Design Guide: TIDM-2008/TIDM-1007

Bidirectional Interleaved CCM Totem Pole Bridgeless PFC Reference Design Using C2000™ MCU

Description

This reference design illustrates a method to control a bidirectional interleaved continuous conduction mode (CCM) totem pole (TTPL) bridgeless power factor correction (PFC) power stage using a C2000™ microcontroller (MCU) and LMG3410R070. This power topology is capable of bidirectional power flow (PFC and grid-tied inverter) and it uses Gallium Nitride (GaN) devices, which enables higher efficiency and reduction in size of the power supply. The design supports phase shedding and adaptive dead time for efficiency improvements, input cap compensation scheme for improved power factor at light loads, and non-linear voltage loop to reduce voltage spikes under transient in PFC mode. The hardware and software available with this reference design accelerates time to market.

Resources

TIDM-02008	Design Folder
TMS320F280049, TMS320F28075	Product Folder
LMG3410R070, UCC27714, ISO7831	Product Folder
OPA2376, SN74LVC1G3157DRYR, TLV713	Product Folder
C2000WARE-DIGITALPOWER-SDK	Tool Folder

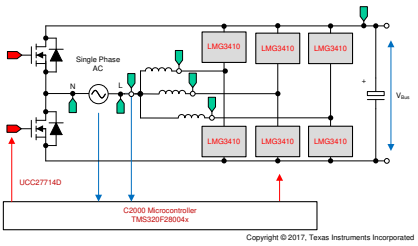
Ask our TI E2E™ support experts

Features

- Interleaved, 3.3-kW, Single-Phase, Bidirectional Bridgeless CCM Totem Pole PFC Stage
- 100-kHz Pulse Width Modulation (PWM) Switching
- Programmable Output Voltage, 380-V DC Output Nominal
- Less Than 2% Total Harmonic Distortion (THD)
- Greater Than 98% Peak Efficiency
- powerSUITE™ Support for Easy Adaptation of Design for User Requirement
- Software Frequency Response Analyzer (SFRA) for Quick Measurement of Open Loop Gain
- Soft Starting of PWM for Reduced Zero Current Spike in TTPL PFC
- Software Support for F28004x Using Driver Library
- Same Source Code Maintained When Running Control Loop on C28x or CLA

Applications

- Onboard Chargers for Electronic Vehicles (EVs)
- Telecom Rectifier
- Drives, Welding, and Other Industrial
- Energy Storage System (ESS)



Note

The TIDM-02008 is the updated version of the TIDM-1007 (unidirectional PFC). These two designs have the same hardware, but the software update in the TIDM-02008 makes it a bidirectional PFC. As the TIDM-02008 includes all of the features provided by the original design, it is a superset of the TIDM-1007, and the technical support for the TIDM-1007 will be continued as a part of the TIDM-02008 updates.

Note

For LMG3410R070 product and availability, refer to LMG3410R070 600-V 70mΩ GaN with integrated driver and protection. This device is not AEC-Q100 qualified. Contact TI for additional information.

1 System Description

Bidirectional interleaved TTPLPFC is an attractive topology for EV chargers, industrial applications and grid-tied energy storage systems with the trend for higher power, higher efficiency, and higher power density. Figure 2-1 shows the implementation of the bidirectional TTPL bridgeless PFC as it is on the TIDM-02008 board.

1.1 Key System Specifications

Table 1-1 describes the bidirectional interleaved CCM TTPL PFC reference design power specifications.

Table 1-1. Key System Specifications

PARAMETER	SPECIFICATION	
	PFC mode	Inverter mode (grid-tied)
Input voltage	AC 120 Vrms VL-N, 60 Hz or AC 230 Vrms VL-N , 50 Hz	380-V DC bus Nominal
Input current	16-A RMS maximum	10-A maximum
Output voltage	380-V DC bus Nominal	AC 120 Vrms VL-N, 60 Hz or AC 230 Vrms VL-N , 50 Hz
Output current	10-A maximum	16-A RMS maximum
Power rating	1.65 KW at single phase 120 Vrms or 3.3 KW at single phase 230 Vrms	
Current THD	<2% at 120-Vrms L-N rated load	
Efficiency	Peak 98.7% at 230-Vrms input, peak >97.7% at 120-Vrms input	Peak 98.3% at 230-Vrms output, peak >97.3% at 120-Vrms output
Primary filter inductor	478 µH	
Output capacitance	880 µF	
PWM switching frequency	100 kHz	

WARNING

TI intends this reference design to be operated in a ***lab environment only and does not consider it to be a finished product*** for general consumer use.

TI Intends this reference design to be used only by ***qualified engineers and technicians*** familiar with risks associated with handling high-voltage electrical and mechanical components, systems, and subsystems.

There are ***accessible high voltages present on the board***. The board operates at voltages and currents that may cause shock, fire, or injury if not properly handled or applied. Use the equipment with necessary caution and appropriate safeguards to avoid injuring yourself or damaging property.

CAUTION

Do not leave the design powered when unattended.

High voltage! There are *accessible high voltages present on the board*. Electric shock is possible. The board operates at voltages and currents that may cause shock, fire, or injury if not properly handled. Use the equipment with necessary caution and appropriate safeguards to avoid injuring yourself or damaging property. For safety, use of isolated test equipment with over-voltage and over-current protection is highly recommended.

TI considers it the user's responsibility to confirm that the voltages and isolation requirements are identified and understood before energizing the board or simulation. *When energized, do not touch the design or components connected to the design*.

Hot surface! Contact may cause burns. Do not touch!

Some components may reach high temperatures >55°C when the board is powered on. The user must not touch the board at any point during operation or immediately after operating, as high temperatures may be present.

2 System Overview

Bidirectional interleaved TTPL PFC is an attractive topology for EV chargers and ESS with the trend for higher power, higher efficiency, and higher power density. Figure 2-1 shows the implementation of the bidirectional TTPL bridgeless PFC as it is in this reference design.

2.1 Block Diagram

Figure 2-1 shows the block diagram of this reference design with key TI components highlighted.

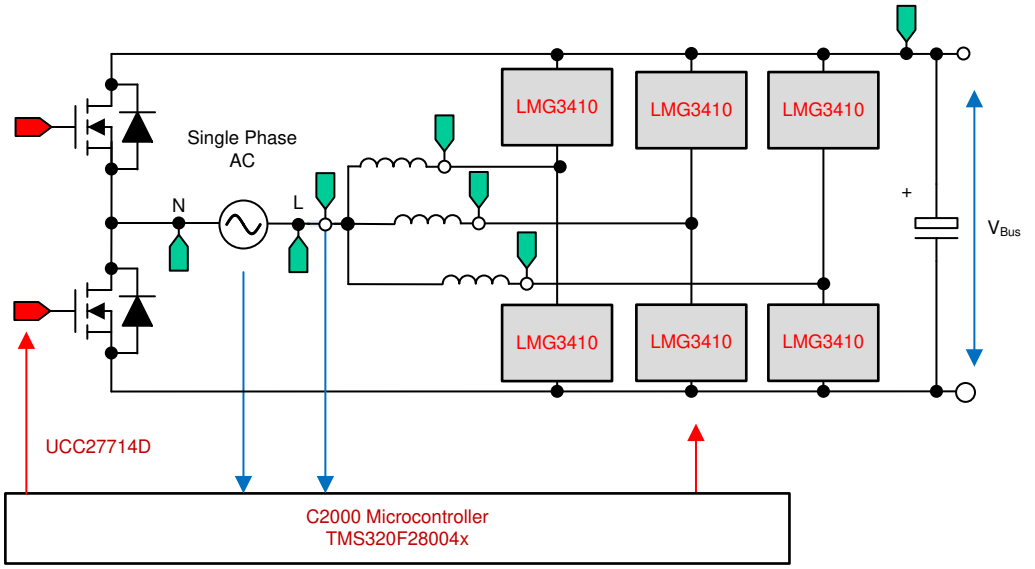


Figure 2-1. Power Topology Block Diagram

2.2 Design Considerations

The following detail the sensing circuit that is on this design. One can also refer to the *calculations.xlsx* file, which is available under the C2000Ware Digital Power SDK Install directory at *<install_location>\solutions\tidm_02008\hardware* for details on sensing circuit.

2.2.1 Input AC Voltage Sensing

The line and the neutral voltages are sensed by resistor divider to the ground of the board as shown in Figure 2-2. The two readings are subtracted on the controller to get the Vac sensing.

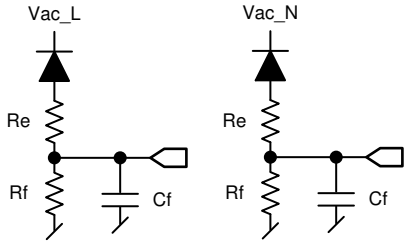


Figure 2-2. Input AC Voltage Sensing

2.2.2 Bus Voltage Sensing

Similarly the bus voltage is sensed by a resistor divider network as shown in Figure 2-3.

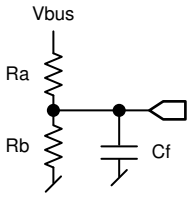


Figure 2-3. Bus Voltage Sensing Circuit

2.2.3 AC Current Sensing

A Hall effect sensor senses the total current. The Hall effect sensor has an inbuilt offset, and the range is different than what ADC can measure. Hence, the voltage is scaled to match the ADC range using the circuit as shown in Figure 2-4.

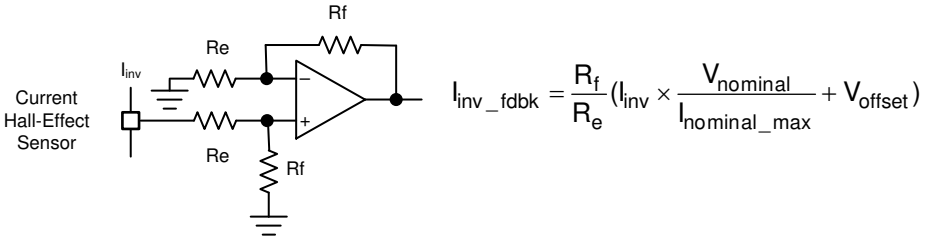


Figure 2-4. Current Sensing Using Hall Effect Sensor

2.2.4 Sense Filter

An RC filter filters the signals before connecting to the controller. A common RC filter is used for all the sensing signals on this design as shown in Figure 2-5.

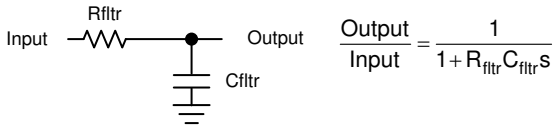


Figure 2-5. RC Filter

2.2.5 Protection (CMPSS)

Most power electronics converters require protection from overcurrent event. For this design multiple comparators are required, and references for the trip must be generated, as shown in Figure 2-6.

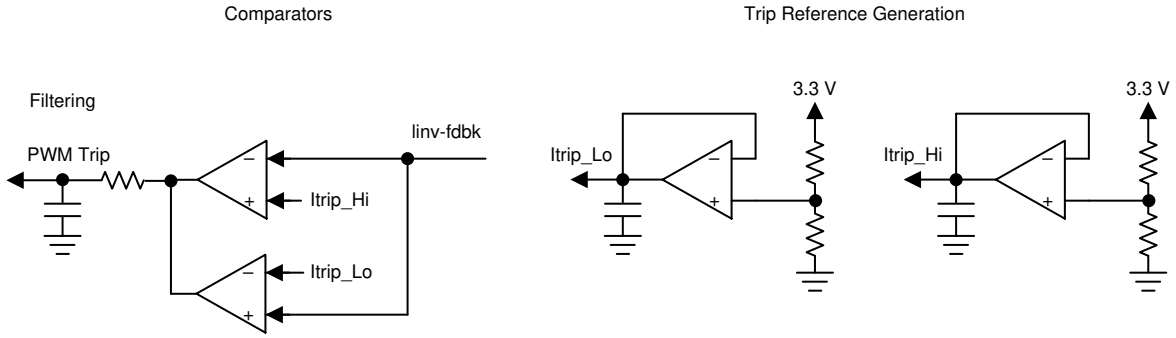


Figure 2-6. Trip Generation for PWM Using Comparators and Reference Generators

All this circuitry is avoided when using C2000 MCUs, such as TMS320F28377D, which have on-chip windowed comparator as part of the CMPSS that are internally connected to the PWM module that can enable fast tripping of the PWM. An on-chip windowed comparator saves board space and cost in the end application as extra components can be avoided using on-chip resources, as shown in Figure 2-7.

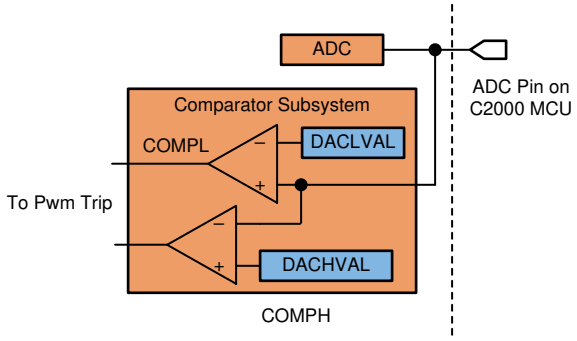


Figure 2-7. CMPSS Used for Overcurrent Protection

2.3 Highlighted Products

2.3.1 C2000™ MCU F28004x

C2000 MCUs are part of an optimized MCU family for real-time control application. Fast and high-quality analog-to-digital controller enables accurate measurement of the current and voltage signals, and an integrated comparator subsystem (CMPSS) provides protection for overcurrent and overvoltage without use of any external devices. The optimized CPU core enables fast execution of control loop. Trigonometric operations are accelerated using the on-chip trigonometric math unit (TMU). The solution also provides an option to use the control law accelerator (CLA) on the F28004x and F2837x. The CLA is a co-processor that can be used to alleviate CPU burden and enable faster-running loops or more functions on the C2000 MCU.

2.3.2 LMG3410R070

The LMG3410R070 single-channel GaN power stage contains a 70-mΩ, 600-V GaN power transistor and specialized driver in an 8-mm × 8-mm QFN package. Direct drive architecture is used to create a normally-off device while providing the native switching performance of the GaN power transistor. When the LMG3410 is unpowered, an integrated low-voltage silicon MOSFET turns the GaN device off through its source. In normal operation, the low-voltage silicon MOSFET is held on continuously while the GaN device is gated directly from an internally-generated negative voltage supply. The integrated driver provides additional protection and convenience features. Fast overcurrent, overtemperature, and undervoltage lockout (UVLO) protections help create a fail-safe system. The device's status is indicated by the FAULT output. An internal 5-V low-dropout regulator can provide up to 5 mA to supply external signal isolators. Finally, externally-adjustable slew rate and a low-inductance QFN package minimize switching loss, drain ringing, and electrical noise generation.

2.3.3 UCC27714

The UCC27714 is a 600-V high-side, low-side gate driver with 4-A source and 4-A sink current capability that is targeted to drive power MOSFETs or IGBTs. The device comprises of one ground-referenced channel (LO) and one floating channel (HO), which is designed for operating with bootstrap supplies. The device features excellent robustness and noise immunity with capability to maintain operational logic at negative voltages of up to –8 VDC on HS pin (at VDD = 12 V).

2.4 System Design Theory

2.4.1 PWM

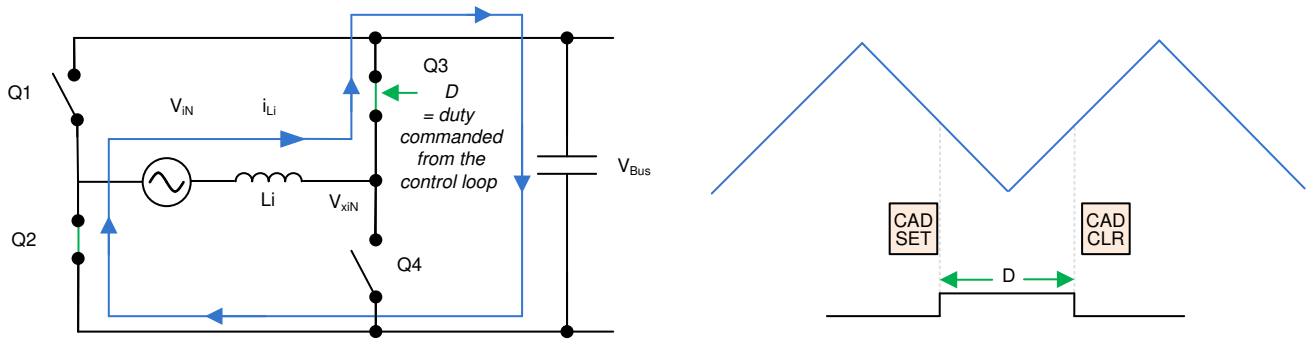


Figure 2-8. Single-Phase Diagram of TTPL PFC

Figure 2-8 shows a simplified diagram of a single phase of the interleaved TTPL PFC topology. To control this rectifier the duty cycle is controlled to regulate the voltage directly. This regulation is possible if the software variable *Duty* or *D* is set so that when it is equal to 1, Q3 is always ON, and the setting makes the voltage V_{xiN} equal to the V_{bus} voltage. When *Duty* is set to 0, Q3 never turns on, and Q4 is always connected to , which makes the V_{xiN} voltage go to 0.

2.4.2 Current Loop Model (PFC and Inverter mode)

The same control loop model applies to both PFC and grid-tied inverter operation. To understand the current loop model, first look at the inductor current closely. In Figure 2-8 the Duty cycle (D) is provided to the PWM modulator, which is connected to the switch Q3 and Q4. From here, Equation 1 is written as:

$$V_{xiN} = D \times V_{bus} \quad (1)$$

Note

When D is set to 1, Q3 is *on* all the time, and when D is 0, Q3 is *off* all the time.

To modulate the current through the inductor, the voltage $V_{xiN}$is regulated using the duty cycle control of Q3 and Q4 switches. It is assumed that the direction of current is positive in the direction from the AC line into the rectifier and that the grid is fairly stiff when using the DC bus feedforward and the AC voltage feedforward. Figure 2-9 shows the simplified current loop, and the current loop plant model is written as Equation 2.

$$H_{p\_i} = \frac{i^*_{Li}}{D} = \frac{1}{K_{v\_gain}} \times K_{i\_gain} \times K_{i\_fltr} \times G_d \times \frac{1}{Z_i} \quad (2)$$

Where:

- K_{v_gain} is the inverse of maximum bus voltage sensed,
- K_{vac_gain} is the inverse of maximum AC voltage sensed, (A factor to normalize this is applied in the feedforward and hence the current loop model is only dependent on the K_{v_gain})
- K_{i_gain} is the inverse of maximum AC current sensed, $\frac{1}{I_{AC\_MaxSense}}$
- K_{i_fltr} is the response of the RC filter connected from the current sensor to the ADC pin
- G_d is the digital delay associated with the PWM update and digital control is the current command
- i^*_{Li} is the current command

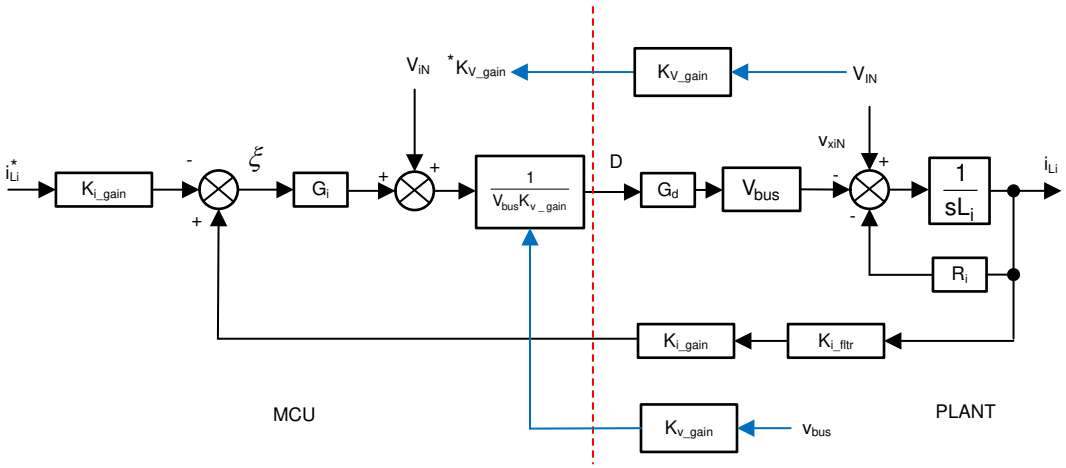


Figure 2-9. Current Loop Control Model

Note

The negative sign on the reference is because the current loop is thought to be regulating the voltage, V_{xiN}. To increase the current, V_{xiN} must be reduced—hence, the opposite sign for reference and feedback in Figure 2-9.

This current loop model is then used to design the current compensator. A simple proportional integral controller is used for the current loop.

Now, in the case of three interleaved phases, the current is simply three times more as the same duty cycle is provided to each leg. Hence, the plant model is given as Equation 3.

$$H_{p\_i} = \frac{i_{Li}^{*}}{D} = 3 \times \frac{1}{K_{v\_gain}} \times K_{i\_gain} \times K_{i\_fltr} \times G_d \times \frac{1}{Z_i} \quad (3)$$

This model is verified on this design using the SFRA library. Figure 2-10 shows the model versus measured open loop frequency response, which shows good correlation between the two.

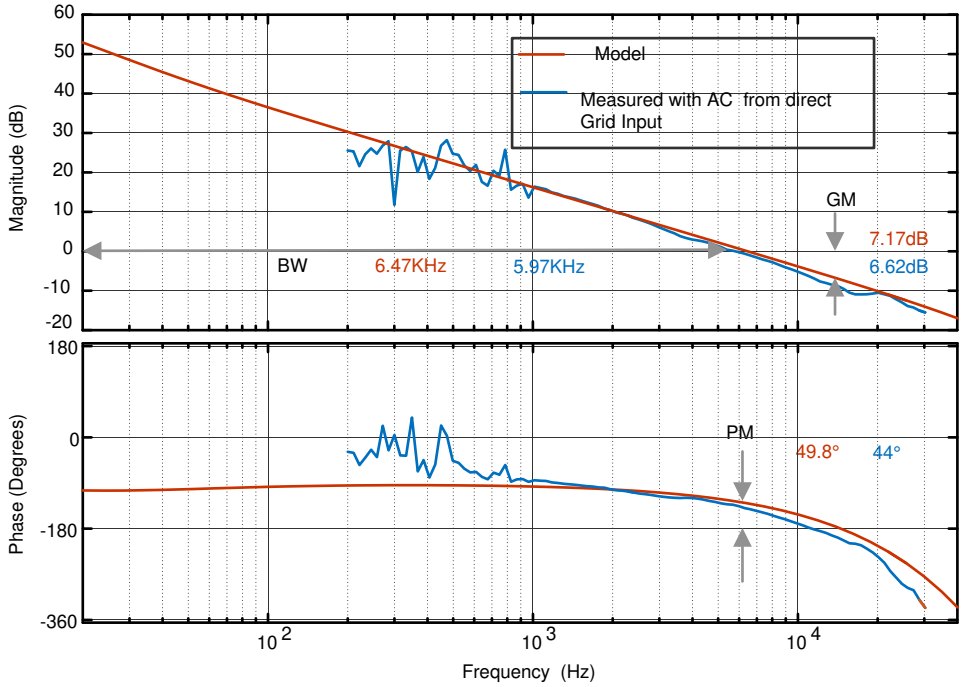


Figure 2-10. Gi, Current Open Loop Gain Measured Verus Modelled

2.4.3 DC Bus Regulation Loop (for PFC mode only)

The DC bus regulation loop is assumed to provide the power reference. The power reference is then divided by the square of the line voltages RMS to provide the conductance, which is further multiplied by the line voltage giving the instantaneous current command.

Small signal model of the DC bus regulation loop is developed by linearizing Equation 4 around the operating point.

$$i_{DC} v_{bus} = \eta V_{Nrms} i_{Nrms} \rightarrow \hat{i}_{DC} = \eta \frac{\overline{V}_{Nrms}}{\overline{V}_{bus}} i_{Li} \quad (4)$$

For a resistive load the bus voltage and current are related as shown in Equation 5.

$$\hat{V}_{bus} = \frac{R_L}{1 + sR_L C_o} \hat{i}_{DC} \quad (5)$$

The DC voltage regulation loop control model can be drawn as shown in Figure 2-11. An additional Vbus feedforward is applied to make the control loop independent of the bus voltage, and hence, the plant model for the bus control can be written as Equation 6.

$$H_{p\_bus} = H_{load} \times \eta \times K_{i\_gain} \times K_{v\_gain} \times K_{v\_flt} \quad (6)$$

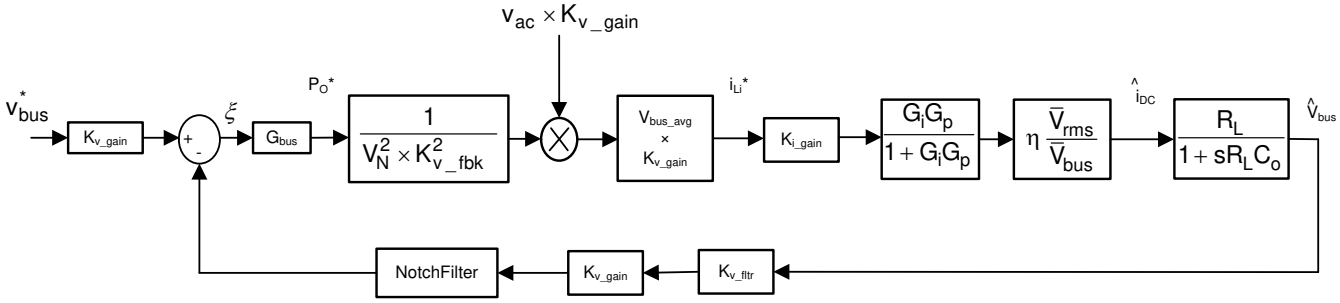


Figure 2-11. DC Voltage Loop Control Model

Using Figure 2-11, a proportional integrator (PI) compensator is designed for the voltage loop. The bandwidth of this loop is kept low as it is in conflict with the THD under steady state.

SFRA library is used to measure the frequency response on the voltage loop and verify the model. Figure 2-12 shows the modelled versus measured plots for the voltage loop.

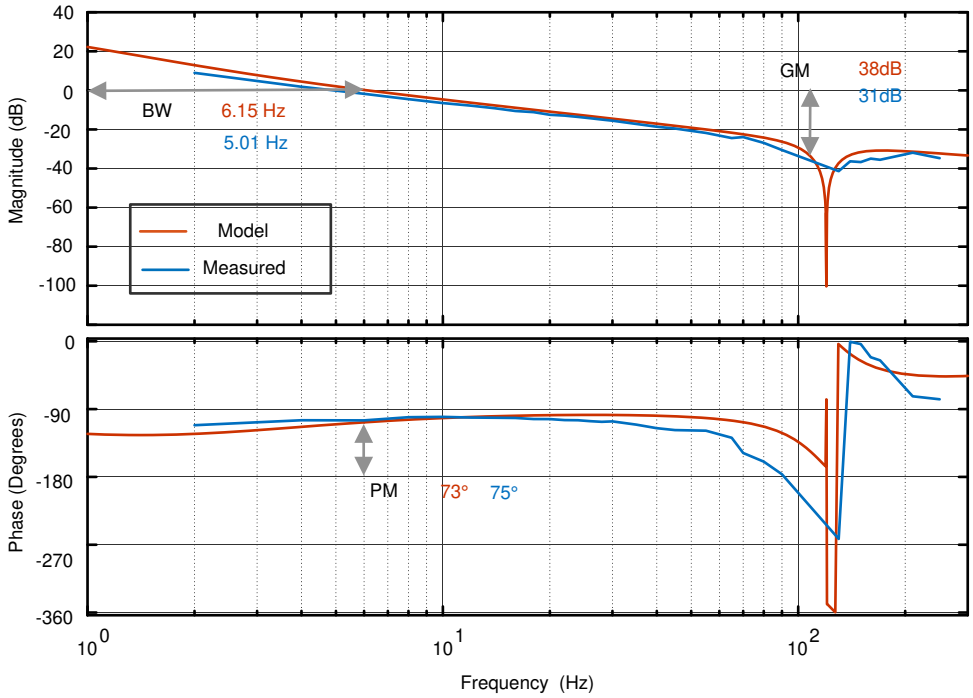


Figure 2-12. Gv, Voltage Loop Modeled Versus Measured

2.4.4 Soft Start Around Zero Crossing for Eliminate or Reduce Current Spike

Zero crossing current spikes is a challenging issue for TTPL PFC topologies. This issue is solved by implementing a soft start scheme with a state machine to turn on and off switches in a particular sequence.

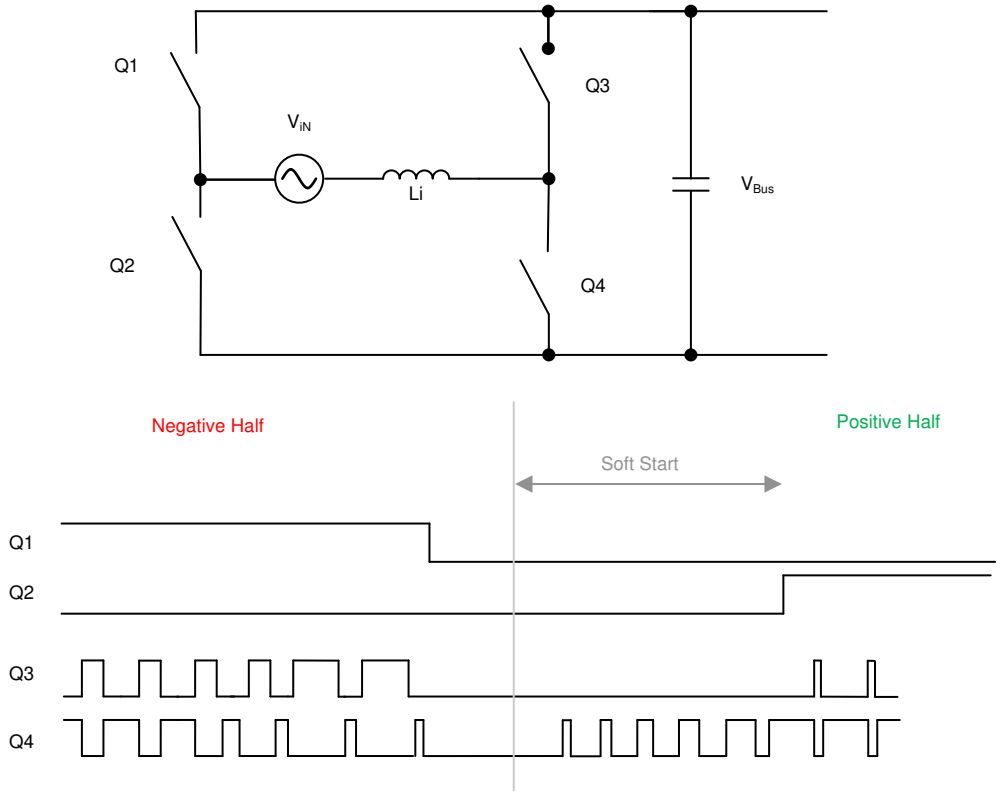


Figure 2-13. PWM Sequence With Soft Starting to Reduce Current Spike at Zero Crossing

Figure 2-13 shows the switching sequence when the AC wave goes from negative to positive. During the negative half Q1 is ON, Q3 is the active FET, and Q4 is the sync FET. During this time the voltage across Q2 is the DC bus voltage. When the AC cycle changes, Q2 must be on 100% or close to 100%. If Q2 is turned ON immediately, a huge positive spike results. Therefore, a soft-start sequence is used to turn Q4 ON as shown in Figure 2-13. The tuning of this soft start depends on the inductance value and other power stage parameters such as device Coss.

Another reason for a negative current spike around zero crossing is the relatively low AC voltage around the zero crossing. When Q3 is turned ON, though the duty cycle is low, a high-voltage difference is applied and can result in a high negative current spike. Therefore, a sufficient delay is applied before Q3 starts switching back again.

Similarly, Q2 is turned on after some delay after the soft start has started.

2.4.5 AC Drop Test

AC drop testing requires accurate detection of ac failure and ac back event to freeze and resume PFC operation. The biggest technical challenge in ac drop test is the PLL synchronization issue when ac gets back. Typically, SW based PLL required several cycles to catch up with the grid phase and therefore it can't provide current reference in phase with the voltage during this transitional period. The ideal ac drop test waveform is shown Figure 2-14. PLL immediately catches the ac grid and resume normal PFC operation as soon as ac gets back.

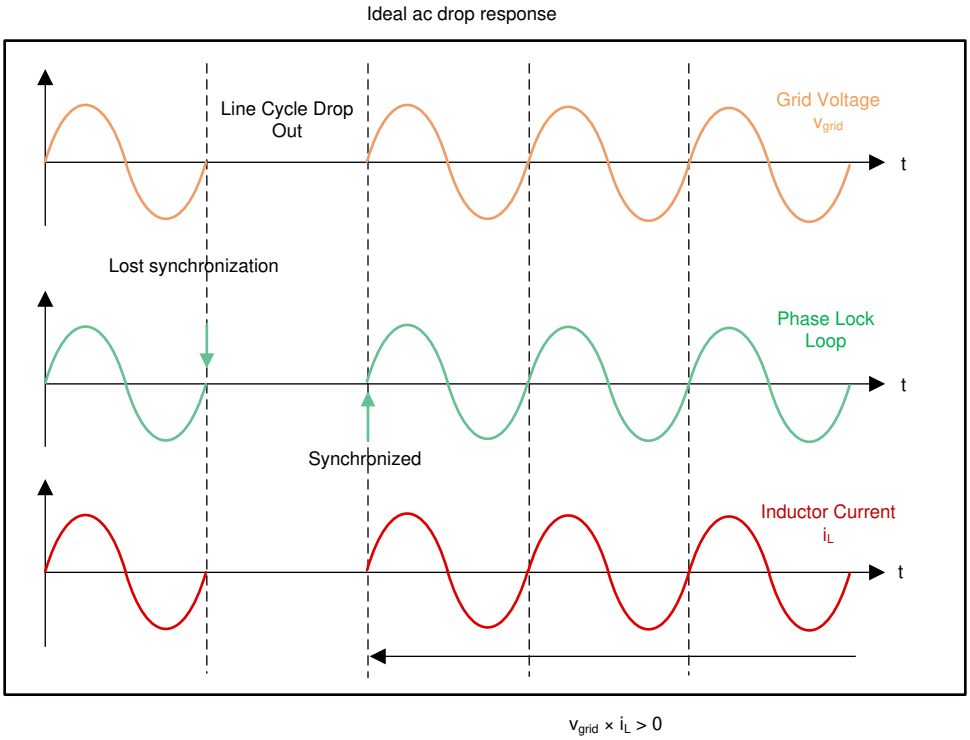


Figure 2-14. Ideal AC Drop Waveforms

Figure 2-15 presents the non-ideal ac drop solution. Due to the SPLL transitioning time, it requires extra time for PLL to work and does not provide fast dc bus recovery.

