Design Guide: TIDM-HV-1PH-DCAC
Voltage Source Inverter Reference Design

Description

This reference design implements single-phase inverter (DC/AC) control using a C2000™ microcontroller (MCU). The design supports two modes of operation for the inverter: a voltage source mode using an output LC filter, and a grid connected mode with an output LCL filter. High-efficiency, low THD, and intuitive software make this design attractive for engineers working on an inverter design for UPS and alternative energy applications such as PV inverters, grid storage, and micro grids. The hardware and software available with this reference design accelerate time to market.

Resources

TIDM-HV-1PH-DCAC	Design Folder
TIEVM-HV-1PH-DCAC	Orderable EVM Tool
TMS320F28377D	Product Folder
TMS320F280049C	Product Folder
AMC1304	Product Folder
OPA4350	Product Folder
UC3865	Product Folder
C2000Ware DigitalPower SDK	Tool Folder

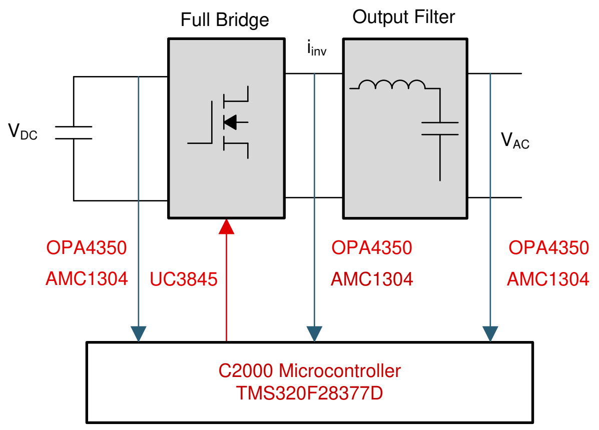
Search Our E2E™ support forums

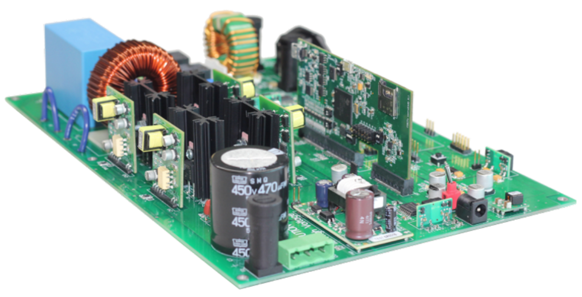
Features

- 380-DC V_{IN}, 110 V_{RMS}, 60 Hz or 22 V_{RMS}
- 50-Hz Output Selectable, 600-VA Max Output
- 98% Peak Efficiency
- 20-kHz Switching
- Low Total Harmonic Distortion (THD)
  - < 1% for Linear Loads and < 3% for Typical Non-Linear Loads With SDFM
  - < 2% for Linear Loads and < 4% for Typical Non-Linear Loads With ADC
- powerSUITE™ Support for Easy Adaption of Design
- Software Frequency Response Analyzer (SFRA) and Compensation Designer for Ease of Turning of Control Loops and Robustness of Design
- Selectable Operation of Controller on Cold or Hot (Isolated or Non-Isolated Side)
- Supports TMS320F28377D and TMS320F280049C

Applications

- Uninterrupted Power Supply (UPS)
- Micro Grids
- Photovoltaic Inverters
- Grid Storage
- Active Rectifier





1 System Description

Voltage source inverters (VSIs) are commonly used in uninterruptible power supplies (UPS) to generate a regulated AC voltage at the output. Control design of such inverter is challenging because of the unknown nature of load that can be connected to the output of the inverter.

This reference design uses devices from the C2000 microcontroller (MCU) family to implement control of a voltage source inverter. An LC output filter is used to filter the switching component in this high-frequency inverter. The firmware of the design is supported in powerSUITE framework, which enables easy adaptation of the software and control design for a custom voltage source inverter.

This reference design features high efficiency, low THD, and intuitive software, which makes it fast and easy to design VSIs. VSIs are increasingly being used in new alternative energy applications such as photovoltaic inverters, micro grids, grid storage, and more.

WARNING

TI intends this design to be operated in a lab environment only and does not consider it to be a finished product for general consumer use. The design is intended to be run at ambient room temperature and is not tested for operation under other ambient temperatures. TI intends this 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 area 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.

WARNING

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.

WARNING

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.

WARNING

Do not leave the design powered when unattended.

1.1 Key System Specifications

Table 1 shows the key system specifications of this reference design.

Table 1. Key System Specifications

VALUE	PARAMETER
Input voltage (V_{IN})	Typical 380-V DC absolute max 400-V DC
Input current (I_{IN})	1.7 A max
Output voltage (V_{OUT})	Typical 110 V_{RMS} or 220 V_{RMS} Absolute max 400 V
Output current (I_{OUT})	Absolute RMS max 5 A Pulse max 10 A
VA rating	Absolute max 600 VA
THDv: Voltage total harmonic distortion	Linear loads: • < 1% when using SDFM-based sensing • < 2% when using ADC-based sensing Typical non-linear loads: • < 3% when using SDFM-based sensing • < 4% when using ADC-based sensing
Efficiency	At 220 V_{RMS}: • Peak 98%, average is approximately 97% At 110 V_{RMS}: • Peak 96.8%, average is approximately 96%
Output inductor	3 mH
Output capacitor	20 µF
Switching frequency	20 kHz

2 System Overview

2.1 Block Diagram

A typical inverter comprises of a full bridge that is constructed with four switches, which can be modulated using pulse width modulation (PWM), and a filter for the high-frequency switching of the bridge, as shown in Figure 1. An inductor capacitor (LC) output filter is used on this reference design.

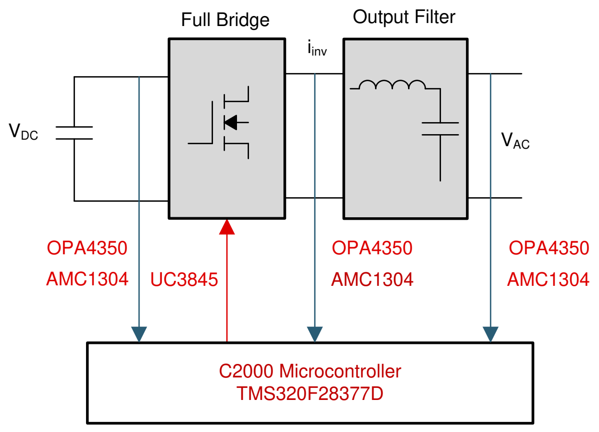


Figure 1. Typical Single Phase Inverter

2.2 System Design Theory

To regulate the output voltage of the inverter, current and voltages must be sensed. The fast and precise on-chip analog-to-digital converters (ADCs) on the C2000 MCU are excellent to sense these signals. Sigma delta-based sensing can provide easy isolation and superior sensing of these signals. Many C2000 MCUs have sigma-delta demodulators built in to decode signals from sigma delta modulators, making their use in an application straightforward.

Once the signals are sensed, the C2000 MCU runs the control algorithm to compute the modulation required for regulated operation. The compensation designer implements the model of the power stage, which makes design of digital control loop coefficients easy. SFRA enables measurement of the frequency response in-circuit to verify the accuracy of the model and ensure robustness of the control.

2.2.1 Modulation Scheme

Popular modulation schemes for the PWM generation include bipolar modulation and unipolar modulation. On this design, a modified unipolar modulation is chosen in which switches Q1 and Q2 are switched at a high frequency and switch Q3 and Q4 are switched at a low frequency (frequency of the AC waveform synthesized). Table 2 lists the switching states used on this reference design. The flexible PWM peripheral of the C2000 MCU enables the generation of these signals easily. Figure 2 shows how the PWM peripheral is configured for the modulation on this reference design. Ensure that the PWM waveform is symmetric around the zero crossing of the AC wave.

Table 2. Switching States Used in TIDM-HV-1PH-DCAC

PARAMETER	Q1	Q2	Q3	Q4	VOLTAGE AT BRIDGE OUTPUT	STATE
Positive half cycle	ON	OFF	OFF	ON	VDC	1
	OFF	ON	OFF	ON	0	2
Negative half cycle	OFF	ON	ON	OFF	–VDC	3
	ON	OFF	ON	OFF	0	4

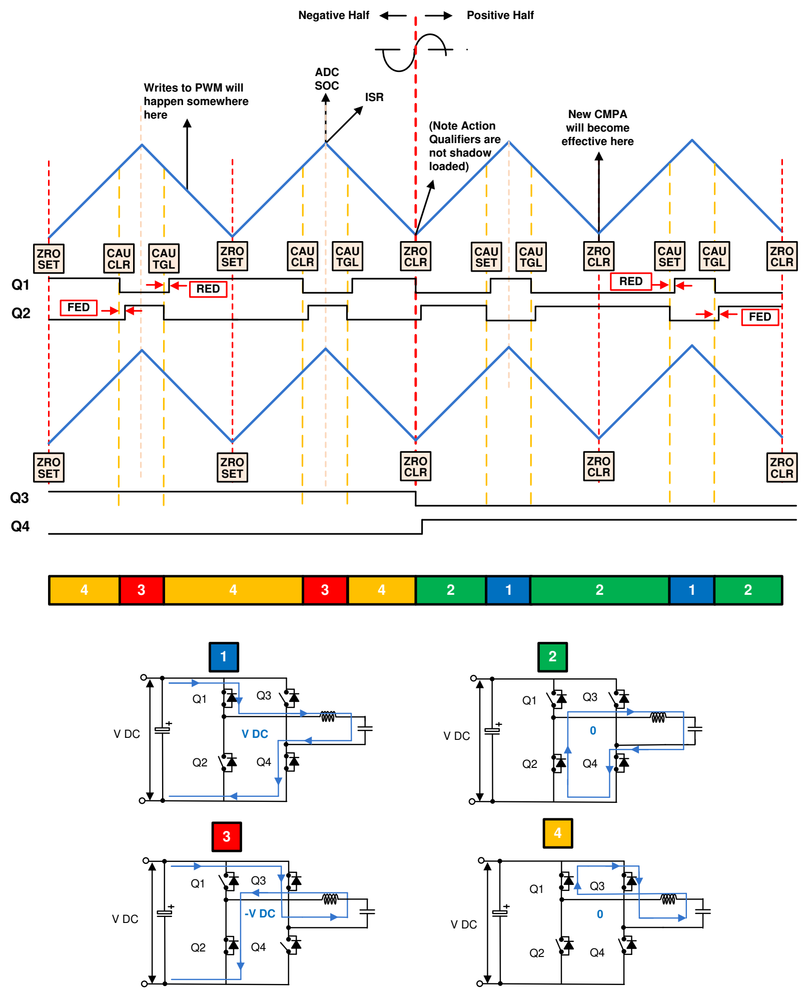

Negative Half
Positive Half
Writes to PWM will happen somewhere here
ADC SOC
ISR
(Note Action Qualifiers are not shadow loaded)
New CMPA will become effective here
ZRO SET
CAU CLR
CAU TGL
ZRO CLR
CAU SET
Q1
Q2
RED
FED
Q3
Q4
4
3
2
1
V DC
-V DC
0

Figure 2. PWM Waveform Generation Using PWM Peripheral on C2000 MCU

2.2.2 Voltage and Current Sensing

To control the inverter stage for desired operation, voltage and current need to be sensed for processing by the digital controller. The design implements sensing scheme based on ADCs and sigma delta filter modules (SDFMs). An Excel® sheet is also provided in the install package to understand the sensing methodology. The Excel sheet is located at *sdk_install_path_<version>\solutions\tidm_hv_1ph_dcac\hardware\baseboard\calculation.xlsx*.

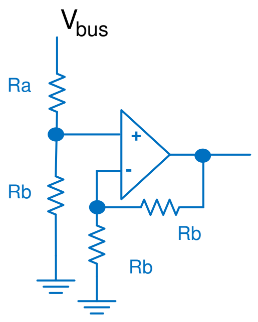
This reference design is supported in powerSUITE framework, which enables entering values of the components in the sensing circuitry resistor dividers (and so on) on the powerSUITE configuration (CFG) page of the solution. This page calculates the maximum sense value, which determines the feedback gain of the control system and is used for the compensation design and tuning.

2.2.2.1 ADC-Based Sensing

On this reference design, the following signals are sensed using the on chip ADC resource. The values shown here can also be entered through the powerSUITE CFG page when ADC-based sensing is selected for the inverter.

2.2.2.1.1 DC Bus Sensing

The high-voltage DC bus is scaled down using a resistor divider. This resistor divider output can be directly fed into the ADC. Figure 3 shows how the op amp stage is used to buffer.

