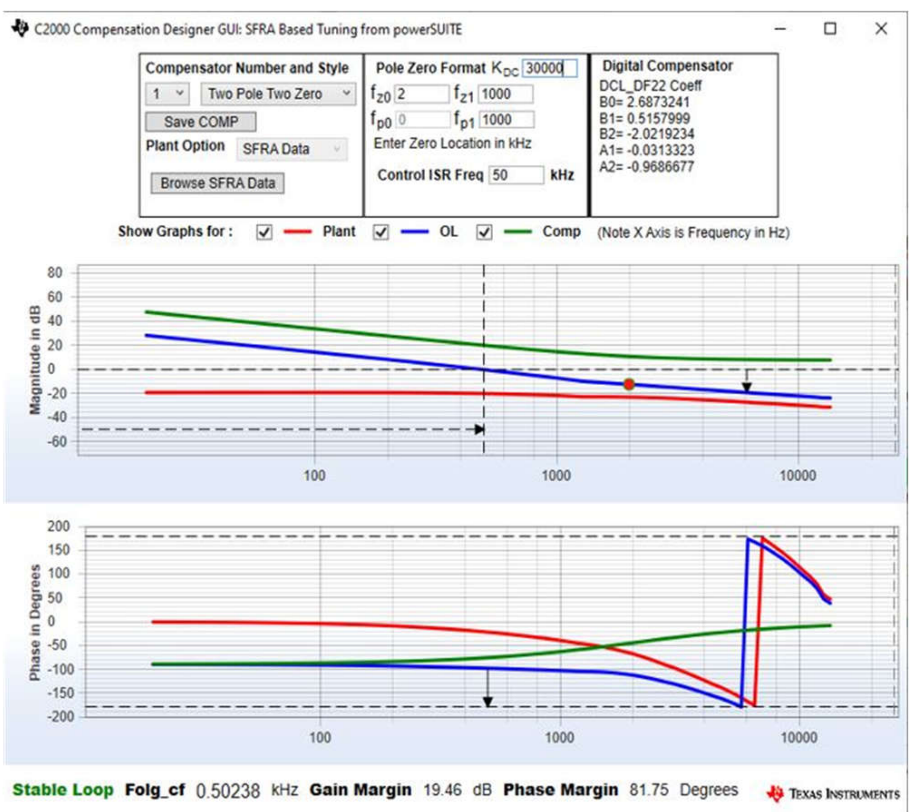


Figure 3-27. Tuning Voltage Loop Using Compensation Designer

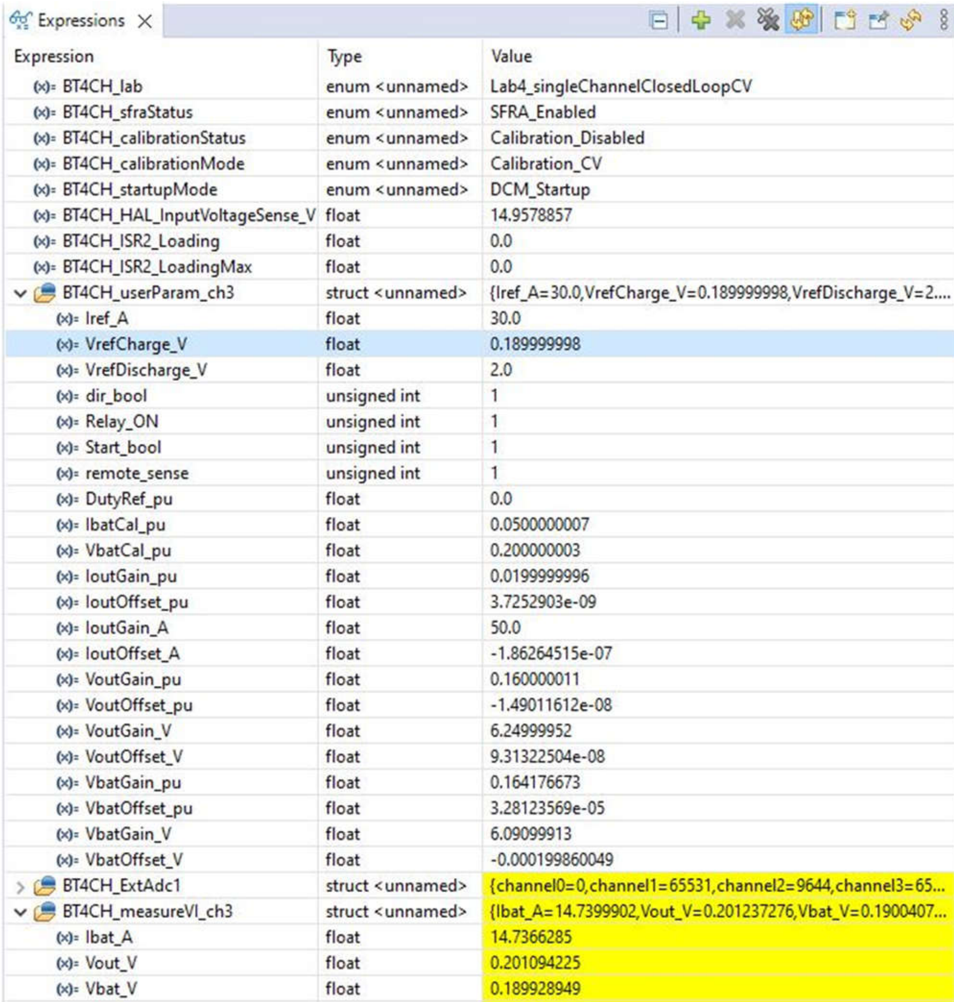
3.4.5.2 Building and Loading the Project and Setting up Debug Environment