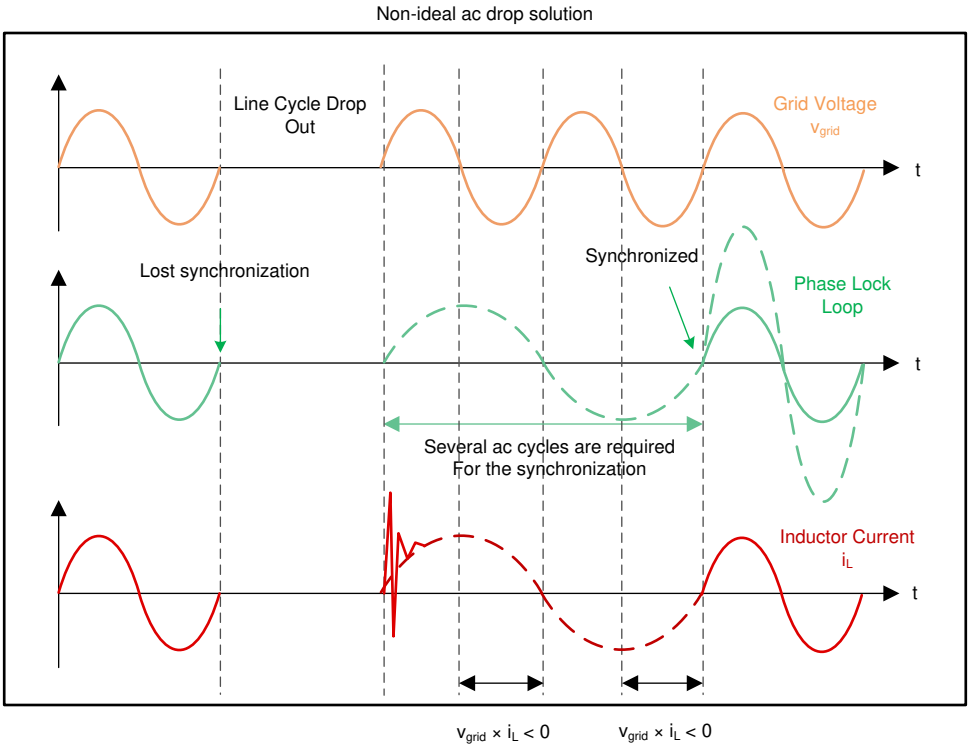


Figure 2-15. Non Ideal AC Drop Waveforms

In TIDM-02008, a virtual ac voltage is introduced. An internal sine wave is generated inside the MCU and the magnitude and the phase angle are synchronized with the actual ac grid when it is available. Once it is synchronized, the virtual signal provides the sinusoidal signal regardless of the actual grid voltage and it can be

utilized even during the ac drop period. As shown in Figure 2-16, virtual ac signal provides the current reference during the transition and once PLL catches up the ac grid, PLL provides the current reference as usual.

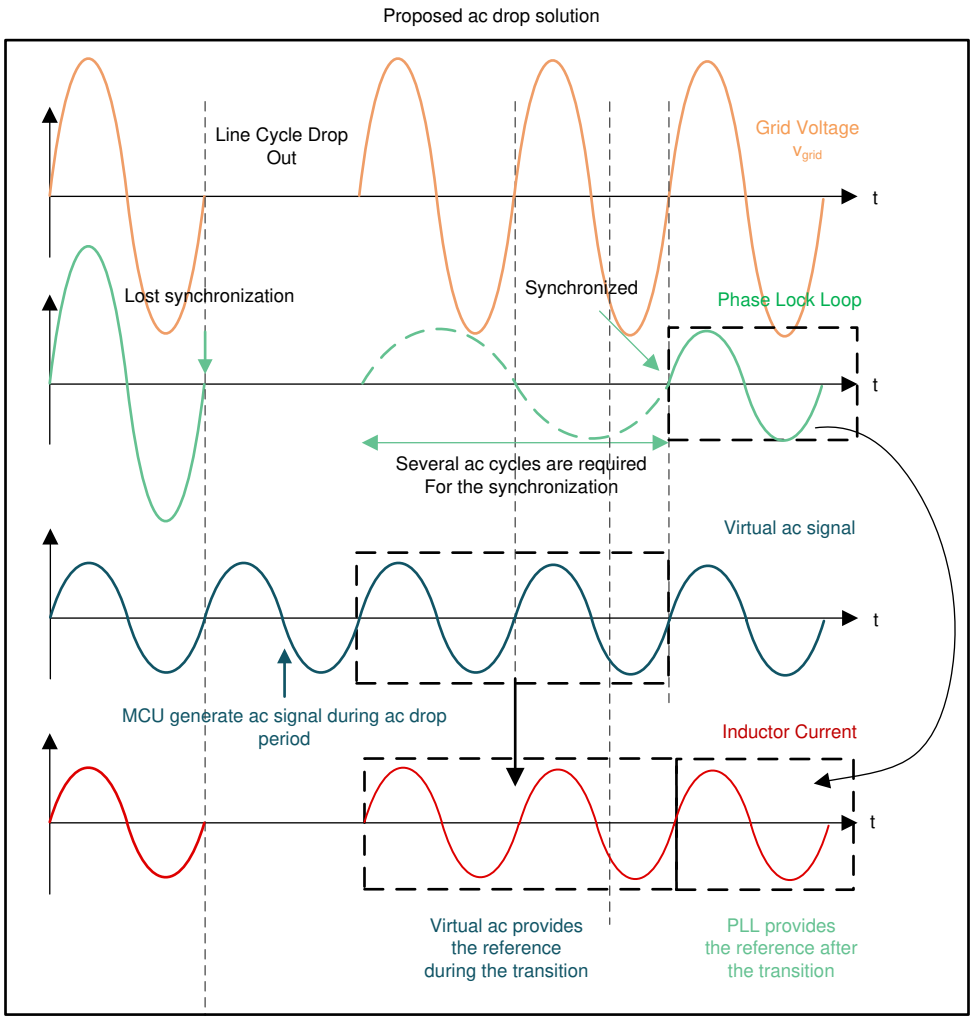


Figure 2-16. Proposed AC Drop Waveforms

A state machine shown in Figure 2-17 monitors ac voltage and makes a decision depending on the ac voltage status.

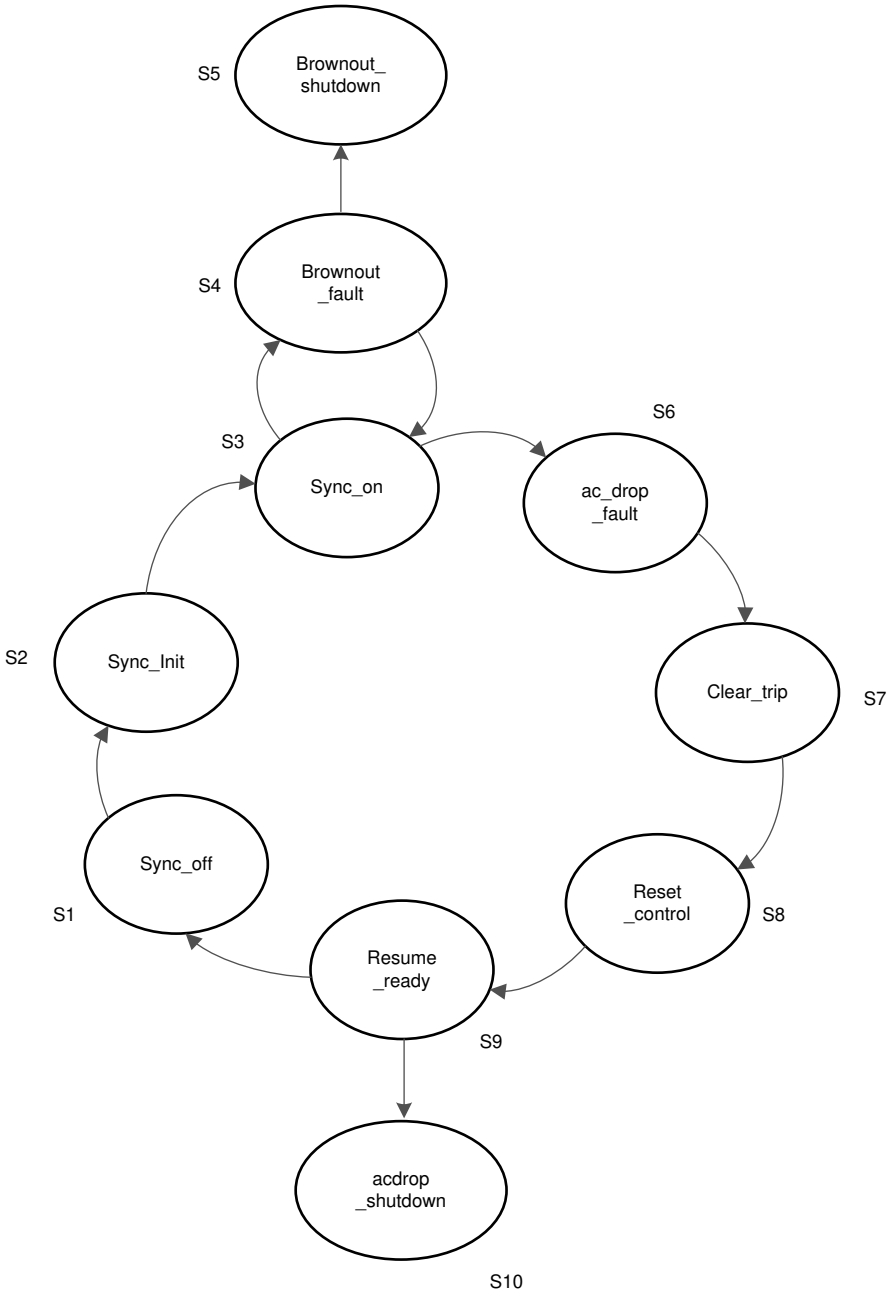


Figure 2-17. AC Drop State Machine

The actual AC drop test waveform is shown inFigure 2-18 It was captured under 900 w load with 120 V ac input.

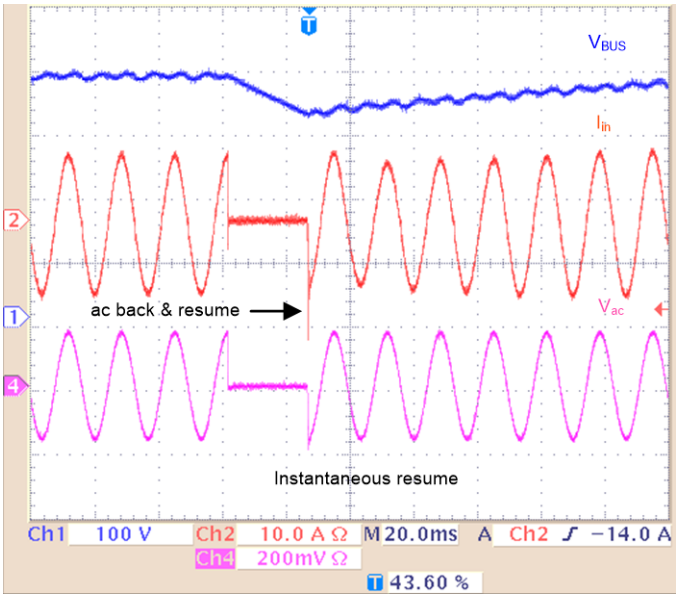


Figure 2-18. AC Drop Test Results Under 900 W

The AC drop test feature can be turned on and off by changing TTPLPFC_AC_DROP in *ttplpfc_user_settings.h*

```
#define TTPLPFC_AC_DROP 1 (1: enable, 0: disable)
```

3 Hardware, Software, Testing Requirements, and Test Results

3.1 Required Hardware and Software

3.1.1 Hardware

This section details the hardware and the different sections on the board. If only using the firmware of the design through powerSUITE, this section may not be valid.

3.1.1.1 Base Board Settings

The design follows a HSEC control card concept, and any device for which HSEC control card is available from the C2000 MCU product family can be potentially used on this design. The key resources used for controlling the power stage on the MCU are listed in Table 3-1. Figure 3-1 shows the key power stage and connectors on the design board. Table 3-2 lists the key connectors and their functions. To get started:

1. Make sure no power source is connected to the design.
2. Insert the control card in the J600 slot.
3. Connect a 12-V, 1-A DC power supply at TP604. For the ground terminal, use TP606 . Do not power up the supply.
4. Connect a 5-V, 1-A DC power supply at TP608. For the ground terminal, use TP609. Do not power up the supply.
5. Turn both the 12-V and 5-V power supply ON. The LED on the control card lights up and indicates the device is powered.

Note

The bias for the MCU is separated from the power stage, which enables safe bring up of the system in this set of instructions. The bias supply design that works with this converter is PMP20396

6. To connect JTAG, use a USB cable from the control card and connect it into a host computer.
7. For PFC mode, a single phase AC power supply can be connected to the input J100. Optionally in some labs a DC source may be required to test out the system safely. A resistive load of approximately 500 Ω and 400 W should be connected to the output at J104.
8. For Inverter mode, A DC power supply can be connected to the input J100. A resistive load of approximately 125 Ω and 200 W should be connected to the output at J104.
9. Current and voltage probes can be connected to observe the input current, input voltage, and output voltages, as shown in Figure 3-1.

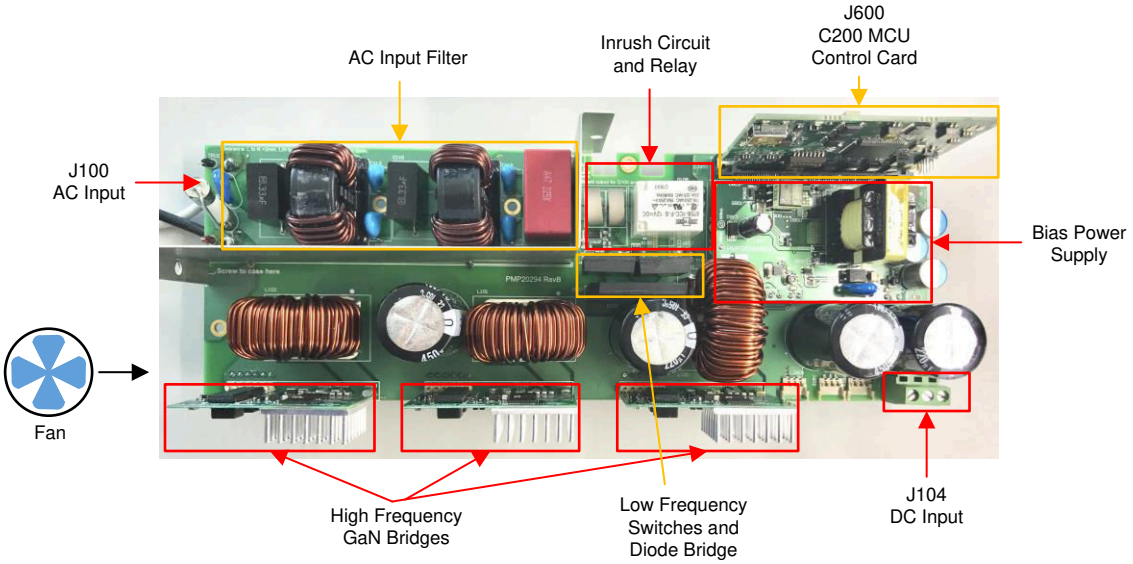


Figure 3-1. Board Overview

Table 3-1. Key Controller Peripherals Used for Control of Power Stage on Board

SIGNAL NAME	HSEC PIN NUMBER	FUNCTION
PWM-1A	49	PWM: low-frequency MOSFET leg, high-side switch
PWM-1B	51	PWM: low-frequency MOSFET leg, low-side switch
PWM-2A	53	PWM: high frequency GaN leg, high side switch, phase one
PWM-2B	55	PWM: high-frequency GaN leg, low-side switch, phase one
PWM-3A	50	PWM: high-frequency GaN leg, high-side switch, phase two
PWM-3B	52	PWM: high-frequency GaN leg, low-side switch, phase two
PWM-4A	54	PWM: high-frequency GaN leg, high-side switch, phase three
PWM-4B	56	PWM: high-frequency GaN leg, low-side switch, phase three
Iac	18	ADC with CMPSS: AC return current measurement
IL1	15	ADC with CMPSS : inductor current measurement Ph1
IL2	21	ADC with CMPSS : inductor current measurement Ph2
IL3	25	ADC with CMPSS : inductor current measurement Ph3
VL	20	ADC: AC voltage line
VN	17	ADC: AC voltage neutral
Vbus	24	ADC: bus voltage
In Rush Relay	57	GPIO: used to control the inrush relay
GaN Fault 1	58	GPIO: GaN fault signal phase one
GaN Fault 2	60	GPIO: GaN fault signal phase two
GaN Fault 3	62	GPIO: GaN fault signal phase three
AC Current Sense Gain Change	63	GPIO: controls the gain stage

Table 3-2. Key Connectors and Function

CONNECTOR NAME	FUNCTION	
	PFC	Inverter
J100	Input AC voltage	Output AC voltage
J104	Output DC bus voltage	Input DC bus Voltage
TP604	Input bias supply, 12-VDC, 1 A	
TP608	Input bias supply, 5-VDC, 1 A	
TP606/TP609	GND	
J600	HSEC control card connector slot	

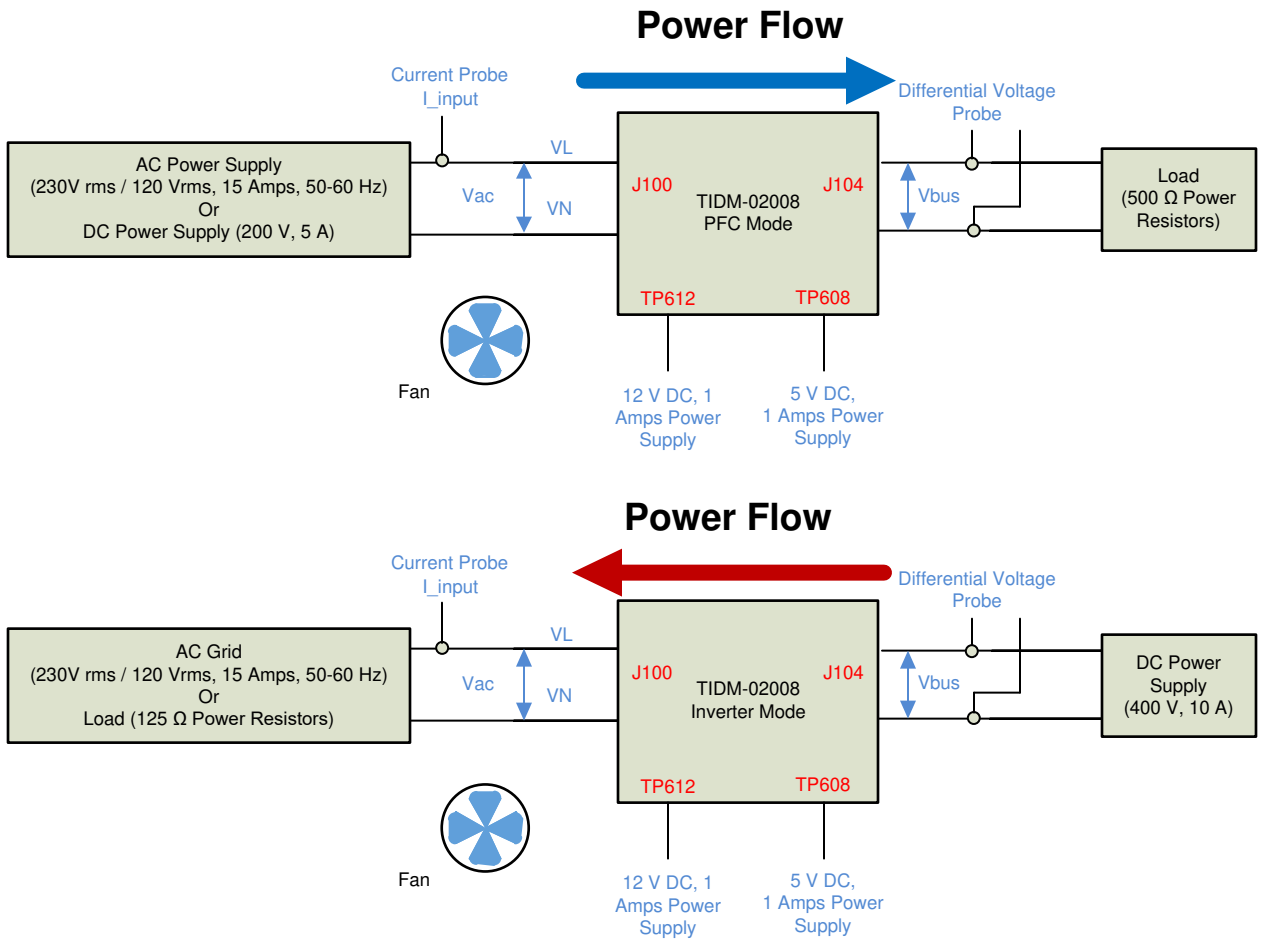


Figure 3-2. Hardware Setup to Run Software (PFC and Inverter Mode)

3.1.1.2 Control Card Settings

Certain settings on the device control card are required to communicate over JTAG and use the isolated UART port. The user must also provide a correct ADC reference voltage. The following are the required settings for revision A of the F280049M control card. One can also refer to the info sheet located inside C2000Ware at *<install_path>\c2000ware\boards\controlcards\TMDSCNCD280049C* or alternatively get it from the document http://www.ti.com/lit/pdf/spruic4

1. S1:A on the control card must be set on both ends to “ON (up) ” position to enable JTAG connection to the device and UART connection for SFRA GUI. If this switch is “OFF (down)” one cannot use the isolated JTAG built in on the control card nor can SFRA GUI communicate to the device.
2. J1:A is the connector for the USB cable that is used to communicate to the device from a host PC on which Code Composer Studio™ (CCS) runs.
3. A 3.3-V reference is desired for the control loop tuning on this design. Internal reference of the F28004x is used and for this S8 switch must be moved to the left i.e. pointing to VREFHI
4. A capacitor is connected between the isolated grounds on the control card, C26:A. It is advised to remove this capacitor for the best performance of this reference design.

3.1.2 Software

The software of this design is available inside C2000Ware Digital Power SDK and is supported inside the powerSUITE framework.

3.1.2.1 Opening Project Inside CCS

To start:

1. Install CCS from the *Code Composer Studio (CCS) Integrated Development Environment (IDE)* tools folder, CCSV9.3 or above is recommended.

2. Install C2000Ware DigitalPower SDK at the *C2000Ware Digital Power SDK* tools folder.
   - Note: powerSUITE is installed with the SDK in the default install.
3. Go to *View → Resource Explorer*. Below the TI Resource Explorer, go to *C2000Ware DigitalPower SDK*.

To open the reference design software as it is (opens firmware as it was run on this design and hardware, requires the board to be exactly the same as this reference design).

1. Under *C2000Ware DigitalPower SDK*, select *Development Kits → CCM Totem Pole PFC TIDM-02008*, and click on *Run <Import> Project*.
2. These steps import the project, and the development kit or designs page show up. This page can be used to browse all the information on the design including this user guide, test reports, hardware design files, and so forth.
3. Click *Import <device_name> Project*.
4. This action imports the project into the workspace environment, and a main.syscfg page with a GUI similar to Figure 3-3shows up.

Open reference design software for adaptation. The user can modify power stage parameters, which are then used to create the model of the power stage in Compensation Designer and can also modify scaling values for voltages and currents for a custom design.

1. Under*C2000Ware Digital Power SDK*click on *powerSUITE →Solution Adapter Tool* ().
2. Select *Single Phase CCM Totem Pole PFC* from the list of solutions presented.
3. Select the device this solution must run on the next page.
4. Once the icon is clicked, a pop-up window shows up asking for a location to create the project. One can also save the project inside the workspace itself. Once the location is specified, a project is created, and a GUI page appears with modifiable options for the solution (Figure 3-3).
5. This GUI can be used to change the parameters for an adapted solution, like power rating, inductance, capacitance, sensing circuit parameters, and so forth.

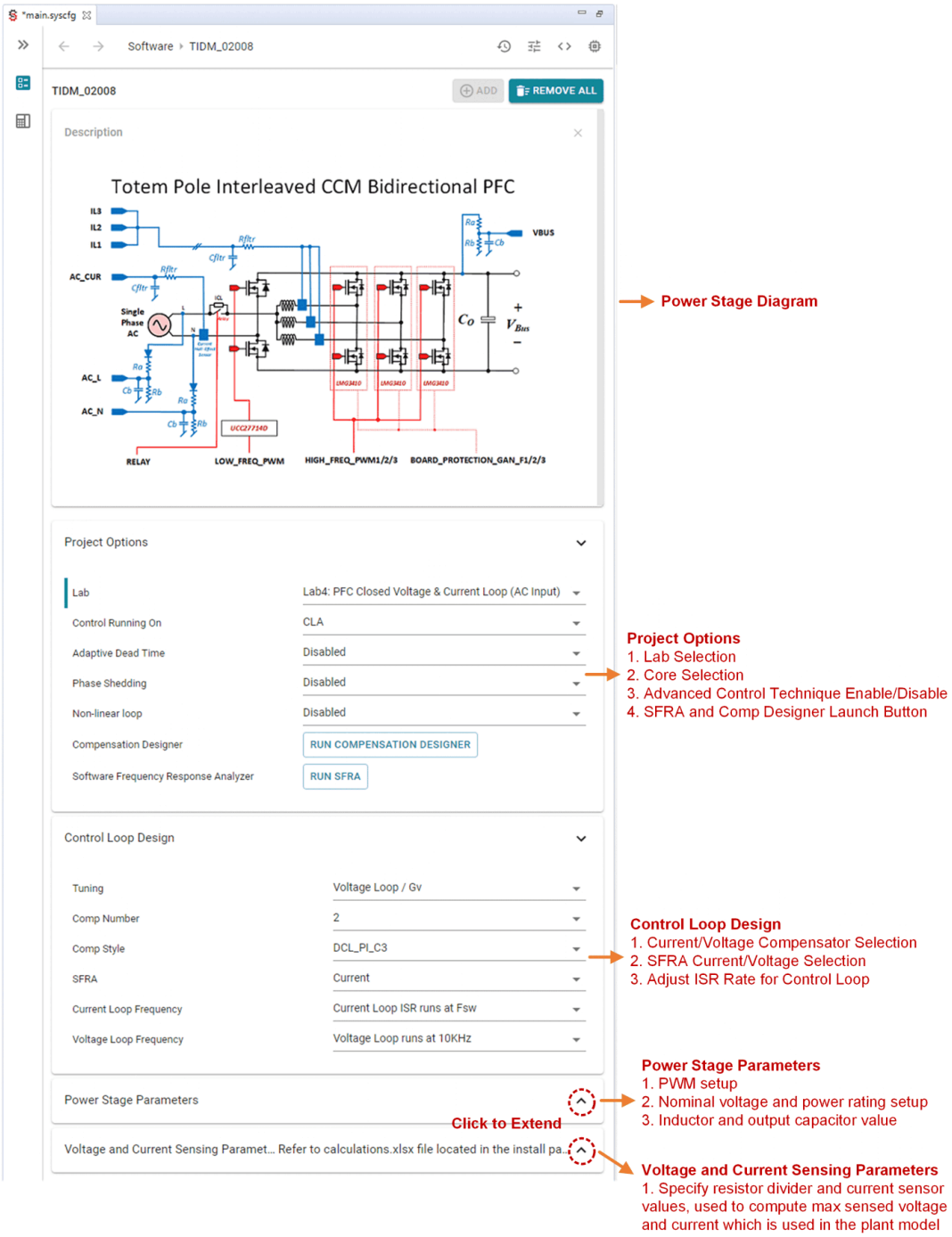


Figure 3-3. powerSUITE Page for CCM TTPL PFC Solution

3.1.2.2 Project Structure

Once the project is imported, the project explorer appears inside CCS as shown in Figure 3-4.

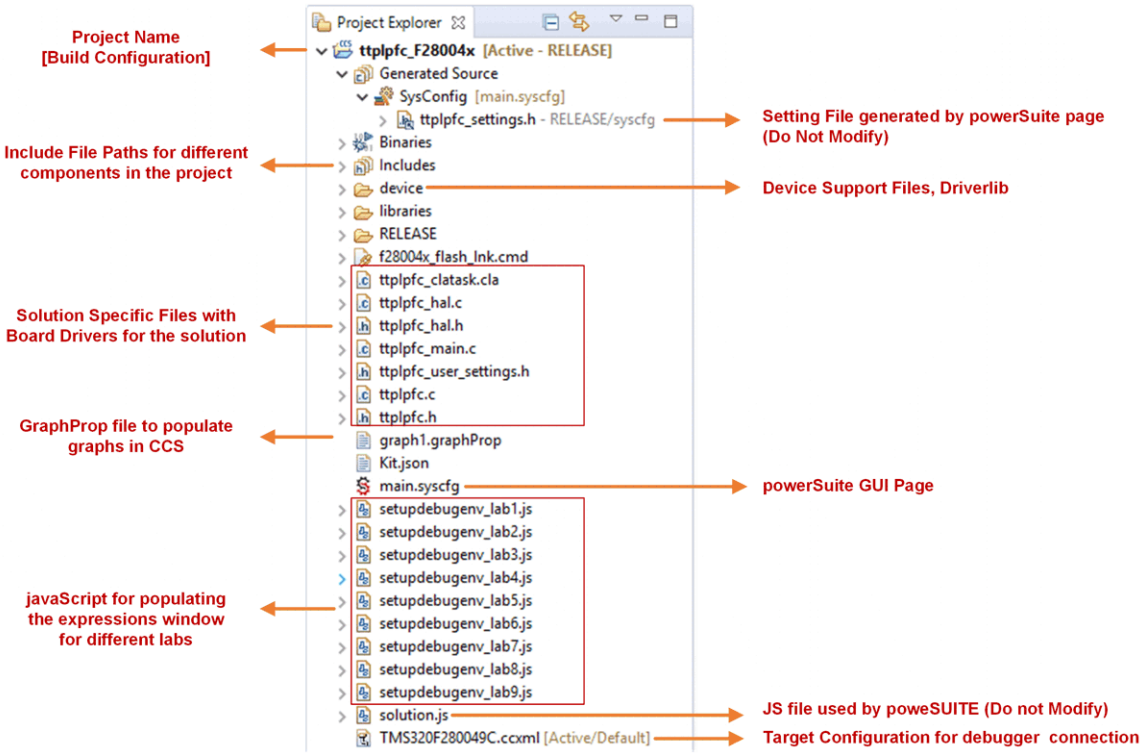


Figure 3-4. Project Explorer View of Solution Project

The general structure of the project is shown in Figure 3-5.

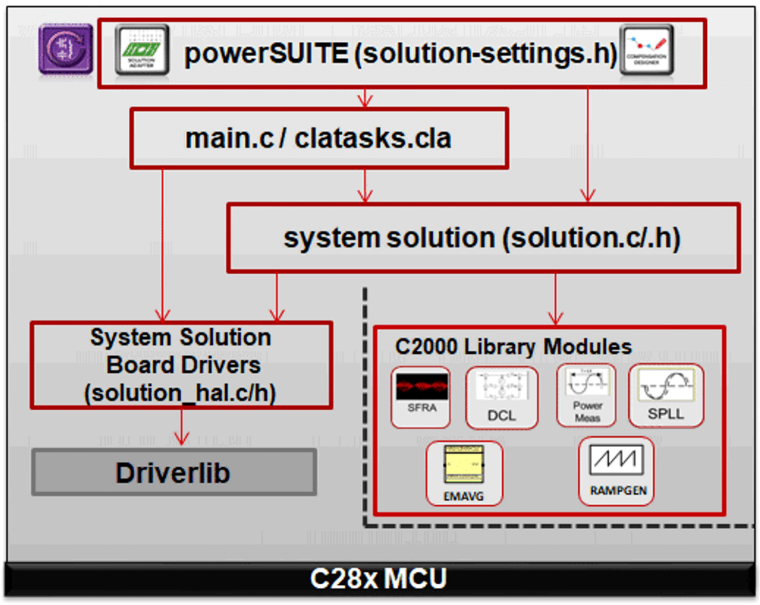


Figure 3-5. Project Structure Overview

Note

Figure 3-5 shows the project for F28004x; however, if a different device is chosen from the powerSUITE page, the structure is similar.

Solution specific and device independent files are *<solution>.c/h*. This file consist of the main.c file of the project and is responsible for the control structure of the solution.

For this design <solution> is *ttplpfc*.

Board-specific and device-specific files are *<solution>_hal.c/h*. This file consists of device specific drivers to run the solution.

The powerSUITE page can be opened by clicking on the *main.syscfg* file, listed under the project explorer. The powerSUITE page generates the *<solution>_settings.h* file. This file is the only file used in the compile of the project that is generated by the powerSUITE page. The user must not modify this file manually as the changes are overwritten by powerSUITE every time the project is saved. User can modify several settings in *<solution>_user_settings.h* file.

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

The setupdebugenv_build.js are provided to autopoulate the watch window variables for different builds

The *.graphProp files is provided to auto populate settings for the data logger graph

The project consists of an interrupt service routine, which is called every PWM cycle, and a current controller is run inside this ISR. In addition to this, there is a slower ISR of approximately 10 kHz that is called for running the voltage loop and the instrumentation ISR. A few background tasks (A0-A4 and B0-B4) are called in a polling fashion and can be used to run slow tasks for which absolute timing accuracy is not required, such as SFRA background and so on.