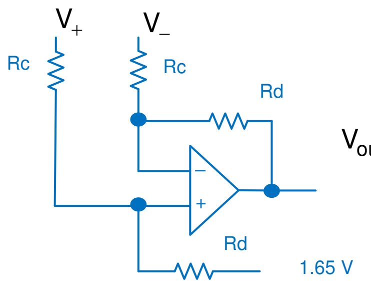


$$V_{bus\_fdbk} = \frac{2R_b}{(R_a + R_b)} V_{bus}$$

Figure 3. DC Bus Sensing Using Resistor Divider and Op Amp

2.2.2.1.2 AC Output Voltage Sensing

The AC output voltage is sensed differentially using resistor dividers and op amps, as shown in Figure 4. An offset voltage is added to the signal to enable measurement using the ADC, which can only convert positive voltages.

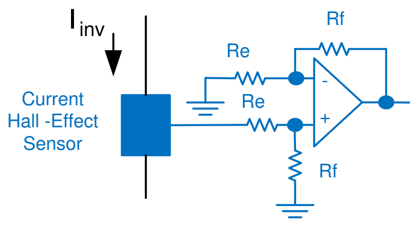


$$V_{out\_fdbk} = \frac{R_d}{R_c}(V_+ - V_-) + 1.65$$

Figure 4. AC Output Voltage Differential Sensing Using Resistor Divider and Op Amp

2.2.2.1.3 Inductor Current Sensing

A Hall effect sensor is used to sense the current through the inductor. The Hall effect sensor has a built-in offset, and the range is different than what ADC can measure. The voltage is scaled to match the ADC range using the circuit shown in Figure 5.

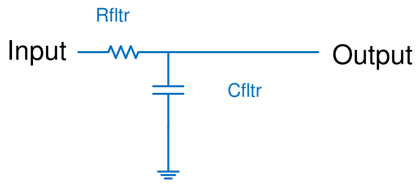


$$I_{inv\_fdbk} = \frac{R_f}{R_e}\left(I_{inv} \times \frac{V_{nominal}}{I_{nominal\_max}} + V_{offset}\right)$$

Figure 5. Current Sense Using the Hall Effect Sensor

2.2.2.1.4 Sense Filter

An RC filter is used to filter the signals before being connected to the inverter. Figure 6 shows a common RC filter used for all the sensing signals on this design.



$$\frac{\text{Output}}{\text{Input}} = \frac{1}{1 + R_{\text{fltr}} C_{\text{fltr}} s}$$

Figure 6. RC Filter

2.2.2.1.5 Protection (Windowed Comparators)

Most power electronics converters need protection from an overcurrent event. For this reason, multiple comparators are needed, and references for the current and voltage trip need to be generated (see Figure 7).

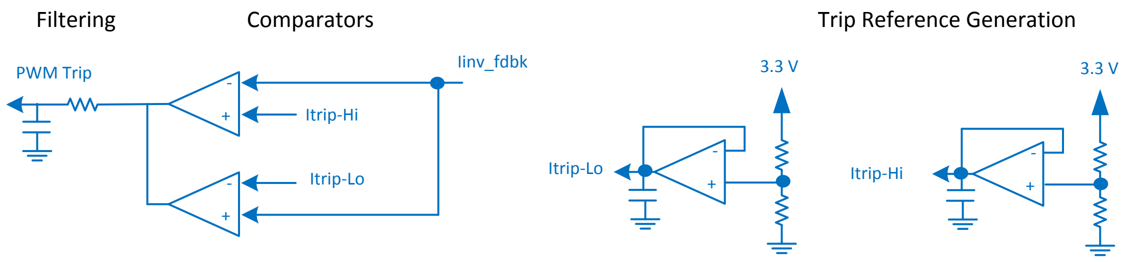


Figure 7. Trip Generation for PWM Using Comparators and Reference Generators

All of this circuitry is avoided when using C2000 MCUs such as the TMS320F28377D device, which has a on-chip windowed comparator that is internally connected to the PWM module that can enable fast tripping of the PWM. This comparator saves board space and cost in the end application as extra components can be avoided using on-chip resources, as shown in Figure 8.

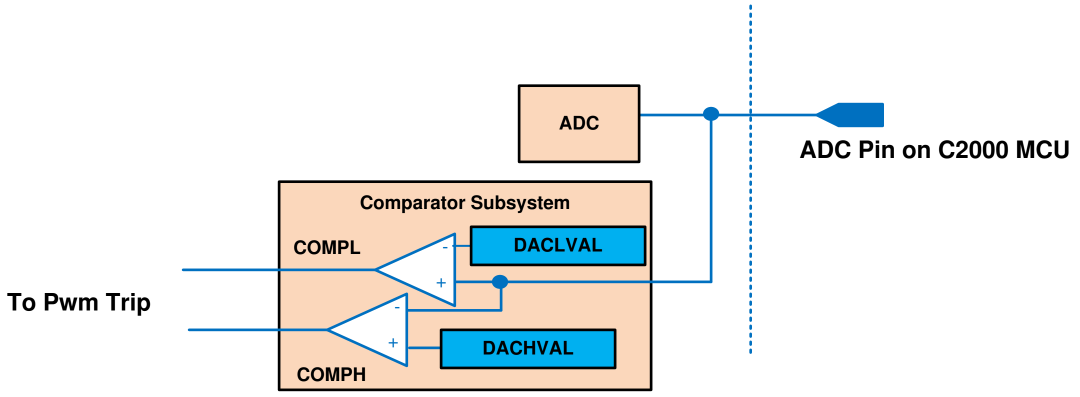


Figure 8. Comparator Subsystem Used for Overcurrent Protection

2.2.2.2 SDFM-Based Sensing

In this reference design, the following signals are sensed using the SDFM demodulator. The AMC1304 is used to generate the sigma delta stream. The clock for the modulator is generated from the ECAP peripheral on the C2000 MCU. The AMC1304 modulator senses the signal in an isolated fashion and is very useful when designing inverters in which the controller needs to be on the isolated and cold side. The values shown in the following subsections can also be entered through the powerSUITE CFG page when SDFM-based sensing is selected for the inverter.

2.2.2.2.1 Isolated Output Current and Capacitor Current Sensing

A shunt resistor senses the capacitor current and the output current on this reference design. The voltage across the shunt resistor is fed into the AMC1304 sigma-delta modulator, which generates the sigma-delta stream that is decoded by the SDFM demodulator present on the C2000 MCU. The inductor current is deduced from the capacitor and the output current readings. Figure 9 shows the SDFM-based isolated current sensing using a shunt resistor.

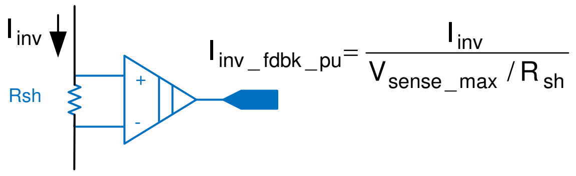

I_{inv}
Rsh
+
-
$I_{inv\_fdbk\_pu} = \frac{I_{inv}}{V_{sense\_max} / R_{sh}}$

Figure 9. SDFM-Based Isolated Current Sensing Using a Shunt Resistor

2.2.2.2.2 Isolated Output Voltage and DC Bus Sensing

A resistor divide network senses the DC bus and output voltage using the SDFM modulators. The differential input resistance of the SDFM modulator must be accounted for when interpreting the demodulated signals. Figure 10 shows the SDFM-based isolated voltage sensing for DC bus and output voltage.

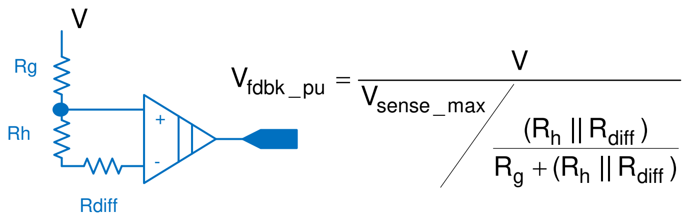

V
Rg
Rh
Rdiff
+
-
$V_{fdbk\_pu} = \frac{V}{V_{sense\_max} \Big/ \frac{(R_h \parallel R_{diff})}{R_g + (R_h \parallel R_{diff})}}$

Figure 10. SDFM-Based Isolated Voltage Sensing for DC Bus and Output Voltage

2.2.2.2.3 Protection

In addition to the data filter, which can be used to demodulate the SDFM stream generated by the modulator with specified oversampling rate (OSR) and filter order (SINC1, SINC2, SINC3), the SDFM module has additional comparator filters that can be programmed with much lower OSR and filter order to enable fast trip of the PWM.

2.2.2.2.4 SDFM Clock Generation

The ECAP module generates the clock for the SDFM modulator AMC1304. This clock is routed outside from the ECAP module using the OutputXbar and then routed back in to the SDFM CLK pins on the device. For details on usage of SDFM and CLK pins on this reference design, see Table 3.

2.2.2.2.5 SDFM Filter Reset Generation and Syncing to the Inverter PWM

SDFM provides a continuous stream of data. This data is then demodulated by the C2000 SDFM peripheral. Most control applications require the sampling of the data to be centered deterministically around the switching waveform (that is, the controlling PWM). The C2000 MCU provides a mechanism to generate this sync signal to the SDFM demodulator. The exact mechanism of the sync can be different on different devices. The following sections detail the sync mechanism as provided on several the C2000 devices.

2.2.2.2.6 TMS320F2837x/TMS320F2807x

On these devices, the PWM11 is tied to the SDFM reset generation; hence, the sync generation involves propagation of the sync from the inverter PWM to the PWM11 module. As the SDFM data is only valid 3 OSR time periods after the sync is provided, determine the time to read the SDFM data. Figure 11 shows the SDFM filter reset being generated from the PWM module and the ISR trigger to read the SDFM registers.

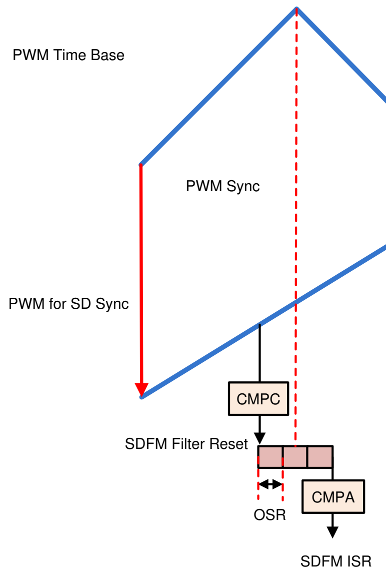


Figure 11. SDFM Filter Reset Being Generated From PWM Module and ISR Trigger to Read SDF Registers

2.2.3 Control Scheme

Figure 12 shows a cascaded control loop scheme, which controls the output voltage of the inverter. Input DC bus voltage V_{bus}, inductor current i_i, and output voltage V_o are sensed by the MCU.

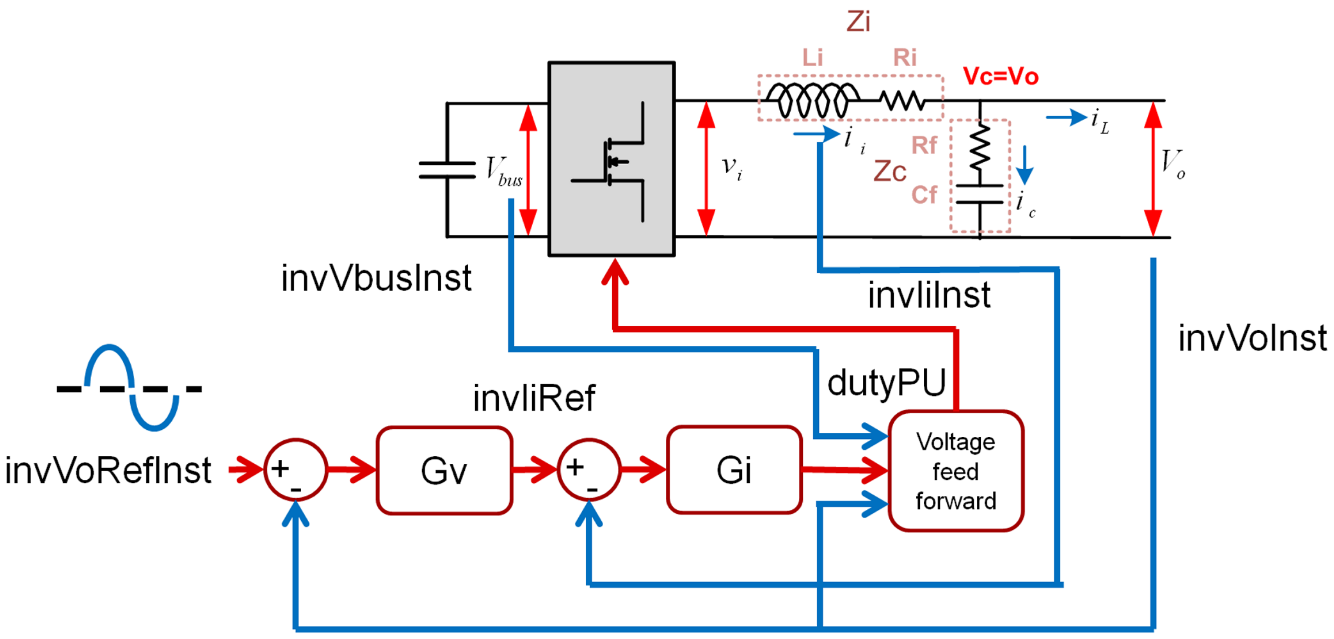


Figure 12. Control Scheme Used for Output Voltage Control

First, a reference sinusoidal value *invVoRefInst* is generated. This value is compared with the sensed output voltage *invVoInst* and the error fed into the voltage compensator Gv. In case of AC voltage control, the tracking error needs to be zeroed for the AC frequency; hence, a proportional resonant controller is used in Gv to zero the fundamental voltage error.