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.

3.4.5.3 Running the Code

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

1. Use the test setup shown in Section 3.3.2.
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` = 25.0
   - `BT4CH_userParam_chX->vrefCharge_V` = 0.12
   - Set the `BT4CH_userParam_chX->en_bool` = 1
   - See the Figure 3-28 for the *Expression Window* settings
5. The `BT4CH_measureVI_chX` variable shows output current and voltage of the DC/DC converter. Vbatsense_V display value is close to vrefCharge_V with ±0.5mV 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.



Expressions

Expression	Type	Value
BT4CH_lab	enum <unnamed>	Lab4_singleChannelClosedLoopCV
BT4CH_sfraStatus	enum <unnamed>	SFRA_Enabled
BT4CH_calibrationStatus	enum <unnamed>	Calibration_Disabled
BT4CH_calibrationMode	enum <unnamed>	Calibration_CV
BT4CH_startupMode	enum <unnamed>	DCM_Startup
BT4CH_HAL_InputVoltageSense_V	float	14.9578857
BT4CH_ISR2_Loading	float	0.0
BT4CH_ISR2_LoadingMax	float	0.0
BT4CH_userParam_ch3	struct <unnamed>	{Iref_A=30.0,VrefCharge_V=0.189999998,VrefDischarge_V=2....
Iref_A	float	30.0
VrefCharge_V	float	0.189999998
VrefDischarge_V	float	2.0
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	0.0500000007
VbatCal_pu	float	0.200000003
IoutGain_pu	float	0.0199999996
IoutOffset_pu	float	3.7252903e-09
IoutGain_A	float	50.0
IoutOffset_A	float	-1.86264515e-07
VoutGain_pu	float	0.160000011
VoutOffset_pu	float	-1.49011612e-08
VoutGain_V	float	6.24999952
VoutOffset_V	float	9.31322504e-08
VbatGain_pu	float	0.164176673
VbatOffset_pu	float	3.28123569e-05
VbatGain_V	float	6.09099913
VbatOffset_V	float	-0.000199860049
BT4CH_ExtAdc1	struct <unnamed>	{channel0=0,channel1=65531,channel2=9644,channel3=65...
BT4CH_measureVI_ch3	struct <unnamed>	{Ibat_A=14.7399902,Vout_V=0.201237276,Vbat_V=0.1900407...
Ibat_A	float	14.7366285
Vout_V	float	0.201094225
Vbat_V	float	0.189928949