Figure 3-6 shows the software flow diagram of the firmware

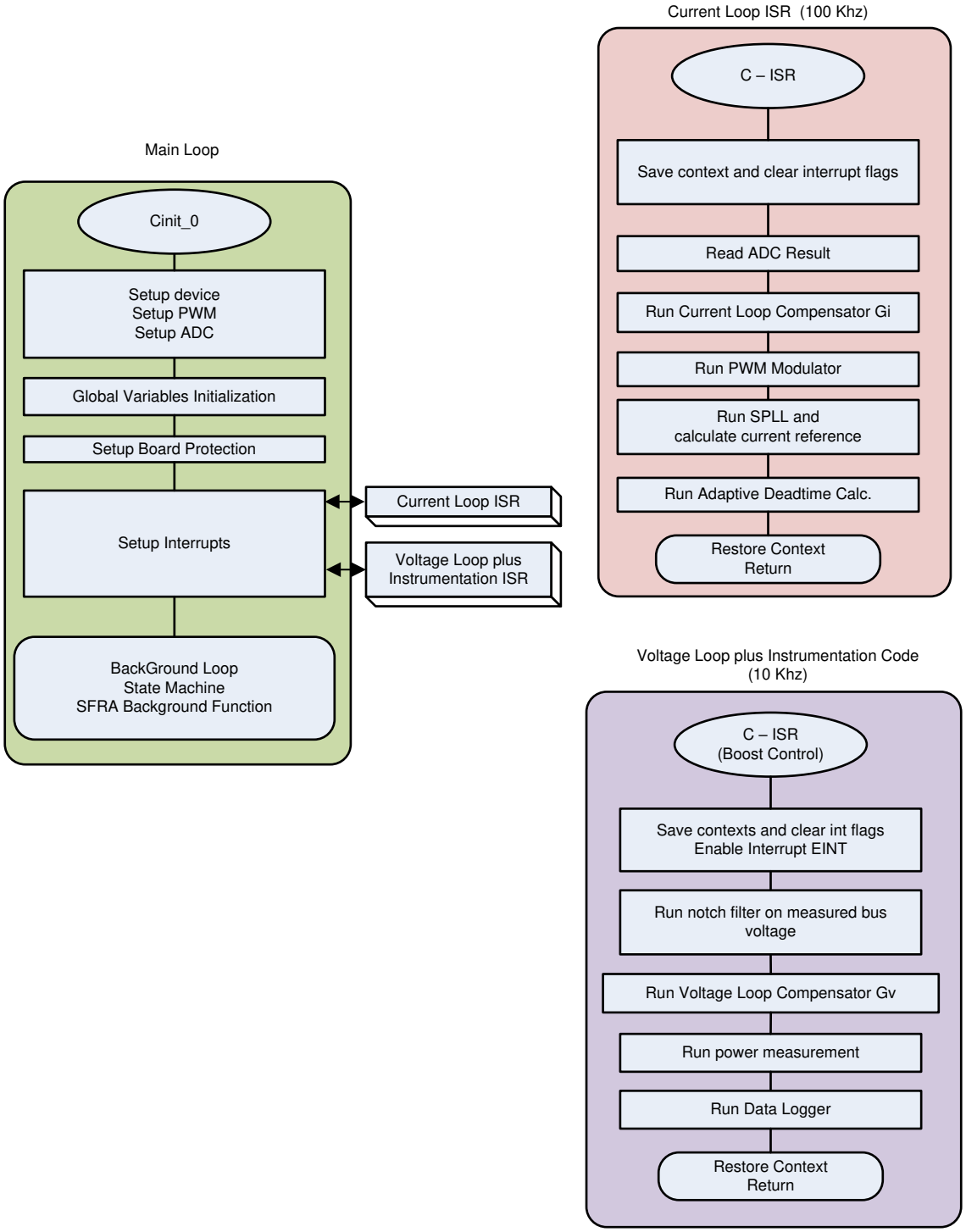


Figure 3-6. Project Structure Image

To simplify the system bring up and design the software of this reference design is organized in nine labs. The lab from 1 to 4 are designed to validate PFC operation and the lab from 5 to 9 are designed to validate the inverter operation.

- PFC Operation
  - Lab 1: Open Loop (DC input)
  - Lab 2: Closed Current Loop (DC input)
  - Lab 3: Closed Current Loop (AC input)

  - Lab 4: Closed Voltage and Current Loop (AC input)

- Inverter Operation
  - Lab 5: Open Loop (DC output)
  - Lab 6: Open Loop (AC output)
  - Lab 7: Closed Current Loop (DC output, Resistive load)
  - Lab 8: Closed Current Loop (AC output, Resistive load)
  - Lab 9: Closed Current Loop (AC output , Gried-tied condition)

These labs are detailed in Section 3.1.2.5. If using the reference design hardware, make sure the hardware setup is completed as outlined in Section 3.1.1.

3.1.2.3 Using CLA on C2000 MCU to Alleviate CPU Burden

The control law accelerator (CLA) is a co-processor available on the C2000 MCU family of devices. This co-processor enables offloading the control-ISR functions from the main C28x CPU core.

To run the control ISR on the CLA for solutions supported in powerSUITE, selection is achieved through a drop-down menu on the powerSUITE CFG page. The software structure of the powerSUITE solution is designed such that offloading the task to the CLA is simply a drop-down menu selection. The code is not duplicated and a single source for the solution algorithm is maintained even when code is run on the CLA or the C28x. This configuration enables flexible debugging of the solution.

The CLA features of each device varies slightly. For example, on the F2837xD, F2837xS, and F2807x, the CLA can support only one task at a given time, and there is no nesting capability. This configuration means that the task is not interruptible. Only one ISR can be offloaded to the CLA. On the F28004x, the CLA supports a background task from which a regular CLA task can nest. This configuration enables offloading two ISRs on the CLA.

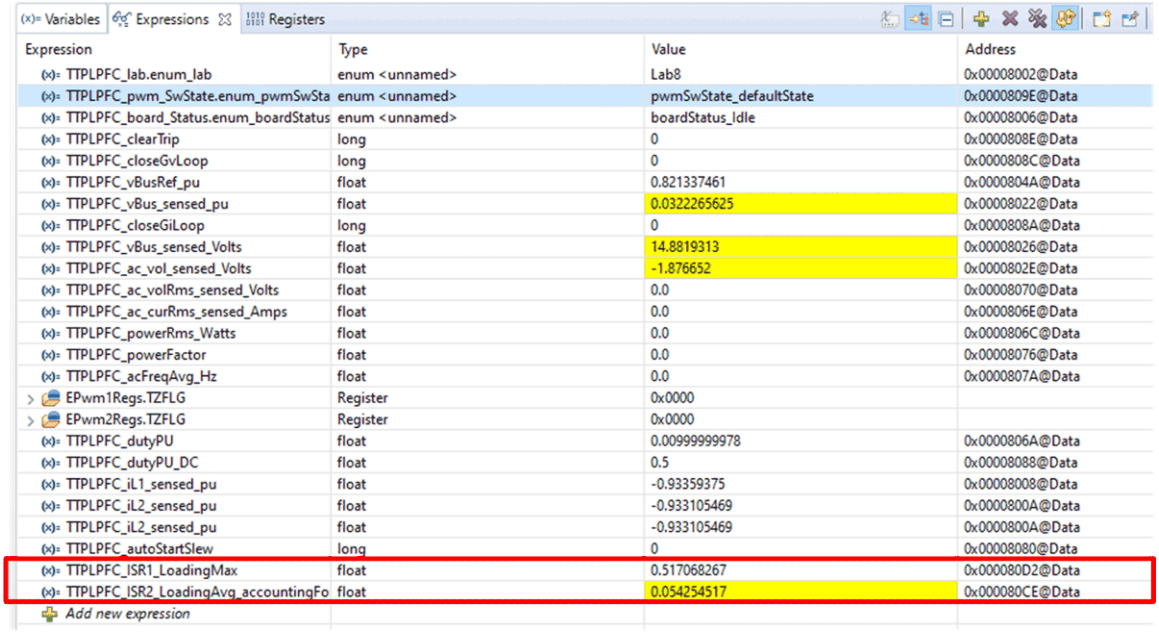
The CLA supports a background task from which it can nest into a CLA task. This configuration allows offloading two ISR functions to the CLA. For the F28004x, both the control ISR (100 kHz) for the current loop and the voltage loop and instrumentation ISR (10 kHz) are offloaded to the CLA.

For more information on the CLA, visit the CLA Hands-On Workshop and the respective device technical reference manuals.

3.1.2.4 CPU and CLA Utilization and Memory Allocation

The CPU utilization can be monitored by toggling GPIOs and capturing the waveforms using oscilloscope. Each ISR includes profiling functions that set GPIO pin high at the beginning of ISR and set GPIO pin low at the end of ISR. However, this method is no longer accurate when ISRs are nested.

To overcome the drawback of the oscilloscope-based method, XBAR and ECAP module are utilized to capture the toggling instant of GPIOs and MCU calculates ISR loading that accommodates nesting. Furthermore, this method provides ISR loadings directly on watch window and therefore, oscilloscope is not required. ISR1(100 kHz) is designed for inner current loop control. The outer voltage loop and instrumentations are implemented on ISR2(10 kHz). ISR1 and ISR2 loadins are presented in TTPLPFC_ISR1_LoadingMaxand TTPLPFC_ISR2_LoadingAvg_accountingForNesting respectively. Figure X captures the watch window in and when controls are running on CPU.



Expression	Type	Value	Address
TTPLPFC_lab.enum_lab	enum <unnamed>	Lab8	0x00008002@Data
TTPLPFC_pwm_SwState.enum_pwmSwSta	enum <unnamed>	pwmSwState_defaultState	0x0000809E@Data
TTPLPFC_board_Status.enum_boardStatus	enum <unnamed>	boardStatus_Idle	0x00008006@Data
TTPLPFC_clearTrip	long	0	0x0000808E@Data
TTPLPFC_closeGvLoop	long	0	0x0000808C@Data
TTPLPFC_vBusRef_pu	float	0.821337461	0x0000804A@Data
TTPLPFC_vBus_sensed_pu	float	0.0322265625	0x00008022@Data
TTPLPFC_closeGiLoop	long	0	0x0000808A@Data
TTPLPFC_vBus_sensed_Volts	float	14.8819313	0x00008026@Data
TTPLPFC_ac_vol_sensed_Volts	float	-1.876652	0x0000802E@Data
TTPLPFC_ac_volRms_sensed_Volts	float	0.0	0x00008070@Data
TTPLPFC_ac_curRms_sensed_Amps	float	0.0	0x0000806E@Data
TTPLPFC_powerRms_Watts	float	0.0	0x0000806C@Data
TTPLPFC_powerFactor	float	0.0	0x00008076@Data
TTPLPFC_acFreqAvg_Hz	float	0.0	0x0000807A@Data
EPwm1Regs.TZFLG	Register	0x0000	
EPwm2Regs.TZFLG	Register	0x0000	
TTPLPFC_dutyPU	float	0.00999999978	0x0000806A@Data
TTPLPFC_dutyPU_DC	float	0.5	0x00008088@Data
TTPLPFC_iL1_sensed_pu	float	-0.93359375	0x00008008@Data
TTPLPFC_iL2_sensed_pu	float	-0.933105469	0x0000800A@Data
TTPLPFC_iL2_sensed_pu	float	-0.933105469	0x0000800A@Data
TTPLPFC_autoStartSlew	long	0	0x00008080@Data
TTPLPFC_ISR1_LoadingMax	float	0.517068267	0x000080D2@Data
TTPLPFC_ISR2_LoadingAvg_accountingFo	float	0.054254517	0x000080CE@Data
Add new expression			

Figure 3-7. Watch expression for eCAP profiling

ISR loadings with advanced options enabled (phase shedding, adaptive dead time, non-linear loop, SFRA) can be measured in the same way by configuring main.syscfg. The ISR loadings for the worst case scenarios were captured in the table

	ISR1 (100 kHz)	ISR2 (10 kHz)
CPU utilization (Advanced options: All Off)	53%	6 %
CPU utilization (Advanced options: All On)	65%	9%

The total CPU usage is approximately 59 % without advanced options. If all the advance options are enabled, the total CPU usage is about 74 %. With the CLA option, the CPU burden is reduced to 0% when both ISRs are offloaded to the CLA. The worst case ISR loadings on CLA is shown in the table

	ISR1 (100 kHz)	ISR2 (10 kHz)
CLA utilization (Advanced options: All Off)	57 %	9 %
CLA utilization (Advanced options: All On)	79%	12 %

The advanced options obviously increase CPU usage due to additional computations. Other than that, the compiler optimization level, phase lock loop (PLL) method for grid synchronization also impact the CPU usage. The ISR loadings on Table x and y are captured with NOTCH SPLL (**#define** SPLL_METHOD_SELECT SPLL_1PH_NOTCH_SEL) and the compiler optimization level is 3. The reference for code optimization can be found at C2000™ C28x Optimization Guide.

The memory allocation is shown in Figure 3-8

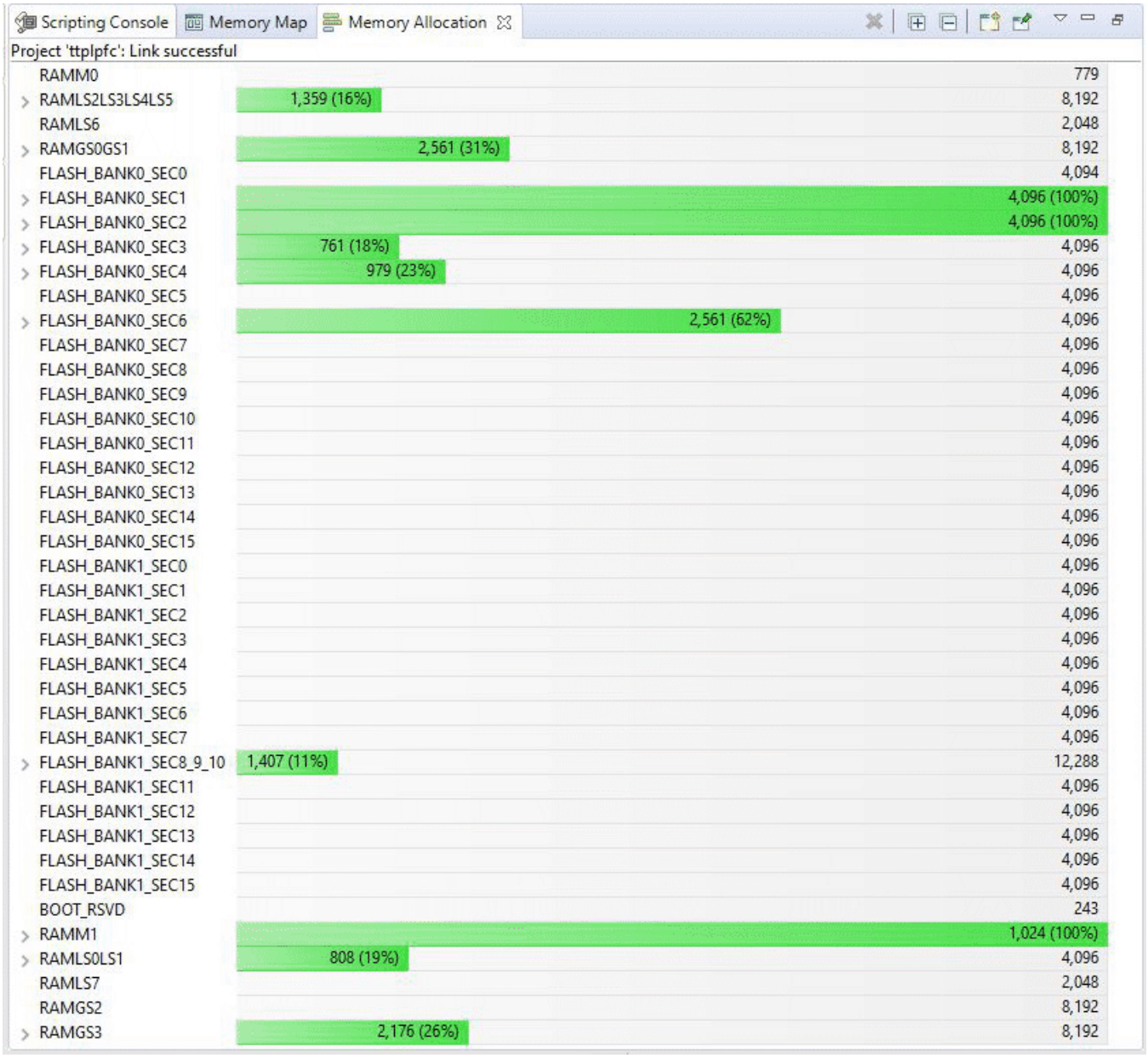


Figure 3-8. TIDM-02008 Memory Allocation

3.1.2.5 Running the Project

3.1.2.5.1 Lab 1: Open Loop, DC (PFC Mode)

In this lab the board is excited in open loop fashion with a fixed duty cycle. The duty cycle is controlled with dutyPU_DC variable. This build verifies the sensing of feedback values from the power stage and also operation of the PWM gate driver and ensures there are no hardware issues. Additionally calibration of input and output voltage sensing can be performed in this build. The software structure for this build is shown in Figure 3-9. There are two ISR in the system: fast ISR for the current loop and a slower ISR to run the voltage loop and instrumentation functions. Modules that are run in each ISR are shown in Figure 3-9.

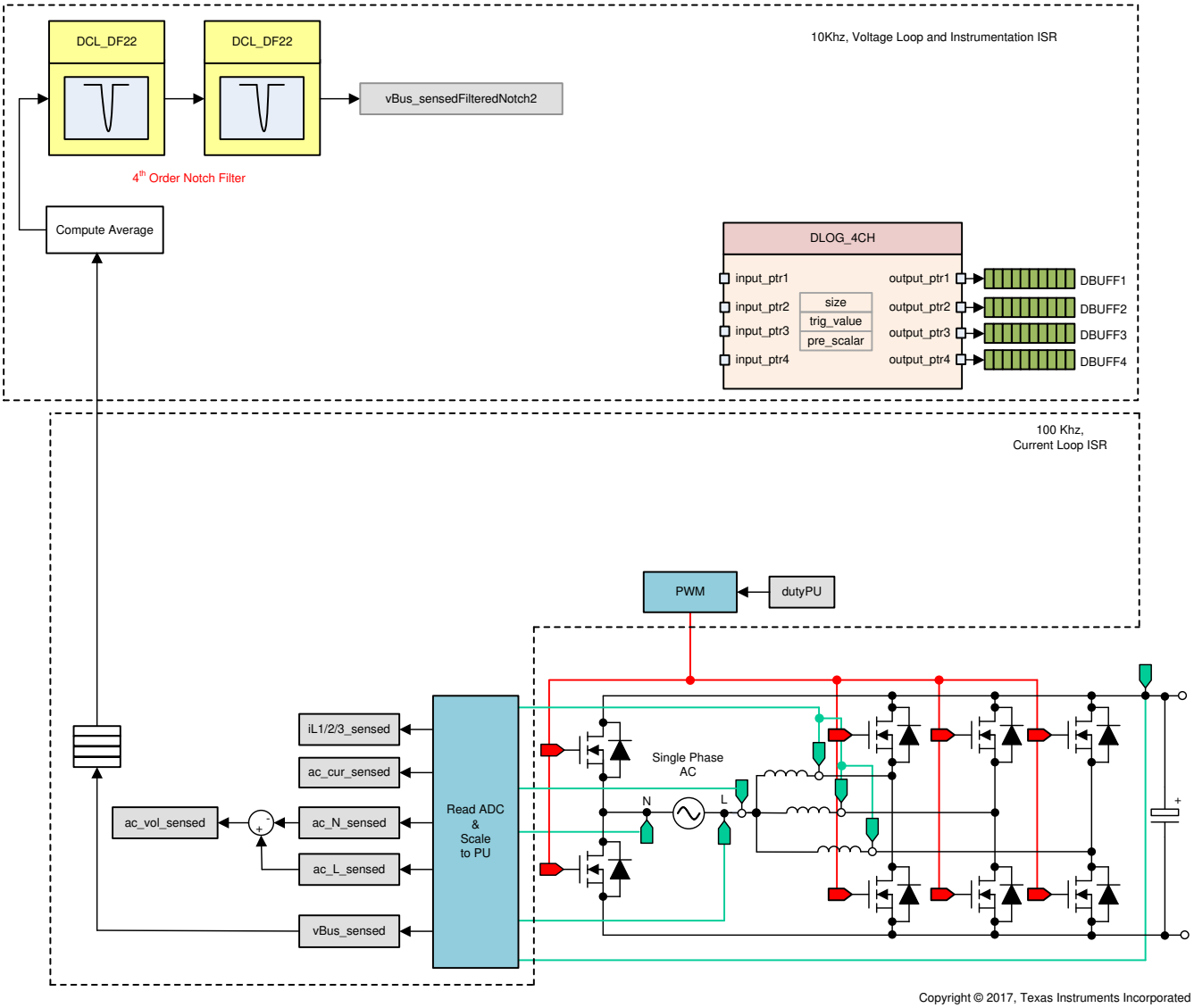


Figure 3-9. Lab 1 Control Software Diagram: Open Loop Project

Hardware setup is assumed to be similar to what was outline in the previous section. Figure 3-10 recaps the hardware setup for Lab 1 test.

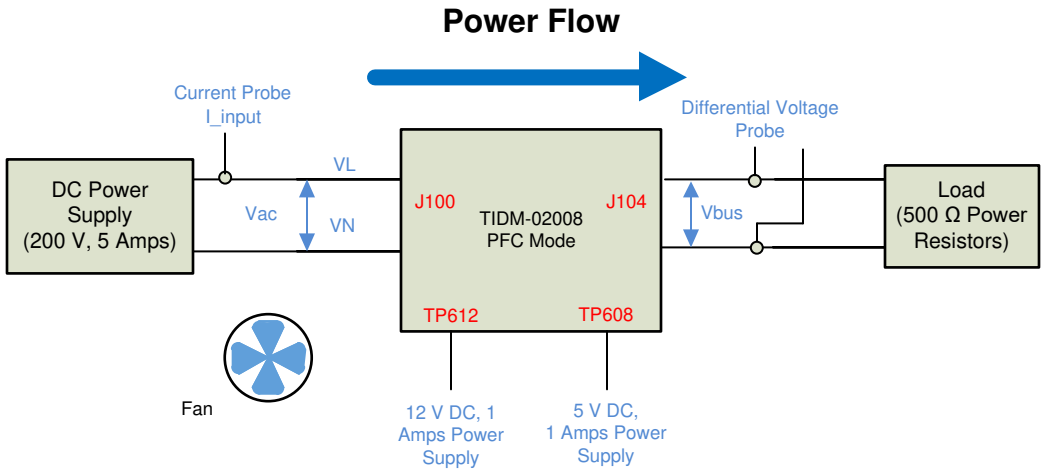


Figure 3-10. HW Setup for Lab 1

3.1.2.5.1.1 Setting Software Options for LAB 1

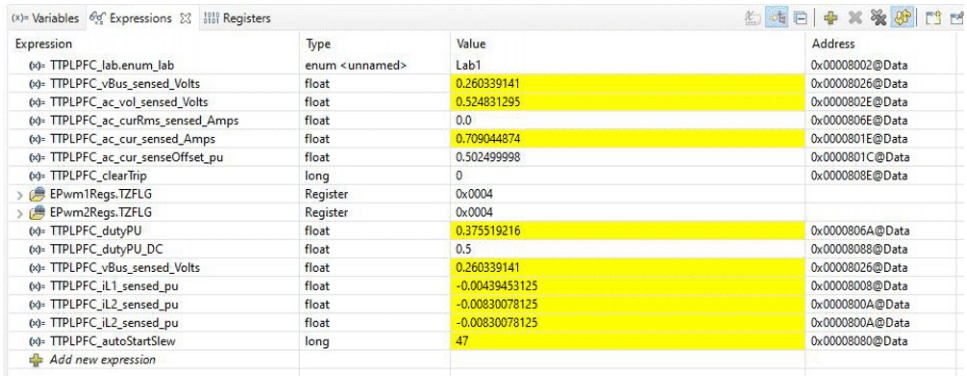
1. powerSUITE Settings : On the powerSUITE page select under *Project Options* section:
   - Select *Lab1* under the Lab option.
   - Also disable the other options such as *Non Linear Voltage Loop, Adaptive Deadtime and Phase shedding*
2. If this is an adapted solution, edit the setting under *Voltage and Current Sensing Parameters* . One can refer to the *calculations.xlsx* file which is available under the C2000Ware DigitalPower SDK Install directory at *<install_location>\solutions\tidm_02008\hardware* for details on sensing circuit and how max range is computed for the powerSUITE page
3. Under Power Stage Parameters specify the switching frequency, the dead band, and the power rating. Save the page.

3.1.2.5.1.2 Building and Loading Project

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 trageteonfigs (Figure 3-4).
4. Then click *Run → Debug*. This action launches a debugging session. In the case of dual CPU devices, a window may appear to select the CPU that the debug must be performed. In this case, select CPU1.
5. The project then loads on the device, and CCS debug view becomes active. The code halts at the start of the main routine.

3.1.2.5.1.3 Setup Debug Environment Windows

1. To add the variables in the watch and expressions window, click *View* → *Scripting Console* to open the scripting console dialog box. On the upper-right corner of this console, click on open then browse to the *setupdebugenv_lab1.js* script file located inside the project folder. This script file populates the watch window with appropriate variables required to debug the system. Click on the Continuous Refresh button on the watch window to enable continuous update of values from the controller. The watch window appears as shown in Figure 3-11.



Variables | Expressions | Registers

Expression	Type	Value	Address
TTPLPFC_lab.enum_lab	enum <unnamed>	Lab1	0x00008002@Data
TTPLPFC_vBus_sensed_Volts	float	0.260339141	0x00008026@Data
TTPLPFC_ac_vol_sensed_Volts	float	0.524831295	0x0000802E@Data
TTPLPFC_ac_curRms_sensed_Amps	float	0.0	0x0000806E@Data
TTPLPFC_ac_cur_sensed_Amps	float	0.709044874	0x0000801E@Data
TTPLPFC_ac_cur_senseOffset_pu	float	0.502499998	0x0000801C@Data
TTPLPFC_clearTrip	long	0	0x0000808E@Data
EPwm1Regs.TZFLG	Register	0x0004	
EPwm2Regs.TZFLG	Register	0x0004	
TTPLPFC_dutyPU	float	0.375519216	0x0000806A@Data
TTPLPFC_dutyPU_DC	float	0.5	0x00008088@Data
TTPLPFC_vBus_sensed_Volts	float	0.260339141	0x00008026@Data
TTPLPFC_iL1_sensed_pu	float	-0.00439453125	0x00008008@Data
TTPLPFC_iL2_sensed_pu	float	-0.00830078125	0x0000800A@Data
TTPLPFC_iL2_sensed_pu	float	-0.00830078125	0x0000800A@Data
TTPLPFC_autoStartSlew	long	47	0x00008080@Data
Add new expression			

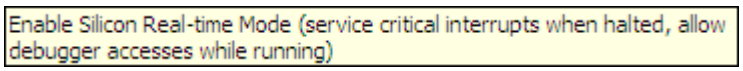
Figure 3-11. Lab 1 Expressions View

2. Run the project by clicking on 
3. Now Halt the processor by using the *Halt* button on the toolbar (

3.1.2.5.1.4 Using Real-Time Emulation

Real-time emulation is a special emulation feature that allows windows within CCS to be updated while the MCU is running. This feature allows graphs and watch views to update but also allows the user to change values in watch or memory windows and see the effect of these changes in the system without halting the processor.

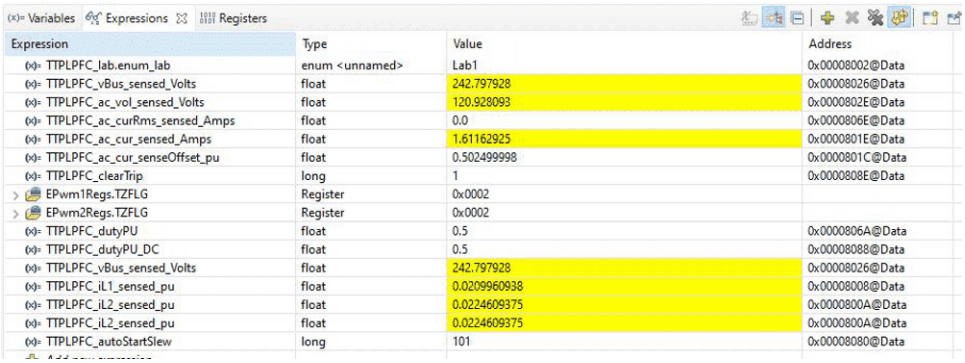
1. Enable real-time mode by hovering the mouse on the buttons on the horizontal toolbar and clicking the  button.



2. A message box may appear. If so, select *YES* to enable debug events. This action sets bit 1 (DGBM bit) of status register 1 (ST1) to a *0*. The DGBM is the debug enable mask bit. When the DGBM bit is set to *0*, memory and register values can be passed to the host processor for updating the debugger windows.

3.1.2.5.1.5 Running Code

1. Now run the project again by clicking on  .
2. In a few seconds the inrush relay clicks, the software is programmed to do so in the lab1. The trip clears, and a duty cycle of 0.5 is applied.
3. In the watch view, check if the TTPLPFC_ac_vol_sensed_Volts, TTPLPFC_vBus_sensed_Volts, TTPLPFC_ac_cur_sensed_Amps, variables are updating, periodically.
   - Note: As no power is applied right now, this value is close to zero.
4. Now slowly increase the input DC voltage from zero to 120 V. The output voltage shows a boosted voltage as a steady duty cycle of 0.5 PU is applied as default setting. If a high current is drawn, verify if the voltage terminals are swapped. If true, reduce the voltage to zero first and correct the issue before resuming the test
5. Verifying the voltage sensing: Make sure *TTPLPFC_ac_vol_sensed_Volts* and *TTPLPFC_vBus_sensed_Volts* display the correct values, for 120-V DC input, TTPLPFC_vBus_sensed_Volts is close to 240V. This verifies the voltage sensing of the board in some manner.
6. Verifying the current sensing: Notice the *TTPLPFC_ac_cur_sensed_Amps* for the given test condition; this value is close to 1 A.



Variables | Expressions | Registers

Expression	Type	Value	Address
TTPLPFC_lab.enum_lab	enum <unnamed>	Lab1	0x00008002@Data
TTPLPFC_vBus_sensed_Volts	float	242.797928	0x00008026@Data
TTPLPFC_ac_vol_sensed_Volts	float	120.928093	0x0000802E@Data
TTPLPFC_ac_curRms_sensed_Amps	float	0.0	0x0000806E@Data
TTPLPFC_ac_cur_sensed_Amps	float	1.61162925	0x0000801E@Data
TTPLPFC_ac_cur_senseOffset_pu	float	0.502499998	0x0000801C@Data
TTPLPFC_clearTrip	long	1	0x0000808E@Data
EPwm1Regs.TZFLG	Register	0x0002	
EPwm2Regs.TZFLG	Register	0x0002	
TTPLPFC_dutyPU	float	0.5	0x0000806A@Data
TTPLPFC_dutyPU_DC	float	0.5	0x00008088@Data
TTPLPFC_vBus_sensed_Volts	float	242.797928	0x00008026@Data
TTPLPFC_iL1_sensed_pu	float	0.0209960938	0x00008008@Data
TTPLPFC_iL2_sensed_pu	float	0.0224609375	0x0000800A@Data
TTPLPFC_iL2_sensed_pu	float	0.0224609375	0x0000800A@Data
TTPLPFC_autoStartSlew	long	101	0x00008080@Data

Figure 3-12. Lab 1: Watch Expression Showing Measured Voltage and Currents

7. This verifies at a basic level the PWM driver and connection of hardware, user can change the dutyPU_DC variable to see operation under various boost conditions.
8. Once finished, reduce the input voltage to zero and watch for the bus voltages to reduce down to zero.
9. This completes the check for this build, the following items are verified on successful completion of this build:
   - Sensing of voltages and currents and scaling to be correct
   - Interrupt generation and execution of the Lab 1 code in the current loop ISR and Voltage Loop Instrumentation ISR
   - PWM driver and switching

   If any issue is observed a careful inspection of the hardware may be required to eliminate any build issues and so forth.
10. The controller can now be halted, and the debug connection terminated.
11. Fully halting the MCU when in real-time mode is a two-step process. First halt the processor by using the *Halt* button on the toolbar () or by using *Target* → *Halt*. Then take the MCU out of real-time mode by clicking on  . Finally, reset the MCU by clicking on  .
12. Close CCS debug session by clicking on *Terminate Debug Session* (*Target* → *Terminate all*).
    

3.1.2.5.2 Lab 2: Closed Current Loop DC (PFC)

In this build, Lab 2, the inner current loop is closed that is the inductor current is controlled using a current compensator Gi. Both DC bus and output voltage feed forward are applied to the output of this current compensator to generate the duty cycle of the inverter, Equation 7. This makes the plant for the current compensator simple and a proportional (P) controller can be used to tune the loop of the inner current. The

model for the current loop was derived in section Section 2.4.2. Complete software diagram for this build is illustrated in Figure 3-13.

$$duty1PU = \frac{(ac\_cur\_meas - ac\_cur\_ref\_inst) \times Gi + ac\_vol\_sensed}{vBus\_sensed} \quad (7)$$

Complete software diagram for this lab as illustrated in Figure 3-13.

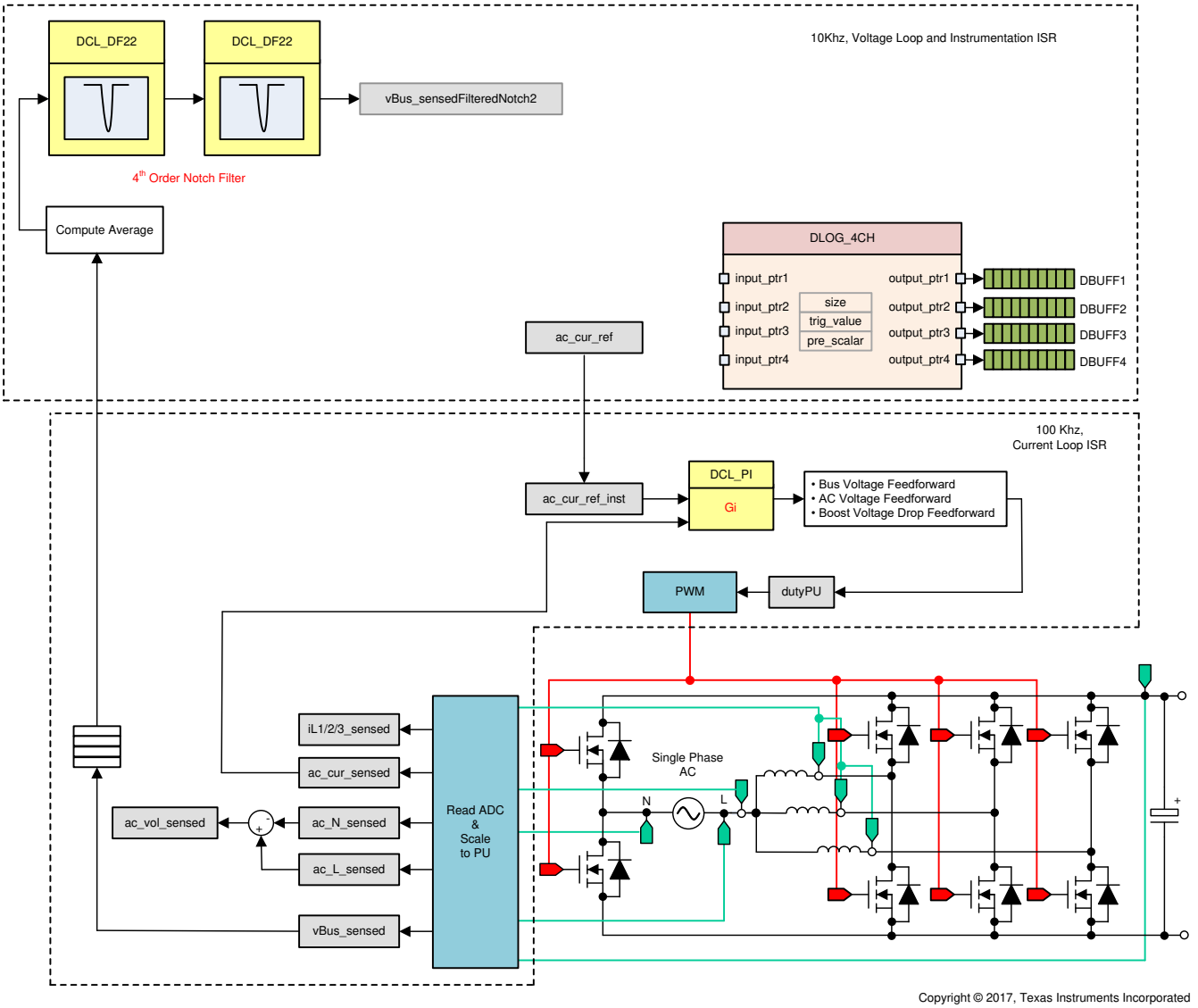


Figure 3-13. Lab 2 Control Software Diagram: Closed Current Loop

3.1.2.5.2.1 Setting Software Options for Lab 2

1. Make sure the hardware is setup as outlined in Figure 3-10 . Do not supply any high voltage (HV) power to the board yet.
2. powerSUITE Settings : On the powerSUITE page select under *Project Options* section:
   - Select *lab2* under Lab option.
   - Select input to be DC under INPUT options
   - Also disable the other options such as *Non Linear Voltage Loop, Adaptive Deadtime and Phase shedding*
3. Assuming all other options are same as specified earlier in Section 3.1.2.5.1.1
4. Under *Control Loop Design*, options for the current loop tuning automatically be selected (*Tuning → Current Loop → COMP1 → DCL_PI_C1*). Now click on the *Compensation Designer* icon ( ).