NOTE: A proportional integral controller zeroes error at DC and is not able to eliminate steady state error when the reference is a sinusoidal signal.

Additional resonant controllers are added to the voltage compensator to zero the error at harmonic frequency of the fundamental frequency being generated. A lead lag compensator is also added to the voltage compensator to improve the phase margin in the reference design.

$$G_V = \left(K_{pV\_1H} + \sum_{n=1}^{N} \frac{K_{iV\_nH} 2\omega_{rcV\_nH} s}{s^2 + 2\omega_{rc\_nH} S + \omega_{o\_nH}^2} \right) G_{Lead\_Lag} \tag{1}$$

Where N is the total number of harmonic compensators added in the voltage control loop. A total of four compensators that compensate the first harmonic, third harmonic, fifth harmonic, and seventh harmonic are used in this reference design. The compensation designer models the voltage loop plant and enables tuning of the voltage loop compensator coefficients through the powerSUITE CFG page.

Figure 13 shows the control model used to model the voltage source inverter operation.

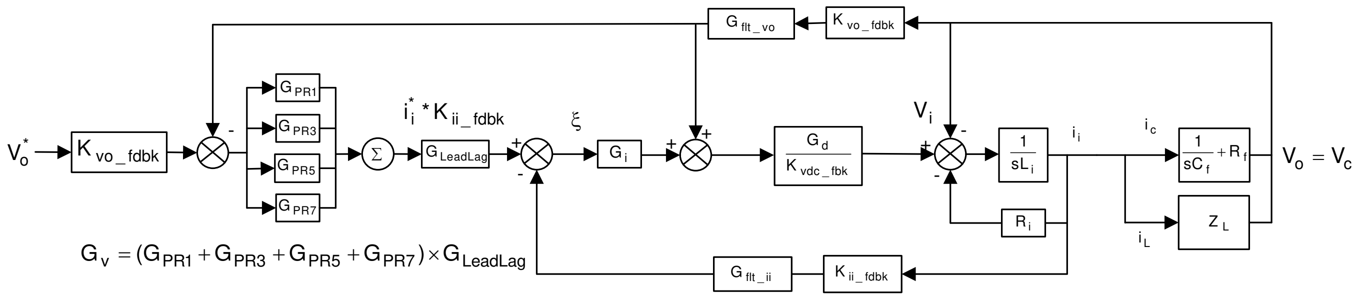


Figure 13. Control Diagram for Voltage Source Inverter

The design of the voltage loop is aided by an inner current loop. The voltage compensator generates a current reference (*invIiRefInst*) for the current loop in which a current compensator Gi is used. Both DC bus voltage and output voltage feedforward in the current loop make the plant simple, and the PI controller can be used to tune the current compensator. The plant model of the inverter current loop is available inside compensation designer, which can be invoked from the powerSUITE page.

2.2.4 Inductor Design

The primary role of the inductor (Li) in the output filter is to filter out the switching frequency harmonics. The design of an inductor, amongst other factors, depends on the calculation of the current ripple and choosing a material for the core that can tolerate the calculated current ripple. Figure 14 shows one switching cycle waveform of the inverter output voltage V_i with regards to the inductor current.

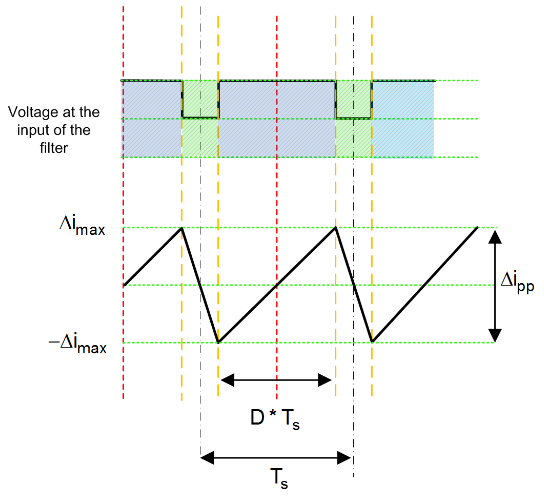


Figure 14. Current Ripple Calculation

The voltage across the inductor is given by Equation 2:

$$V = L_i \times \frac{di}{dt} \tag{2}$$

For the full-bridge inverter with an AC output, write the equation as:

$$=> (V_{Bus} - V_O) = L_i \times \frac{\Delta i_{pp}}{D \times T_s} \tag{3}$$

Where $T_s = \frac{1}{F_{sw}}$ is the switching period. Now, rearrange the current ripple at any instant in the AC waveform, given as:

$$=> \Delta i_{pp} = \frac{D \times T_s \times (V_{Bus} - V_O)}{L_i} \tag{4}$$

Assuming the modulation index to be m_a, the duty cycle is given as:

$$D(\omega t) = m_a \times \sin(\omega t) \tag{5}$$

The output of the inverter must match the AC voltage as it is safe to assume:

$$V_O = V_{DC} \times D \tag{6}$$

Therefore,

$$\Delta i_{pp} = \frac{V_{Bus} \times T_s \times m_a \times \sin(\omega t) \times (1 - m_a \sin(\omega t))}{L_i} \tag{7}$$

As seen in Equation 7, the peak ripple is a factor of where the inverter is in the sinusoidal waveform (for example, the modulation index). To find the modulation index where the maximum ripple is present, differentiate Equation 7 with regards to time to get Equation 8, and equate to zero.

$$\frac{d(\Delta i_{pp})}{dt} = K\{\cos(\omega t)(1 - m_a \sin(\omega t)) - m_a \sin(\omega t) * \cos(\omega t)\} = 0 \tag{8}$$

$$=> \sin(\omega t) = \frac{1}{2\, m_a}$$

then gives the modulation index for which the ripple is maximum, substituting back in Equation 7. The inductance value required to tolerate the ripple is shown in Equation 9 and Equation 10:

$$\Delta i_{pp}\Big|_{max} = \frac{V_{Bus} \times T_s}{4 \times L_i} \tag{9}$$

$$L_i = \frac{V_{Bus}}{4 \times F_{sw} \times \Delta i_{pp}\Big|_{max}} \tag{10}$$

For this design, the rating is 600 VA, the switching frequency is 20 kHz, and the bus voltage is 380 V. Assume that the ripple is 20% and is tolerable by the inductor core, and the minimum inductance required is calculated as:

$$L = \frac{380}{4 \times 20000 \times 5.45 \times 1.414 \times 0.20} = 3.08 \text{ mH} \tag{11}$$

These calculations are also provided inside an Excel sheet for convenience located at: *location of Excel sheet to C:\ti\c2000\C2000Ware_DigitalPower_SDK_<version>\solutions\tidm_hv_1ph_dcac\hardware\baseboard\calculation.xlsx sheet → UPS Li & Cf Sel*

An appropriate core must be selected with these values in mind, and the inductor is designed to meet the inductance value.

2.2.5 Capacitance Selection

The output inductor and capacitor form a low pass filter that filters out the switching frequency. To get good switching frequency attenuation the cut off frequency is kept at F_{sw}/10 or lower.

3 Hardware, Firmware, Testing Requirements, and Test Results

3.1 Required Hardware and Firmware

3.1.1 Hardware

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

NOTE: This reference design is also available for order as TIEVM-HV-1PH-DCAC. Note for the 15-V DC power supply, 15 W is not shipped with the design and must be arranged for by the user. A two-pronged power supply is recommended so it is truly floating and isolated. Cables, loads, oscilloscopes, and current probes must be arranged for by the user and connected to this EVM according to the user guide instructions and observing local compliance and standards for wiring. Use isolated power supplies.

3.1.1.1 Base Board Settings

This reference design follows an HSEC control card concept, and any device for which the HSEC control card is available from the C2000 MCU product family can be potentially used on the design. The key resources used for controlling the power stage on the MCU are listed in Table 3. Figure 15 shows the key power stage and connectors on the reference design, and Table 4 lists the key connectors and their functions. To get started:

1. Confirm that no power source is connected to the design.
2. Confirm that the output filter is correct for the mode that the device will run in. For example, voltage source inverter uses an LC filter. The L2 and L2N slot must be jumper wired as shown in Figure 11.
3. Ensure that the capacitor is 20 µF by checking the marking on the capacitor.
4. Insert the control card in the J15-J16 slot.
5. Insert a jumper at J10 if not already populated.
6. Connect a 15-V DC, 1-A power supply at J2.
7. Insert a jumper at J4 if not already populated. The LED lights on the base board and control card will light up to indicate that the device is powered up.

NOTE: The bias for the MCU is separated from the power stage, enabling safe bring up of the system.

8. To connect JTAG, use a USB cable from the control card and connect it to a host computer.
9. A DC source can also be connected to the J17, but do not apply power at this point.
10. Connect a resistive load of approximately 100 Ω to the output from J1.

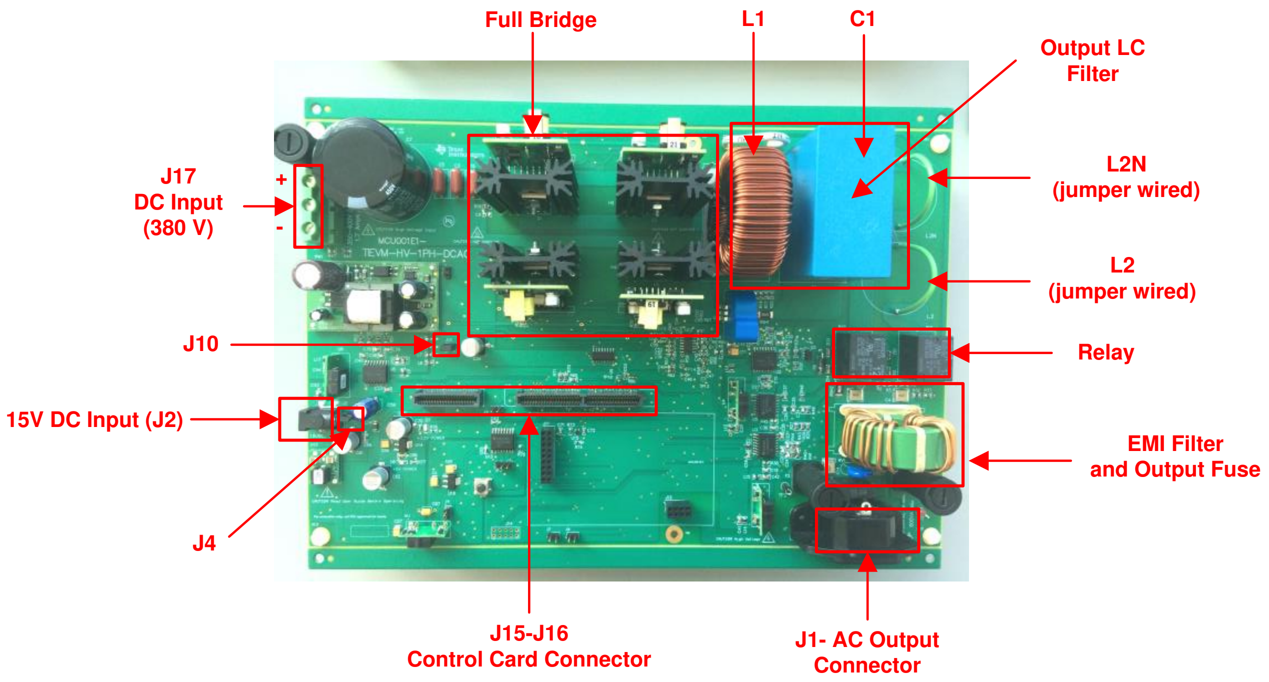


Figure 15. Board Overview

Table 3. Key Controller Peripherals Used for Control on the Full Bridge on the Board

SIGNAL NAME	HSEC PIN NUMBER	FUNCTION
PWM-1A	49	PWM: Inverter drive
PWM-1B	51	PWM: Inverter drive
PWM-2A	53	PWM: Inverter drive
PWM-2B	55	PWM: Inverter drive
I.inv	15	ADC: Inductor current measurement
1.65V	17	ADC: Reference voltage generated on the board
Bus.V	21	ADC: DC bus sensed on the board
Line.V	25	ADC: AC voltage sensing
PLC_RX	27, 12	ADC: PLC ADC pin
SD_Data_CapI	99	SDFM: Data from the SDFM modulator for the capacitor current feedback
SD_Data_GridI	103	SDFM: Data from the SDFM modulator for output current
SD_Data_Vbus	100	SDFM: Data from the SDFM modulator for the DC bus voltage
SD_Data_GridV	107	SDFM: Data from the SDFM modulator for the grid voltage
SD_CLK_GridV / SD_CLK_GridI / SD_CLK_CapI	50, 101, 105, 109	SDFM: Data from the SDFM modulator for the grid voltage
SD_CLK_Vbus	102, 54	SDFM: Clock from the SDFM modulator used for Vbus measurement. The clock is generated from ECAP1 module which is brought out using the output xBar
OPRLY	52	GPIO: relay gpio output
SW-ON	56	GPIO: switch gpio input

Table 4. Key Connectors and Their Function

CONNECTOR NAME	FUNCTION
J17	Used to connect the high-voltage DC bus at the input
J2	Supplies the bias power supply for the control card and the circuitry for sensing on the base board
J4	Can be used to disconnect bias power
J1	AC connector to connect the output to load
J15-J16	HSEC control card slot
J10	Supplies the DC bias power supply to the isolated gate drivers, and the drivers must be populated

Figure 16 shows a block diagram of a typical setup of this reference design under test.