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

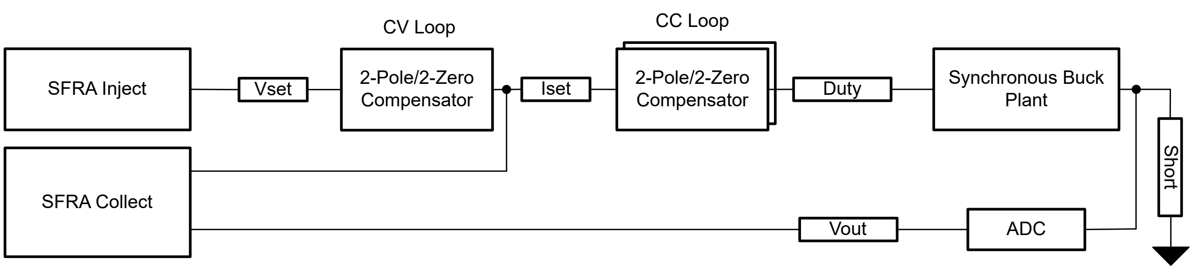


Figure 3-29. SFRA Setup for Closed-Loop Voltage Control

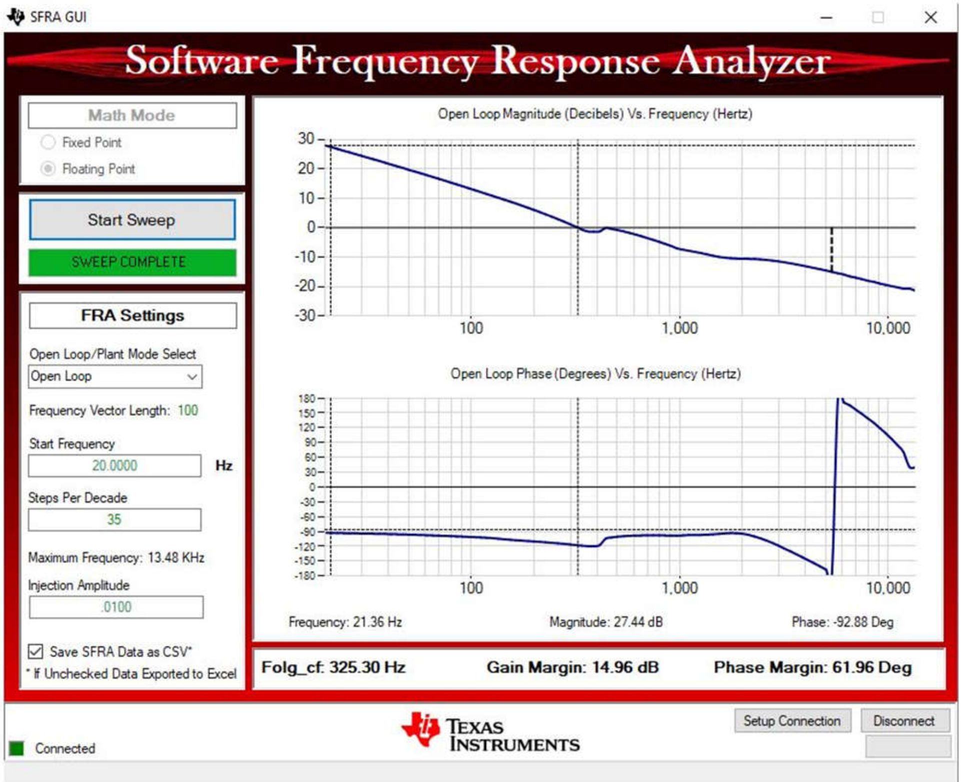


Figure 3-30. Voltage Control Closed-Loop Frequency Response

3.4.6 Lab 5. Closed Loop Current and Voltage Control Four Channels

3.4.6.1 Setting Software Options for Lab 5

1. Use the test setup shown in Figure 3-11.
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
    - Enable all channels
    - Set the SFRA Enable/Disable to 1
    - Save the page
4. When using non-powerSuite version of the project, the above settings are directly modified in the `solution_settings.h` file.