3.1.2.5.2.2 Designing Current Loop Compensator

1. Compensation Designer launches with the model of the current loop plant with parameters specified on the powerSUITE page. PI-based controller can be tuned from a pole zero perspective to ensure stable closed loop operation. Stability of the system when using the designed compensator can be verified by observing the gain and phase margins on the open loop transfer function plot in the Compensation Designer, as shown in Figure 3-14.

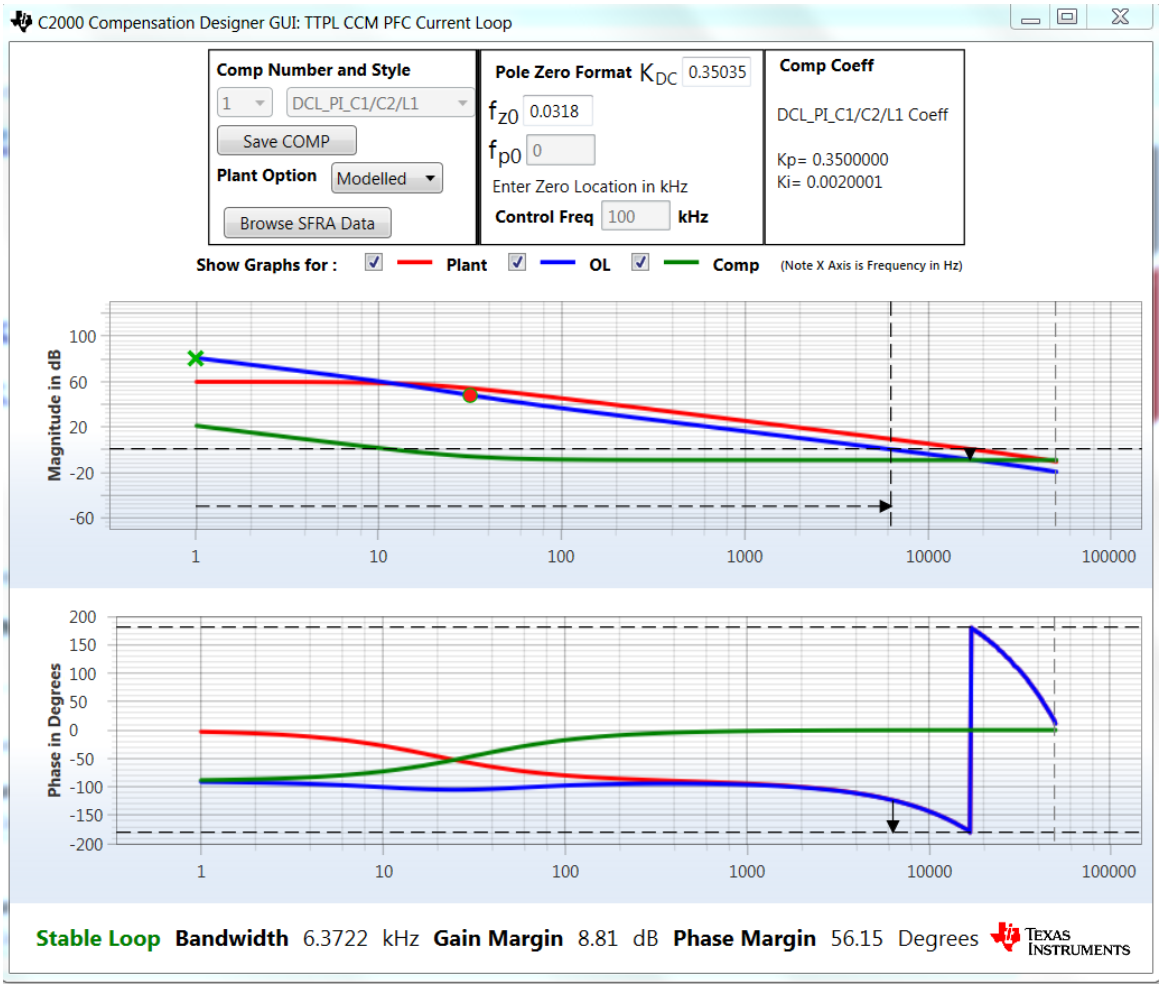


Figure 3-14. Current Loop Design Using Compensation Designer

2. Once satisfied with the open loop gain, click on *Save COMP*. This action saves the compensator values into the project.
   - Note: If the project was not selected from the solution adapter, changes to the compensator are not allowed. Select the solution through the solution adapter.
3. Close the Compensation Designer, and return to the powerSUITE page.

3.1.2.5.2.3 Building and Loading Project and Setting up Debug

1. Right click on the project name, and click *Rebuild Project*. The project builds successfully. Click *Run* → *Debug*, which launches a debugging session. In the case of dual CPU devices, a window may appear to select the CPU the debug must be performed. In this case, select CPU1. The project then loads on the device, and CCS debug view becomes active. The code halts at the start of the main routine.
2. To add the variables in the watch and expressions window click *View* → *Scripting Console* to open the scripting console dialog box. On the upper right corner of this console, click on *Open* to browse to the *setupdebugenv_lab2.js* script file, which is located inside the project folder. This file populates the watch window with appropriate variables required to debug the system. Click on *Continuous Refresh* button ( ) on the watch window to enable continuous update of values from the controller. The watch window appears as Figure 3-15.

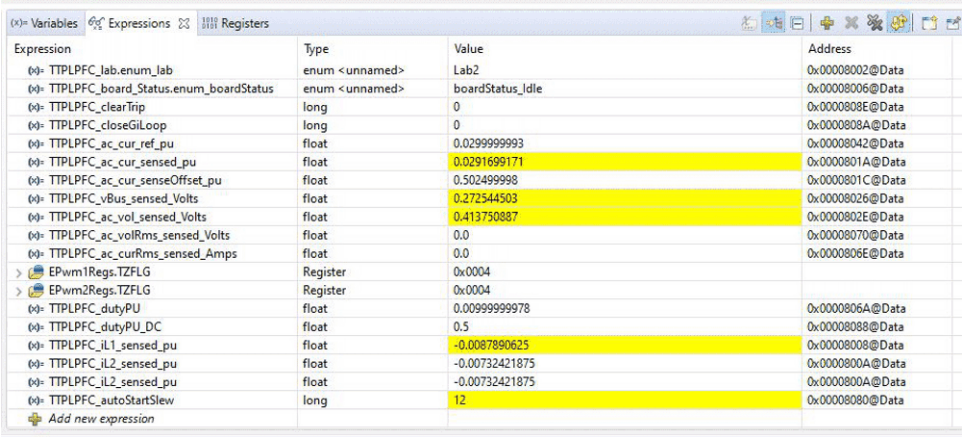
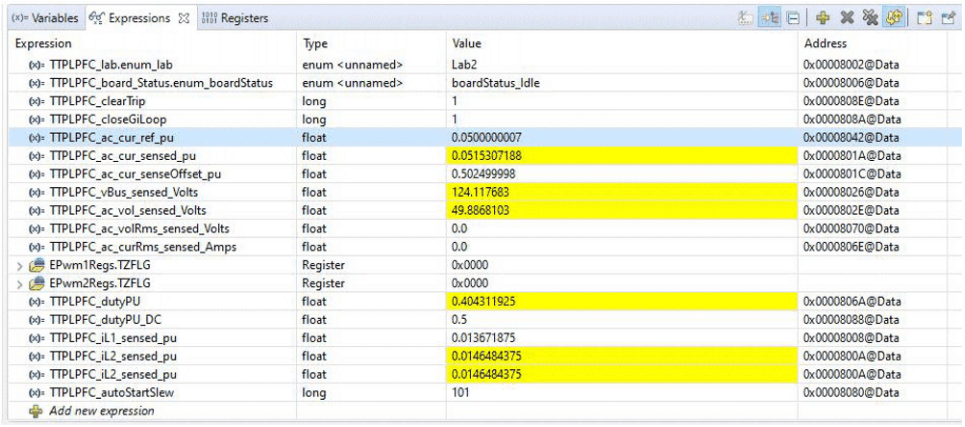


Figure 3-15. Lab 2: Closed Current Loop Expressions View

3. Enable real-time mode by hovering the mouse on the buttons on the horizontal toolbar, and clicking the button.
4. Run the project by clicking on
5. Now Halt the processor by using the *Halt* button on the toolbar (

3.1.2.5.2.4 Running Code

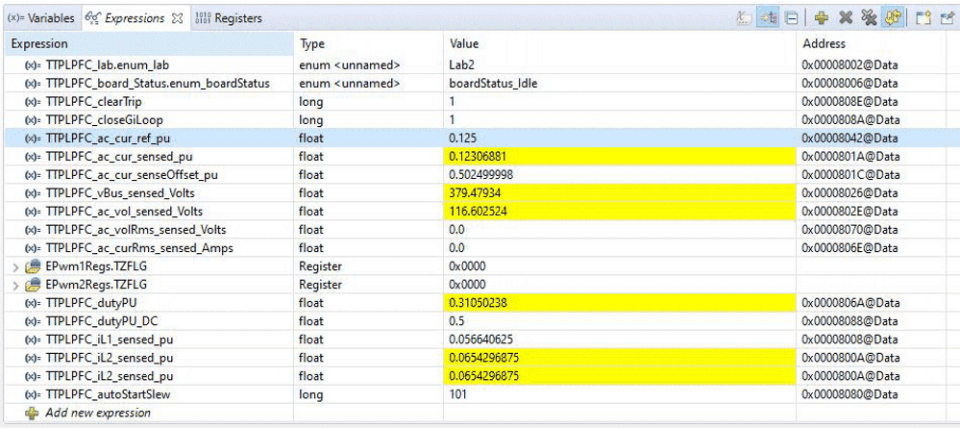
1. The project is programmed to drive the inrush relay and clear the trip after a set amount of time, that is, autoStartSlew==100. The software is programmed to do so in this lab. An input voltage must be applied after hitting run and before this autoslew counter reaches 100. If the counter reaches 100, before voltage is applied at the input, the code must be reset. For which the controller must be brought out of real time mode, a reset performed and restarted. Repeat steps from 3
2. Now run the project by clicking .
3. Apply an input voltage of approximately 50 V before the TTPLPFC_autoStartSlew reaches 100. As soon TTPLPFC_autoStartSlew reaches 100, the inrush relay is triggered, and PWM trip is cleared along with closing the current loop flag.



Expression	Type	Value	Address
TTPLPFC_lab.enum_lab	enum <unnamed>	Lab2	0x00008002@Data
TTPLPFC_board_Status.enum_boardStatus	enum <unnamed>	boardStatus_Idle	0x00008006@Data
TTPLPFC_clearTrip	long	1	0x0000808E@Data
TTPLPFC_closeGiLoop	long	1	0x0000808A@Data
TTPLPFC_ac_cur_ref_pu	float	0.0500000007	0x00008042@Data
TTPLPFC_ac_cur_sensed_pu	float	0.0515307188	0x0000801A@Data
TTPLPFC_ac_cur_senseOffset_pu	float	0.502499998	0x0000801C@Data
TTPLPFC_vBus_sensed_Volts	float	124.117683	0x00008026@Data
TTPLPFC_ac_vol_sensed_Volts	float	49.8868103	0x0000802E@Data
TTPLPFC_ac_volRms_sensed_Volts	float	0.0	0x00008070@Data
TTPLPFC_ac_curRms_sensed_Amps	float	0.0	0x0000806E@Data
EPwm1Regs.TZFLG	Register	0x0000	
EPwm2Regs.TZFLG	Register	0x0000	
TTPLPFC_dutyPU	float	0.404311925	0x0000806A@Data
TTPLPFC_dutyPU_DC	float	0.5	0x00008088@Data
TTPLPFC_iL1_sensed_pu	float	0.013671875	0x00008008@Data
TTPLPFC_iL2_sensed_pu	float	0.0146484375	0x0000800A@Data
TTPLPFC_iL2_sensed_pu	float	0.0146484375	0x0000800A@Data
TTPLPFC_autoStartSlew	long	101	0x00008080@Data
Add new expression			

Figure 3-16. Watch Expression, Lab 2, After Closed Current Loop Operation Begins

4. The input current regulates around 0.7 A, and the output voltage boosts to approximately 128 V.
5. Now slowly increase TTPLPFC_ac_cur_ref_pu to 0.1, that is, 2.5-A input.
6. Next slowly increase Vin = 120 V, and the output voltage will be greater than 350 V.



Expression	Type	Value	Address
TTPLPFC_lab.enum_lab	enum <unnamed>	Lab2	0x00008002@Data
TTPLPFC_board_Status.enum_boardStatus	enum <unnamed>	boardStatus_Idle	0x00008006@Data
TTPLPFC_clearTrip	long	1	0x0000808E@Data
TTPLPFC_closeGiLoop	long	1	0x0000808A@Data
TTPLPFC_ac_cur_ref_pu	float	0.125	0x00008042@Data
TTPLPFC_ac_cur_sensed_pu	float	0.12306881	0x0000801A@Data
TTPLPFC_ac_cur_senseOffset_pu	float	0.502499998	0x0000801C@Data
TTPLPFC_vBus_sensed_Volts	float	379.47934	0x00008026@Data
TTPLPFC_ac_vol_sensed_Volts	float	116.602524	0x0000802E@Data
TTPLPFC_ac_volRms_sensed_Volts	float	0.0	0x00008070@Data
TTPLPFC_ac_curRms_sensed_Amps	float	0.0	0x0000806E@Data
EPwm1Regs.TZFLG	Register	0x0000	
EPwm2Regs.TZFLG	Register	0x0000	
TTPLPFC_dutyPU	float	0.31050238	0x0000806A@Data
TTPLPFC_dutyPU_DC	float	0.5	0x00008088@Data
TTPLPFC_iL1_sensed_pu	float	0.056640625	0x00008008@Data
TTPLPFC_iL2_sensed_pu	float	0.0654296875	0x0000800A@Data
TTPLPFC_iL2_sensed_pu	float	0.0654296875	0x0000800A@Data
TTPLPFC_autoStartSlew	long	101	0x00008080@Data
Add new expression			

Figure 3-17. Watch Expression, Lab 2, After Closed Current Loop Operation Begins at Full Voltage

7. SFRA is integrated in the software of this build to verify the designed compensator provides enough gain and phase margin by measuring on hardware. To run the SFRA keep the project running, and from the cfg page, click on the *SFRA* icon. SFRA GUI appears.
8. Select the options for the device on the SFRA GUI. For example, for F28004x select floating point. Click on *Setup Connection*. On the pop-up window uncheck the boot on connect option, and select an appropriate COM port. Ensure *Boot on Connect* is deselected. Click *OK*. Return to the SFRA GUI, and click *Connect*.

9. 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. Activity can be monitored by seeing the progress bar on the SFRA GUI and also checking the flashing of blue LED on the back on the control card that indicates UART activity. Once complete, a graph with the open loop plot appears. Comparing this with the measured plots, there is good correlation between the modeled and measured as shown in Figure 3-18. This verifies that the designed compensator is indeed stable and the model accurate. Note: the deviation at low frequency, less than 200 Hz, is expected and is a known phenomena, also the measurement shown here was taken with a DC source, if an AC source is used to emulate a DC source the output impedance of the AC source can result in deviations from the graph plotted below

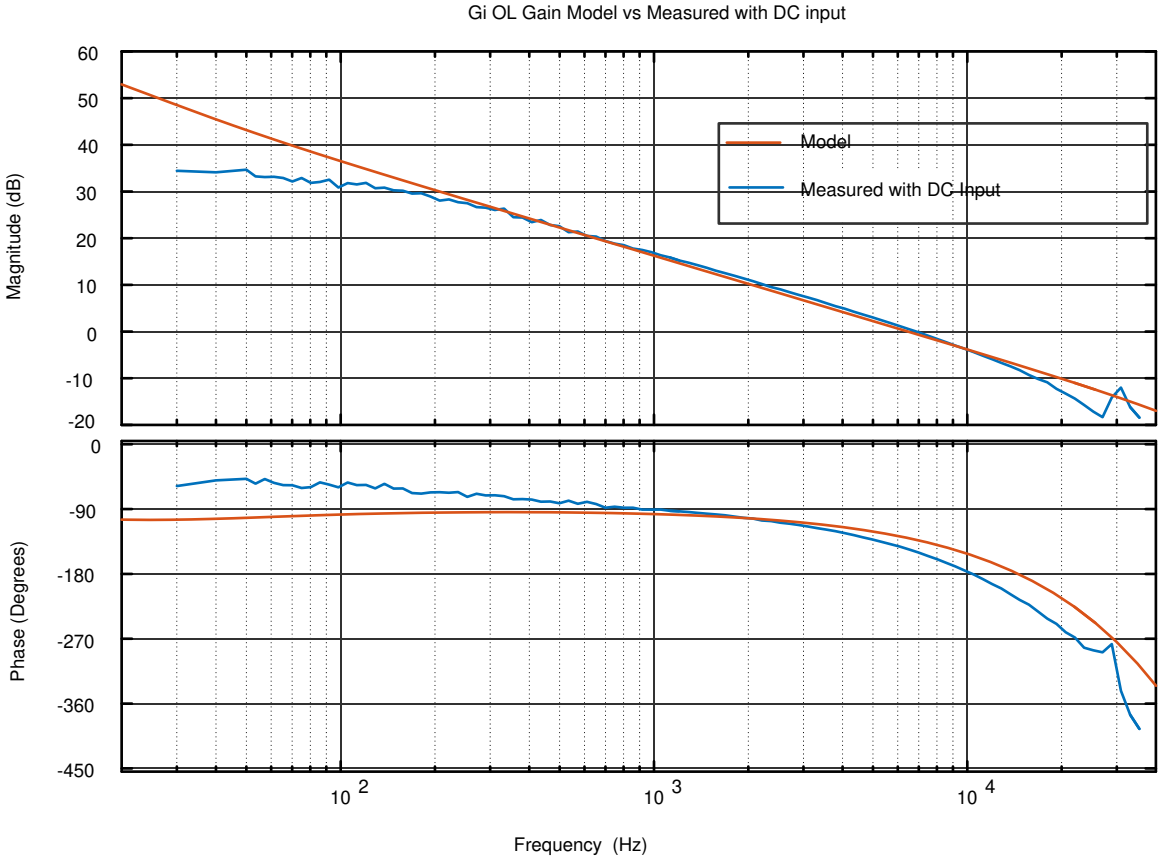


Figure 3-18. SFRA Run vs Modeled Closed Current Loop, Open Loop Gain

The frequency response data is also saved in the project folder under an SFRA data folder and is time stamped with the time of the SFRA run.

10. Optionally one can use the measured frequency response of the plant to design the current compensator by clicking on the Compensation Designer again from the syscfg page and choose *SFRA Data* for plant option on the GUI. This uses the measured plant information to design the compensator. This option can be used to fine tune the compensation. By default the compensation designer points to the latest SFRA run. If a previous SFRA run plant information must be used the user can select the SFRAData.csv file by browsing to it by clicking on *Browse SFRA Data*.
11. This action verifies the current compensator design.
12. To bring the system to a safe stop, bring the input DC voltage down to zero, observe the TTPLPFC_vBus_sensed_Volts comes down to zero as well.
13. Fully halting the MCU when in real-time mode is a two-step process. First halt the processor by using the *Halt* button on the toolbar ( ) or by using *Target* → *Halt*. Then take the MCU out of real-time mode by clicking on . Finally, reset the MCU ( ) .
14. Close the CCS debug session by clicking *Terminate Debug Session* (*Target* → *Terminate all*).

3.1.2.5.3 Lab 3: Closed Current Loop, AC (PFC)

In Lab 3 the inner current loop is closed, that is, the inductor current is controlled using a current compensator Gi. Both DC bus and output voltage feedforward are applied to the output of this current compensator++ to generate the duty cycle of the inverter along with soft start for PWM around the zero crossing

Complete software diagram for this build as illustrated in Figure 3-19.

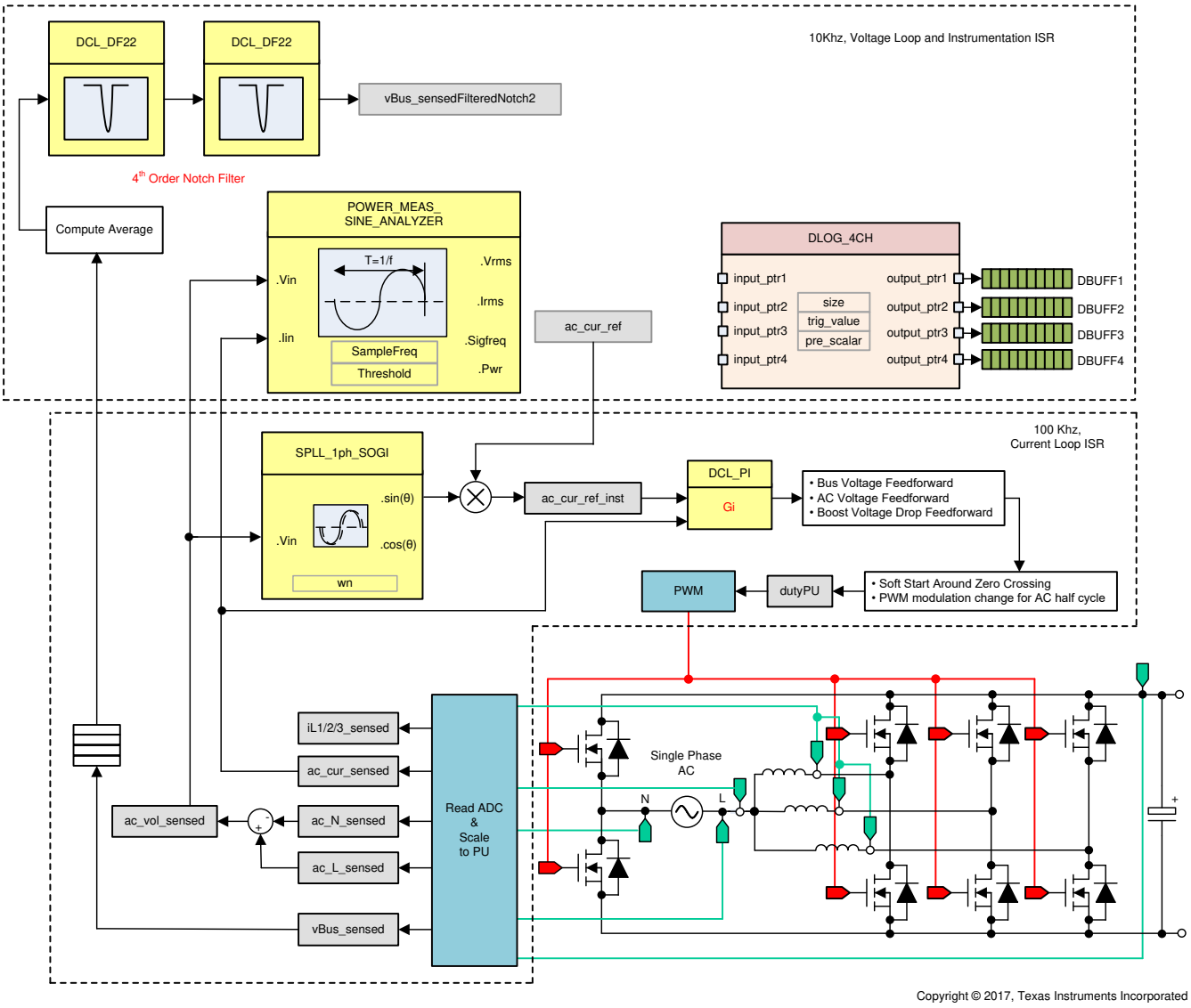


Figure 3-19. Lab 3 Control Software Diagram: Closed Current Loop AC

To run this lab make sure the hardware is setup as shown in Figure 3-20. Do not supply any HV power yet to the board.

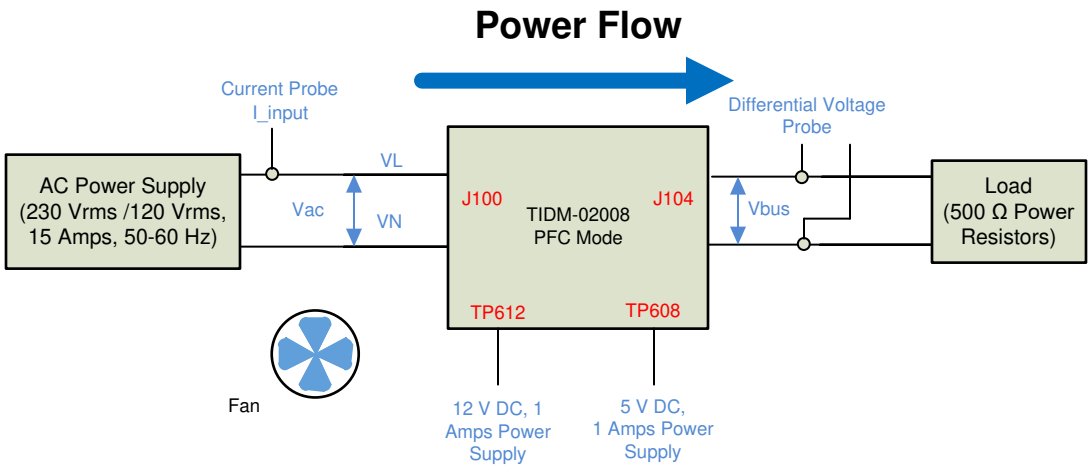


Figure 3-20. HW Setup for AC Input

3.1.2.5.3.1 Setting Software Options for Lab 3

1. powerSUITE Settings: On the *powerSUITE* page assuming options were selected at outlined in 2under *Project Options* section, select *Lab 3*. Save the page.
2. Current compensator from the previous build is re-used in this lab so no additional steps are required for tuning the current loop in the lab.

3.1.2.5.3.2 Building and Loading Project and Setting up Debug

1. Right click on the project name, and click *Rebuild Project*. The project builds successfully. Click *Run* → *Debug*, which launches a debugging session. In the case of dual CPU devices, a window may appear to select the CPU the debug must be performed. In this case, select CPU1. The project then loads on the device, and CCS debug view becomes active. The code halts at the start of the main routine.
2. To add the variables in the watch and expressions window click *View* → *Scripting Console* to open the scripting console dialog box. On the upper right corner of this console, click on *Open* to browse to the *setupdebugenv_lab3.js* script file, which is located inside the project folder. This file populates the watch window with appropriate variables required to debug the system. Click on *Continuous Refresh* button ( ) on the watch window to enable continuous update of values from the controller. The watch window appears as 3.

(x)= Variables | Expressions | Registers

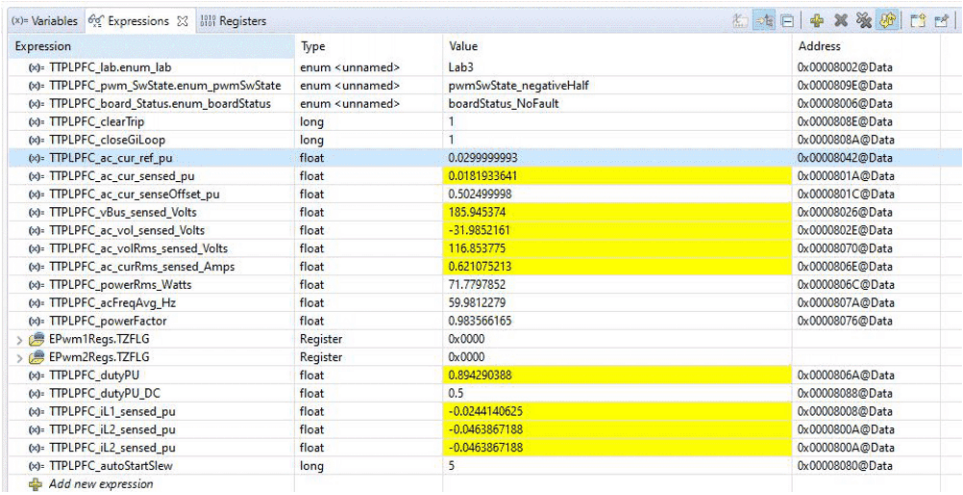
Expression	Type	Value	Address
TTPLPFC_lab.enum_lab	enum <unnamed>	Lab3	0x00008002@Data
TTPLPFC_pwm_SwState.enum_pwmSwState	enum <unnamed>	pwmSwState_defaultState	0x0000809E@Data
TTPLPFC_board_Status.enum_boardStatus	enum <unnamed>	boardStatus_InputUnderVoltageTrip	0x00008006@Data
TTPLPFC_clearTrip	long	0	0x0000808E@Data
TTPLPFC_closeGiLoop	long	0	0x0000808A@Data
TTPLPFC_ac_cur_ref_pu	float	0.0299999993	0x00008042@Data
TTPLPFC_ac_cur_sensed_pu	float	0.0283173919	0x0000801A@Data
TTPLPFC_ac_cur_senseOffset_pu	float	0.502499998	0x0000801C@Data
TTPLPFC_vBus_sensed_Volts	float	0.280229211	0x00008026@Data
TTPLPFC_ac_vol_sensed_Volts	float	-0.792785585	0x0000802E@Data
TTPLPFC_ac_volRms_sensed_Volts	float	0.0	0x00008070@Data
TTPLPFC_ac_curRms_sensed_Amps	float	0.0	0x0000806E@Data
TTPLPFC_powerRms_Watts	float	0.0	0x0000806C@Data
TTPLPFC_acFreqAvg_Hz	float	0.0	0x0000807A@Data
TTPLPFC_powerFactor	float	0.0	0x00008076@Data
EPwm1Regs.TZFLG	Register	0x0004	
EPwm2Regs.TZFLG	Register	0x0004	
TTPLPFC_dutyPU	float	0.00999999978	0x0000806A@Data
TTPLPFC_dutyPU_DC	float	0.5	0x00008088@Data
TTPLPFC_iL1_sensed_pu	float	-0.00927734375	0x00008008@Data
TTPLPFC_iL2_sensed_pu	float	-0.00732421875	0x0000800A@Data
TTPLPFC_iL2_sensed_pu	float	-0.00732421875	0x0000800A@Data
TTPLPFC_autoStartSlew	long	0	0x00008080@Data

Figure 3-21. Lab 3: Closed Current Loop Expressions View

3. Enable real-time mode by hovering the mouse on the buttons on the horizontal toolbar, and clicking the button.

3.1.2.5.3.3 Running Code

1. The project is programmed to wait for input voltage to exceed approximately 70Vrms to drive the in rush relay, and clear the trip.
2. Run the project by clicking  .
3. Now apply an input voltage of approximately 120 V, the board comes out of the undervoltage condition and inrush relay is driven. The trip clears, and a small amount of current of approximately 0.55-A RMS is drawn. The watch window looks similar to Figure 3-22. The bus voltage is close to 180 V.



Expression	Type	Value	Address
TTPLPFC_lab.enum_lab	enum <unnamed>	Lab3	0x00008002@Data
TTPLPFC_pwm_SwState.enum_pwmSwState	enum <unnamed>	pwmSwState_negativeHalf	0x0000809E@Data
TTPLPFC_board_Status.enum_boardStatus	enum <unnamed>	boardStatus_NoFault	0x00008006@Data
TTPLPFC_clearTrip	long	1	0x0000808E@Data
TTPLPFC_closeGiLoop	long	1	0x0000808A@Data
TTPLPFC_ac_cur_ref_pu	float	0.0299999993	0x00008042@Data
TTPLPFC_ac_cur_sensed_pu	float	0.0181933641	0x0000801A@Data
TTPLPFC_ac_cur_senseOffset_pu	float	0.502499998	0x0000801C@Data
TTPLPFC_vBus_sensed_Volts	float	185.945374	0x00008026@Data
TTPLPFC_ac_vol_sensed_Volts	float	-31.9852161	0x0000802E@Data
TTPLPFC_ac_volRms_sensed_Volts	float	116.853775	0x00008070@Data
TTPLPFC_ac_curRms_sensed_Amps	float	0.621075213	0x0000806E@Data
TTPLPFC_powerRms_Watts	float	71.7797852	0x0000806C@Data
TTPLPFC_acFreqAvg_Hz	float	59.9812279	0x0000807A@Data
TTPLPFC_powerFactor	float	0.983566165	0x00008076@Data
EPwm1Regs.TZFLG	Register	0x0000	
EPwm2Regs.TZFLG	Register	0x0000	
TTPLPFC_dutyPU	float	0.894290388	0x0000806A@Data
TTPLPFC_dutyPU_DC	float	0.5	0x00008088@Data
TTPLPFC_iL1_sensed_pu	float	-0.0244140625	0x00008008@Data
TTPLPFC_iL2_sensed_pu	float	-0.0463867188	0x0000800A@Data
TTPLPFC_iL2_sensed_pu	float	-0.0463867188	0x0000800A@Data
TTPLPFC_autoStartSlew	long	5	0x00008080@Data
Add new expression			

Figure 3-22. Watch Expression, Lab 3, AC After Closed Current Loop Operation Begins

4. Now slowly increase TTPLPFC_ac_cur_ref_pu to 0.14, that is, 2.4-A input, and the bus voltage rises to 380 V. The voltage and current waveform are shown in Figure 3-23.