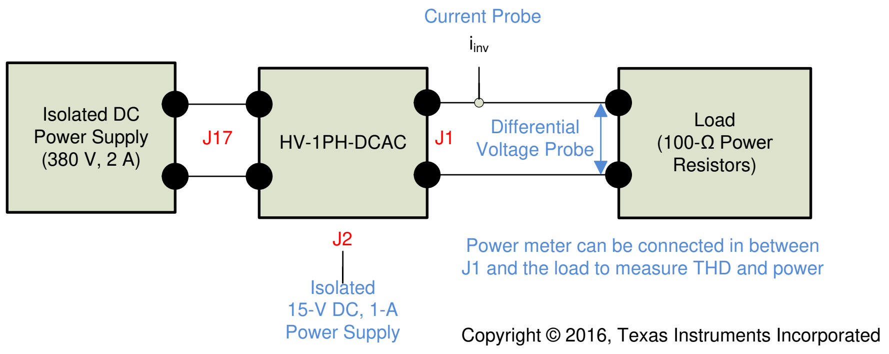

Current Probe
i_{inv}
Isolated DC Power Supply (380 V, 2 A)
J17
HV-1PH-DCAC
J1
Differential Voltage Probe
Load (100-Ω Power Resistors)
J2
Isolated 15-V DC, 1-A Power Supply
Power meter can be connected in between J1 and the load to measure THD and power
Copyright © 2016, Texas Instruments Incorporated

Figure 16. Hardware Setup to Run Software

3.1.1.2 *Control Card Settings*

Certain settings on the device control card are needed to communicate over JTAG; use the isolated UART port, and provide a correct ADC reference voltage. The following settings are required on revision 1.1 of the TMS320F28377D control card. Refer to the info sheet located inside C2000Ware at *<sdk_install_path>c2000Ware\boards\controlCARDs\TMDSCNCD28377D*.

1. Set A:SW1 on both ends of the control card to the *ON* (up) position to enable the JTAG connection to the device and UART connection for the SFRA GUI. If the switch is *OFF* (down), the user cannot use the isolated JTAG built in the control card, nor can the SFRA GUI communicate with the device.
2. Connect the USB cable to A:J1 to communicate to the device from a host PC on which Code Composer Studio™ (CCS) runs.
3. For the control loop, set the appropriate jumpers to provide a 3.3-V reference externally to the on-chip ADC.

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 settings are required for revision A of the TMS320F280049C control card (refer to the info sheet located at *<sdk_install_path>\c2000ware\boards\controlcards\TMDSCNCD280049C*:

1. Set both ends of S1:A on the control card to *ON* (up) position to enable JTAG connection to the device and UART connection for SFRA GUI. If this switch is *OFF* (down), the user cannot use the isolated JTAG built in on the control card, nor can the SFRA GUI communicate with the device.
2. Connect the USB cable to J1:A to communicate with the device from a host PC on which CCS runs.
3. For the control loop, use the internal reference of the TMS320F28004x and move the S8 switch to the left (that is, pointing to VREFHI).
4. For the best performance of this reference design, remove the capacitor connected between the isolated grounds on the control card, C26:A.
5. GPIO24 through GPIO27 are muxed on the TMS320F280049C control card. To route them to the correct control card pins for the SDFM, put all the switches on SW5 to *OFF* (down) and all the switches on SW6 to *ON* (up).

3.1.1.3 Tips to Connect JTAG USB Cable

High-voltage boards can generate high EMI due to switching action. Even though the JTAG is isolated, some coupling can still occur due to radiated EMI. This coupling can result in a loss of JTAG frequently. To avoid this from happening, perform the following steps:

1. Wind the USB cable around a ferrite bead as shown in Figure 17.

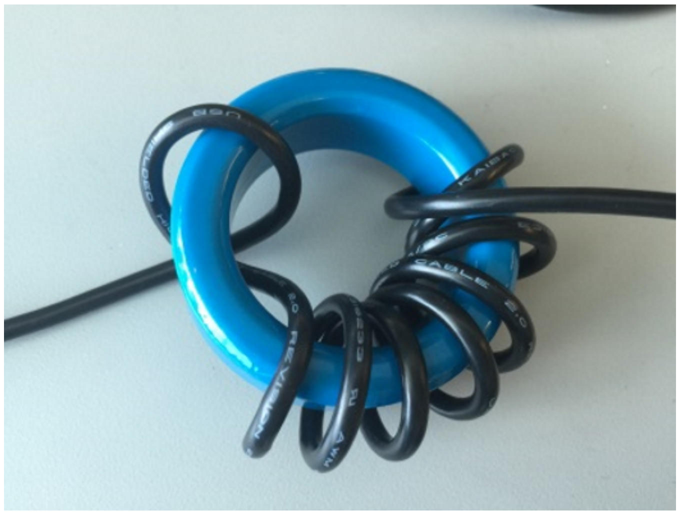


Figure 17. USB Cable Around Ferrite Bead

2. Make sure the USB cable does not cross directly over the high-voltage section. This setup can be ensured on this design by the following connection of the USB cable.

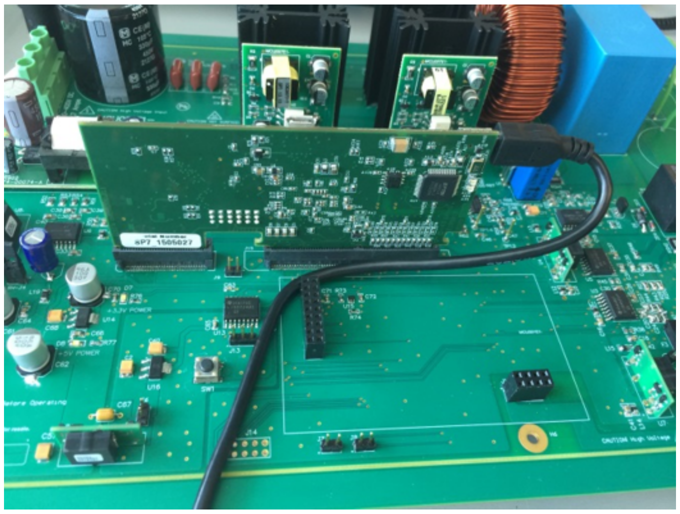


Figure 18. USB Connection on Board

3.1.2 Firmware: powerSUITE™ and Incremental Build Software

NOTE: The firmware for the solution is supported on both the TMS320F283779D and TMS320F280049C devices.

3.1.2.1 Opening the Project Inside Code Composer Studio™

To start:

1. Install CCS (version 9.3 or above).
2. Install the C2000Ware DigitalPower SDK at the C2000Ware Digital Power SDK tool folder.

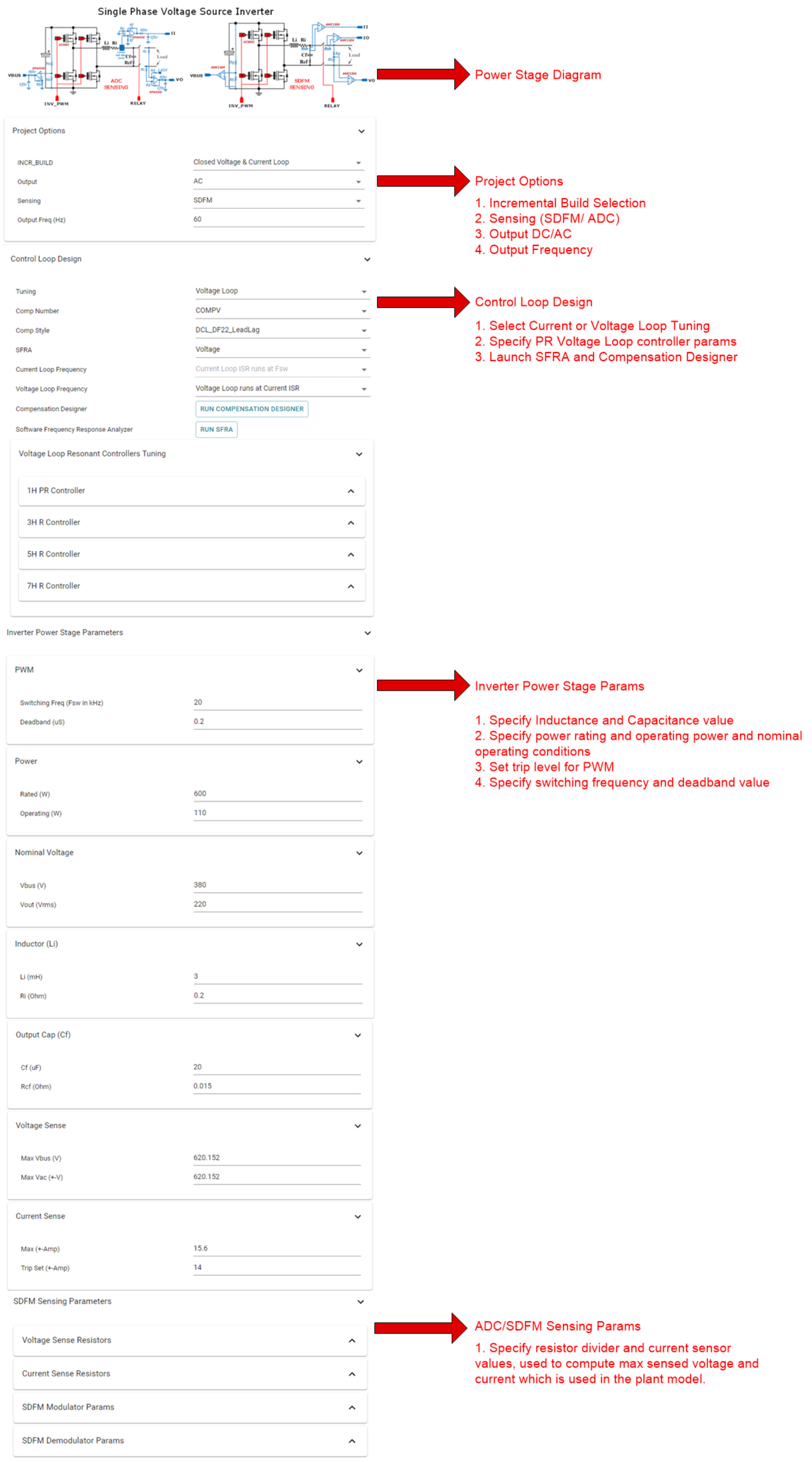
NOTE: powerSUITE is installed with the DigitalPower SDK in the default install.

3. Open CCS, go to *View* → *Resource Explorer*
4. Under the TI *Resource Explorer*, go to *Software* → *C2000Ware DigitalPower SDK - <version>*.

3.1.2.1.1 Open TI 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 also modify scaling values for voltages and currents.

1. In the resource explorer under C2000Ware Digital Power SDK, click on the *Solution Adapter Tool*
2. Select *Single Phase Inverter: Voltage Source* from the list of solutions presented.
3. The development kit and designs page appear. Use this page to browse all the information on the design including this user guide, test reports, and hardware design files.
4. Click on *Import <device name>Project*.
5. The project imports into the workspace environment. A .cfg page with a GUI similar to Figure 19 appears.
6. Use the GUI to change the parameters for an adopted solution, such as power rating, inductance, capacitance, or sensing circuit parameters if desired.