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

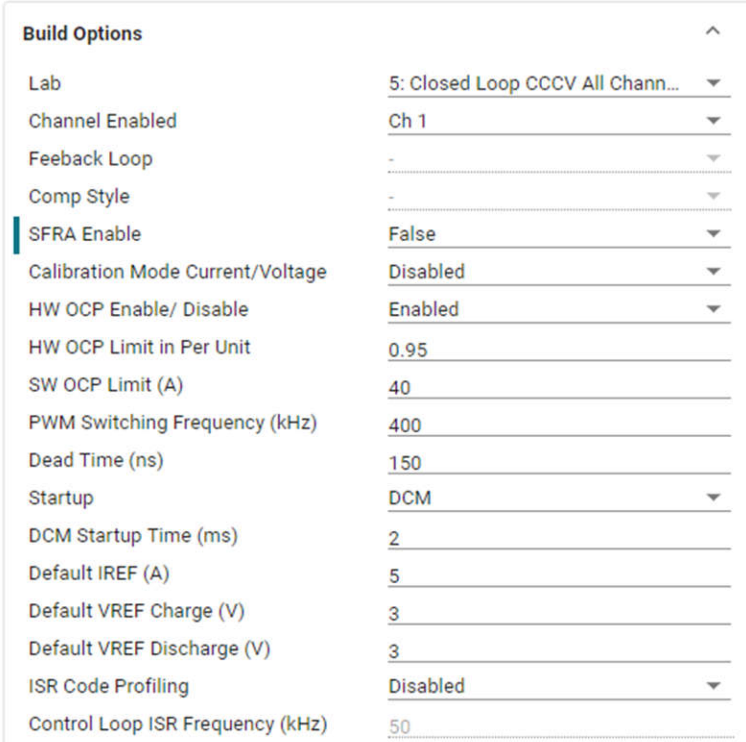


Figure 3-31. Build Options for Lab 5

3.4.6.2 Building and Loading the Project and Setting up Debug Environment

1. Right-click on the project name and click ***Rebuild Project***. The project builds successfully. In the Project Explorer, make sure the correct target configuration file is set as *Active* under targetConfigs.
2. Then, click ***Run*** → ***Debug*** to launch a debugging session. The project then loads on the device and the CCS debug view becomes active. The code halts at the start of the main routine.
3. 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_chAll.js*** script file located inside the project folder. This populates the watch window with the appropriate variables needed to debug the system.
4. Click on the ***Continuous Refresh*** button  on the watch window to enable continuous update of values from the controller.

3.4.6.3 Running the Code

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

1. Use the test setup shown in Section 3.3.2.
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` = 4.2
   - Set the `BT4CH_userParam_chX->en_bool` = 1
   - See the Figure 3-28 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`.

3.4.7 Calibration

1. To run this lab, make sure the hardware is set up as shown in Figure 3-11. The 2-points calibration method is used to calibrate gain and offset errors.
2. To measure current, use an external precision resistor and a DMM, or you can use E-Load current readings. Alternatively, voltage across sense resistors on the TIDA-010090 boards can be used to measure the output current. To measure voltage, use a DMM across the buck converter output voltage and remote sense connections.
3. Open the SYSCONFIG page, select Lab 5, and set *Calibration Mode* to *Current Calibration. Figure 3-32* 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 "0.05" and "0.3", 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)2.5)
   #define BT4CH_IBAT_ACTUAL_CH1_P2_A ((float32_t)15.00)
   ```

   - Repeat the steps for channel 2, 3 and 4.
4. Open the SYSCONFIG page, select Lab 5, and set *Calibration Mode* to Voltage *Calibration*. Figure 3-33 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)1.195)
     #define BT4CH_VBAT_ACTUAL_CH1_P2_V ((float32_t)3.589)
     ```

   - Repeat the steps for channel 2, 3 and 4.
5. Open the SYSCONFIG page, disable the calibration mode.
6. When using non-powerSuite version of the project, *Build Settings* are directly modified in `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)
   ```