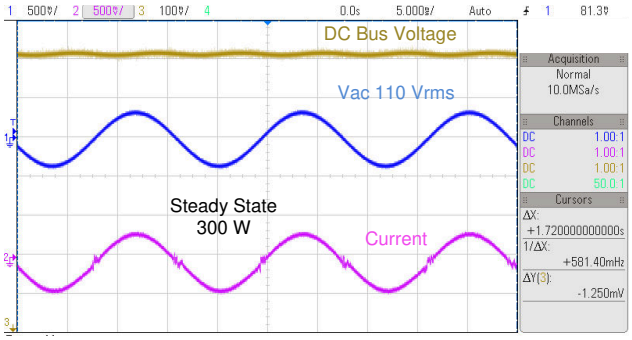


Figure 3-23. Input AC input, Current and Output DC Voltage Waveform

5. SFRA is integrated in the software of this build to verify the designed compensator provides enough gain and phase margin by measuring on hardware. To run the SFRA keep the project running, and from the cfg page, click on the *SFRA* icon. SFRA GUI appears.
6. Select the options for the device on the SFRA GUI. For example, for F28377D select floating point. Click on *Setup Connection*. On the pop-up window uncheck the boot on connect option, and select an appropriate COM port. Click *OK*. Return to the SFRA GUI, and click *Connect*.

7. 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. Activity can be monitored by seeing the progress bar on the SFRA GUI and also checking the flashing of blue LED on the back on the control card that indicates UART activity. Once complete, a graph with the open loop plot appears, Figure 3-24. This is similar to the plot seen under DC conditions; however, some additional noise is visible due to AC harmonic frequencies close to the measured frequencies. The BW, PM, and GM numbers are very similar to the DC case. Note the graph shown in Figure 3-24 was taken with direct grid AC input. When using AC source interaction of the AC source output, impedance can be observed, which can affect the control margins.

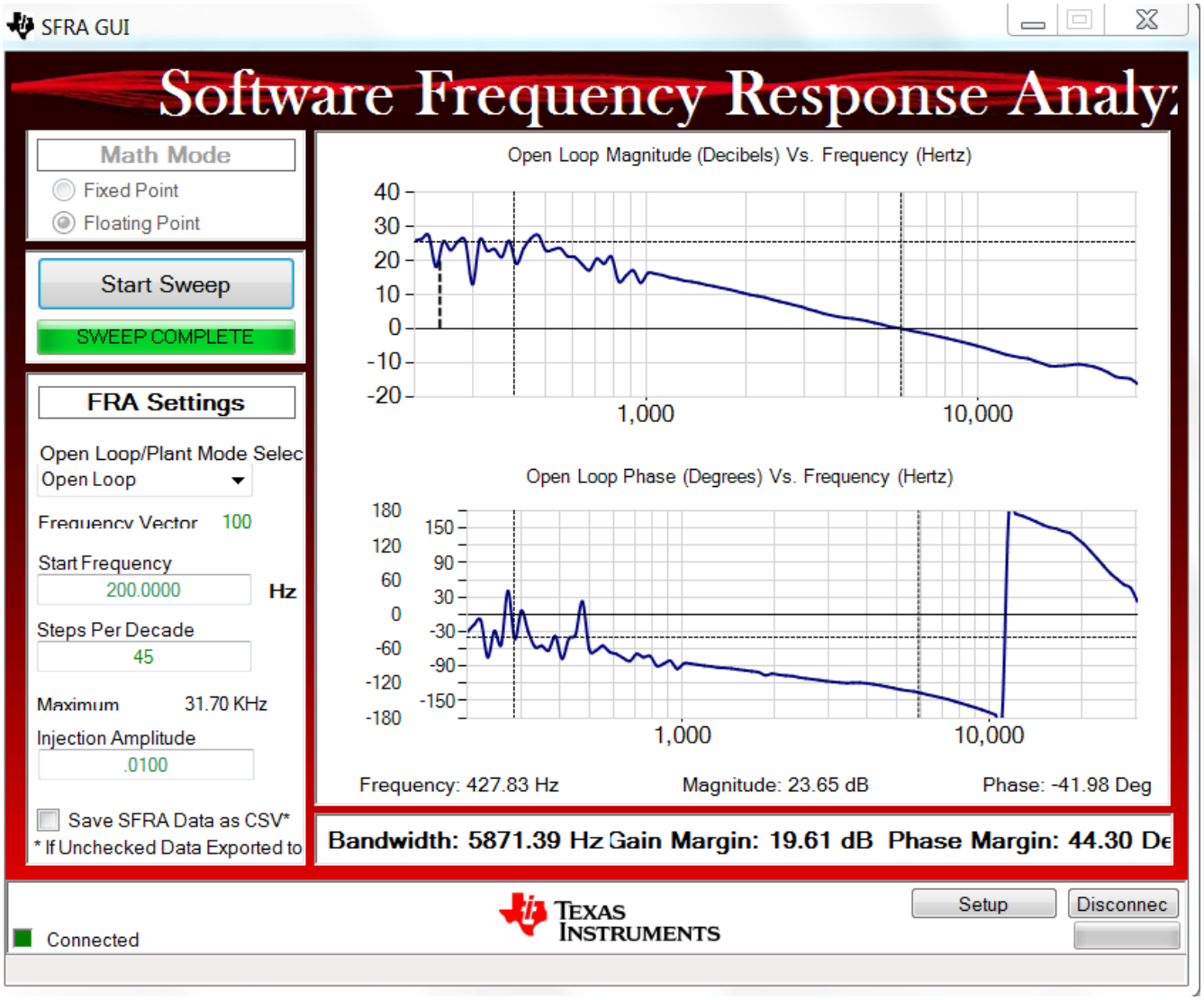


Figure 3-24. SFRA Run, Closed Current Loop, Open Loop Gain

8. To bring the system to a safe stop, switch off the output from the AC power supply thus bring the input AC voltage down to zero, observe the TTPLPFC_vBus_sensed_Volts comes down to zero as well.
9. Fully halting the MCU when in real-time mode is a two-step process. First halt the processor by using the *Halt* button on the toolbar ( ) or by using *Target* → *Halt*. Then take the MCU out of real-time mode by clicking on . Finally, reset the MCU ( ) .
10. Close the CCS debug session by clicking on *Terminate Debug Session* (*Target* → *Terminate all*).

3.1.2.5.4 Lab 4: Closed Voltage and Current Loop (PFC)

In this lab the outer voltage loop is closed with the inner current loop closed. The model of the outer voltage loop was derived in Section 2.4.3. A PI-based compensator is used and tuned through the compensation designer for the outer voltage loop.

Figure 3-25 shows the software diagram for this build.

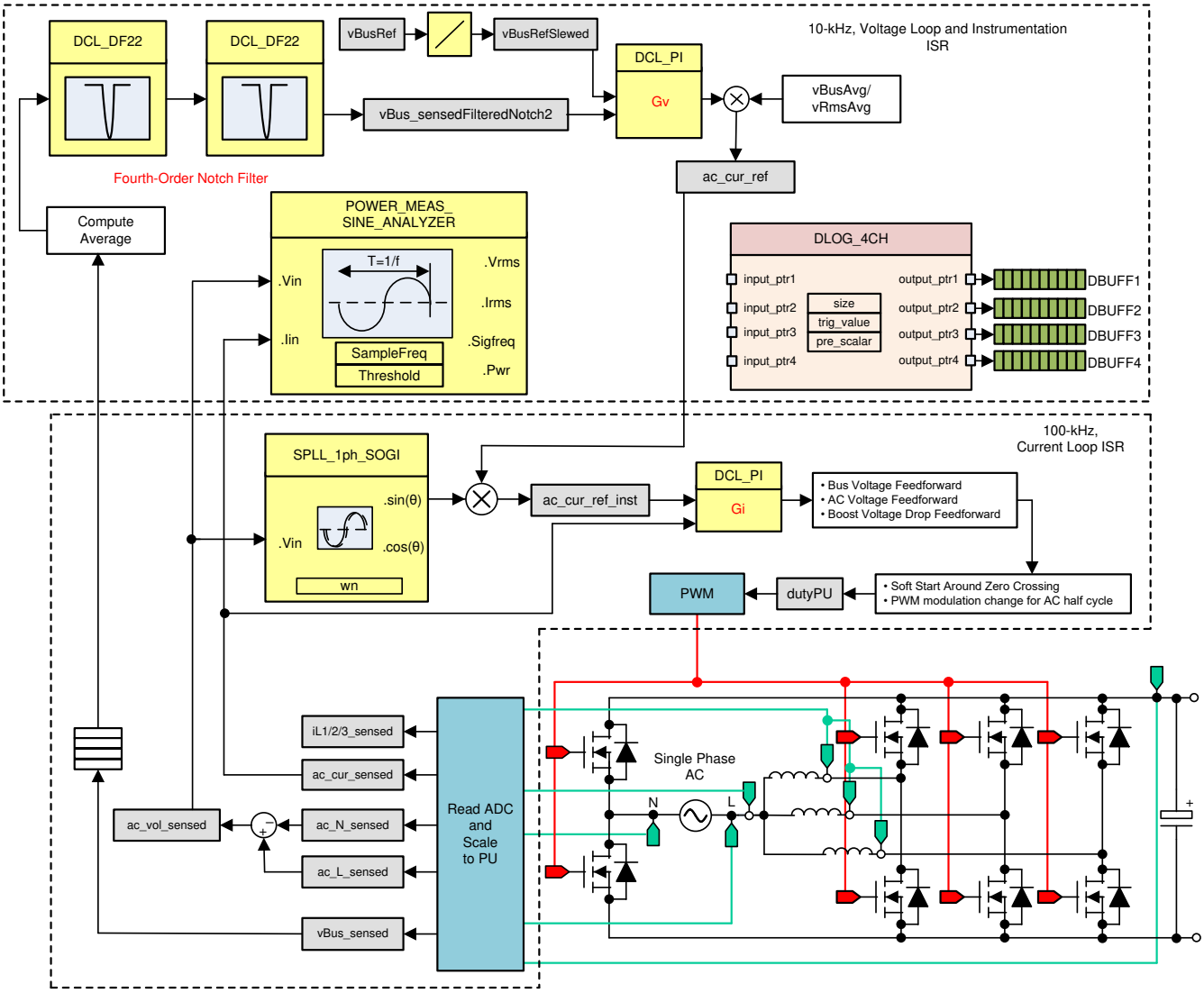


Figure 3-25. Lab 4 Control Diagram: Output Voltage Control With Inner Current Loop

3.1.2.5.4.1 Setting Software Options for Lab 4

1. Make sure the hardware is setup as outlined in Figure 3-20. Do not supply any HV power yet to the board.
2. powerSUITE Settings : On the powerSUITE page select under *Project Options* section:
   - Select *Lab 4* under Lab option.
   - Also disable the other options such as *Non Linear Voltage Loop, Adaptive Deadtime and Phase shedding*
3. Assuming all other options are same as specified earlier in Section 3.1.2.5.1.1
4. Under *Control Loop Design*, select *Tuning* as *Voltage Loop*. Style presets to *DCL PI_C*3 Save the page by *Ctrl* + *S*, and click on the Compensation Designer button ( ).
5. Make sure the load connected at the output of the board is correctly entered on the powerSUITE cfg page because this load value is used in the design of the voltage compensator.

3.1.2.5.4.2 Designing Voltage Loop Compensator

1. Compensation designer then launches with the model of the voltage loop plant, as shown in Figure 3-26. The PI compensator can be edited to get the desired gain and phase margin, keeping in mind the bandwidth of the voltage loop has an inverse relationship with the THD achieved. Typically in a PFC application, this bandwidth is kept at approximately 10 Hz.

Figure 3-26. Voltage Loop PI Compensation Tuning Using Compensation Designer

2. Once satisfied with the compensator design, click on *Save COMP*. This action saves the compensator values into the project.
   - Note: If the project was not selected from the solution adapter, changes to the compensator are not allowed. To design one's own, select the solution through the solution adapter.
3. Close the Compensation Designer, and return to the powerSUITE page. Save using *Ctrl* + *S*.

3.1.2.5.4.3 Building and Loading Project and Setting up Debug

1. Right click on the project name, and click *Rebuild Project*. The project builds successfully. Click *Run* → *Debug*, which launches a debugging session. In the case of dual CPU devices, a window may appear to select the CPU the debug must be performed. In this case, select CPU1. The project then loads on the device, and CCS debug view becomes active. The code halts at the start of the main routine.
2. To add the variables in the watch and expressions window, click *View* → *Scripting Console* to open the scripting console dialog box. On the upper-right corner of this console, click on *Open* to browse to the *setupdebugenv_lab4.js* script file located inside the project folder. This file populates the watch window with appropriate variables required to debug the system. Click on the *Continuous Refresh* button ( ) on the watch window to enable continuous update of values from the controller.
   The watch window appears as shown in Figure 3-27.

Variables | Expressions | Registers

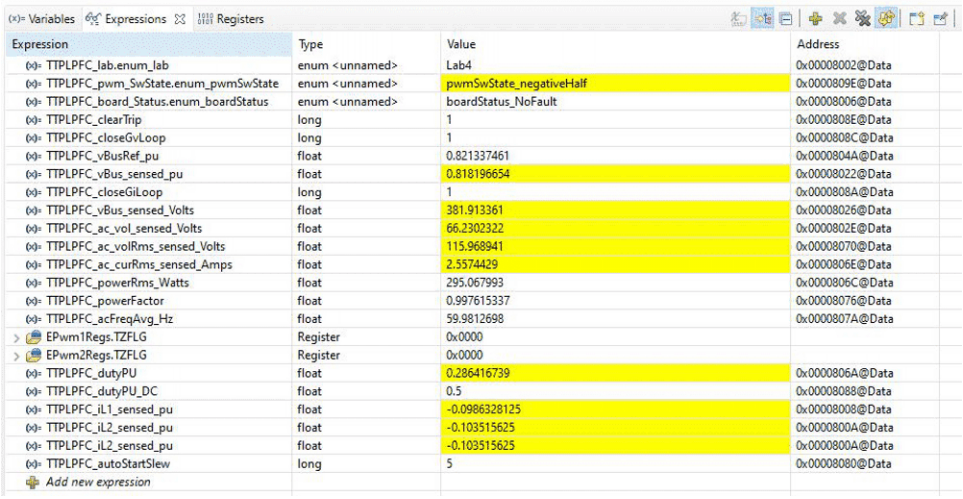
Expression	Type	Value	Address
TTPLPFC_lab.enum_lab	enum <unnamed>	Lab4	0x00008002@Data
TTPLPFC_pwm_SwState.enum_pwmSwState	enum <unnamed>	pwmSwState_defaultState	0x0000809E@Data
TTPLPFC_board_Status.enum_boardStatus	enum <unnamed>	boardStatus_InputUnderVoltageTrip	0x00008006@Data
TTPLPFC_clearTrip	long	0	0x0000808E@Data
TTPLPFC_closeGvLoop	long	0	0x0000808C@Data
TTPLPFC_vBusRef_pu	float	0.821337461	0x0000804A@Data
TTPLPFC_vBus_sensed_pu	float	0.000406901061	0x00008022@Data
TTPLPFC_closeGiLoop	long	0	0x0000808A@Data
TTPLPFC_vBus_sensed_Volts	float	0.270254403	0x00008026@Data
TTPLPFC_ac_vol_sensed_Volts	float	-0.825044274	0x0000802E@Data
TTPLPFC_ac_volRms_sensed_Volts	float	0.0	0x00008070@Data
TTPLPFC_ac_curRms_sensed_Amps	float	0.0	0x0000806E@Data
TTPLPFC_powerRms_Watts	float	0.0	0x0000806C@Data
TTPLPFC_powerFactor	float	0.0	0x00008076@Data
TTPLPFC_acFreqAvg_Hz	float	0.0	0x0000807A@Data
EPwm1Regs.TZFLG	Register	0x0004	
EPwm2Regs.TZFLG	Register	0x0004	
TTPLPFC_dutyPU	float	0.00999999978	0x0000806A@Data
TTPLPFC_dutyPU_DC	float	0.5	0x00008088@Data
TTPLPFC_iL1_sensed_pu	float	-0.001953125	0x00008008@Data
TTPLPFC_iL2_sensed_pu	float	-0.00732421875	0x0000800A@Data
TTPLPFC_iL2_sensed_pu	float	-0.00732421875	0x0000800A@Data
TTPLPFC_autoStartSlew	long	0	0x00008080@Data
Add new expression			

Figure 3-27. Lab 4: Expressions View

3. Enable real-time mode by hovering the mouse on the buttons on the horizontal toolbar and clicking the button.
4. Run the project by clicking on
5. Now Halt the processor by using the *Halt* button on the toolbar ( ).

3.1.2.5.4.4 Running Code

1. The project is programmed to wait for input voltage to excel at approximately 70 Vrms to drive the in rush relay, and clear the trip.
2. Run the project by clicking  .
3. Now apply an input voltage of approximately 120 V. The board comes out of the undervoltage condition and inrush relay is driven. The trip clears, and the output rises to 380-V DC. A sinusoidal current is drawn from the AC input. Figure 3-28 shows the watch window when the program is running at this stage.



Variables | Expressions | Registers

Expression	Type	Value	Address
TTPLPFC_lab.enum_lab	enum <unnamed>	Lab4	0x00008002@Data
TTPLPFC_pwm_SwState.enum_pwmSwState	enum <unnamed>	pwmSwState_negativeHalf	0x0000809E@Data
TTPLPFC_board_Status.enum_boardStatus	enum <unnamed>	boardStatus_NoFault	0x00008006@Data
TTPLPFC_clearTrip	long	1	0x0000808E@Data
TTPLPFC_closeGvLoop	long	1	0x0000808C@Data
TTPLPFC_vBusRef_pu	float	0.821337461	0x0000804A@Data
TTPLPFC_vBus_sensed_pu	float	0.818196654	0x00008022@Data
TTPLPFC_closeGiLoop	long	1	0x0000808A@Data
TTPLPFC_vBus_sensed_Volts	float	381.913361	0x00008026@Data
TTPLPFC_ac_vol_sensed_Volts	float	66.2302322	0x0000802E@Data
TTPLPFC_ac_volRms_sensed_Volts	float	115.968941	0x00008070@Data
TTPLPFC_ac_curRms_sensed_Amps	float	2.5574429	0x0000806E@Data
TTPLPFC_powerRms_Watts	float	295.067993	0x0000806C@Data
TTPLPFC_powerFactor	float	0.997615337	0x00008076@Data
TTPLPFC_acFreqAvg_Hz	float	59.9812698	0x0000807A@Data
EPwm1Regs.TZFLG	Register	0x0000	
EPwm2Regs.TZFLG	Register	0x0000	
TTPLPFC_dutyPU	float	0.286416739	0x0000806A@Data
TTPLPFC_dutyPU_DC	float	0.5	0x00008088@Data
TTPLPFC_iL1_sensed_pu	float	-0.0986328125	0x00008008@Data
TTPLPFC_iL2_sensed_pu	float	-0.103515625	0x0000800A@Data
TTPLPFC_iL2_sensed_pu	float	-0.103515625	0x0000800A@Data
TTPLPFC_autoStartSlew	long	5	0x00008080@Data
Add new expression			

Figure 3-28. Lab 4: Expressions View After AC Voltage is Applied

4. SFRA is integrated in the software of this build to verify the designed compensator provides enough gain and phase margin by measuring on hardware. To run the SFRA, keep the project running, and from the cfg page, click on the *SFRA* icon. SFRA GUI appears.
5. Select the options for the device on the SFRA GUI. For example, for F28004x, select floating point. Click on *Setup Connection*, and on the pop-up window, uncheck the boot on connect option and select an appropriate COM port. Click *OK*. Return to the SFRA GUI, and click *Connect*.

6. 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. Activity can be monitored by seeing the progress bar on the SFRA GUI and checking the flashing of blue LED on the back on the control card that indicates UART activity. Once complete, a graph with the open loop plot appears, as seen in Figure 3-29. This action verifies that the designed compensator is indeed stable.

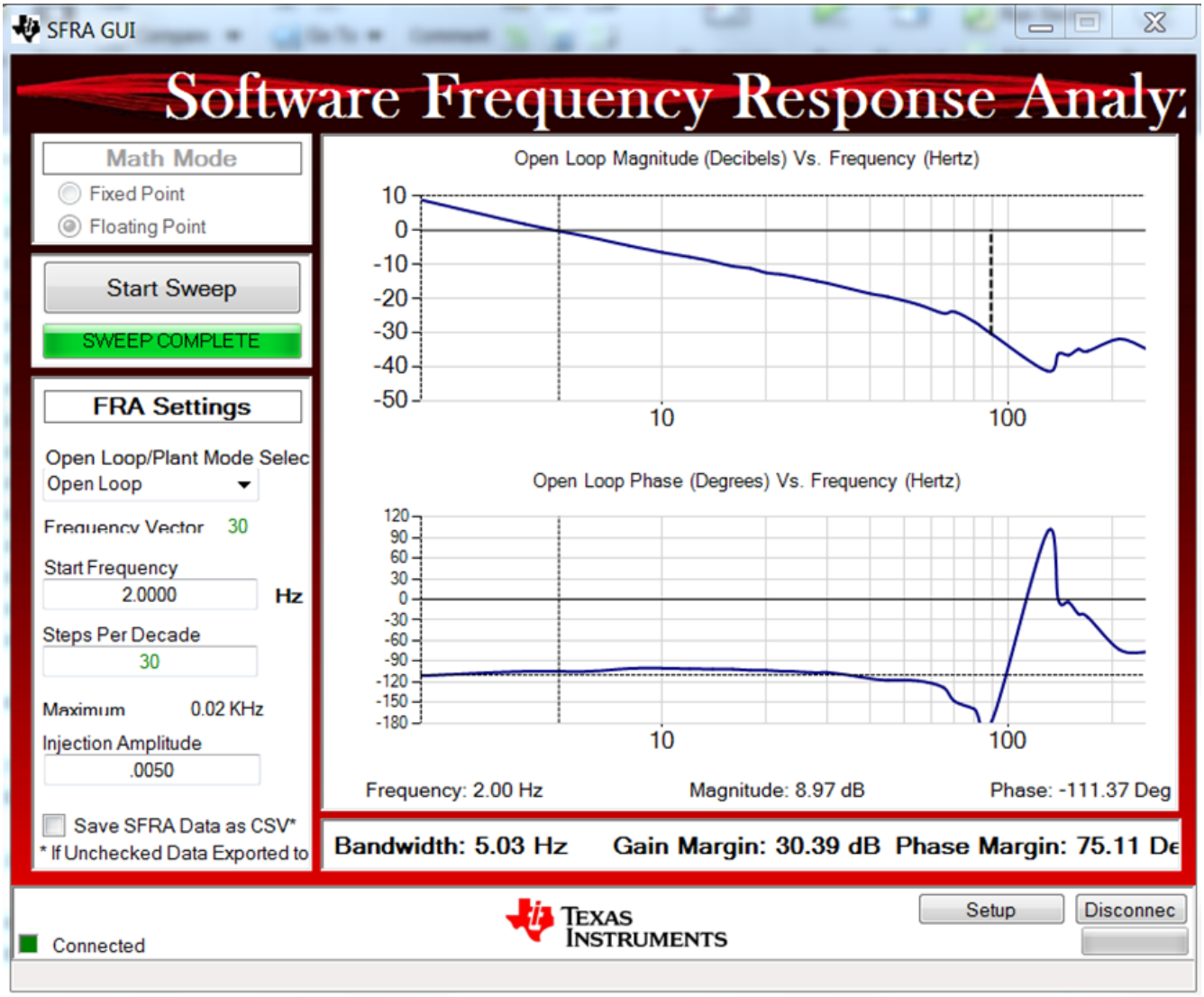


Figure 3-29. SFRA Run on Closed Voltage Loop

The frequency response data is also saved in the project folder under an SFRA data folder and is time stamped with the time of the SFRA run.

Note the measured gain and phase margin are close to the modeled values, as shown in Figure 2-12 .

7. Optionally. Click on the Compensation Designer again from the CFG page, and choose *SFRA Data* for plant option on the GUI. This option uses the measured plant information to design the compensator, and can be used to fine tune the compensation. By default the Compensation Designer points to the latest SFRA run. If a previous SFRA run plant information must be used the user can select the *SFRAData.csv* file by browsing to it by clicking on *Browse SFRA Data*. Close the Compensation Designer to return to the cfg page once done.
8. This verifies the voltage compensator design.
9. To bring the system to a safe stop bring the input AC voltage down to zero, observe the TTPLPFC_vBus_sensed_Volts comes down to zero as well.
10. Fully halting the MCU when in real-time mode is a two-step process. First halt the processor by using the *Halt* button on the toolbar ( ) or by using *Target* → *Halt*. Then take the MCU out of real-time mode by clicking on . Finally, reset the MCU ( ).
11. Close CCS debug session by clicking on *Terminate Debug Session* (*Target* → *Terminate all*).

3.1.2.5.5 Lab 5: Open loop, DC (Inverter)

In this build the board is excited in open loop fashion with a fixed duty cycle for inverter mode operation. The duty cycle is controlled with dutyPU_DC variable. The test procedure is similar to Lab 1. The software structure for this build is the same as the one in Lab 1 shown in Figure 3-9. It should be noted that HW setup for Lab 5 is different from Lab 1. In this build, dc power supply has to be connected to J104 and the resistive load has to be moved to J100.

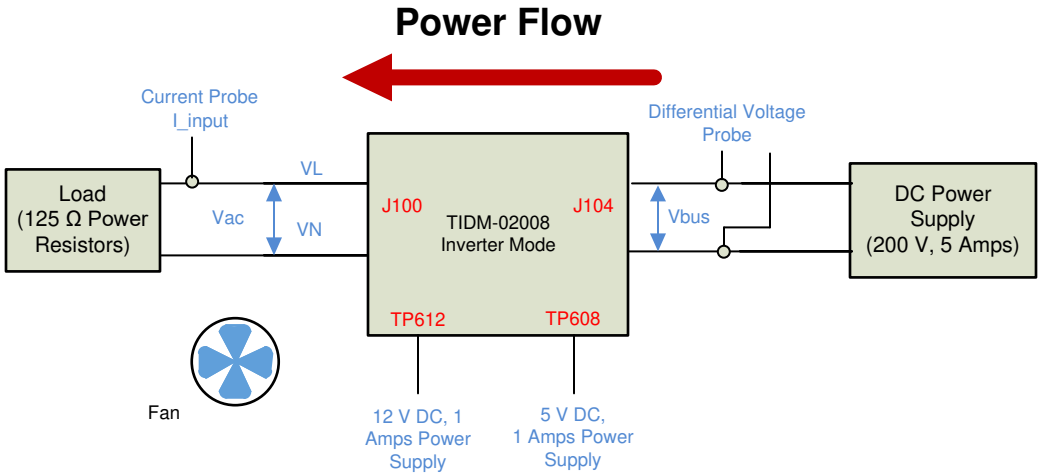


Figure 3-30. HW Setup for Lab 5

3.1.2.5.5.1 Setting Software Options for Lab 5

1. powerSUITE Settings : On the powerSUITE page select under *Project Options* section:
   - Select Lab 5 under lab option.
   - Also disable the other options such as *Non Linear Voltage Loop, Adaptive Deadtime and Phase shedding*
2. If this is an adapted solution, edit the setting under *Voltage and Current Sensing Parameters* . One can refer to the *calculations.xlsx* file which is available under the C2000Ware DigitalPower SDK Install directory at *<install_location>\solutions\tidm_02008\hardware* for details on sensing circuit and how max range is computed for the powerSUITE page
3. Under Power Stage Parameters specify the switching frequency, the dead band, and the power rating. Save the page.

3.1.2.5.5.2 Building and Loading Project

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 tragetconfigs (Figure 3-4).
4. Then click *Run* → *Debug*. This action launches a debugging session. In the case of dual CPU devices, a window may appear to select the CPU that the debug must be performed. In this case, select CPU1.
5. The project then loads on the device, and CCS debug view becomes active. The code halts at the start of the main routine.

3.1.2.5.5.3 Setup Debug Environment Windows

1. To add the variables in the watch and expressions window, click *View* → *Scripting Console* to open the scripting console dialog box. On the upper-right corner of this console, click on open then browse to the *setupdebugenv_lab5.js* script file located inside the project folder. This script file populates the watch window with appropriate variables required to debug the system. Click on the Continuous Refresh button on the watch window to enable continuous update of values from the controller.

Variables | Expressions

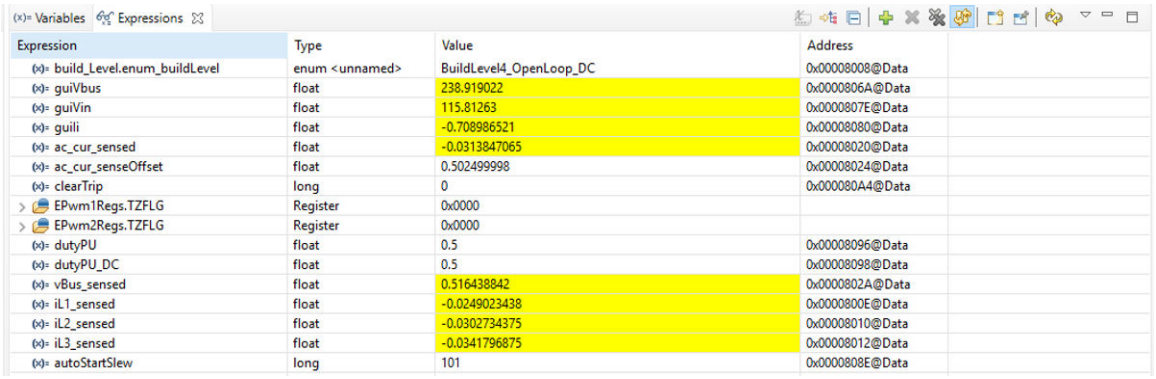
Expression	Type	Value	Address
build_Level.enum_buildLevel	enum <unnamed>	BuildLevel4_OpenLoop_DC	0x00008008@Data
guiVbus	float	0.113195412	0x0000806A@Data
guiVin	float	0.338019282	0x0000807E@Data
guili	float	0.376971513	0x00008080@Data
ac_cur_sensed	float	0.0154055059	0x00008020@Data
ac_cur_senseOffset	float	0.502499998	0x00008024@Data
clearTrip	long	0	0x000080A4@Data
EPwm1Regs.TZFLG	Register	0x0004	
EPwm2Regs.TZFLG	Register	0x0004	
dutyPU	float	0.5	0x00008096@Data
dutyPU_DC	float	0.5	0x00008098@Data
vBus_sensed	float	8.10623169e-05	0x0000802A@Data
iL1_sensed	float	-0.009765625	0x0000800E@Data
iL2_sensed	float	-0.0112304688	0x00008010@Data
iL3_sensed	float	-0.00439453125	0x00008012@Data
autoStartSlew	long	28	0x0000808E@Data

Figure 3-31. Lab 5 Expressions View

2. Run the project by clicking 
3. Now Halt the processor by using the *Halt* button on the toolbar ()

3.1.2.5.5.4 Running Code

1. Now run the project again by clicking on  .
2. In a few seconds the inrush relay clicks, the software is programmed to do so in the lab with DC. The trip clears, and a duty cycle of 0.5 is applied.
3. In the watch view, check if the guiVin, guiVbus, guili, variables are updating, periodically.
   - Note: As no power is applied right now, this value is close to zero.
4. Now slowly increase the input DC voltage from zero to 240 V. The output voltage shows a step down voltage (buck converter operation) as a steady duty cycle of 0.5 PU is applied as default setting. If a high current is drawn, verify if the voltage terminals are swapped. If true, reduce the voltage to zero first and correct the issue before resuming the test
5. Verifying the voltage sensing: Make sure *TTPLPFC_ac_vol_sensed_Volts* and *TTPLPFC_vBus_sensed_Volts* display the correct values, for 240-V DC input, TTPLPFC_ac_vol_sensed_Volts is close to 120V. This verifies the voltage sensing of the board in some manner.
6. Verifying the current sensing: Notice the *TTPLPFC_ac_curRms_sensed_Amps* for the given test condition; this value is close to 1 A.



Expression	Type	Value	Address
build_Level.enum_buildLevel	enum <unnamed>	BuildLevel4_OpenLoop_DC	0x00008008@Data
guiVbus	float	238.919022	0x0000806A@Data
guiVin	float	115.81263	0x0000807E@Data
guili	float	-0.708986521	0x00008080@Data
ac_cur_sensed	float	-0.0313847065	0x00008020@Data
ac_cur_senseOffset	float	0.502499998	0x00008024@Data
clearTrip	long	0	0x000080A4@Data
EPwm1Regs.TZFLG	Register	0x0000	
EPwm2Regs.TZFLG	Register	0x0000	
dutyPU	float	0.5	0x00008096@Data
dutyPU_DC	float	0.5	0x00008098@Data
vBus_sensed	float	0.516438842	0x0000802A@Data
iL1_sensed	float	-0.0249023438	0x0000800E@Data
iL2_sensed	float	-0.0302734375	0x00008010@Data
iL3_sensed	float	-0.0341796875	0x00008012@Data
autoStartSlew	long	101	0x0000808E@Data

Figure 3-32. Lab 5 Watch Expression Showing Measured Voltage and Currents

7. Once finished, reduce the input voltage to zero and watch for the bus voltages to reduce down to zero.
8. This completes the check for this build, the following items are verified on successful completion of this build:
   - Sensing of voltages and currents and scaling to be correct
   - Interrupt generation and execution of the LAb 5 code in the current loop ISR and Voltage Loop Instrumentation ISR
   - PWM driver and switching

   If any issue is observed a careful inspection of the hardware may be required to eliminate any build issues and so forth.
9. The controller can now be halted, and the debug connection terminated.
10. Fully halting the MCU when in real-time mode is a two-step process. First halt the processor by using the *Halt* button on the toolbar ( ) or by using *Target* → *Halt*. Then take the MCU out of real-time mode by clicking on . Finally, reset the MCU by clicking on .
11. Close CCS debug session by clicking on *Terminate Debug Session* (*Target* → *Terminate all*).

3.1.2.5.6 Lab 6: Open loop, AC (Inverter)

In this build the board is excited in open loop fashion with a sinusoidal duty cycle for inverter mode operation. The duty cycle is controlled with dutyPU variable. The test procedure is similar to LAb 5. HW setup is identical to Lab 5 shown in Figure 3-30.