Project Options

INCR_BUILD	Closed Voltage & Current Loop
Output	AC
Sensing	SDFM
Output Freq (Hz)	60

Project Options
1. Incremental Build Selection
2. Sensing (SDFM/ ADC)
3. Output DC/AC
4. Output Frequency

Control Loop Design

Tuning	Voltage Loop
Comp Number	COMPV
Comp Style	DCL_DF22_LeadLag
SFRA	Voltage
Current Loop Frequency	Current Loop ISR runs at Fsw
Voltage Loop Frequency	Voltage Loop runs at Current ISR
Compensation Designer	RUN COMPENSATION DESIGNER
Software Frequency Response Analyzer	RUN SFRA

Voltage Loop Resonant Controllers Tuning
- 1H PR Controller
- 3H R Controller
- 5H R Controller
- 7H R Controller

Control Loop Design
1. Select Current or Voltage Loop Tuning
2. Specify PR Voltage Loop controller params
3. Launch SFRA and Compensation Designer

Inverter Power Stage Parameters

PWM	
Switching Freq (Fsw in kHz)	20
Deadband (uS)	0.2
Power	
Rated (W)	600
Operating (W)	110
Nominal Voltage	
Vbus (V)	380
Vout (Vrms)	220
Inductor (Li)	
Li (mH)	3
Ri (Ohm)	0.2
Output Cap (Cf)	
Cf (uF)	20
Rcf (Ohm)	0.015
Voltage Sense	
Max Vbus (V)	620.152
Max Vac (+-V)	620.152
Current Sense	
Max (+-Amp)	15.6
Trip Set (+-Amp)	14

Inverter Power Stage Params
1. Specify Inductance and Capacitance value
2. Specify power rating and operating power and nominal operating conditions
3. Set trip level for PWM
4. Specify switching frequency and deadband value

SDFM Sensing Parameters
- Voltage Sense Resistors
- Current Sense Resistors
- SDFM Modulator Params
- SDFM Demodulator Params

ADC/SDFM Sensing Params
1. Specify resistor divider and current sensor values, used to compute max sensed voltage and current which is used in the plant model.

Figure 19. powerSUITE Page for Voltage Source Inverter Solution

3.1.2.2 Project Structure

Once the project is imported, the project explorer appears inside CCS, as shown in Figure 20.

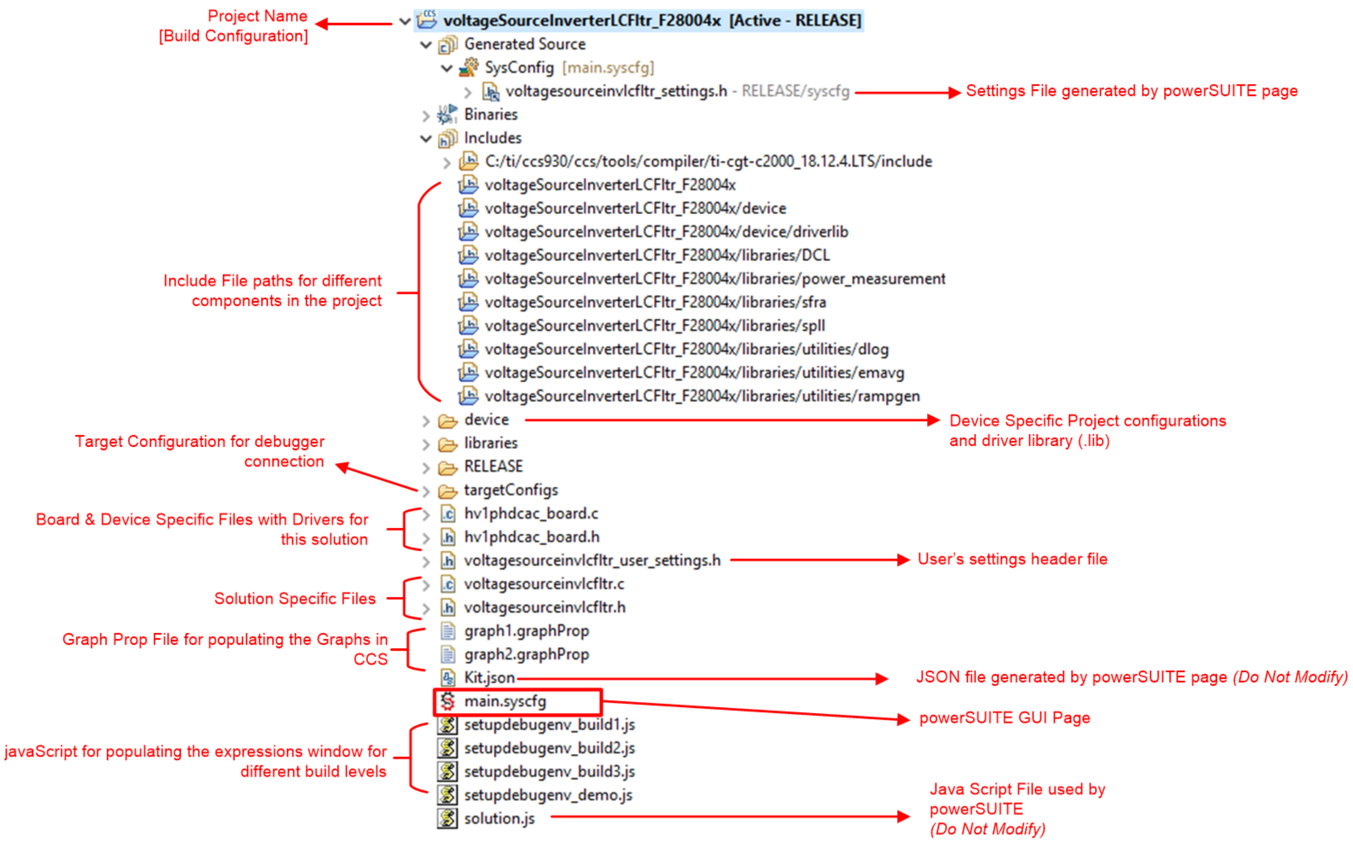
NOTE: Figure 20 shows the project for TMS320F2837x; however, if a different device is chosen from the powerSUITE page, the structure will be similar.

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

Board specific and device specific files are *hv1phdcac_board.c/h*. This file consists of device specific drivers to run the single phase inverter. If the user desired to use a different modulation scheme or use a different device changes are required only to these files, asides from changing the device support files in the project.

The powerSUITE page can be opened by clicking on the *main.syscfg* file, listed under the project explorer. The powerSUITE page generates the *voltagesourceinvlcfltr_settings.h* file. This file is the only file used in the compile of the project that is generated by the powerSUITE page. User defined settings are located in *voltagesourceinvlcfltr_user_settings.h*.

The *Kit.json* and *solution.js* files are used internally by powerSUITE and must also not be modified by the user. Any changes to these files result in project not functioning properly. Figure 20 shows the project explorer view of the solution project.



```
voltageSourceInverterLCFltr_F28004x [Active - RELEASE]
  Generated Source
    SysConfig [main.syscfg]
      voltagesourceinvlcfltr_settings.h - RELEASE/syscfg
  Binaries
  Includes
    C:/ti/ccs930/ccs/tools/compiler/ti-cgt-c2000_18.12.4.LTS/include
    voltageSourceInverterLCFltr_F28004x
    voltageSourceInverterLCFltr_F28004x/device
    voltageSourceInverterLCFltr_F28004x/device/driverlib
    voltageSourceInverterLCFltr_F28004x/libraries/DCL
    voltageSourceInverterLCFltr_F28004x/libraries/power_measurement
    voltageSourceInverterLCFltr_F28004x/libraries/sfra
    voltageSourceInverterLCFltr_F28004x/libraries/spll
    voltageSourceInverterLCFltr_F28004x/libraries/utilities/dlog
    voltageSourceInverterLCFltr_F28004x/libraries/utilities/emavg
    voltageSourceInverterLCFltr_F28004x/libraries/utilities/rampgen
  device
  libraries
  RELEASE
  targetConfigs
  hv1phdcac_board.c
  hv1phdcac_board.h
  voltagesourceinvlcfltr_user_settings.h
  voltagesourceinvlcfltr.c
  voltagesourceinvlcfltr.h
  graph1.graphProp
  graph2.graphProp
  Kit.json
  main.syscfg
  setupdebugenv_build1.js
  setupdebugenv_build2.js
  setupdebugenv_build3.js
  setupdebugenv_demo.js
  solution.js
```

Project Name [Build Configuration]
Settings File generated by powerSUITE page
Include File paths for different components in the project
Device Specific Project configurations and driver library (.lib)
Target Configuration for debugger connection
Board & Device Specific Files with Drivers for this solution
User's settings header file
Solution Specific Files
Graph Prop File for populating the Graphs in CCS
JSON file generated by powerSUITE page (Do Not Modify)
powerSUITE GUI Page
javaScript for populating the expressions window for different build levels
Java Script File used by powerSUITE (Do Not Modify)

Figure 20. Project Explorer View of the Solution Project

The project consists of an interrupt service routine, which is called every PWM cycle called *inverter_ISR()* where the control algorithm is executed. In addition to this, there are background tasks A0-A4, B0-B4, C0-C4, which are called in a polling fashion and can be used to run slow tasks for which absolute timing accuracy is not required.

3.1.2.3 Running the Project

The software of this reference design is organized in three incremental builds and a few options to test the control loop design. The incremental build process simplifies the system bring-up and design. This process is outlined in the following sections. If using the reference design hardware, make sure the hardware setup is completed as outlined in Section 3.1.1.

3.1.2.3.1 Build Level 1—Open Loop

In this build, the inverter is excited in open loop fashion with a fixed modulation index, as shown in Figure 21. First, a ramp generator generates the theta angle, which is then used to compute the sine value. This sine value is multiplied with the *invModIndex* variable, which gives the duty cycle *invDutyPU* with which the inverter full bridge is modulated. Modulation scheme and feedback values from the power stage can be checked in this build to ensure they are correct and there are no hardware issues.

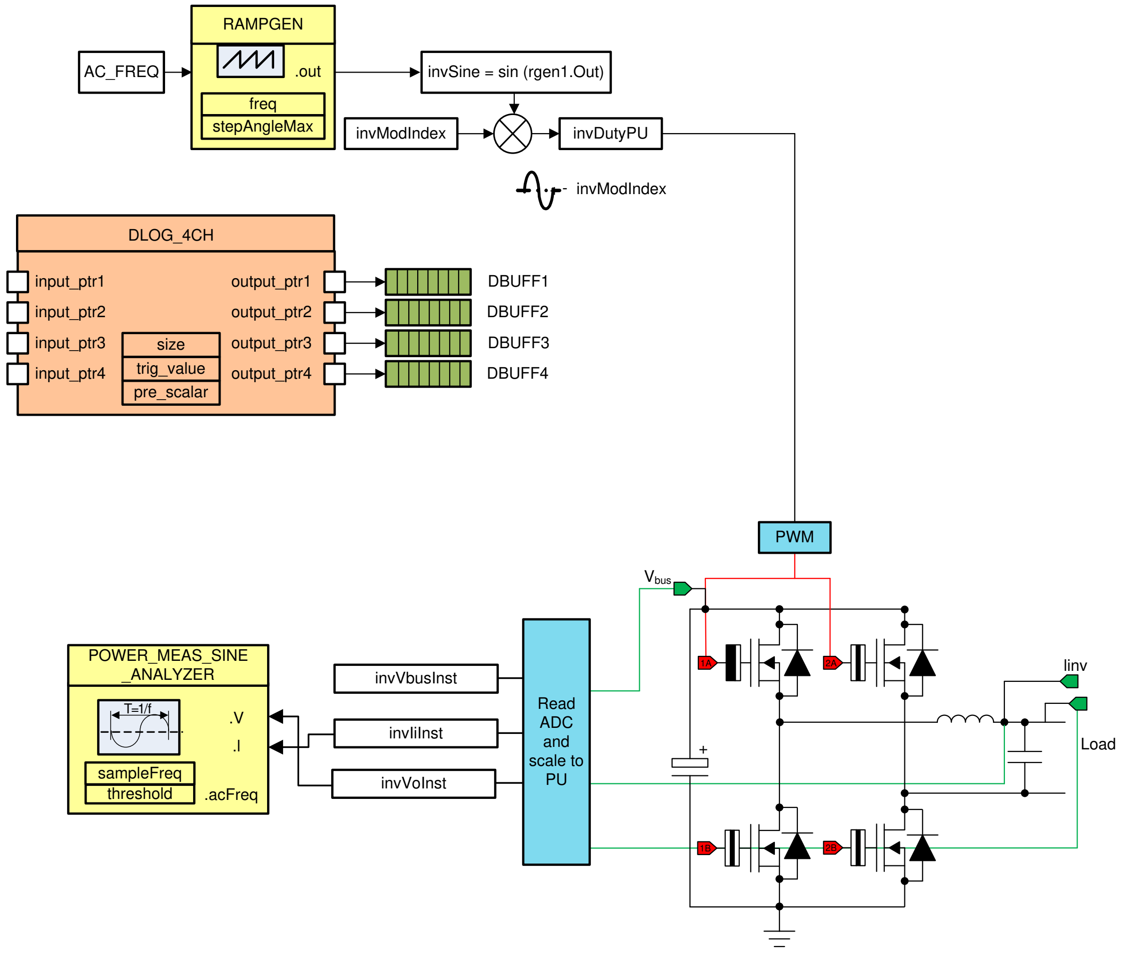


Figure 21. Build Level 1 Control Diagram: Open Loop Project

3.1.2.3.1.1 Setting Software Options for Build 1

1. Make sure the hardware is setup as shown in Figure 16. Do not supply any high-voltage power to the board yet.
2. powerSUITE settings: On the powerSUITE page, select under the *Project Options* section:
   - *Open Loop* for the build level
   - *AC* for the *Output*
   - Either *SDFM* or *ADC* for the sensing method depending on what is in the design (For this design, both *SDFM* and *ADC* sensing methods are present)
   - Enter output frequency as 60 Hz

   If this is an adapted solution, edit the setting under *ADC Sensing Parameters*, *SDFM Sensing Parameters* with the sensing resistors used in each case. Specify the switching frequency, the dead band, and the power rating. Save the page.