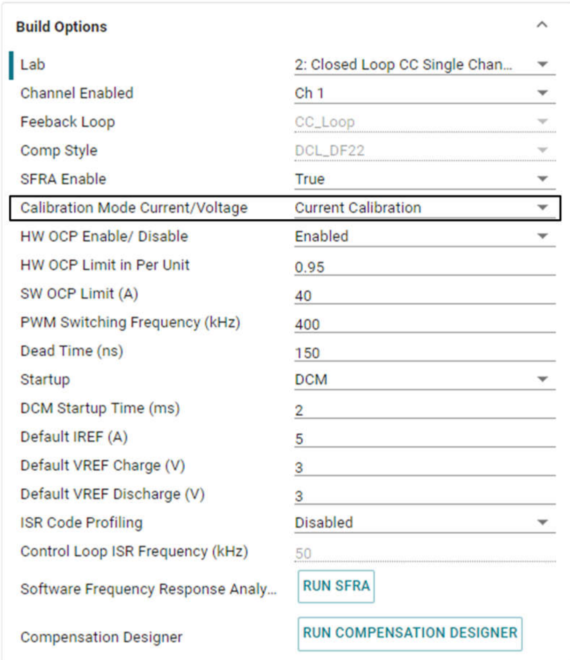


Figure 3-32. Build Options for Current Calibration

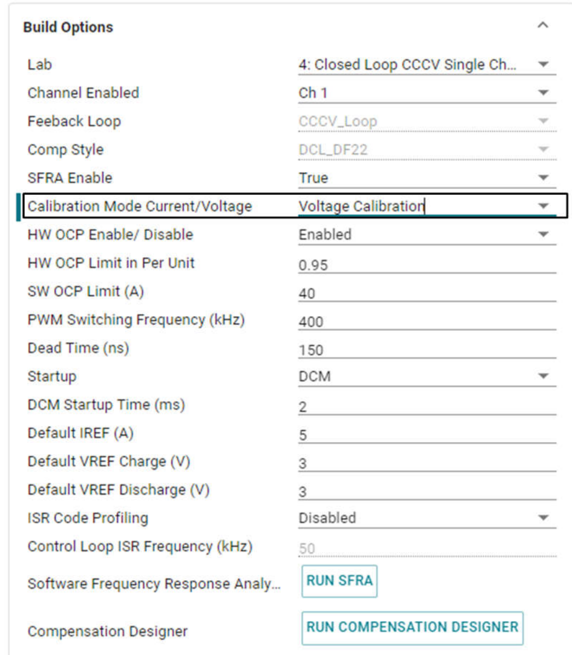


Figure 3-33. Build Options for Voltage Calibration

3.5 Test Results

3.5.1 Current Loop Load Regulation

Table 3-2. Charge Mode Load Regulation

FSR (A)	**50**							
Output Mode	**Charging**				**Discharging**			
TIDA_010090 I_{SET} (A)	0.1	1	10	25	0.1	1	10	25
ELOAD CV Mode	**Current Measured Through the Current Sense Resistor**							
V_{SET} (V)	**I_{actual} (A)**	**I_{actual} (A)**	**I_{actual} (A)**	**I_{actual} (A)**	**I_{actual} (A)**	**I_{actual} (A)**	**I_{actual} (A)**	**I_{actual} (A)**
1.0	0.101	1.000	9.999	24.998	0.099	1.000	9.996	24.994
2.0	0.101	1.000	9.9984	24.997	0.099	0.998	9.996	24.993
3.0	0.1	1.001	9.9984	24.9966	0.098	0.998	9.997	24.994
4.0	0.101	1.001	10.000	24.997	0.1	0.999	9.996	24.993
Error (mA)	1	1	1.6	3.4	2	2	4	7
Error (% FSR)	0.002	0.002	0.0032	0.0068	0.004	0.004	0.008	0.014