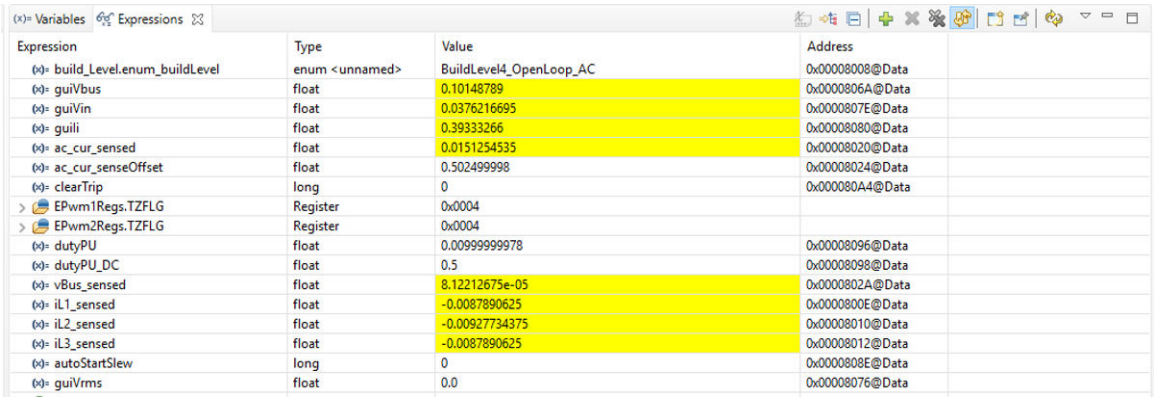
3.1.2.5.6.1 Setting Software Options for Lab 6

Setting Software Options for powerSUITE Settings : On the powerSUITE page select under *Project Options* section:

- Select Lab 6 under Lab option.
- Also disable the other options such as *Non Linear Voltage Loop, Adaptive Deadtime and Phase shedding*

3.1.2.5.6.2 Building and Loading Project and Setting up Debug

1. Right click on the project name, and click *Rebuild Project*. The project builds successfully. Click *Run* → *Debug*, which launches a debugging session. In the case of dual CPU devices, a window may appear to select the CPU the debug must be performed. In this case, select CPU1. The project then loads on the device, and CCS debug view becomes active. The code halts at the start of the main routine.
2. To add the variables in the watch and expressions window click *View* → *Scripting Console* to open the scripting console dialog box. On the upper right corner of this console, click on *Open* to browse to the *setupdebugenv_lab6.js* script file, which is located inside the project folder. This file populates the watch window with appropriate variables required to debug the system. Click on *Continuous Refresh* button ( ) on the watch window to enable continuous update of values from the controller. The watch window appears as 3.



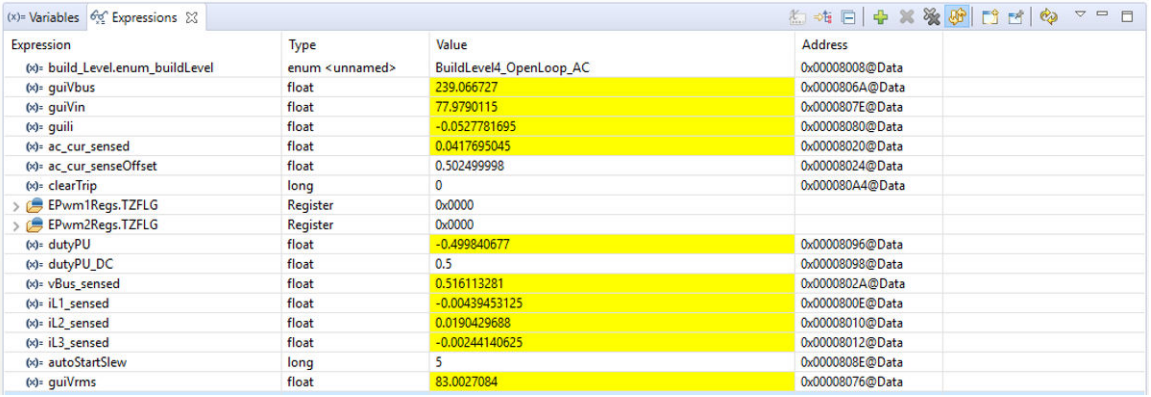
Expression	Type	Value	Address
build_Level.enum_buildLevel	enum <unnamed>	BuildLevel4_OpenLoop_AC	0x00008008@Data
guiVbus	float	0.10148789	0x0000806A@Data
guiVin	float	0.0376216695	0x0000807E@Data
guili	float	0.39333266	0x00008080@Data
ac_cur_sensed	float	0.0151254535	0x00008020@Data
ac_cur_senseOffset	float	0.502499998	0x00008024@Data
clearTrip	long	0	0x000080A4@Data
EPwm1Regs.TZFLG	Register	0x0004	
EPwm2Regs.TZFLG	Register	0x0004	
dutyPU	float	0.00999999978	0x00008096@Data
dutyPU_DC	float	0.5	0x00008098@Data
vBus_sensed	float	8.12212675e-05	0x0000802A@Data
iL1_sensed	float	-0.0087890625	0x0000800E@Data
iL2_sensed	float	-0.00927734375	0x00008010@Data
iL3_sensed	float	-0.0087890625	0x00008012@Data
autoStartSlew	long	0	0x0000808E@Data
guiVrms	float	0.0	0x00008076@Data

Figure 3-33. Lab 6 Expressions View

3. Enable real-time mode by hovering the mouse on the buttons on the horizontal toolbar, and clicking the  button.

3.1.2.5.6.3 Running Code

1. Now run the project again by clicking  .
2. The inrush relay is closed when the DC bus voltage is higher than 110 V. The trip clears, and a sinusodal duty cycle is applied.
3. In the watch view, check if the TTPLPFC_ac_volRms_sensed_Volts, TTPLPFC_vBus_sensed_Volts, TTPLPFC_ac_curRms_sensed_Amps, variables are updating, periodically.
   - Note: As no power is applied right now, this value is close to zero.
4. Now slowly increase the input DC voltage from zero to 240 V.
5. Verifying the voltage sensing: Make sure *TTPLPFC_ac_volRms_sensed_Volts* and *TTPLPFC_vBus_sensed_Volts* display the correct values, for 240-V DC input, TTPLPFC_ac_volRms_sensed_Volts is close to 85V. This verifies the voltage sensing of the board in some manner.
6. Verifying the current sensing: Notice the *TTPLPFC_ac_curRms_sensed_Amps* for the given test condition; this value is close to 0.65 A.



Expression	Type	Value	Address
build_Level.enum_buildLevel	enum <unnamed>	BuildLevel4_OpenLoop_AC	0x00008008@Data
guiVbus	float	239.066727	0x0000806A@Data
guiVin	float	77.9790115	0x0000807E@Data
guili	float	-0.0527781695	0x00008080@Data
ac_cur_sensed	float	0.0417695045	0x00008020@Data
ac_cur_senseOffset	float	0.502499998	0x00008024@Data
clearTrip	long	0	0x000080A4@Data
EPwm1Regs.TZFLG	Register	0x0000	
EPwm2Regs.TZFLG	Register	0x0000	
dutyPU	float	-0.499840677	0x00008096@Data
dutyPU_DC	float	0.5	0x00008098@Data
vBus_sensed	float	0.516113281	0x0000802A@Data
iL1_sensed	float	-0.00439453125	0x0000800E@Data
iL2_sensed	float	0.0190429688	0x00008010@Data
iL3_sensed	float	-0.00244140625	0x00008012@Data
autoStartSlew	long	5	0x0000808E@Data
guiVrms	float	83.0027084	0x00008076@Data

Figure 3-34. Lab 6 Watch Expression Showing Measured Voltage and Currents

7. The controller can now be halted, and the debug connection terminated.
8. Fully halting the MCU when in real-time mode is a two-step process. First halt the processor by using the *Halt* button on the toolbar () or by using *Target* → *Halt*. Then take the MCU out of real-time mode by clicking on  . Finally, reset the MCU by clicking on  .
9. Close CCS debug session by clicking on *Terminate Debug Session* (*Target* → *Terminate all*).

3.1.2.5.7 Lab 7: Closed Current Loop, DC (Inverter with resistive load)

In this Lab, the inner current loop is closed that is the inductor current is controlled using a current compensator Gi. The current loop model is identical to PFC mode operation shown in Section 3.1.2.5.2 but the polarity of the current reference is opposite to PFC mode (negatvie current reference).

3.1.2.5.7.1 Setting Software Options for Lab 7

1. Make sure the hardware is setup as outlined in Figure 3-30. Do not supply any high voltage (HV) power to the board yet.
2. powerSUITE Settings : On the powerSUITE page select under *Project Options* section:
   - Select Lab 7 under Lab option.
   - Select input to be DC under PFC INPUT/INV Output options
   - Also disable the other options such as *Non Linear Voltage Loop, Adaptive Deadtime and Phase shedding*
3. Assuming all other options are same as specified earlier in Section 3.1.2.5.5.1.
4. Under *Control Loop Design*, options for the current loop tuning automatically be selected (*Tuning* → *Current Loop* → *COMP1* → *DCL_PI_C1*). Now click on the *Compensation Designer* icon ( ).

3.1.2.5.7.2 Designing Current Loop Compensator

The compensator design for inverter mode is identical to PFC mode. The compensator design procedures are provided in Section 3.1.2.5.2.2.

3.1.2.5.7.3 Building and Loading Project and Setting up Debug

1. Right click the project name, and click *Rebuild Project*. The project builds successfully. Click *Run* → *Debug*, which launches a debugging session. In the case of dual CPU devices, a window may appear to select the CPU the debug must be performed. In this case, select CPU1. The project then loads on the device, and CCS debug view becomes active. The code halts at the start of the main routine.
2. To add the variables in the watch and expressions window click *View* → *Scripting Console* to open the scripting console dialog box. On the upper right corner of this console, click on *Open* to browse to the *setupdebugenv_lab7.js* script file, which is located inside the project folder. This file populates the watch window with appropriate variables required to debug the system. Click on *Continuous Refresh* button ( ) on the watch window to enable continuous update of values from the controller. The watch window appears as Figure 3-35.

Variables | Expressions

Expression	Type	Value	Address
build_Level.enum_buildLevel	enum <unnamed>	BuildLevel5_CurrentLoop_DC	0x00008008@Data
pwm_SwState.enum_pwmSwState	enum <unnamed>	pwmSwState_defaultState	0x000080B6@Data
board_Status.enum_boardStatus	enum <unnamed>	boardStatus_Idle	0x0000800C@Data
clearTrip	long	0	0x000080A4@Data
closeGiLoop	long	0	0x000080A0@Data
ac_cur_ref	float	0.0299999993	0x00008046@Data
ac_cur_sensed	float	0.0154101253	0x00008020@Data
ac_cur_senseOffset	float	0.502499998	0x00008024@Data
guiVbus	float	0.0884667262	0x0000806A@Data
guiVin	float	-0.867004335	0x0000807E@Data
guiVrms	float	0.0	0x00008076@Data
guiIrms	float	0.0	0x00008074@Data
guiPrms	float	0.0	0x00008072@Data
guiFreqAvg	float	0.0	0x00008088@Data
guiPowerFactor	float	0.0	0x00008084@Data
EPwm1Regs.TZFLG	Register	0x0004	
EPwm2Regs.TZFLG	Register	0x0004	
dutyPU	float	0.00999999978	0x00008096@Data
dutyPU_DC	float	0.5	0x00008098@Data
iL1_sensed	float	-0.0087890625	0x0000800E@Data
iL2_sensed	float	-0.0112304688	0x00008010@Data
iL3_sensed	float	-0.00732421875	0x00008012@Data
autoStartSlew	long	25	0x0000808E@Data
thetaOffset	float	0.0	0x00008028@Data

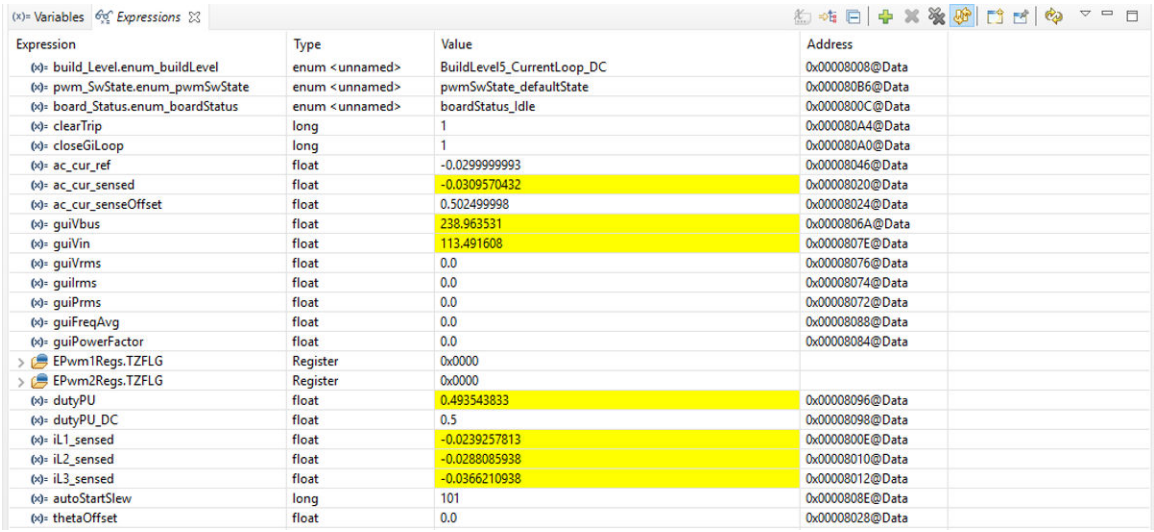
Figure 3-35. Lab 7: Closed Current Loop Expressions View

3. Enable real-time mode by hovering the mouse on the buttons on the horizontal toolbar, and clicking the button.
4. Run the project by clicking on

5. Now Halt the processor by using the *Halt* button on the toolbar ()

3.1.2.5.7.4 Running Code

1. The project is programmed to drive the inrush relay and clear the trip after a set amount of time, that is, TTPLPFC_autoStartSlew==100. The software is programmed to do so in the lab with DC. An input voltage must be applied after hitting run and before this autoslew counter reaches 100. If the counter reaches 100, before voltage is applied at the input, the code must be reset. For which the controller must be brought out of real time mode, a reset performed and restarted.
2. Now run the project by clicking  .
3. Apply an input voltage of approximately 120 V before the autoStartSlew reaches 100. As soon autoStartSlew reaches 100, the inrush relay is triggered, and PWM trip is cleared along with closing the current loop flag.
4. TTPLPFC_ac_cur_ref is set to -0.03 by default and the output voltage is close to 120 V and output current is close to 1 A.
5. Now slowly increase Vbus=240 V and see if output voltage and current stay in 120 V and 1 A.



Expression	Type	Value	Address
build_Level.enum_buildLevel	enum <unnamed>	BuildLevel5_CurrentLoop_DC	0x00008008@Data
pwm_SwState.enum_pwmSwState	enum <unnamed>	pwmSwState_defaultState	0x000080B6@Data
board_Status.enum_boardStatus	enum <unnamed>	boardStatus_Idle	0x0000800C@Data
clearTrip	long	1	0x000080A4@Data
closeGiLoop	long	1	0x000080A0@Data
ac_cur_ref	float	-0.0299999993	0x00008046@Data
ac_cur_sensed	float	-0.0309570432	0x00008020@Data
ac_cur_senseOffset	float	0.502499998	0x00008024@Data
guiVbus	float	238.963531	0x0000806A@Data
guiVin	float	113.491608	0x0000807E@Data
guiVrms	float	0.0	0x00008076@Data
guiIrms	float	0.0	0x00008074@Data
guiPrms	float	0.0	0x00008072@Data
guiFreqAvg	float	0.0	0x00008088@Data
guiPowerFactor	float	0.0	0x00008084@Data
EPwm1Regs.TZFLG	Register	0x0000	
EPwm2Regs.TZFLG	Register	0x0000	
dutyPU	float	0.493543833	0x00008096@Data
dutyPU_DC	float	0.5	0x00008098@Data
iL1_sensed	float	-0.0239257813	0x0000800E@Data
iL2_sensed	float	-0.0288085938	0x00008010@Data
iL3_sensed	float	-0.0366210938	0x00008012@Data
autoStartSlew	long	101	0x0000808E@Data
thetaOffset	float	0.0	0x00008028@Data

Figure 3-36. Watch Expression, LAb 7, After Closed Current Loop

6. In this build, SFRA can measure the frequency response. However the resistive load changes the plant pole location and the measurement result is different from Figure 3-24. SFRA for inverter mode will be revisited in Lab 8.
7. To bring the system to a safe stop, bring the input DC voltage down to zero, observe the TTPLPFC_vBus_sensed_Volts comes down to zero as well.
8. Fully halting the MCU when in real-time mode is a two-step process. First halt the processor by using the *Halt* button on the toolbar () or by using *Target* → *Halt*. Then take the MCU out of real-time mode by clicking on  . Finally, reset the MCU () .
9. Close the CCS debug session by clicking on *Terminate Debug Session* (*Target* → *Terminate all*). 

3.1.2.5.8 Lab 8: Closed Current Loop, AC (Inverter with resistive load)

In this lab, the inner current loop is closed that is the inductor current is controlled using a current compensator Gi. Both DC bus and output voltage feedforward are applied to the output of this current compensator to generate the duty cycle of the inverter along with soft start for PWM around the zero crossing The current loop model is identical to PFC mode operation shown in Section 3.1.2.5.2 except for the direction of the reference current.

3.1.2.5.8.1 Setting Software Options for Lab 8

1. Make sure the hardware is setup as outlined in Figure 3-10. Do not supply any high voltage (HV) power to the board yet.

2. powerSUITE Settings : On the powerSUITE page select under *Project Options* section:
   - Select Lab 8 under Lab option.
   - Also disable the other options such as *Non Linear Voltage Loop, Adaptive Deadtime and Phase shedding*
3. Assuming all other options are same as specified earlier in Section 3.1.2.5.5.1 .
4. Under *Control Loop Design*, options for the current loop tuning automatically be selected (*Tuning → Current Loop → COMP1 → DCL_PI_C1*). Now click on the *Compensation Designer* icon ( ).

3.1.2.5.8.2 Building and Loading Project and Setting up Debug

1. Right click on the project name, and click *Rebuild Project*. The project builds successfully. Click *Run → Debug*, which launches a debugging session. In the case of dual CPU devices, a window may appear to select the CPU the debug must be performed. In this case, select CPU1. The project then loads on the device, and CCS debug view becomes active. The code halts at the start of the main routine.
2. To add the variables in the watch and expressions window click *View → Scripting Console* to open the scripting console dialog box. On the upper right corner of this console, click on *Open* to browse to the *setupdebugenv_lab8.js* script file, which is located inside the project folder. This file populates the watch window with appropriate variables required to debug the system. Click on *Continuous Refresh* button ( ) on the watch window to enable continuous update of values from the controller. The watch window appears as Figure 3-35.

Variables | Expressions

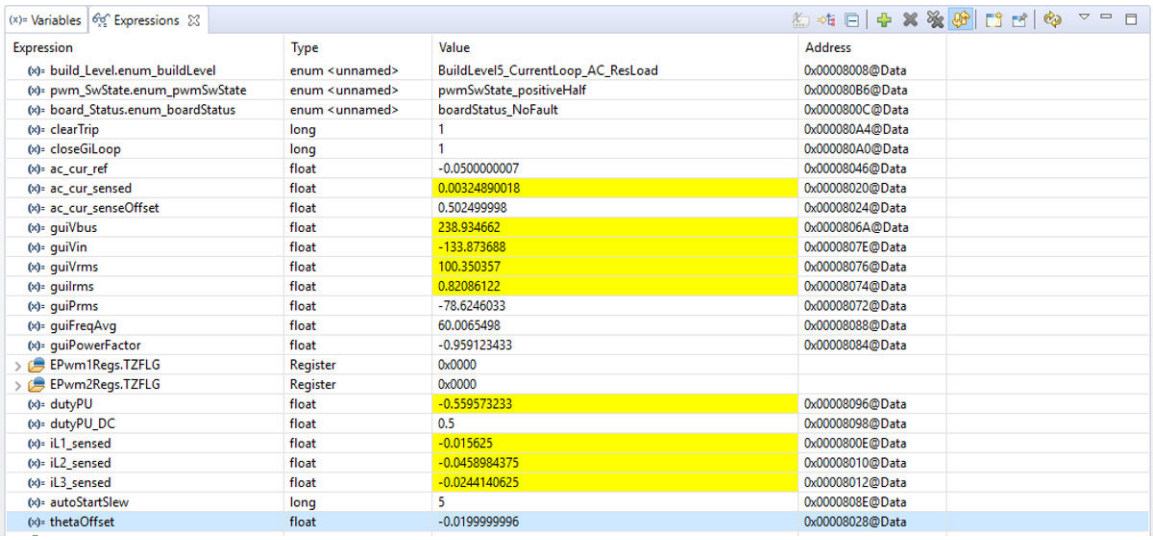
Expression	Type	Value	Address
build_Level.enum_buildLevel	enum <unnamed>	BuildLevel5_CurrentLoop_AC_ResLoad	0x00008008@Data
pwm_SwState.enum_pwmSwState	enum <unnamed>	pwmSwState_positiveHalf	0x000080B6@Data
board_Status.enum_boardStatus	enum <unnamed>	boardStatus_NoFault	0x0000800C@Data
clearTrip	long	1	0x000080A4@Data
closeGiLoop	long	1	0x000080A0@Data
ac_cur_ref	float	-0.0500000007	0x00008046@Data
ac_cur_sensed	float	0.00324890018	0x00008020@Data
ac_cur_senseOffset	float	0.502499998	0x00008024@Data
guiVbus	float	238.934662	0x0000806A@Data
guiVin	float	-133.873688	0x0000807E@Data
guiVrms	float	100.350357	0x00008076@Data
guiIrms	float	0.82086122	0x00008074@Data
guiPrms	float	-78.6246033	0x00008072@Data
guiFreqAvg	float	60.0065498	0x00008088@Data
guiPowerFactor	float	-0.959123433	0x00008084@Data
EPwm1Regs.TZFLG	Register	0x0000	
EPwm2Regs.TZFLG	Register	0x0000	
dutyPU	float	-0.559573233	0x00008096@Data
dutyPU_DC	float	0.5	0x00008098@Data
iL1_sensed	float	-0.015625	0x0000800E@Data
iL2_sensed	float	-0.0458984375	0x00008010@Data
iL3_sensed	float	-0.0244140625	0x00008012@Data
autoStartSlew	long	5	0x0000808E@Data
thetaOffset	float	-0.0199999996	0x00008028@Data

Figure 3-37. Lab 8: Closed Current Loop Expressions View

3. Enable real-time mode by hovering the mouse on the buttons on the horizontal toolbar, and clicking the button.
4. Run the project by clicking on
5. Now Halt the processor by using the *Halt* button on the toolbar

3.1.2.5.8.3 Running Code

1. The project is programmed to wait for dc bus voltage to exceed approximately 110V to drive the in rush relay, and clear the trip.
2. Run the project by clicking .
3. Now apply an input voltage of approximately 120 V, the board comes out of the undervoltage condition and inrush relay is driven and change TTPLPFC_pwmSwState from pwmSwState_defaultState to pwmSwState_normalOperation to enable pwm output. **The pwm output is turned off by default and the inverter does not work without selecting this option**.
4. Slowly increase the bus voltage to 240 V and set TTPLPFC_ac_cur_ref = -0.05. The watch window looks similar to Figure 3-22. The RMS value of inverter output voltage and current are close to 100 V and 0.8 A respectively.



Expression	Type	Value	Address
build_Level.enum_buildLevel	enum <unnamed>	BuildLevel5_CurrentLoop_AC_ResLoad	0x00008008@Data
pwm_SwState.enum_pwmSwState	enum <unnamed>	pwmSwState_positiveHalf	0x000080B6@Data
board_Status.enum_boardStatus	enum <unnamed>	boardStatus_NoFault	0x0000800C@Data
clearTrip	long	1	0x000080A4@Data
closeGiLoop	long	1	0x000080A0@Data
ac_cur_ref	float	-0.0500000007	0x00008046@Data
ac_cur_sensed	float	0.00324890018	0x00008020@Data
ac_cur_senseOffset	float	0.502499998	0x00008024@Data
guiVbus	float	238.934662	0x0000806A@Data
guiVin	float	-133.873688	0x0000807E@Data
guiVrms	float	100.350357	0x00008076@Data
guiIrms	float	0.82086122	0x00008074@Data
guiPrms	float	-78.6246033	0x00008072@Data
guiFreqAvg	float	60.0065498	0x00008088@Data
guiPowerFactor	float	-0.959123433	0x00008084@Data
EPwm1Regs.TZFLG	Register	0x0000	
EPwm2Regs.TZFLG	Register	0x0000	
dutyPU	float	-0.559573233	0x00008096@Data
dutyPU_DC	float	0.5	0x00008098@Data
iL1_sensed	float	-0.015625	0x0000800E@Data
iL2_sensed	float	-0.0458984375	0x00008010@Data
iL3_sensed	float	-0.0244140625	0x00008012@Data
autoStartSlew	long	5	0x0000808E@Data
thetaOffset	float	-0.0199999996	0x00008028@Data

Figure 3-38. Watch Expression, Lab 8, AC After Closed Current Loop

5. To bring the system to a safe stop, switch off the output from the AC power supply thus bring the input AC voltage down to zero, observe the TTPLPFC_ac_volRms_sensed_Volts comes down to zero as well.
6. Fully halting the MCU when in real-time mode is a two-step process. First halt the processor by using the *Halt* button on the toolbar ( ) or by using *Target* → *Halt*. Then take the MCU out of real-time mode by clicking on . Finally, reset the MCU ( ) .
7. Close the CCS debug session by clicking on *Terminate Debug Session* (*Target* → *Terminate all*).

3.1.2.5.9 Lab 9: Closed Current Loop (Grid Connected Inverter)

In this lab, the inner current loop is closed under the grid-tied condition. Both DC bus and output voltage feedforward are applied to the output of this current compensator to generate the duty cycle of the inverter along with soft start for PWM around the zero crossing .The current loop model is identical to PFC mode operation shown in Section 3.1.2.5.2.

3.1.2.5.9.1 Setting Software Options for Lab 9

1. powerSUITE Settings : On the powerSUITE page select under *Project Options* section:
   - Select Lab 9 under lab option.
   - Also disable the other options such as *Non Linear Voltage Loop, Adaptive Deadtime and Phase shedding*
2. Assuming all other options are same as specified earlier in Section 3.1.2.5.5.1 .
3. Under *Control Loop Design*, options for the current loop tuning automatically be selected (*Tuning* → *Current Loop* → *COMP1* → *DCL_PI_C1*). Now click on the *Compensation Designer* icon ( ).

3.1.2.5.9.2 Building and Loading Project and Setting up Debug

1. Right click on the project name, and click *Rebuild Project*. The project builds successfully. Click *Run* → *Debug*, which launches a debugging session. In the case of dual CPU devices, a window may appear to select the CPU the debug must be performed. In this case, select CPU1. The project then loads on the device, and CCS debug view becomes active. The code halts at the start of the main routine.
2. To add the variables in the watch and expressions window click *View* → *Scripting Console* to open the scripting console dialog box. On the upper right corner of this console, click on *Open* to browse to the *setupdebugenv_lab9.js* script file, which is located inside the project folder. This file populates the watch window with appropriate variables required to debug the system. Click on *Continuous Refresh* button ( ) on the watch window to enable continuous update of values from the controller. The watch window appears as Figure 3-35.

Expression	Type	Value	Address
build_Level.enum_buildLevel	enum <unnamed>	BuildLevel6_CurrentLoop_GridConnected	0x00008008@Data
pwm_SwState.enum_pwmSwState	enum <unnamed>	pwmSwState_defaultState	0x000080B6@Data
board_Status.enum_boardStatus	enum <unnamed>	boardStatus_InputUnderVoltageTrip	0x0000800C@Data
clearTrip	long	0	0x000080A4@Data
closeGiLoop	long	0	0x000080A0@Data
ac_cur_ref	float	0.0299999993	0x00008046@Data
ac_cur_sensed	float	0.0174335241	0x00008020@Data
ac_cur_senseOffset	float	0.502499998	0x00008024@Data
guiVbus	float	0.0863133222	0x0000806A@Data
guiVin	float	0.0752274767	0x0000807E@Data
guiVrms	float	0.0	0x00008076@Data
guiIrms	float	0.0	0x00008074@Data
guiPrms	float	0.0	0x00008072@Data
guiFreqAvg	float	0.0	0x00008088@Data
guiPowerFactor	float	0.0	0x00008084@Data
EPwm1Regs.TZFLG	Register	0x0004	
EPwm2Regs.TZFLG	Register	0x0004	
dutyPU	float	0.00999999978	0x00008096@Data
dutyPU_DC	float	0.5	0x00008098@Data
iL1_sensed	float	-0.0068359375	0x0000800E@Data
iL2_sensed	float	-0.00927734375	0x00008010@Data
iL3_sensed	float	-0.00537109375	0x00008012@Data
autoStartSlew	long	0	0x0000808E@Data
thetaOffset	float	0.0	0x00008028@Data

Figure 3-39. Lab 9: Closed Current Loop (Grid-connected) Expressions View

3. Enable real-time mode by hovering the mouse on the buttons on the horizontal toolbar, and clicking the button.
4. Run the project by clicking on
5. Now Halt the processor by using the *Halt* button on the toolbar (

3.1.2.5.9.3 Running Code: Emulated Grid-tied Condition (Verification purpose only)

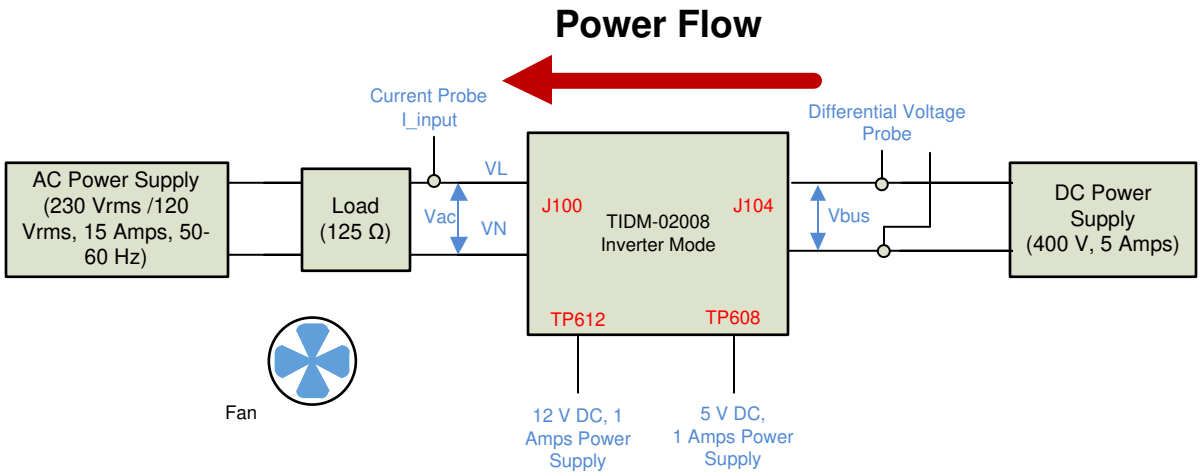


Figure 3-40. HW setup for Build 6 Emulated Grid Condition

1. The project is programmed to wait for dc bus voltage and ac source voltage to exceed approximately 340 V and 75 Vrms to drive the in rush relay, and clear the trip.
2. Run the project by clicking .
3. Now apply an input voltage of approximately 340 V, the board comes out of the undervoltage condition. To run the solution, ac voltage in the output of the inverter has to be higher than 75 V rms and then inrush relay is driven. TTPLPFC_ac_cur_ref is set to -0.03 by default.
4. Change TTPLPFC_pwmSwState from pwmSwState_defaultState to pwmSwState_normalOperation to enable pwm output. **The pwm output is turned off by default and the inverter does not work without selecting this option.**
5. Slowly increase TTPLPFC_ac_cur_ref to -0.05 and inverter output current is close to 0.8 A in RMS value. The output voltage of the inverter is determined by the ac voltage source.

6. As the source impedance of the emulgated grid condition is not as small as the actual grid, the bandwidth of the current loop is much lower than the designed target and the system suffers from huge current spikes near zero crossing.

Variables | Expressions

Expression	Type	Value	Address
build_Level.enum_buildLevel	enum <unnamed>	BuildLevel6_CurrentLoop_GridConnected	0x00008008@Data
pwm_SwState.enum_pwmSwState	enum <unnamed>	pwmSwState_negativeHalf	0x000080B6@Data
board_Status.enum_boardStatus	enum <unnamed>	boardStatus_NoFault	0x0000800C@Data
clearTrip	long	1	0x000080A4@Data
closeGiLoop	long	1	0x000080A0@Data
ac_cur_ref	float	-0.0299999993	0x00008046@Data
ac_cur_sensed	float	0.0243725181	0x00008020@Data
ac_cur_senseOffset	float	0.502499998	0x00008024@Data
guiVbus	float	378.007446	0x0000806A@Data
guiVin	float	-63.893631	0x0000807E@Data
guiVrms	float	118.721397	0x00008076@Data
guiIrms	float	0.476669252	0x00008074@Data
guiPrms	float	-51.2441063	0x00008072@Data
guiFreqAvg	float	60.0667419	0x00008088@Data
guiPowerFactor	float	-0.880266964	0x00008084@Data
EPwm1Regs.TZFLG	Register	0x0000	
EPwm2Regs.TZFLG	Register	0x0000	
dutyPU	float	-0.0921395048	0x00008096@Data
dutyPU_DC	float	0.5	0x00008098@Data
iL1_sensed	float	-0.0190429688	0x0000800E@Data
iL2_sensed	float	0.00927734375	0x00008010@Data
iL3_sensed	float	0.0078125	0x00008012@Data
autoStartSlew	long	5	0x0000808E@Data
thetaOffset	float	-0.00999999978	0x00008028@Data

Figure 3-41. Lab 9: Closed Current Loop after close the current loop

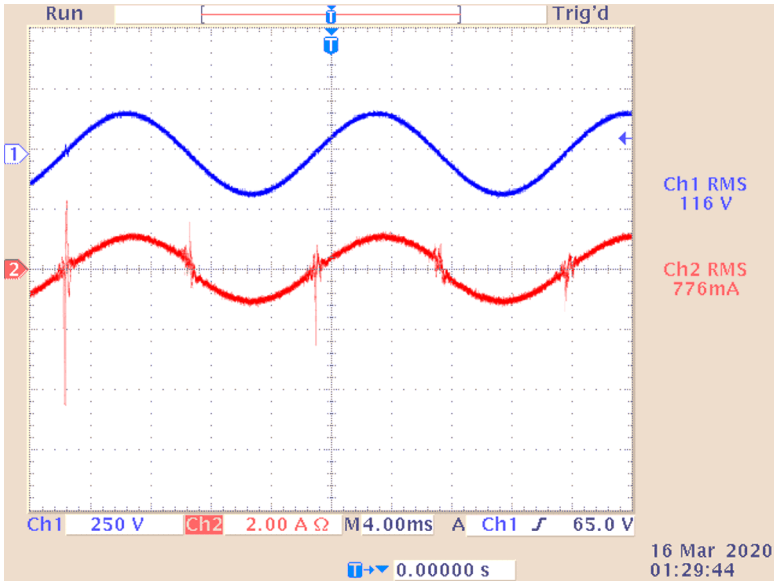


Figure 3-42. Voltage and current waveform (Lab 9 Emulated Grid Condition)

7. To bring the system to a safe stop, switch off the output from the AC power supply first and switch off the input DC power supply subsequently. This will eliminate the risk of the undesired reverse power flow in the inverter mode. Bring input AC voltage down to zero and observe the TTPLPFC_ac_volRms_sensed_Volts comes down to zero as well. After AC voltage is fully switched off, the the input DC voltage has to be down to zero, observe the TTPLPFC_vBus_sensed_Volts becomes zero.
8. Fully halting the MCU when in real-time mode is a two-step process. First halt the processor by using the *Halt* button on the toolbar ( ) or by using *Target* → *Halt*. Then take the MCU out of real-time mode by clicking on . Finally, reset the MCU ( ) .
9. Close the CCS debug session by clicking *Terminate Debug Session* (*Target* → *Terminate all*).