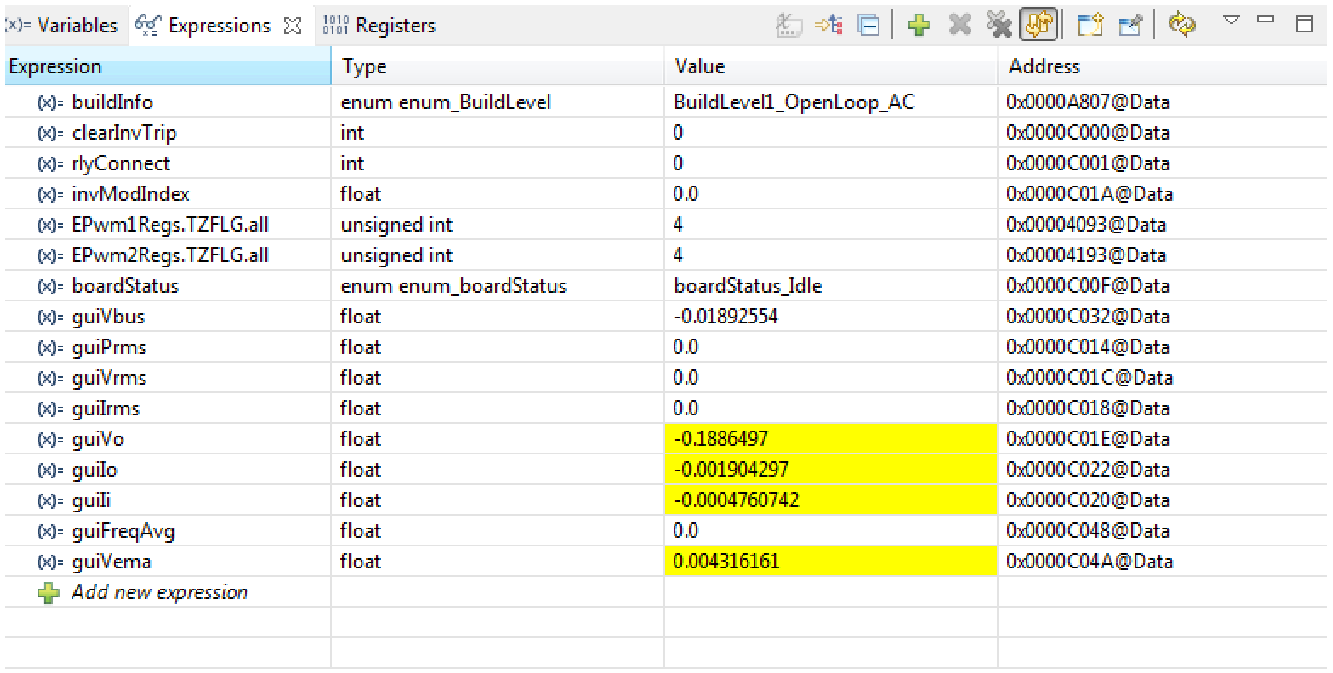
3.1.2.3.1.2 Building and Loading the Project

1. Right click on the project name and click *Rebuild Project*.
2. The project will build successfully.
3. In the *Project Explorer*, make sure the correct target configuration file is set as *Active*, as shown in Figure 20.
4. Click *Run* → *Debug* to launch a debugging session. A window can appear to select the CPU the debug needs to be performed on in case of dual CPU devices. In this case, select *CPU1*.

3.1.2.3.1.3 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.
2. On the upper right corner of the console, click on open to browse to the setupdebugenv_build1.js script file inside the project folder.
3. The watch window will be populated with the appropriate variables needed to debug the system.
4. Click on the Continuous Refresh button on the watch window to enable continuous update of values from the controller.

   Figure 22 shows how the watch window will appear after these steps are taken.

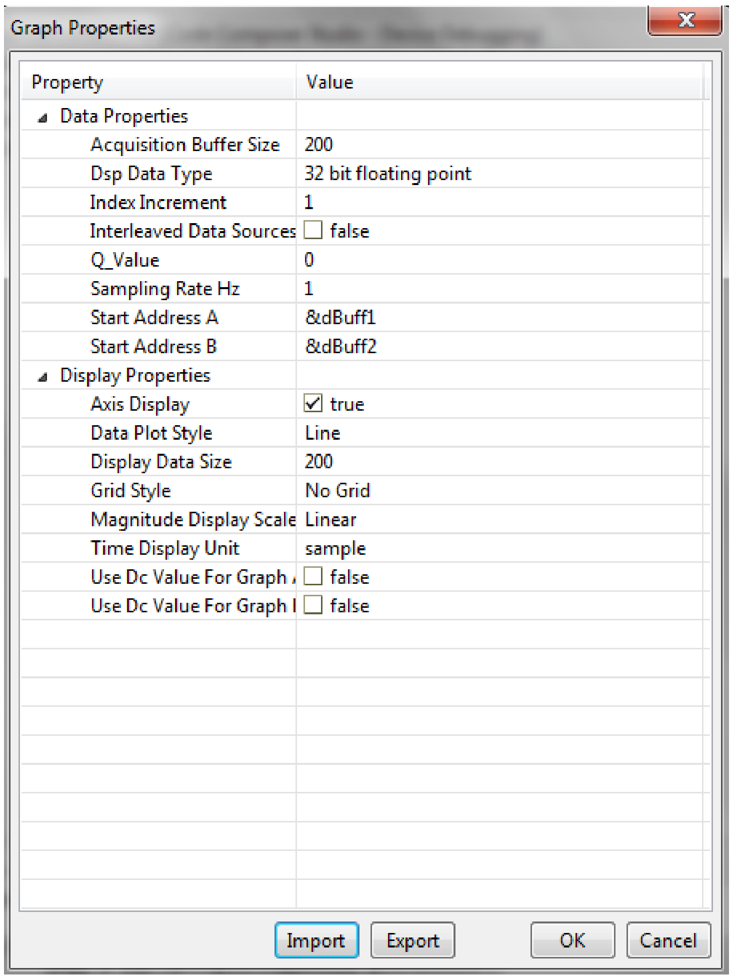


Expression	Type	Value	Address
buildInfo	enum enum_BuildLevel	BuildLevel1_OpenLoop_AC	0x0000A807@Data
clearInvTrip	int	0	0x0000C000@Data
rlyConnect	int	0	0x0000C001@Data
invModIndex	float	0.0	0x0000C01A@Data
EPwm1Regs.TZFLG.all	unsigned int	4	0x00004093@Data
EPwm2Regs.TZFLG.all	unsigned int	4	0x00004193@Data
boardStatus	enum enum_boardStatus	boardStatus_Idle	0x0000C00F@Data
guiVbus	float	-0.01892554	0x0000C032@Data
guiPrms	float	0.0	0x0000C014@Data
guiVrms	float	0.0	0x0000C01C@Data
guiIrms	float	0.0	0x0000C018@Data
guiVo	float	-0.1886497	0x0000C01E@Data
guiIo	float	-0.001904297	0x0000C022@Data
guiIi	float	-0.0004760742	0x0000C020@Data
guiFreqAvg	float	0.0	0x0000C048@Data
guiVema	float	0.004316161	0x0000C04A@Data
Add new expression			

Figure 22. Build Level 1 Expressions View

5. The inverter current and voltage measurements can also be verified by viewing the data in the graph window. These values are logged in the inverterISR() routine.
6. Go to *Tools* → *Graph* → *DualTime* and click on *Import* and point to the *graph1.GraphProp file* inside the project folder.
7. The graph will populate in the properties window.
8. Alternatively, the user can enter the values as shown in Figure 23.
9. Click *OK* once the entries are verified.
10. Two graphs will appear in the CCS.
11. Click on Continuous Refresh on the graphs.

A second set of graphs can be added by importing *graph.2.GraphProp file*.


Graph Properties

Property	Value
Data Properties	
Acquisition Buffer Size	200
Dsp Data Type	32 bit floating point
Index Increment	1
Interleaved Data Sources	☐ false
Q_Value	0
Sampling Rate Hz	1
Start Address A	&dBuff1
Start Address B	&dBuff2
Display Properties	
Axis Display	✓ true
Data Plot Style	Line
Display Data Size	200
Grid Style	No Grid
Magnitude Display Scale	Linear
Time Display Unit	sample
Use Dc Value For Graph /	☐ false
Use Dc Value For Graph I	☐ false

Figure 23. Graph Settings

3.1.2.3.1.4 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 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 button Enable Silicon Real-time Mode (service critical interrupts when halted, allow debugger accesses while running).
2. A message box *may* appear.
3. If the message box appears, select *Yes* to enable debug events.
4. Clicking *Yes* sets bit 1 (DGM but) of status register 1 (ST1) to 0.

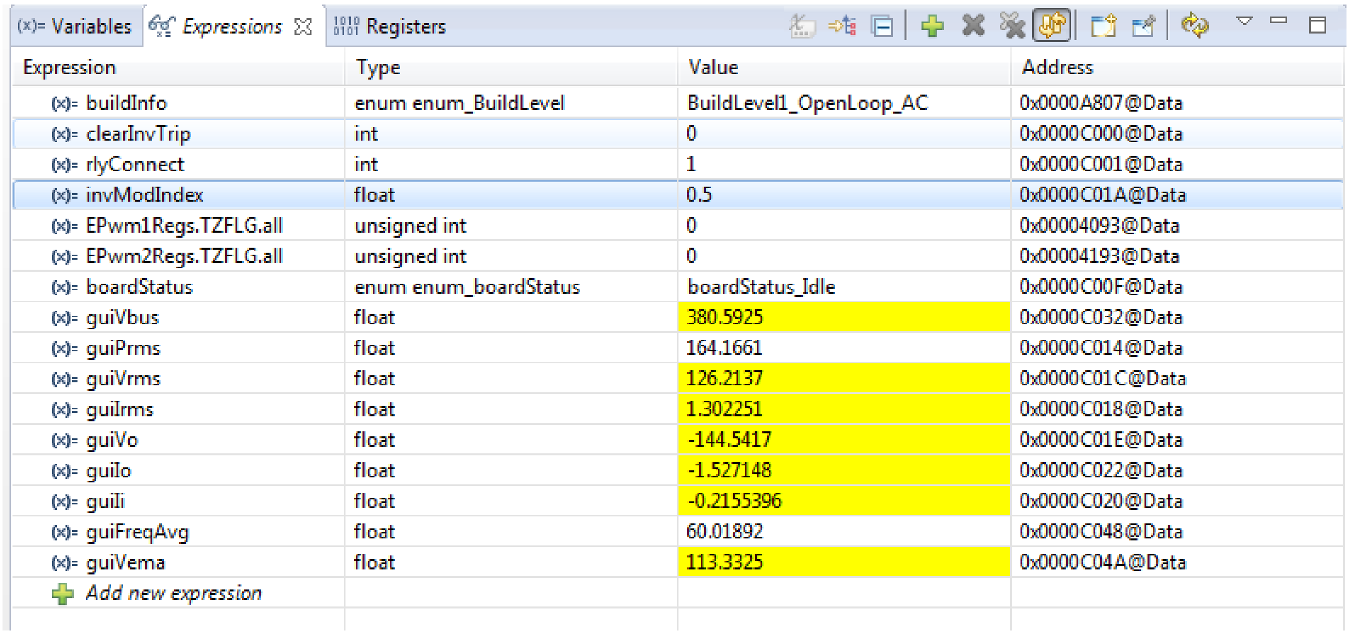
NOTE: The DGBM is the debug enable mask bit. When the DGBM bit is set to 0, the memory and register values are passed to the host processor for updating the debugging windows.