3.5.2 Current Loop Linearity Test

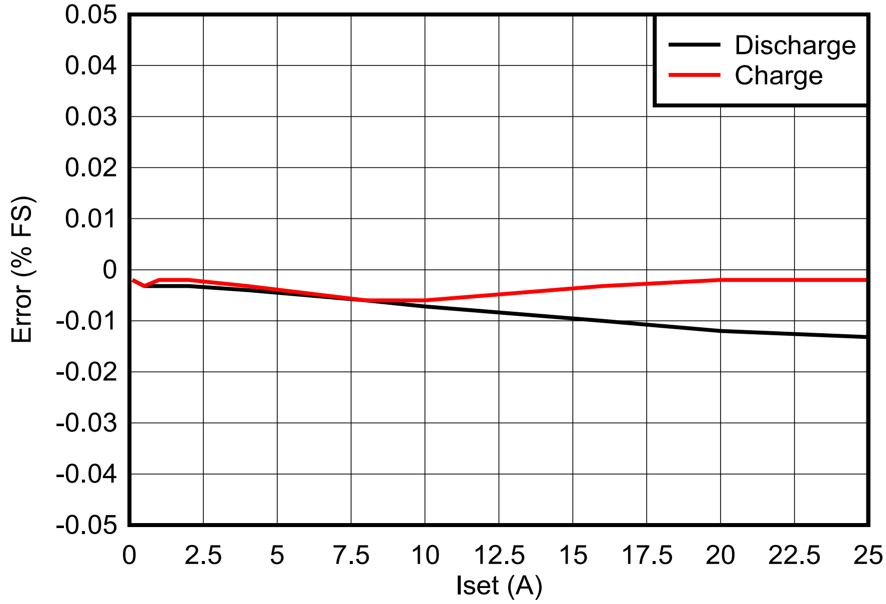


Figure 3-34. Current Loop Linearity Test

3.5.3 Voltage Loop Linearity Test

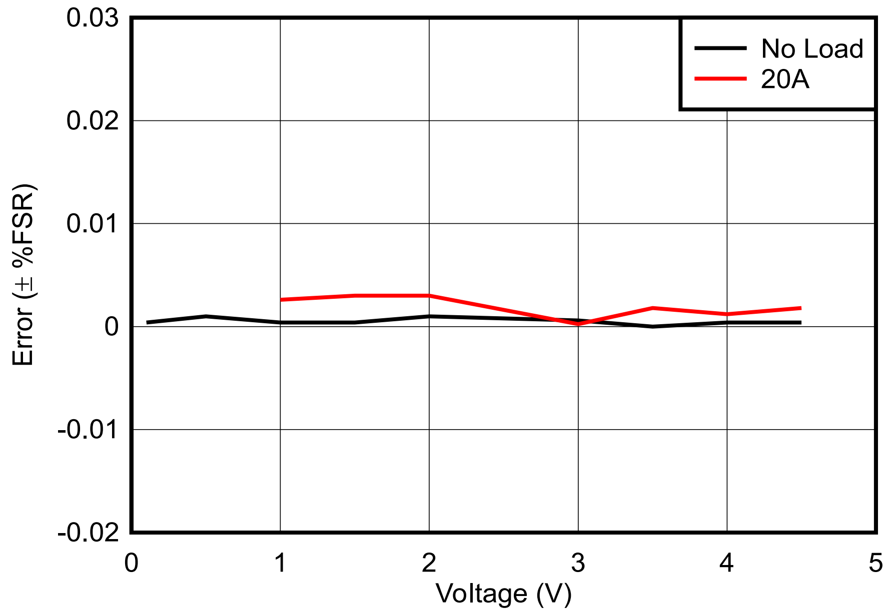


Figure 3-35. Voltage Loop Linearity Test

3.5.4 DCM Start-Up

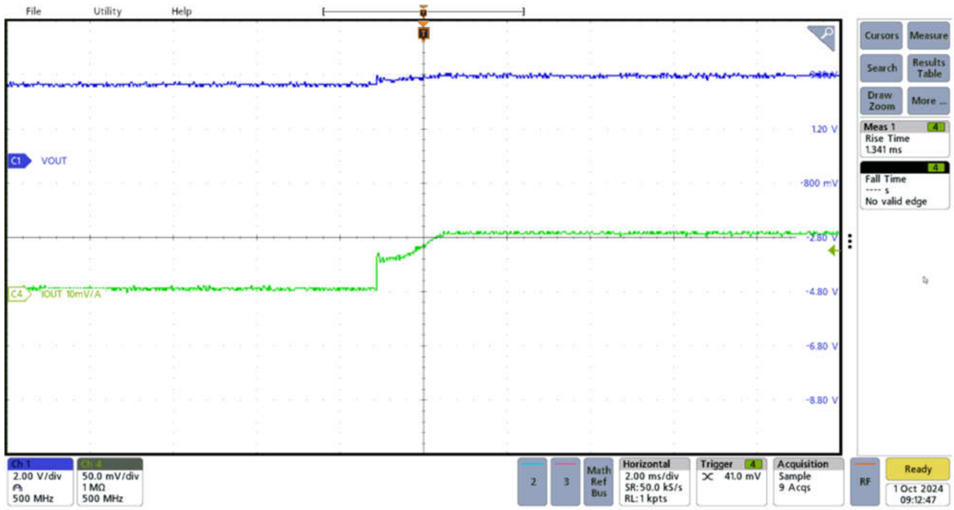


Figure 3-36. Transient Response at 5A Start-Up

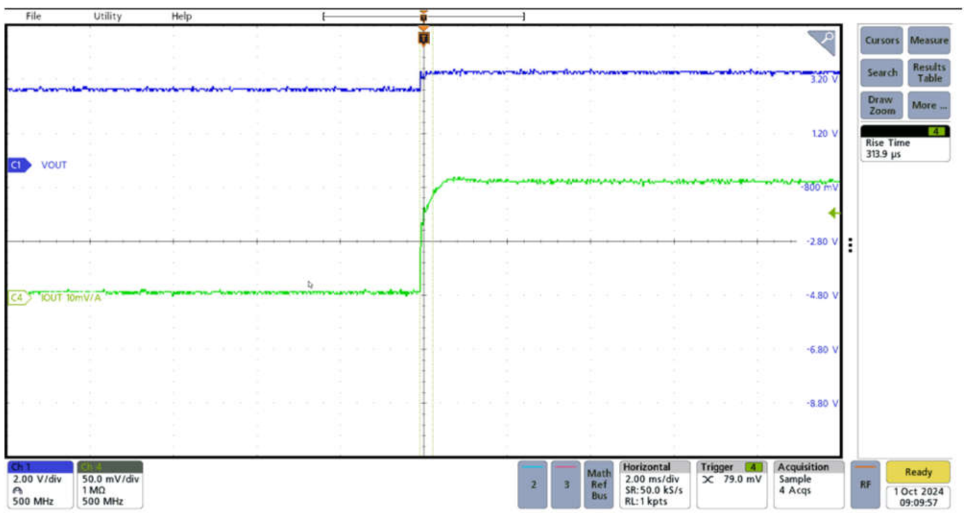


Figure 3-37. Transient Response at 10A Start-Up

3.5.5 Bidirectional Current Switching Time

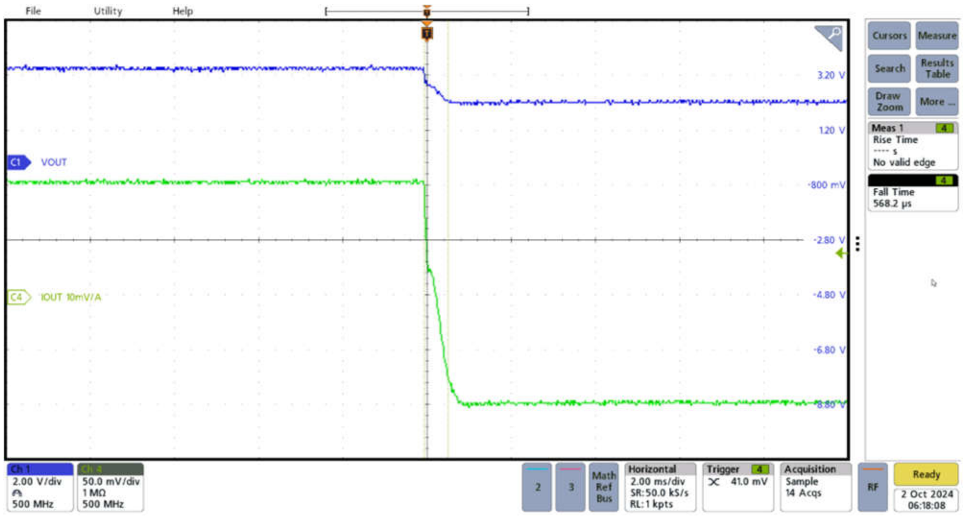


Figure 3-38. Charge to Discharge Transient Response

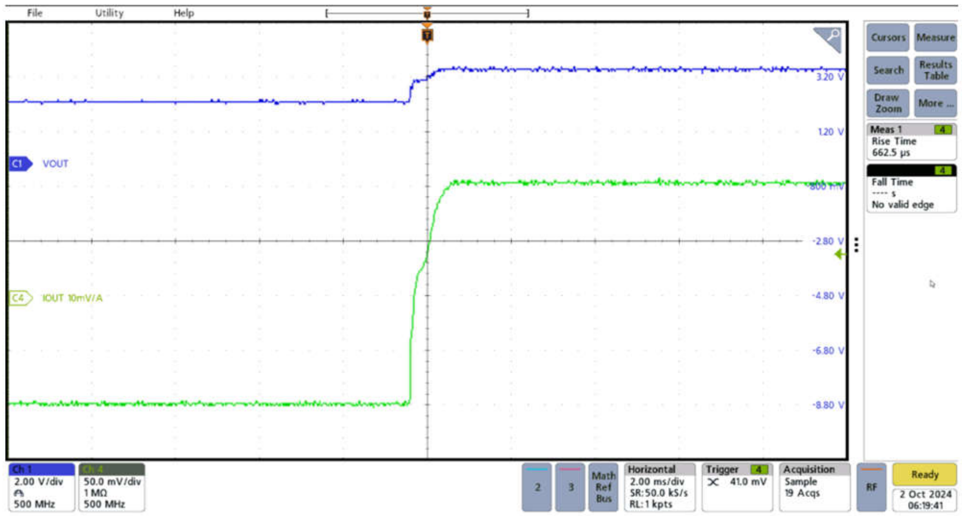


Figure 3-39. Discharge to Charge Transient Response