3.1.2.5.9.4 Running Code: Grid-tied Condition

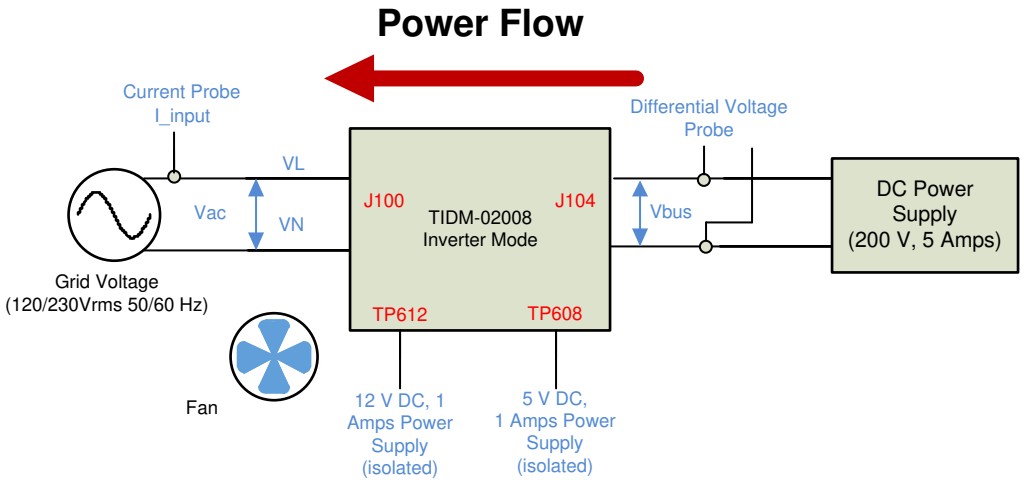


Figure 3-43. Lab 9 Grid Connected Condition

1. The project is programmed to wait for dc bus voltage and ac source voltage to exceed approximately 340 V and 75 Vrms to drive the in rush relay, and clear the trip.
2. Run the project by clicking  .
3. Now apply an input voltage of approximately 340 V, the board comes out of the undervoltage condition. To run the solution, ac voltage has to be higher than 75 V rms and inrush relay is driven.
4. Change pwmSwState from pwmSwState_defaultState to pwmSwState_normalOperation to enable pwm output. **The pwm output is turned off by default and the inverter does not work without selecting this option.**
5. Then slowly increase the TTPLPFC_ac_cur_ref to -0.3 and inverter output current is close to 5.3 A in RMS value. The voltage of inverter output is determined by the grid.
6. SFRA is integrated in the software of this build to verify the designed compensator provides enough gain and phase margin by measuring on hardware. To run the SFRA keep the project running, and from the cfg page, click the *SFRA* icon. SFRA GUI appears.

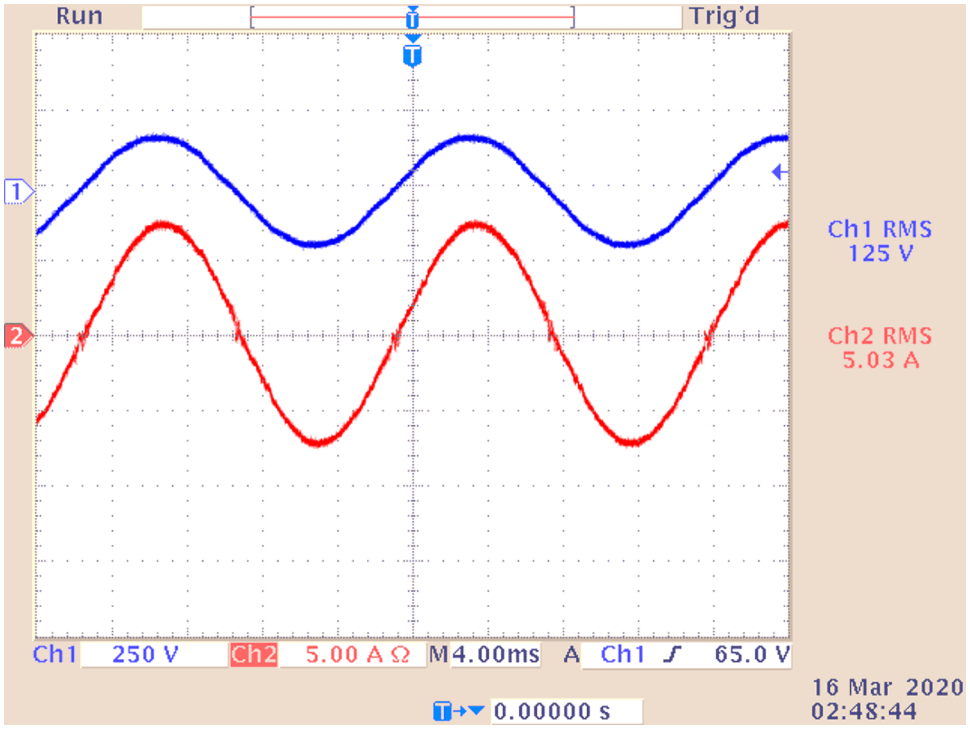


Figure 3-44. Voltage and current waveform (Lab 9 Grid-Connected Condition)

7. Select the options for the device on the SFRA GUI. For example, for F28377D select floating point. Click on *Setup Connection*. On the pop-up window uncheck the boot on connect option, and select an appropriate COM port. Click *OK*. Return to the SFRA GUI, and click *Connect*.

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. Activity can be monitored by seeing the progress bar on the SFRA GUI and also checking the flashing of blue LED on the back on the control card that indicates UART activity. Once complete, a graph with the open loop plot appears, Figure 3-24. This is similar to the plot seen under DC conditions; however, some additional noise is visible due to AC harmonic frequencies close to the measured frequencies. The BW, PM, and GM numbers are very similar to the DC case. Note the graph shown in Figure 3-24 was taken with direct grid AC input. When using AC source interaction of the AC source output, impedance can be observed, which can affect the control margins.

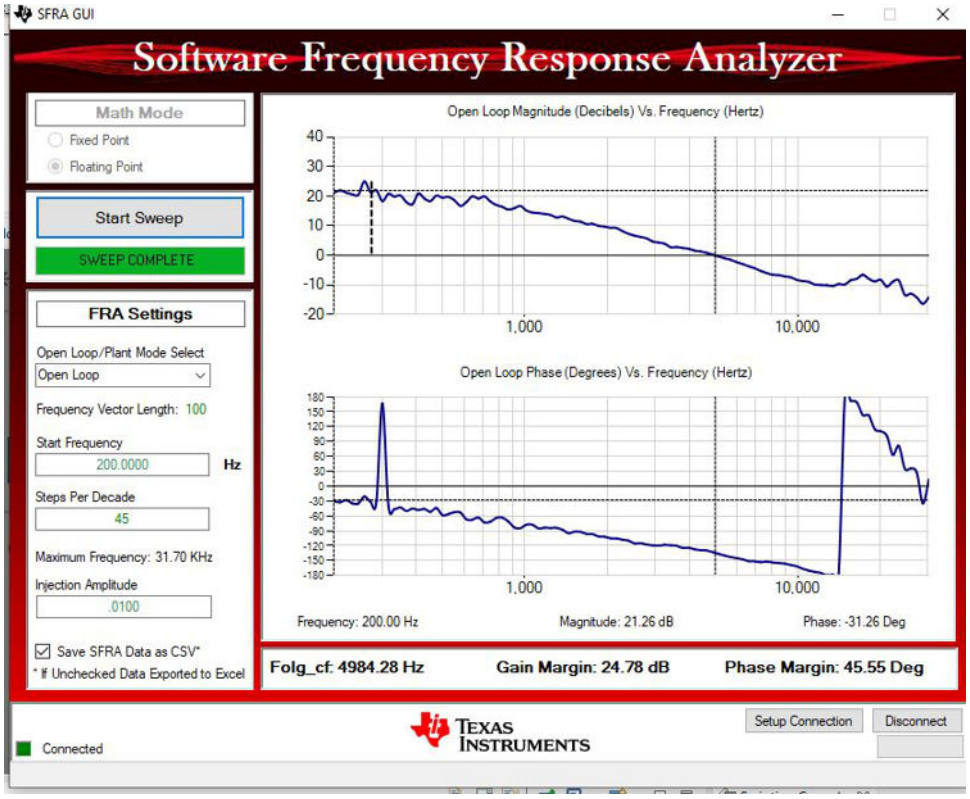


Figure 3-45. SFRA Run, Closed Current Loop, Open Loop Gain (Inverter mode)

9. To bring the system to a safe stop, switch off the output from the AC grid first and switch off the input DC power supply subsequently. This will eliminate the risk of the undesired reverse power flow in the inverter mode. Turn off the grid voltage and observe the TTPLPFC_ac_volRms_sensed_Volts comes down to zero as well. After the grid voltage is fully switched off, the the input DC voltage has to be down to zero, observe the TTPLPFC_vBus_sensed_Volts becomes zero
10. Fully halting the MCU when in real-time mode is a two-step process. First halt the processor by using the *Halt* button on the toolbar ( ) or by using *Target* → *Halt*. Then take the MCU out of real-time mode by clicking on . Finally, reset the MCU ( ) .
11. Close the CCS debug session by clicking on *Terminate Debug Session* (*Target* → *Terminate all*).

3.1.2.6 Running Code on CLA

This solution is supported with an option to run the code on the CLA. This option is selected using a drop-down box under project option on the powerSUITE main.cfg page. Running on CLA can be selected for any lab options.

Note

SFRA library does not support CLA, hence the SFRA cannot be run when using CLA.

DLOG is also not used when using CLA, hence the datalogging graphs will not work when using CLA.

Once the option is changed, the CFG file must be saved and the project re-compiled. Once recompiled, follow the steps as outlined in the specific lab documentation.

Depending on the device, for example for F28004x CLA supports CLA tasks and a background task, thus both 100kHz ISR and 10kHz ISR can be offloaded to the CLA. By default if the selection from the powerSUITE page is made the faster ISR is moved to the CLA task and the slower ISR is moved to the background task by default. If the user does not want to run the 10kHz ISR on the CLA, the option to do so is available under the <solution>_user_settings.h file.

```
#if TTPLPFC_CONTROL_RUNNING_ON == CLA_CORE
#define TTPLPFC_INSTRUMENTATION_ISR_RUNNING_ON CLA_CORE
#else
#define TTPLPFC_INSTRUMENTATION_ISR_RUNNING_ON C28x_CORE
#endif
```

3.1.2.7 Advanced Options

In this section some advanced setting that improve the power factor are discussed one by one and their relative impact quantified with test results

3.1.2.7.1 Input Cap Compensation for PF Improvement Under Light Load

Input cap causes PF degradation if the current reference is maintained perfectly in sync with the voltage as shown in Equation 8 and in Figure 3-46(a).

$$i_{ref\_dpllvc} = i^{*}_{ref} \sin(\omega t) - i_{input\_cap\_comp} \cos(\omega t) \tag{8}$$

The current reference can be adjusted with vectors to offset the PF degradation, Equation 9, Figure 3-46(b) . As the angle from the phase locked loop is used to compute the vector this technique is called the digital phase locked loop vector cancellation (DPLLVC) technique. This improves PF at light load and high line significantly.

$$i_{ref\_DPLLVC} = i^{*}_{ref} \sin(\omega t) - i_{input\_cap\_comp} \cos(\omega t) \tag{9}$$

The amount of correction applied depends on the input capacitor value, for example on this design the input cap is 2.2 µF. Which means at high line a capacitive current equal to $\frac{220}{\frac{1}{2} \times pi \times 60 \times 2.2\,\mu F} = 0.1823\ A$ is drawn. Under light loads, this is significant amount of current and causes power factor loss. The current sensor gain on this design is approximately 24 A, so this translates to an adjustment of approximately 0.01 pu for the high-line condition.

Furthermore, there is a tracking error under low power conditions. This tracking error can be offset by an adjustment to the current command shown in Figure 3-46©). The amount of this tracking error is adjusted empirically for the best performance from the system. Thus the total current reference is given by Equation 10.

$$i_{ref\_DPLLVC\_TC} = i^{*}_{ref} \sin(\omega t) - i_{input\_cap\_comp} \cos(\omega t) - i_{tracking\_error}\Big|_{power\_dependent} \cos(\omega t) \quad (10)$$

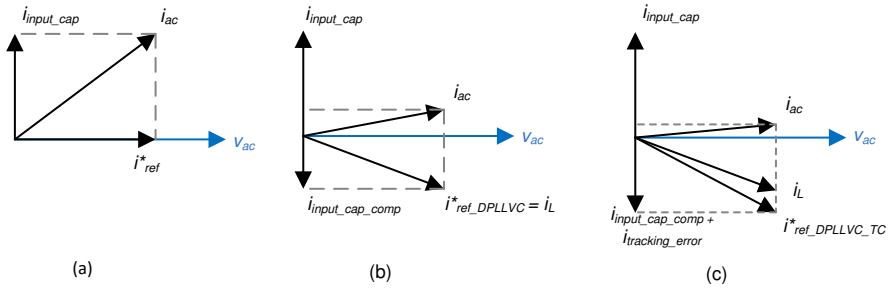


Figure 3-46. Power Factor (a) No Adjustment (b) DPLLVC ©) DPLLVC Plus Tracking Error Compensation

The result for PF improvement are graphed in Figure 3-47.

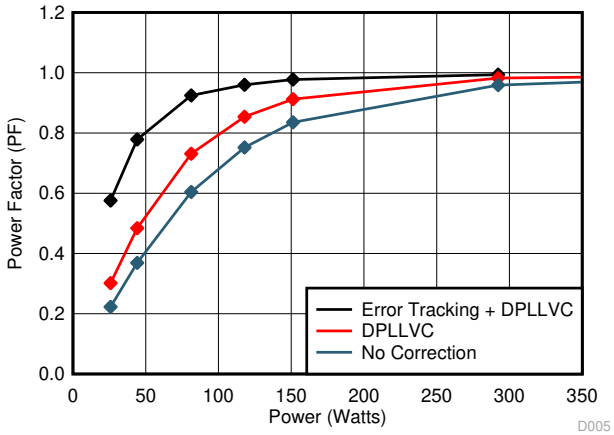


Figure 3-47. PF Graph vs Power at 220 Vrms Highlighting Improvement With Input Cap Current Compensation

3.1.2.7.2

This feature can be turned off or on by writing to the TTPLPFC_INPUT_CAP_COMPENSATION #define variable in the <solution>_user_settings.h file. This setting can be modified by the user. The value of the adjustment is controlled using the #defines TTPLPFC_HIGH_LINE_INPUT_CAP_COMP_ADJUST & TTPLPFC_LOW_LINE_INPUT_CAP_COMP_ADJUST. These are in per-unit format and how this value is determined was explained earlier in this section.

```
#define TTPLPFC_INPUT_CAP_COMPENSATION 1
#define TTPLPFC_HIGH_LINE_INPUT_CAP_COMP_ADJUST -0.02
#define TTPLPFC_LOW_LINE_INPUT_CAP_COMP_ADJUST  -0.01
```

3.1.2.7.3 Adaptive Dead Time for Efficiency Improvements

In continuous conduction mode, the dead-time control for synchronous rectification is critical in terms of short-circuit protection and efficiency. With the optimal dead time, the risk of shoot through can be eliminated and it also prevents an excessive conduction loss from body diode conduction of Sync FET. Therefore, the goal of optimal dead time is not to turn on the Active FET and Sync FET simultaneously while minimize a redundant third quadrant conduction of Sync FET.

This optimal dead time can be calculated from the measured current and the device output capacitance, this is given by Equation 11.

$$t_{deadtime\_optimal} = \frac{2C_{oss}V_{out}}{i_{L\_peak}} \quad (11)$$

Figure 3-48 shows the block diagram for implementation.

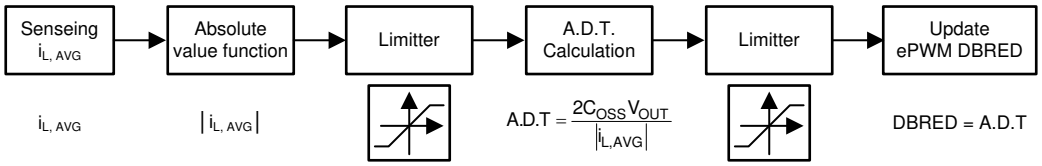


Figure 3-48. Adaptive Dead-Time Implementation

The option enables power saving, which is shown for the high line case in Figure 3-49 compared to a fixed dead time from which it can be inferred that avoiding the shoot through at low power levels results in significant power savings. However, once the shoot through is avoided the power savings drop first and then progressively increase as power increases and the diode conduction time is reduced by implementing adaptive dead-time adjustment.

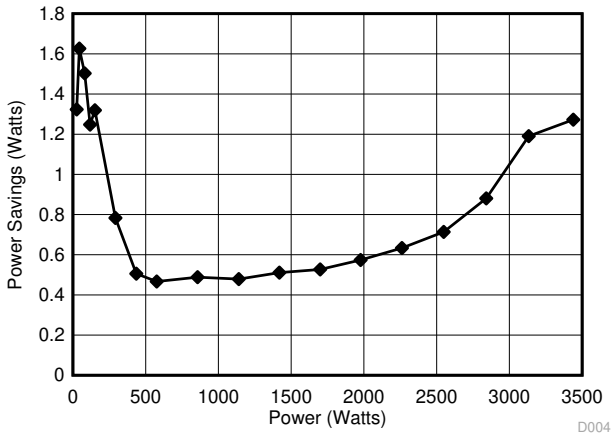


Figure 3-49. Power Savings With Adaptive Dead-Time at High Line 230 Vrms

To enable adaptive dead time, select the drop down box under *Project Options* on the *powerSUITE* page of the solution. For FED the fixed value that is specified on the main.syscfg is used. When adaptive dead time is enabled the RED is modulated and the minimum and maximum bounds are specified in the *<solution>_user_settings.h*. The following are the #define that can be adjusted:

```
#define TTPLPFC_HF_FET_COSS  (float32_t) 0.000000000145
#define TTPLPFC_PFC_DEADBAND_RED_MIN_US (float32_t) 0.020
#define TTPLPFC_PFC_DEADBAND_RED_MAX_US (float32_t) 0.200
```

Following these adjustments the project must be saved, re-compiled, and loaded on the controller when this option is changed. Hardware setup and software instructions as outlined in Section 3.1.2.5.4 can be followed to see the behavior of the board and measure efficiency.

3.1.2.7.4 Phase Shedding for Efficiency Improvements

Phase shedding can be an effective technique to improve efficiency in interleaved application by optimizing for the conduction and the switching losses. In this design there are three phases, so three different configurations are possible as shown in Figure 3-50.

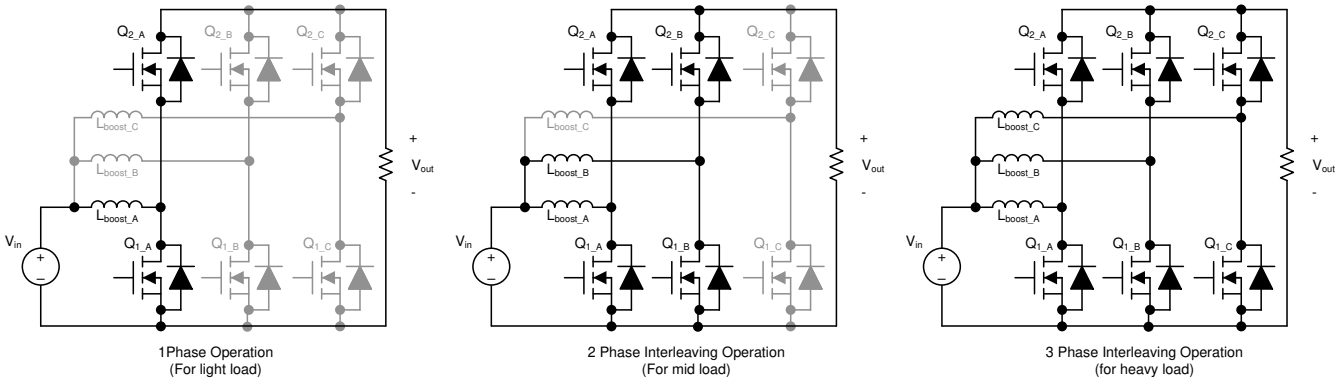


Figure 3-50. Phase Shedding Options on TTPL PFC

In each of these modes the phase shift between each of them must be adjusted. When in two-phase mode, a 180° phase shift is desired between the PWMs, and when in three-phase mode, a 120° of phase shift is desired.

The decision to do phase shedding can be made on different parameters, such as the RMS current, power, the peak inductor current, and so on. When using RMS current the change of phases can be significantly delayed. Figure 3-51 shows phase shedding when the decision is based on RMS current. Code takes multiple AC cycles before the phases are added .

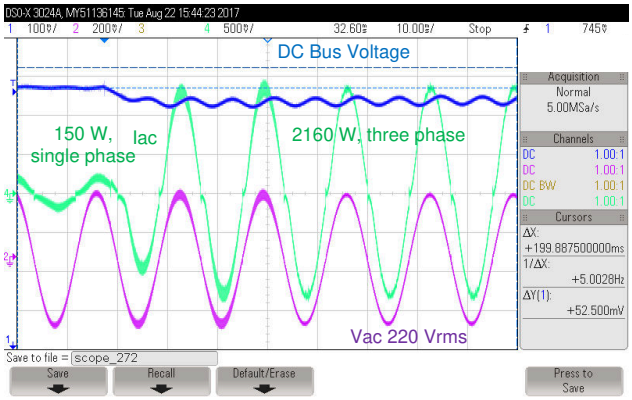


Figure 3-51. Waveform When Decision to Add Phases is Based on RMS Calculation

This delay may not be acceptable for many application. Thus, the voltage controller output is chosen as the decision point to drop or add phases. A state machine is constructed as shown in Figure 3-52, with some hysteresis built around the phase shedding points.

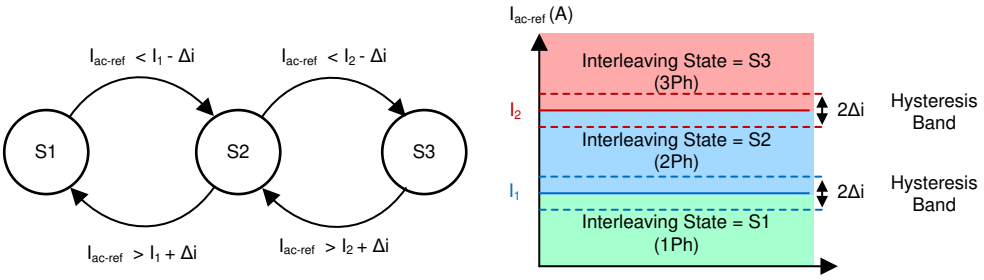


Figure 3-52. State Machine for Phase Shedding Control

Bringing a phase in and out can cause in-advertent pulses to be generated. Hence the implementation to drop and add phase is done through the GPIO and PWM peripheral switch using the GPIO pin Mux registers. All PWM capable pins are configured and GPIO outputs and driven low. Now based on how many phases must be applied the GPIO pin mux is changed accordingly. It is safe to enable and disable the phases using the GPIO pin mux switch at any point in the AC cycle as the registers in the PWM are shadowed. Figure 3-53 shows details of the implementation of phase shedding on the C2000 MCU.

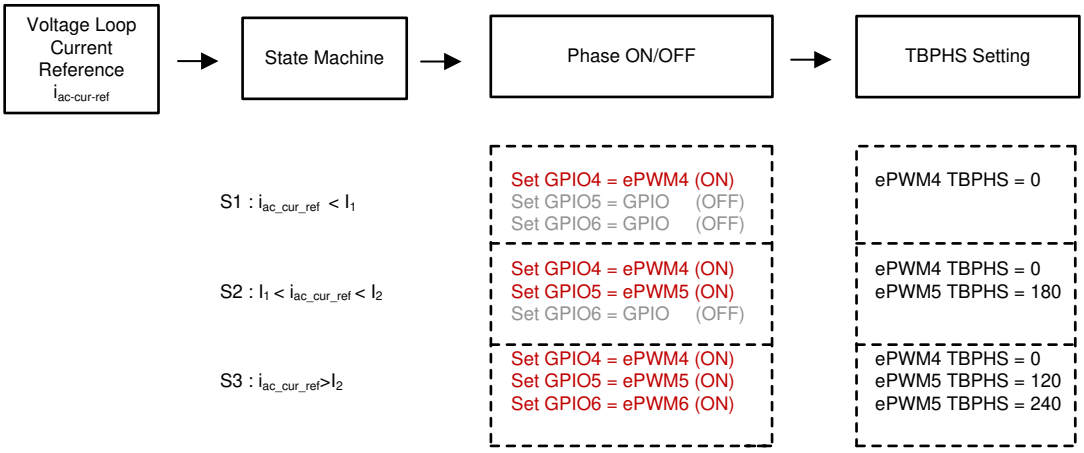


Figure 3-53. Implementation of Phase Shedding on TTPL PFC Using C2000™ MCU

The phase shedding can be set in the powerSuite page by selecting enable under the Phase shedding option. The points at which phases are brought in and out are set by changing the *PHASE_SHEDDING_1PH_2PH_TRANSITION_CURRENT* and *PHASE_SHEDDING_2PH_3PH_TRANSITION_CURRENT* define, which correspond to I1 and I2 as shown in Figure 3-53. Recompile the code, load the code, and repeat the steps as outlined in Section 3.1.2.5.4 to test this feature. With this feature implemented, under transients the phases are dropped and added quickly.

Figure 3-54 and Figure 3-55 show the transient at 110 Vrms of 1.3 KW to 150 W and vice versa. The phases are added back quickly and dropped quickly under transients as the decision is based on the voltage loop generated current reference.

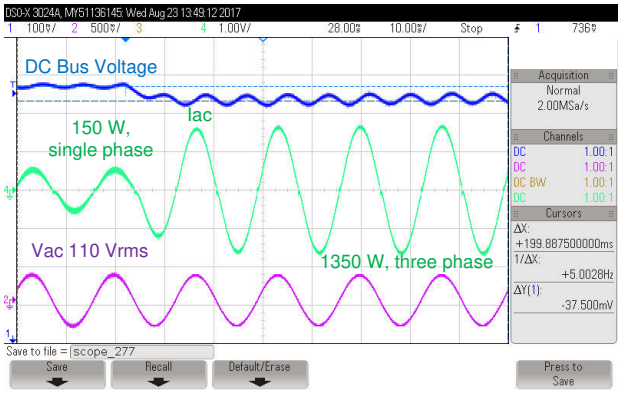


Figure 3-54. Phase Added Quickly Under Transient at 120 Vrms, 60 Hz

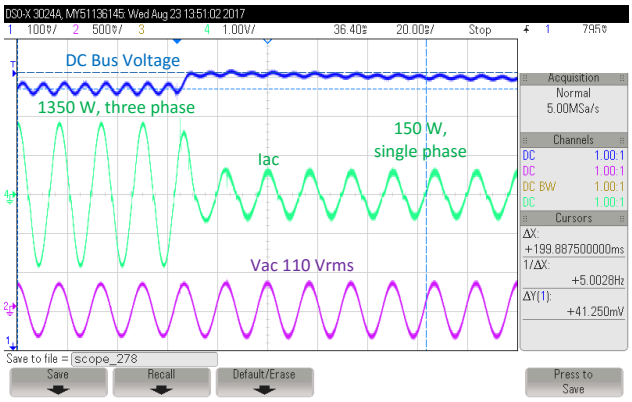


Figure 3-55. Phase Shed Quickly Under Transient at 120 Vrms, 60 Hz

Similarly at high line, a transient greater than 2 KW is applied, and the phase goes from single to three phase almost instantly. Figure 3-56 and Figure 3-56 show the test waveform at high line under transient of 2.16 KW to 150 W and vice versa.

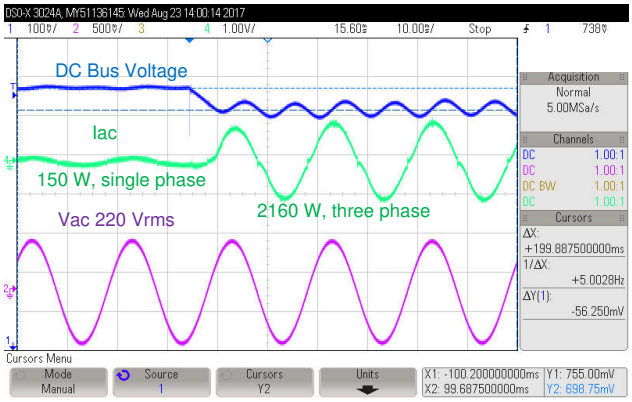


Figure 3-56. Phase Added Quickly Under Transient at 230 Vrms, 50 Hz

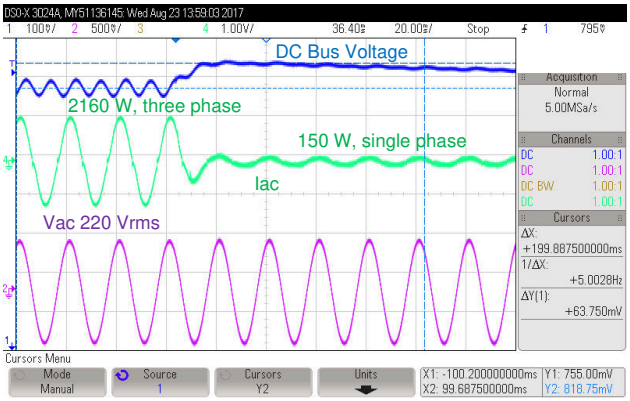


Figure 3-57. Phase Shed Quickly Under Transient at 230 Vrms, 50 Hz

Figure 3-58 shows the efficiency improvement at 230 Vrms with phase shedding

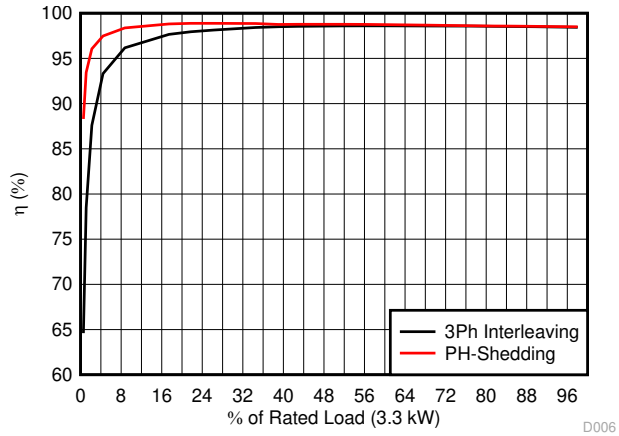


Figure 3-58. Efficiency Comparison at 230 Vrms With Phase Shedding and Without

3.1.2.7.5 Non-Linear Voltage Loop for Transient Reduction

The PFC stage control is composed of an inner current loop, which tries to follow the input voltage and an outer voltage loop that tries to maintain a constant DC bus voltage at the output. The voltage loop is thus in conflict with the current loop and hence must be designed to be very low bandwidth (approximately 10 Hz) in order to achieve good power factor. The slow voltage loop results in significant overshoot and undershoot under transients (see Figure 3-59).

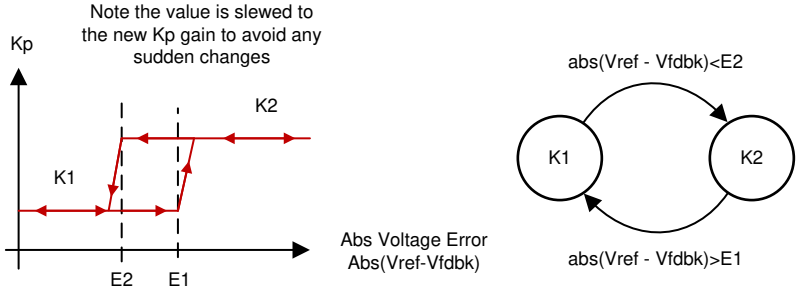


Figure 3-59. Non-Linear Voltage Loop with Hysteresis

To improve voltage overshoot and undershoot, while maintaining good power factor a non-linear voltage control loop is implemented as shown in Figure 3-60. A hysteresis band is added in the non-linear voltage loop to avoid oscillation between high-gain and low-gain mode. Furthermore the gain change is slewed to avoid any sudden changes. Figure 3-61 shows the result with non-linear voltage loop.