3.1.2.3.1.5 Running the Code

1. Run the project by clicking the green arrow button.
2. In the watch view, check if the *guiVbus*, *guiIi*, *guiIo*, and *guiVo* variables are updating periodically.
3. Set the value of *rlyConnect* to 1, which will connect the relay, and the user will hear a clicking sound.
4. Set the *clearInvTrip* variable to 1.
5. *EmPwm1Regs.TZFLG.all* appears.
6. Set *EPwm1Regs* to zero, and the *boardStatus* will update to *boardStatus_NoFault*.
7. Set the *invModIndex* to 0.5.
8. With a resistance of 100 Ω connected at the output, first raise the input DC bus up slowly to 50 V. Observe the current and voltage waveform on the scope. AC current and voltage waveform should appear albeit at a low voltage. Now, slowly increase the DC bus to 380 V.
9. Observe the AC waveform on the oscilloscope for the voltage and current.
10. Observe the clean AC waveform (owing to the low frequency switching, a sharp pulse around the zero crossing is expected).
11. Check the frequency of the generated AC waveform.
12. Confirm that the frequency matches the value entered on the powerSUITE page.
13. Confirm the AC measurement is correct by viewing the *gui_Vrms* and *gui_Irms* values.
14. For the DC bus of 380 V, and a 100-Ω output with a 0.5 inverter modulation index, the output voltage will be close to guiVrms = 133, and the current will be guiIrms = 1.35 A for SDFM sensing; for ADC sensing, the measured current is going to be higher because the inductor current is measured instead of the output current.

NOTE: If there are inconsistencies in the measured valued and the actual values, confirm that the hardware and enter the correct values on the powerSUITE page for the scaling of voltages and currents.

Figure 24 shows the build level 1 expressions view with power measurement.



(x)= Variables | Expressions | Registers

Expression	Type	Value	Address
buildInfo	enum enum_BuildLevel	BuildLevel1_OpenLoop_AC	0x0000A807@Data
clearInvTrip	int	0	0x0000C000@Data
rlyConnect	int	1	0x0000C001@Data
invModIndex	float	0.5	0x0000C01A@Data
EPwm1Regs.TZFLG.all	unsigned int	0	0x00004093@Data
EPwm2Regs.TZFLG.all	unsigned int	0	0x00004193@Data
boardStatus	enum enum_boardStatus	boardStatus_Idle	0x0000C00F@Data
guiVbus	float	380.5925	0x0000C032@Data
guiPrms	float	164.1661	0x0000C014@Data
guiVrms	float	126.2137	0x0000C01C@Data
guiIrms	float	1.302251	0x0000C018@Data
guiVo	float	-144.5417	0x0000C01E@Data
guiIo	float	-1.527148	0x0000C022@Data
guiIi	float	-0.2155396	0x0000C020@Data
guiFreqAvg	float	60.01892	0x0000C048@Data
guiVema	float	113.3325	0x0000C04A@Data
Add new expression			

Figure 24. Build Level 1 Expressions View With Power Measurement

Figure 25 shows the graph window inside CCS that was set up previously (see Figure 23). The measured voltage and current AC waveform can be observed on these graphs.

Figure 25 shows the graph imported in the build level 1 through the graph1.graphprop file, displaying the measured per-unit voltage and current values.

If nothing is observed in the graph, put dlog1.status in the Expression window, and if it is set to 0, set it to 1. Also, if multiple AC cycle need to be observed, enter dlog1.prescalar to be greater than 1; for example, it can be set to be 5.

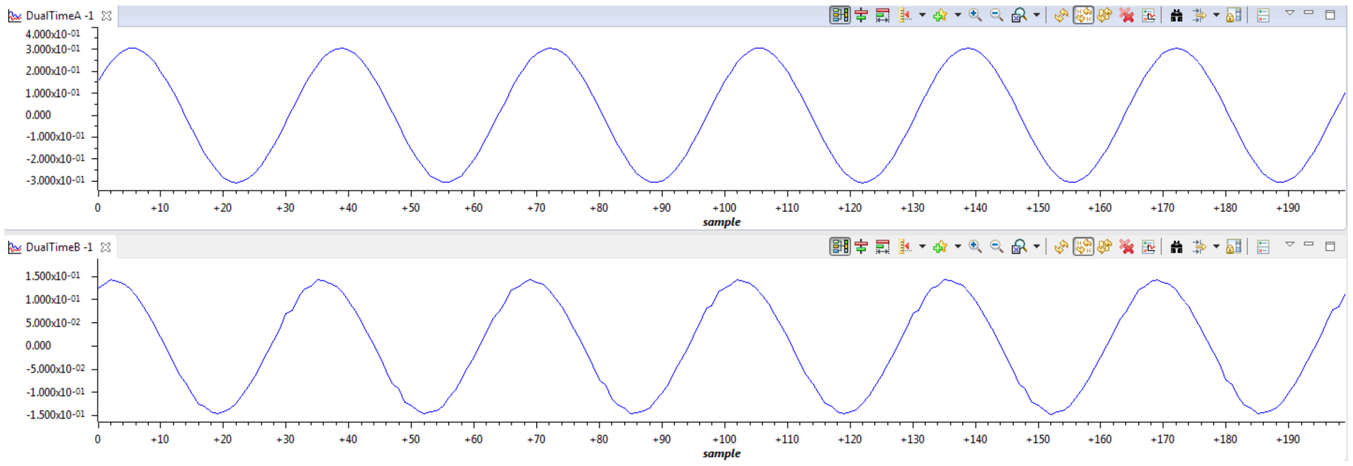


Figure 25. Build Level 1 Graph.1.GraphProp File Showing Measured Per-Unit Voltage and Current Values

15. The AC voltage may be modulated by changing the modulation index through the watch window.
16. The check for this build is now completed.
17. Verify the following items on successful completion of the build.
    a. Inverter modulation scheme and generation of correct AC waveform.
    b. Sensing of voltages, currents, and scalings are correct.
    c. Interrupt generation and execution of the build 1 code in the inverter ISR.
18. To power down, set *invModIndex* to zero.
19. Set *rlyConnect* to zero.
20. Slowly decrease the DC bus voltage to 0 V.
21. The controller can now be halted, and the debug connection is terminated.

NOTE: In case CCS loses connection at any point, bring the system to a safe stop by bringing the input voltage to zero and then disconnecting the JTAG cable and reconnecting.

3.1.2.3.2 Build Level 2—Close Current Loop

In Build 1, the open loop operation of the inverter is verified. In this build, Build 2, the inner current loop is closed. For example, the inductor current is controlled using a current compensator G_i. 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, as shown in Equation 12. The plant for the current compensator is simple to use, and a proportional integral (PI) controller may be used to tune the loop of the inner current.

$$invDutyPU = \frac{(invIiRefInst - invIiInst) \times G_i + invVoInst}{invVbusInst} \tag{12}$$

Figure 26 shows the build level 2 control diagram of the closed current loop.

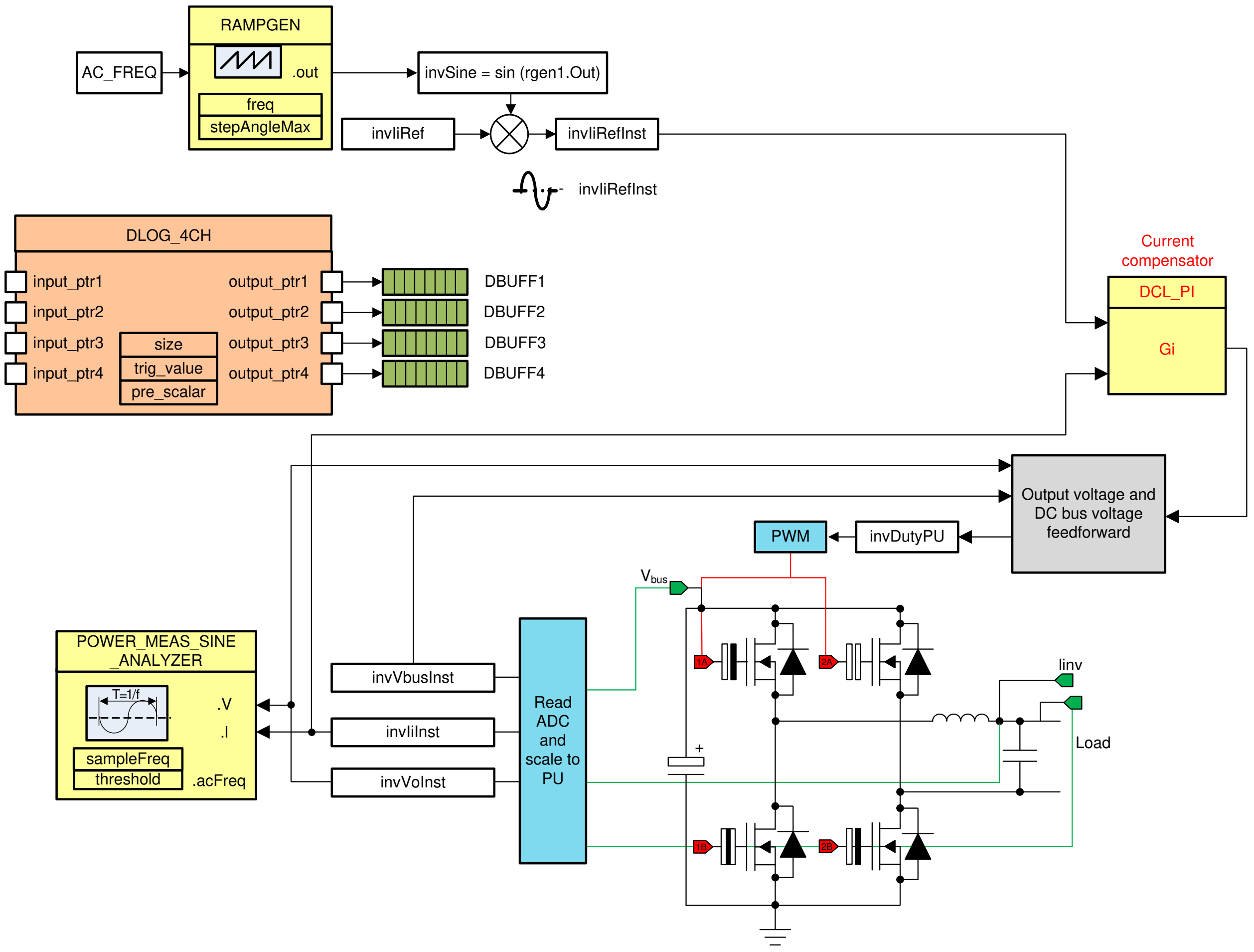


Figure 26. Build Level 2 Control Diagram: Closed Current Loop

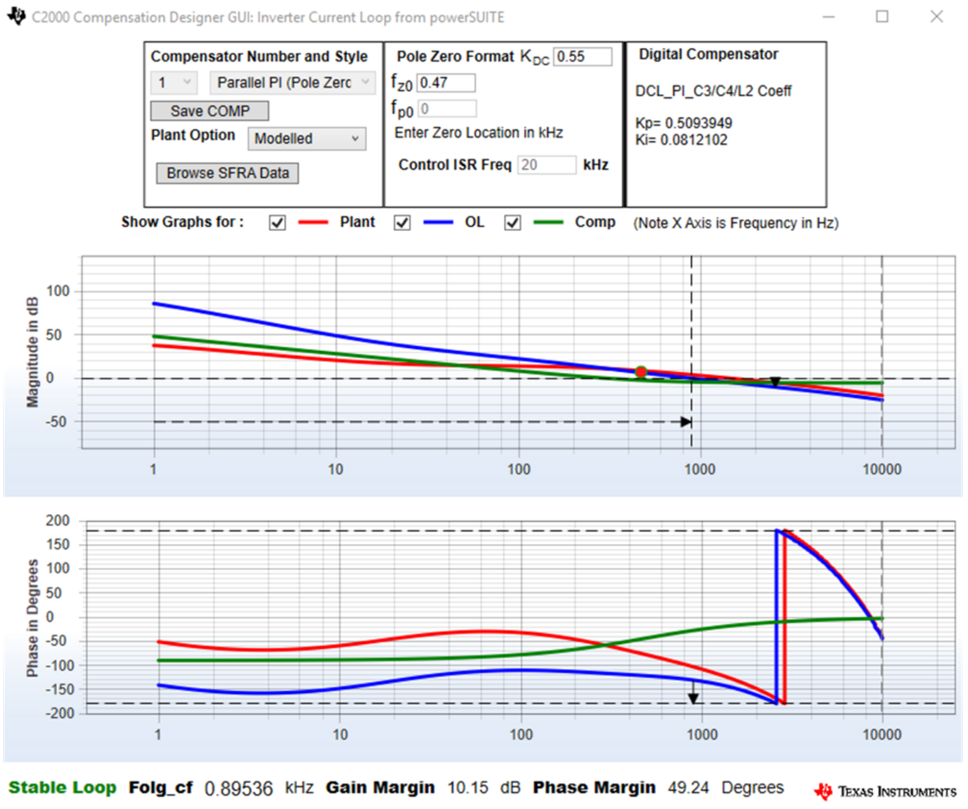
3.1.2.3.2.1 Setting Software Options for Build 2

1. Make sure the hardware is set up as outlined in the previous sections (see Figure 16). Do not supply any high-voltage power yet to the board.
2. powerSUITE settings: On the powerSUITE page select under "Project Options" section:
   - "Closed Current Loop" for the build level
   - "DC" for the output
   - Either "SDFM" or "ADC" for sensing method depending on what is on the design (On this design, both SDFM and ADC sensing methods are present).