3.5.6 Thermal Performance

Figure 3-40 shows thermal image of the reference design when two channels are sourcing 20A, and the other two channels are sinking 20A. Channel 1 is shorted with Channel 2, and Channel 3 is shorted with Channel 4 for this measurement.

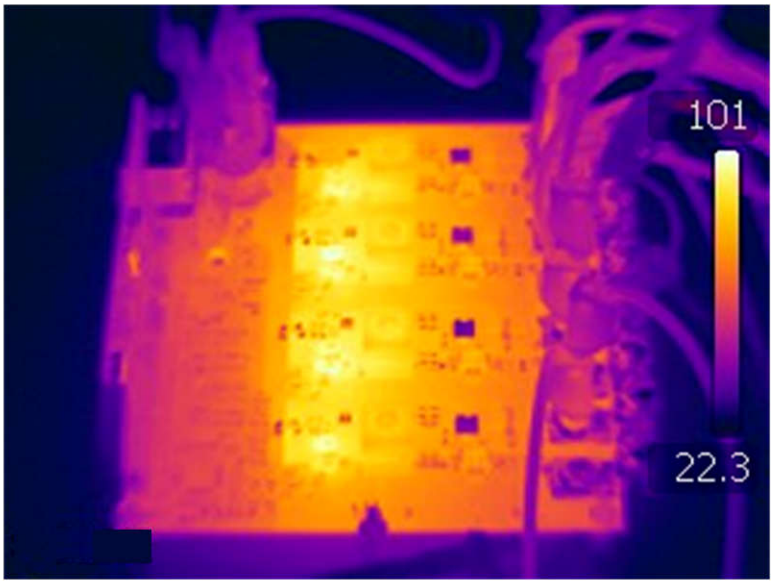


Figure 3-40. Thermal Image

4 Design and Documentation Support

4.1 Design Files

4.1.1 Schematics

To download the schematics, see the design files at TIDA-010090.

4.1.2 BOM

To download the bill of materials (BOM), see the design files at TIDA-010090.

4.2 Tools and Software

Tools

TMDSCNCD28P65X	TSM320F28P65x 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, *ADS8588S 16-Bit, High-Speed, 8-Channel, Simultaneous-Sampling ADC 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

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

5 About the Author

SHAURY ANAND is a systems engineer at Texas Instruments, where he is responsible for developing reference designs for Test and Measurement applications. Shaury earned his bachelor’s degree (B.Tech) in electrical engineering from the Indian Institute of Technology, Roorkee.

The author thanks JINGQUAN ZHU, ETHAN YU, SEN WANG, OZINO ODHARO, and TIM PRICE for the support given with this reference design.

IMPORTANT NOTICE AND DISCLAIMER

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