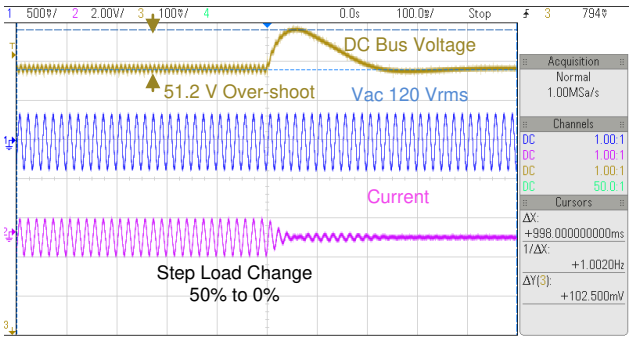


Figure 3-60. Voltage Transient Without Non-Linear Voltage Loop, Vin 120 Vrms, 880 W to 0 W Transient, Overshoot 51.2 V

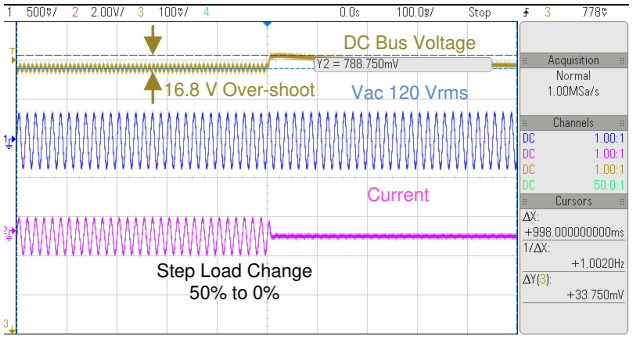


Figure 3-61. Voltage Transient With Non-Linear Voltage Loop, Vin 120 Vrms, 880 W to 0 W Transient, Overshoot 16.8 V

To enable non-linear voltage loop, select the drop down box under *Project Options* on the *powerSUITE* page of the solution. Default value of five times the gain is applied for the proportional term under transient condition. This value can be adjusted under the *<solution>_user_settings.h* file by modifying the *NON_LINEAR_V_LOOP_KP_MULTIPLIER* define. The project must be saved, re-compiled, and loaded on the controller when this option is changed. Hardware setup and software instructions for the lab 4 can be followed to see the behavior of the board under transients.

3.1.2.7.6 Software Phase Locked Loop Methods: SOGI - FLL

To meet industry standards, the design must be tested under frequency transients. This poses a problem when using PLL angle for the drive of the PWM signal if the PLL cannot adapt to the frequency change. A frequency locked loop scheme, as proposed in *Grid Synchronization of Power Converters Using Multiple Second Order Generalized Integrators* [1], can be used to make the software phase locked loop frequency adaptive.

The module follows the same basic structure as the SOGI PLL module, with implementation discussed in *Software PLL Design Using C2000 MCUs Single Phase Grid Connected Inverter* [2]. A frequency adaptive feature through a frequency locked loop is added as shown in Figure 3-62.

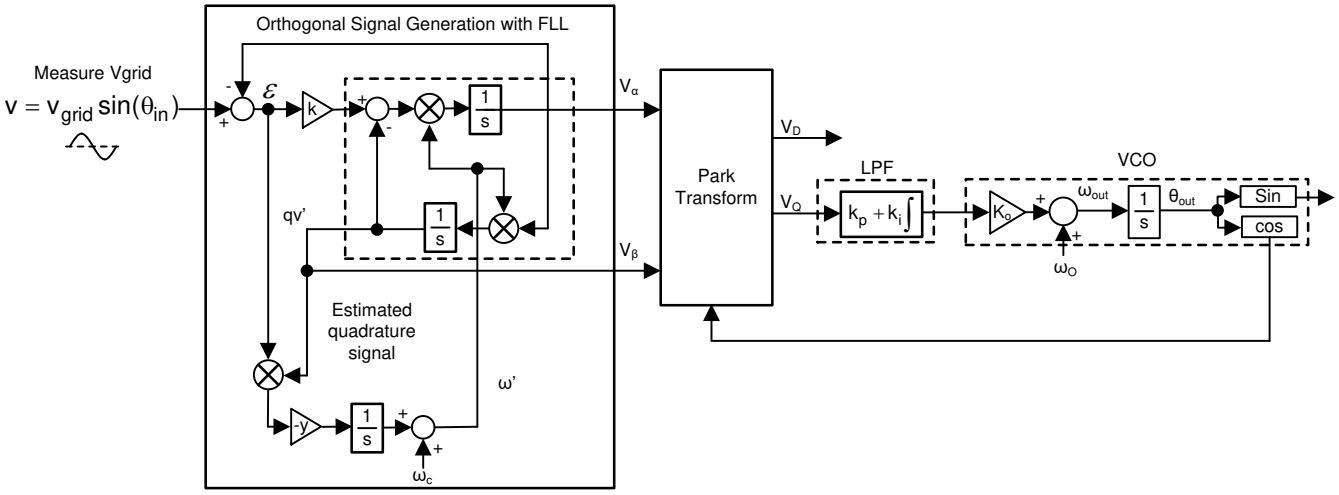


Figure 3-62. Software Phase Locked Loop Structure Based on Second Order Generalized Integrator and Frequency Locked Loop

To select a PLL method go to the *<solution>_user_settings.h*. The following are the #define that can be adjusted:

```
#define TTPLPFC_SPLL_METHOD_SELECT TTPLPFC_SPLL_1PH_SOGI_FLL_SEL
```

Following this adjustment the project must be saved, re-compiled, and loaded on the controller. Hardware setup and software instructions for the Lab 4 can be followed to see the behavior of the board with the new PLL scheme

3.2 Testing and Results

3.2.1 Test Results at Input 120 Vrms, 60 Hz, Output 380-V DC

3.2.1.1 Startup

The startup sequence of the power stage is shown in Figure 3-63 with input single phase of 120 Vrms VL-N, an output bus regulated at 380 V, and a 1.6-KW load and no load.

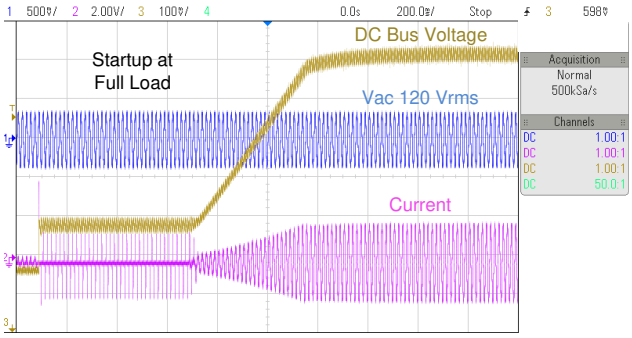


Figure 3-63. Startup of PFC Operation at 120-Vac IN, 380-V DC OUT and 1.6-KW Load

Figure 3-64 shows the startup under no load.

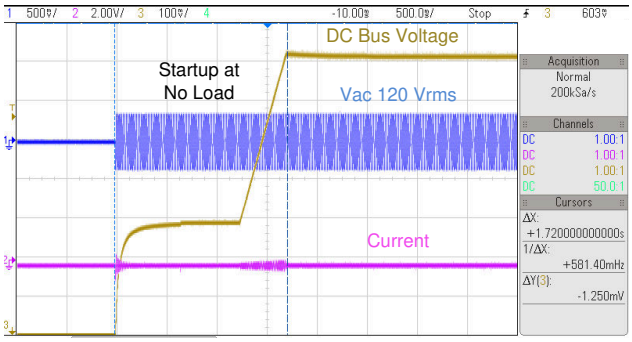


Figure 3-64. Startup of PFC at 120-Vac IN, 380-V DC Output, and 0% Load

3.2.1.2 Steady State Condition

Steady state current waveforms are shown in Figure 3-65, Figure 3-66, and Figure 3-67 at different load conditions. Phase shedding is disabled for these readings.

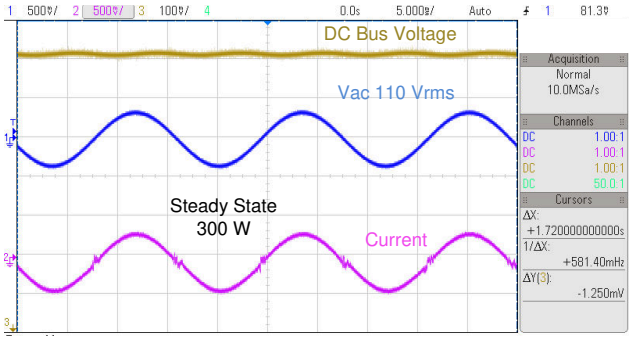


Figure 3-65. Steady State 120-Vac IN, 380-V DC OUT 300W, iTHD 5.5%

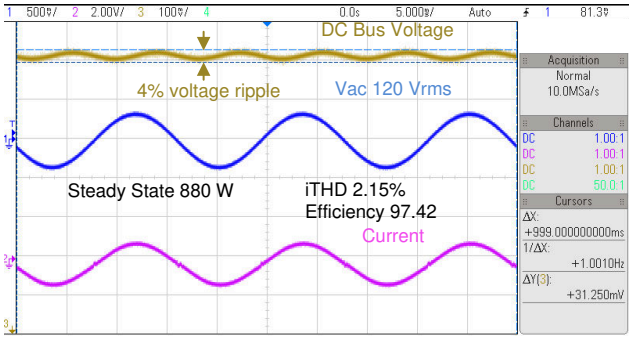


Figure 3-66. Steady State 120-Vac IN, 380-V DC OUT 880W, iTHD 2.15%

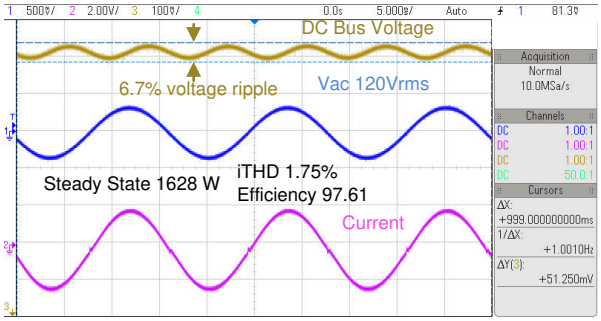


Figure 3-67. Steady State 120-Vac IN, 380-V DC OUT, 1.674-KW, iTHD 1.75%

Table 3-3 lists the detailed test results of this design under varying load conditions with 120-Vac input and 380-V DC output. For this data, phase shedding is disabled, adaptive dead time is enabled, 100 ns is chosen as the fixed dead time for the hard-switched edge, and the soft switching edge dead time varies between 20 ns and 200 ns.

Table 3-3. Detailed Test Results with 120-Vac IN, 380-V DC OUT, and Different Power Levels

Vin (V RMS)	Vout (V)	Pin (W)	Iout (A)	Pout (W)	EFFICIENCY %	iTHD%	PF	% RATED LOAD	THETA OFFSET	GI KP
120.05	382.02	154.27	0.375	143.47	92.98	10.54	0.9927	9.0	-0.014	0.35
119.86	382.01	301.30	0.750	286.36	95.14	5.50	0.9974	17.9	-0.01	0.35
119.49	382.01	444.40	1.120	427.76	96.30	4.16	0.9987	26.7	-0.01	0.35
119.42	382.03	579.10	1.469	561.40	96.94	2.89	0.9950	35.1	-0.01	0.35
119.16	382.02	721.30	1.837	701.80	97.30	2.42	0.9995	43.9	-0.01	0.35
119.02	382.05	863.00	2.202	841.50	97.52	2.15	0.9995	52.6	0	0.35
118.78	381.96	1007.20	2.573	983.30	97.64	1.92	0.9995	61.5	0	0.35
118.63	382.08	1152.00	2.944	1125.30	97.69	1.82	0.9995	70.3	0	0.35
118.40	382.08	1298.40	3.319	1268.20	97.70	1.72	0.9994	79.3	0	0.35
118.25	382.08	1442.00	3.685	1408.30	97.69	1.87	0.9991	88.0	0	0.3
118.03	382.08	1593.80	4.071	1555.50	97.65	1.80	0.9991	97.2	0	0.3
117.98	382.05	1716.40	4.449	1674.80	97.61	1.75	0.9991	104.7	0	0.3

3.2.1.3 Transient Test With Step Load Change

3.2.1.3.1 0% to 50% Load Step Change

Figure 3-68 shows the transient response when input is 120 Vrms and a load step of 50% is applied to the power stage

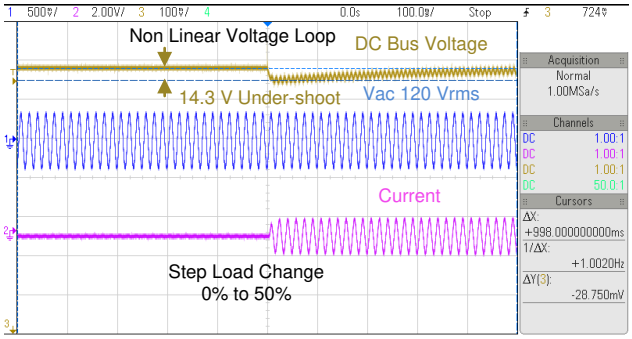


Figure 3-68. Transient Response, 120 Vrms, 60 Hz, 0% to 50% Load Step

3.2.1.3.2 50% to 100% Load Step Change

Figure 3-69 shows the transient response when input is 120 Vrms and load is stepped up from 50% to 100%

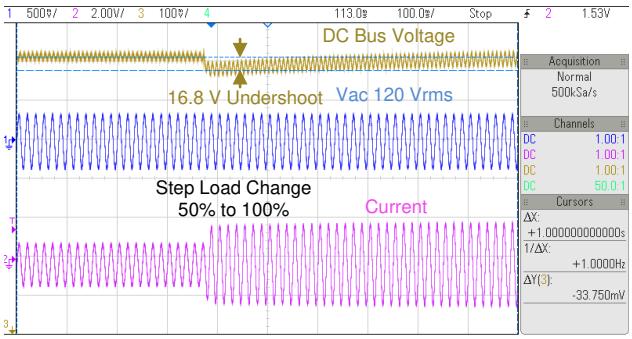


Figure 3-69. Transient Response, 120 Vrms, 60 Hz, 50% to 100% Load Step

3.2.1.3.3 100% to 50% Load Step Change

Figure 3-70 shows the transient response when the input is 120 Vrms and load is stepped down from 100% to 50%.

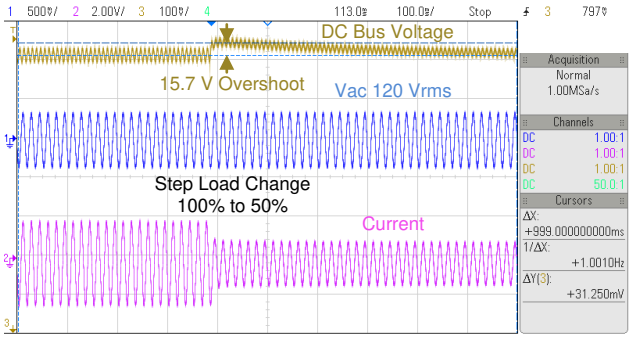


Figure 3-70. Transient Response, 120 Vrms, 60 Hz, 100% to 50% Load Step

3.2.1.3.4 50% to 100% Load Step Change

Figure 3-71 shows the transient response when input is 120Vrms and load is stepped down from 50% to 0%.

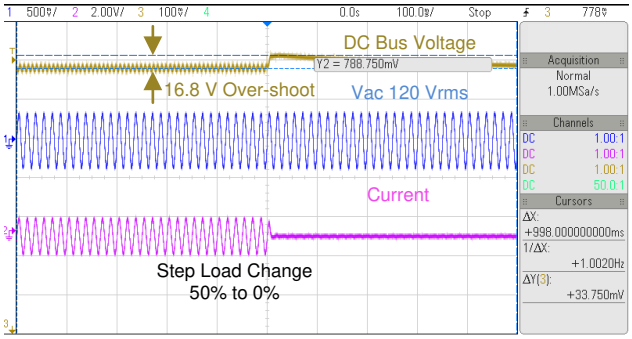


Figure 3-71. Transient Response, 120 Vrms, 60 Hz, 50% to 0% Load Step

3.2.2 Test Results at Input 230 Vrms, 50 Hz, Output 380 V DC

3.2.2.1 Startup

Figure 3-72 shows the startup sequence of the power stage with input single phase 230-Vrms VL-N and output bus regulated at 380V and a 880-W load.

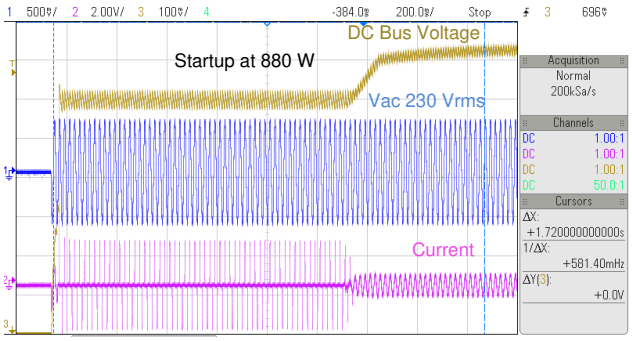


Figure 3-72. Startup of PFC Operation at 230-Vac IN, 380-V DC OUT at 880-W Load

Figure 3-73 shows the startup of PFC at no load at 230 Vrms.

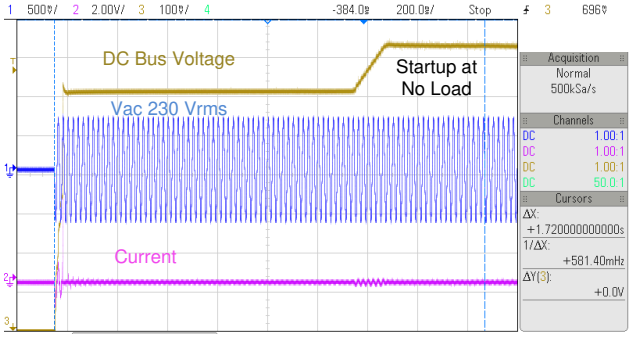


Figure 3-73. Startup of PFC Operation at 230-Vac IN, 380-V DC OUT at no Load

3.2.2.2 Steady State Condition

Figure 3-74, Figure 3-75, and Figure 3-76 show the steady state current waveform at different load conditions. Phase shedding is disabled for these readings.

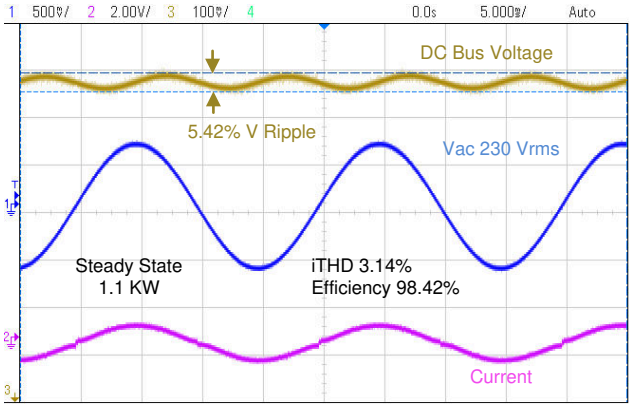


Figure 3-74. Steady State 230-Vac IN, 380-V DC OUT, 1.1KW, iTHD 3.14%

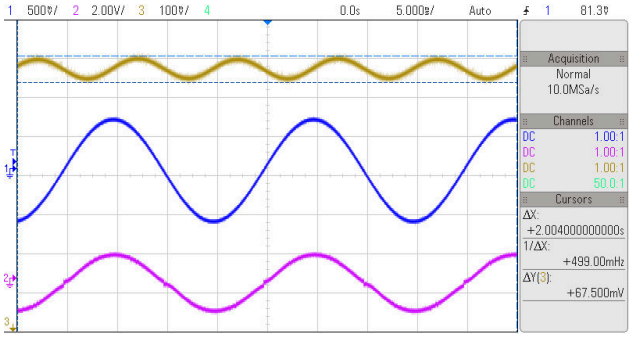


Figure 3-75. Steady State 230-Vac IN, 380-V DC OUT, 2.2KW, iTHD 2.62%

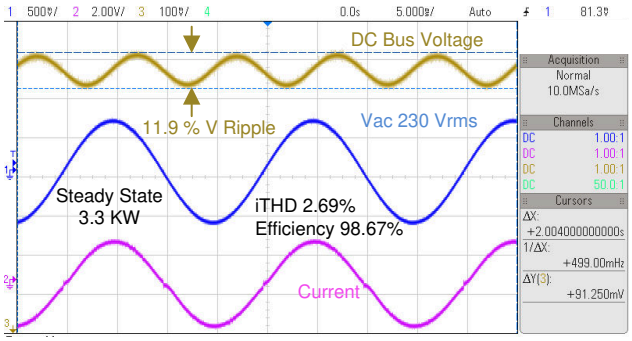


Figure 3-76. Steady State 230-Vac IN, 380-V DC OUT, 3.3KW, iTHD 2.69%

Table 3-4 lists the detailed test results of this design under varying load conditions with 230-Vac input and 380-V DC output. For the following data, phase shedding is disabled, adaptive dead time is enabled, 100 ns is chosen as the fixed dead time for the hard switched edge, and the soft-switching edge dead time varies between 20 ns to 200 ns.

Table 3-4. Detailed Test Results With 230-Vac IN, 380-V DC OUT and Different Power Levels

Vin (V RMS)	Vout (V)	Pin (W)	Iout (A)	Pout (W)	EFFICIENCY %	iTHD%	PF	% RATED LOAD	THETA OFFSET	Gi Kp
230.68	381.98	151.28	0.372	142.16	94.03	18.20	0.9775	4.4	-0.025	0.35
230.43	382.00	292.24	0.736	281.41	96.29	9.15	0.9936	8.8	-0.02	0.35
230.25	382.03	435.90	1.109	423.62	97.18	6.12	0.9938	13.2	-0.01	0.35
230.06	382.06	576.40	1.473	562.86	97.66	4.85	0.9972	17.6	-0.01	0.35
229.80	382.05	856.80	2.201	841.00	98.15	4.16	0.9974	26.3	0	0.35
229.70	382.11	1140.10	2.935	1121.90	98.42	3.14	0.9989	35.1	0	0.35
229.52	382.08	1418.80	3.659	1398.40	98.57	2.42	0.9993	43.7	0	0.3
229.28	382.08	1699.20	4.386	1676.40	98.66	2.74	0.9995	52.4	0	0.3
229.06	382.09	1977.70	5.106	1951.90	98.71	2.56	0.9996	61.0	0	0.3
229.09	382.11	2261.50	5.840	2232.40	98.73	2.62	0.9995	69.8	0	0.25
228.91	382.11	2548.30	6.580	2515.60	98.73	2.50	0.9994	78.6	0	0.25
228.86	382.14	2840.60	7.332	2803.20	98.71	2.89	0.9990	87.6	0	0.2
228.51	382.12	3132.80	8.083	3091.10	98.69	2.80	0.9989	96.6	0	0.2
228.22	382.03	3439.10	8.873	3392.30	98.65	2.69	0.9988	106.0	0	0.2

3.2.2.3 Transient Test With Step Load Change

Following sections show the transient test results with step load change.

3.2.2.3.1 33% to 100% Load Step Change

Figure 3-77 shows the transient response when input is 230 Vrms and load is stepped down from 100% to 33%.

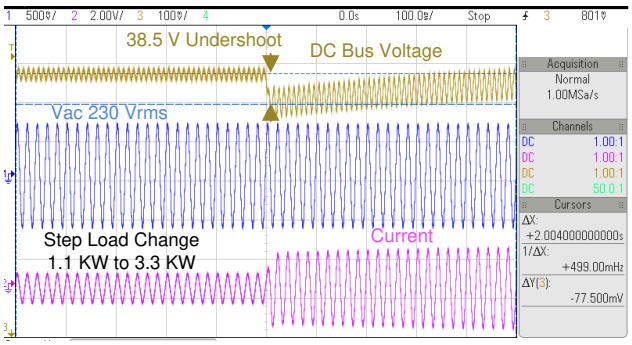


Figure 3-77. Transient Response, 230 Vrms 50 Hz, 33% to 100% Load Step

3.2.2.3.2 100% to 33% Load Step Change

Figure 3-78 shows the transient response when input is 230 Vrms and load is stepped down from 100% to 33%.

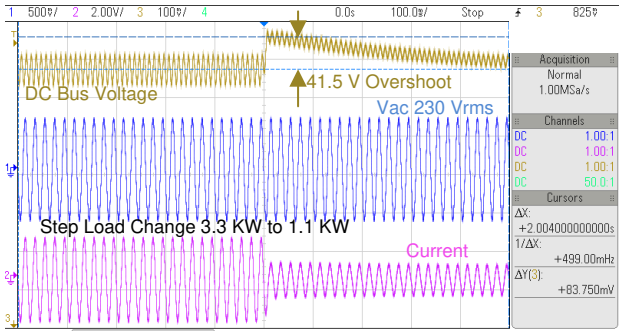


Figure 3-78. Transient Response, 230 Vrms 50 Hz, 100% to 33% Load Step

3.2.3 Test Results Graphs

Figure 3-79 shows the efficiency data plotted under different test conditions for this design.

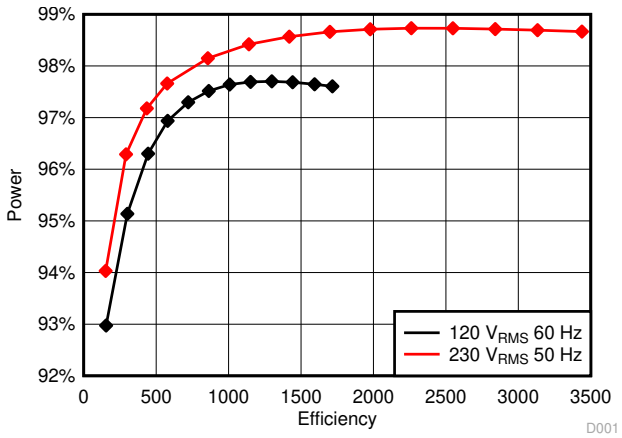


Figure 3-79. Efficiency at 230-Vrms Input and 120-Vrms Input

Figure 3-80 shows the THD data plotted under these test conditions.

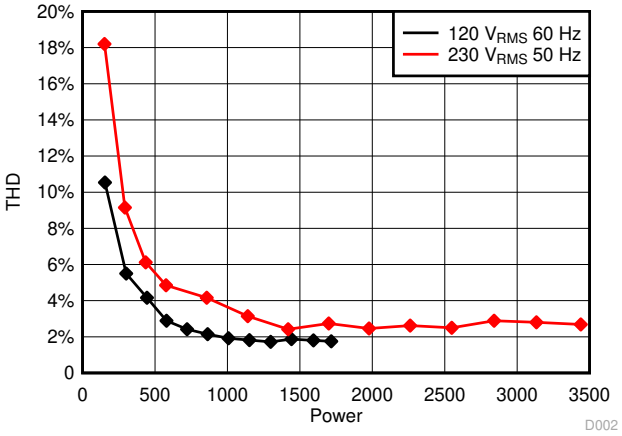


Figure 3-80. THD at 230-Vrms Input and 120-Vrms Input

Figure 3-81 shows the PF data plotted under these test conditions.

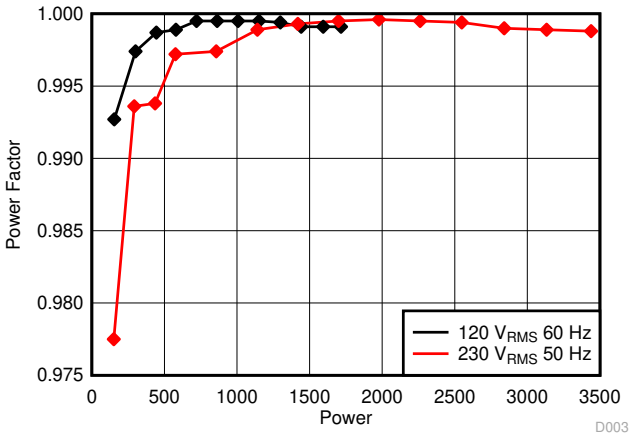


Figure 3-81. PF at 230 Vrms and 120Vrms With Varying Load

4 Design Files

4.1 Schematics

To download the schematics, see the design files at TIDM-1007.

4.2 Bill of Materials

To download the bill of materials (BOM), see the design files at TIDM-1007.

4.3 PCB Layout Recommendations

4.3.1 Layout Prints

To download the layer plots, see the design files at TIDM-1007.

4.4 Altium Project

To download the Altium project files, see the design files at TIDM-1007.

4.5 Gerber Files

To download the Gerber files, see the design files at TIDM-1007.

4.6 Assembly Drawings

To download the assembly drawings, see the design files at TIDM-1007.

5 Software Files

To download the software files, see the design files at TIDM-1007.

6 Related Documentation

- Texas Instruments, *How to reduced current spike at AC zero crossing for totem-pole PFC Technical Brief*
- Z. Ye, A. Aguilar, Y. Bolurian and B. Daugherty, *GaN FET-Based High CCM Totem-Pole Bridgeless PFC*, Texas Instruments Power Supply Design Seminar, 2014-15.
- L. Xue, Z. Shen, D. Boroyevich and P. Mattavelli, *GaN-based High Frequency Totem-Pole Bridgeless PFC Design with Digital Implementation*, IEEE 2015 Applied Power Electronics Conference, 2015, pp. 759-766.
- H.-S. Youn, J.-B. Lee, J.-I. Black and G.-W. Moon, *A Digital Phase Leading Filter Current Compensation (PLFCC) Technique for CCM Boost PFC Converter to Improve PF in High Ligne Voltage and Light Load Condition*, IEEE Transactions on Power Eelectronics , vol. 31, no. 9, pp. 6596-6606, 2016.
- D. M. V. d. Sype, K. D. Gusseme, A. P. M. V. d. Bossche and J. A. Melkebeek, *Duty-Ratio Feeforward for Digitally Controlled Boost PFC Converters*, IEEE Transactions on Industrial Electronics, vol. 52, no. 1, pp. 108-115, February 2005.
- "Rodriguez, P.,Luna, A., Candela, I., Teodorescu, R., and Blaabjerg, F. *Grid Synchronization of Power Converters Using Multiple Second Order Generalized Integrators* , In Proceedings of IEEE industrial Electronics Conference (IECON'08), November 2008, pp 755-760"
- Texas Instruments, *Software PLL Design Using C2000 MCUs Single Phase Grid Connected Inverter Application Report*
- Texas Instruments, *TMS320F28004x Piccolo™ Microcontrollers Data Sheet*
- Texas Instruments, *LMG3410 600-V 12-A Single Channel GaN Power Stage Data Sheet*

6.1 Trademarks

7 About the Author

MANISH BHARDWAJ is a Systems Application Engineer with C2000 Microcontrollers System Solutions Group at Texas Instruments, where he is responsible for developing reference design solutions for digital power, motor control, and solar power applications. Before joining TI in 2009, Manish received his Masters of Science in Electrical and Computer Engineering from Georgia Institute of Technology, Atlanta in 2008 and his Bachelor of Engineering from Netaji Subhash Institute of Technology, University of Delhi, India in 2007.

Jongwan Kim is a Systems Application Engineer with C2000 Microcontrollers System Solutions Group at Texas Instruments, where he is responsible for developing reference design solutions for digital power applications. Before joining TI in 2019, Jongwan received his Ph.D. degree in Electrical Engineering from Virginia Tech, Blacksburg in 2019 and received his B.S. and M.S. degree in Electrical Engineering from Korea University, Seoul, Korea, in 2013 and 2015.

8 Revision History

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

Changes from Revision D (October 2020) to Revision E (April 2021)	Page
• Added AC Drop Test topic	12

Changes from Revision C (March 2020) to Revision D (October 2020)	Page
• Changed *incremental build* to *lab* throughout the document	3
• Changed *build level* to *lab* throughout the document	3
• Changed *Build Level 3* to *Lab 4* throughout the document	3
• Changed powerSUITE Page for CCM TTPL PFC Solution image	19
• Changed Project Explorer View of Solution Project image	23
• Changed Project Project Structure Overview image	23
• Changed *pfc1ph3ilttpl* to *ttplpfc* throughout document	23
• Changed Incremental Build to Lab	23
• Added CPU and CLU Utilization and Memory Allocation section	26
• Changed Lab 1 Expressions View image	31
• Changed Lab 1: Watch Expression Showing Measured Voltage and Currents image	32
• Changed Lab 2: Closed Current Loop Expressions View image	35
• Changed Watch Expression, Lab 2: After Closed Current Loop Operations Begin image	35
• Changed Watch Expression, Lab 2: After Closed Current Loop Operations Begins at Full Voltage image	35
• Changed Lab 3: Closed Current Loop Expressions View image	40
• Changed Watch Expression, Lab 3, AC After Closed Current Loop Operation Begins image	41
• Changed Lab 4: Expressions View image	45
• Changed Lab 4: Expressions View After AC Voltage is Applied image	46
• Changed Lab 5 Expressions View image	49
• Changed Lab 5 Watch Expression Showing Measured Voltage and Currents image	49
• Changed Lab 6 Expressions View image	50
• Changed Lab 6 Watch Expression Showing Measured Voltage and Currents image	51
• Changed Lab 7: Closed Current Loop Expressions View image	52
• Changed Watch Expression, Lap 8, AC After Closed Current Loop image	53
• Changed Lab 9: Closed Current Loop (Grid-connected) Expressions View image	54

Changes from Revision B (June 2018) to Revision C (March 2020)	Page
• Changed LMG34310 to LMG3410R070 throughout document	1
• Added ...is capable of bidirectional power flow (PFC and grid-tied inverter)	1
• Changed TIDM-01007 to TIDM-020008	1
• Added Bidirectional	1
• Added Energy Storage System (ESS)	1
• Added Inverter mode column to Table 1	3
• Added Equation 2 list items	8
• Added for PFC mode only to subsection title	10
• Changed Figure 15: Hardware Setup to Run Software (PFC and Inverter Mode)	17
• Changed main.syscfg support	19
• Changed version 7.4 to CCSV9.3	19
• Changed Figure 16: powerSUITE Page for CCM TTPL PFC Solution	19
• Changed Figure 17: Project Explorer View of Solution Project	23
• Added INCR_BUILD 4, 5, and 6 for Inverter operation	23
• Changed Figure 21: HW Setup for Build Level 1	28
• Changed Figure 31: HW Setup for AC Input	38
• Added instructions to run INCR_BUILD4 (DC)	48
• Changed Figure 41: HW Setup for Build Level 4	48
• Changed Figure 42: Build Level 4 dc Expressions View	49
• Changed Figure 43: Build Level 4 dc: Watch Expression Showing Measured Voltage and Currents	49
• Added instructions to run INCR_BUILD4 (AC)	50
• Changed Figure 44: Build Level 4 ac Expressions View	50
• Changed Figure 45: Build Level 4 ac: Watch Expression Showing Measured Voltage and Currents	51
• Added instructions to run INCR_BUILD5 (DC)	52
• Changed Figure 46: Build Level 5 dc: Closed Current Loop Expressions View	52
• Changed Figure 47: Watch Expression, Build Level 5, DC After Closed Current Loop	53
• Added instructions to run INCR_BUILD5 (AC)	53
• Changed Figure 48: Build Level 5: Closed Current Loop Expressions View	54

Changes from Revision A (March 2018) to Revision B (May 2018) **Page**

Changes from Revision * (November 2017) to Revision A (March 2018) **Page**

IMPORTANT NOTICE AND DISCLAIMER

TI PROVIDES TECHNICAL AND RELIABILITY DATA (INCLUDING DATA SHEETS), 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 will 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 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.

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

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