   Save the page.
3. Under "Control Loop Design", select Tuning → Current Loop, Comp Number → COMPI, Comp Style → DCL_PI_C3, and click on the Compensation Designer icon.

3.1.2.3.2.2 Designing Current Loop Compensator

Compensation designer launches with the model of the current loop plant for the inverter, and parameters are specified on the powerSUITE page. Compensator values can be changed to ensure stable closed loop operation. Stability of the system when using the designed compensator must be verified by observing the gain and phase margins on the open loop transfer function plot in the compensation designer, as shown in Figure 27.


C2000 Compensation Designer GUI: Inverter Current Loop from powerSUITE

Compensator Number and Style: 1, Parallel PI (Pole Zero
Plant Option: Modelled
Pole Zero Format K_{DC} 0.55
f_{z0} 0.47
f_{p0} 0
Enter Zero Location in kHz
Control ISR Freq 20 kHz
Digital Compensator
DCL_PI_C3/C4/L2 Coeff
Kp= 0.5093949
Ki= 0.0812102
Show Graphs for : ✓ Plant ✓ OL ✓ Comp (Note X Axis is Frequency in Hz)

Stable Loop Folg_cf 0.89536 kHz Gain Margin 10.15 dB Phase Margin 49.24 Degrees

Figure 27. Current Loop Design Using Compensation Designer

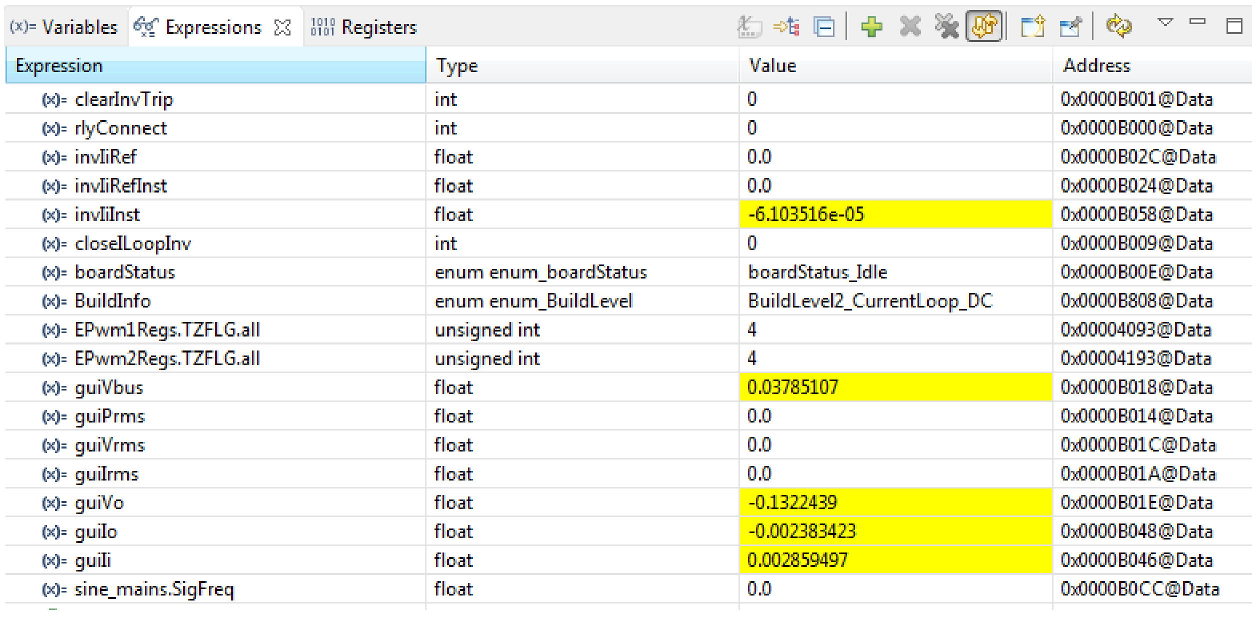
1. Once satisfied with the compensation design, click on *Save Comp*. This option saves the compensator values into the project.

NOTE: If the project was not selected from the solution adapter, changes to the compensator will not be allowed. The user is not able to design their own select solution through the solution adapter.

2. Close the compensation designer.
3. Return to the powerSUITE page.

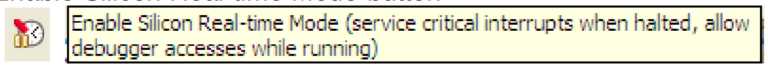
3.1.2.3.2.3 Building and Loading the Project and Setting up Debug

1. Right click on the project name, and click *Rebuild Project*.
2. The project will build successfully.
3. Click *Run* → *Debug* to launch a debugging session.
4. A window to select the CPU may appear.
5. A debug needs to be performed in case of dual CPU devices.
6. Select CPU1.
7. The project will load on the device, and CCS debug view will become active.
8. The code will halt at the start of the main routine.
9. To add variables in the watch and expressions window, click *View* → *Scripting Console* to open the scripting console dialog box.
10. On the upper right corner of the console, click *open* to browse the *setupdebugenv_build2.js* script file located inside the project folder.
11. The watch window will populate with the appropriate variables needed to debug the system.
12. Click on the *Continuos Refresh* button on the watch window to enable a continuous update of values from the controller. Figure 28 shows how the watch window will appear.



Expression	Type	Value	Address
(x)= clearInvTrip	int	0	0x0000B001@Data
(x)= rlyConnect	int	0	0x0000B000@Data
(x)= invIiRef	float	0.0	0x0000B02C@Data
(x)= invIiRefInst	float	0.0	0x0000B024@Data
(x)= invIiInst	float	-6.103516e-05	0x0000B058@Data
(x)= closeILoopInv	int	0	0x0000B009@Data
(x)= boardStatus	enum enum_boardStatus	boardStatus_Idle	0x0000B00E@Data
(x)= BuildInfo	enum enum_BuildLevel	BuildLevel2_CurrentLoop_DC	0x0000B808@Data
(x)= EPwm1Regs.TZFLG.all	unsigned int	4	0x00004093@Data
(x)= EPwm2Regs.TZFLG.all	unsigned int	4	0x00004193@Data
(x)= guiVbus	float	0.03785107	0x0000B018@Data
(x)= guiPrms	float	0.0	0x0000B014@Data
(x)= guiVrms	float	0.0	0x0000B01C@Data
(x)= guiIrms	float	0.0	0x0000B01A@Data
(x)= guiVo	float	-0.1322439	0x0000B01E@Data
(x)= guiIo	float	-0.002383423	0x0000B048@Data
(x)= guiIi	float	0.002859497	0x0000B046@Data
(x)= sine_mains.SigFreq	float	0.0	0x0000B0CC@Data

Figure 28. Build Level 1 Expressions View

13. As this is a DC check, the graph window is not used.
14. Enable real-time mode by hovering the mouse on the buttons on the horizontal toolbar, and click the *Enable Silicon Real-time Mode* button 
Enable Silicon Real-time Mode (service critical interrupts when halted, allow debugger accesses while running).

3.1.2.3.3 Running the Code

1. Run the project by clicking the button.
2. Clear the inverter trip by setting the *clearInvTrip* variable to 1.
3. Set the *rlyConnect* variable to 1 to connect the relay.
4. Set a small current command of 0.02 for *invIiRef*.
5. Slowly raise the DC bus.
6. Observe the *invIiInst* rises slowly, and then regulates at the value set by *invIiRef*.
7. For the 100-Ω output load, the inverter will begin regulating the current once the DC bus is raised about 50 V. Increasing the DC Bus further will not result in increase in the *invIiInst*, which verifies closed loop operation.
8. Once the closed loop operation is verified, for example, *invIirefInst* closely matches *invIiRef*, the DC bus must be raised to the operating voltage, such as 380 V.

NOTE: If the closed loop operation is not verified, the user must reduce the DC bus immediately, and verify Build 1 to see if all the voltage and current sensing parameters are correct. The user must also visit the compensation designer again to verify that the system has enough gain at DC.

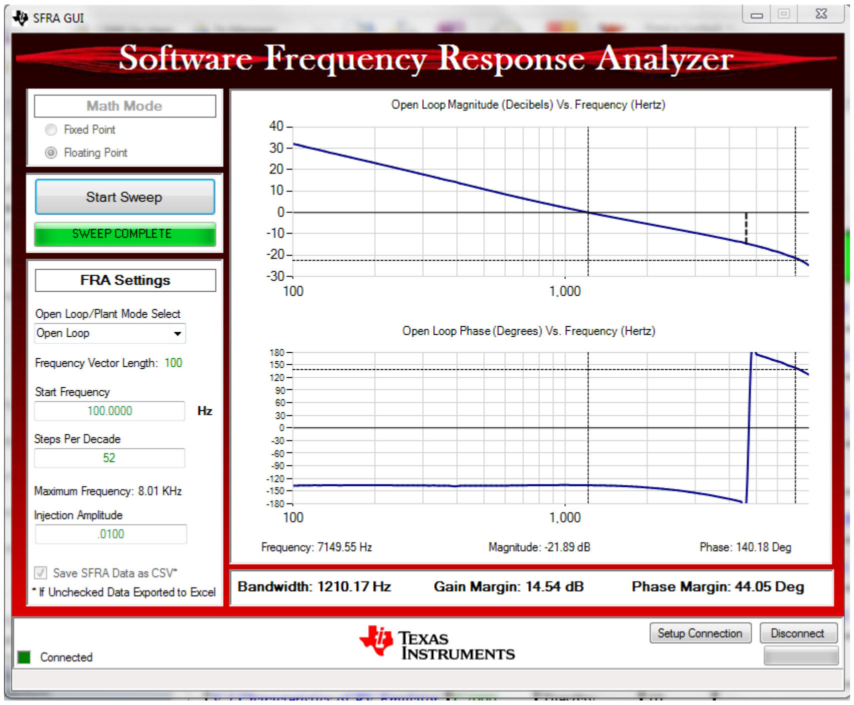
9. After raising the DC bus to 380 V, the invIiRef must be increased in steps of 0.02 pu to 0.08 pu, while continually monitoring closed loop operation (that is, with invIiInst matched invIiRef at every step change, one can monitor these on the oscilloscope as well to verify settling time, overshoot, and so on).
10. This design at 380-V DC input with 0.08-pu current invIiRef set will correspond to approximately 1.3 Amps of current in case of SDFM monitored using a guiIi variable, which with a 100-Ω load, will equal 170 W of power.

NOTE: SFRA is integrated in the software of this build to verify the designed compensator provides enough gain and phase margin by measuring on hardware.

11. To run the SFRA, keep the project running and from the CFG page, click on the SFRA icon.
12. SFRA GUI will appear.
13. Select the options for the device on the GUI; for example, for TMS320F28377D, select *floating point*.
14. Click on *setup connection*.
15. On the pop-up window, *uncheck* the *boot on connect* option.
16. Select an appropriate COM port, and click *OK*.
17. Return to the SFRA GUI.
18. Click *connect*.
19. The SFRA GUI will connect to the device.
20. A SFRA sweep can now be started by clicking on Start Sweep.
21. The complete SFRA sweep will take a few minutes to complete.
22. Activity may be monitored by viewing the progress bar on the SFRA GUI, and also by checking the flashing of the blue LED light on the back of the control card that indicates UART activity.

    If these blue LEDs stop blinking, it indicates a loss of communication. Under this situation, close SFRA GUI, and restart the connection process. If the situation persists, bring the system to a safe stop by setting the invIiRef to 0 and bringing the DC bus voltage down to 0. Terminate the debug session, followed by connecting and reconnecting the JTAG USB cable to reset the JTAG circuit and relaunch a debug session by following the steps outlined previously to rerun the SFRA.

23. Once complete, a graph will open a loop plot, as shown in Figure 29. This verifies that the designed compensator is stable.


SFRA GUI
Software Frequency Response Analyzer
Math Mode
Start Sweep
SWEEP COMPLETE
FRA Settings
Open Loop Magnitude (Decibels) Vs. Frequency (Hertz)
Open Loop Phase (Degrees) Vs. Frequency (Hertz)
Setup Connection
Disconnect
Connected

Fixed Point
[✓ Floating Point]

Open Loop/Plant Mode Select: Open Loop

Frequency Vector Length: 100

Start Frequency: 100.0000 Hz

Steps Per Decade: 52

Maximum Frequency: 8.01 KHz

Injection Amplitude: .0100

[✓ Save SFRA Data as CSV*]

* If Unchecked Data Exported to Excel

Frequency: 7149.55 Hz Magnitude: -21.89 dB Phase: 140.18 Deg

Bandwidth: 1210.17 Hz Gain Margin: 14.54 dB Phase Margin: 44.05 Deg

Figure 29. SFRA Run on Closed Current 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.

The measured gain and phase margin are better than the modeled values, indicating slight differences in modeled and measured response. In this case, the difference is of gain reduction which decreases the bandwidth, but improves the margins. However, depending on the estimate of the parameters of the power stage, the margins may shift either way and it is essential to measure the response to ensure robust operation.

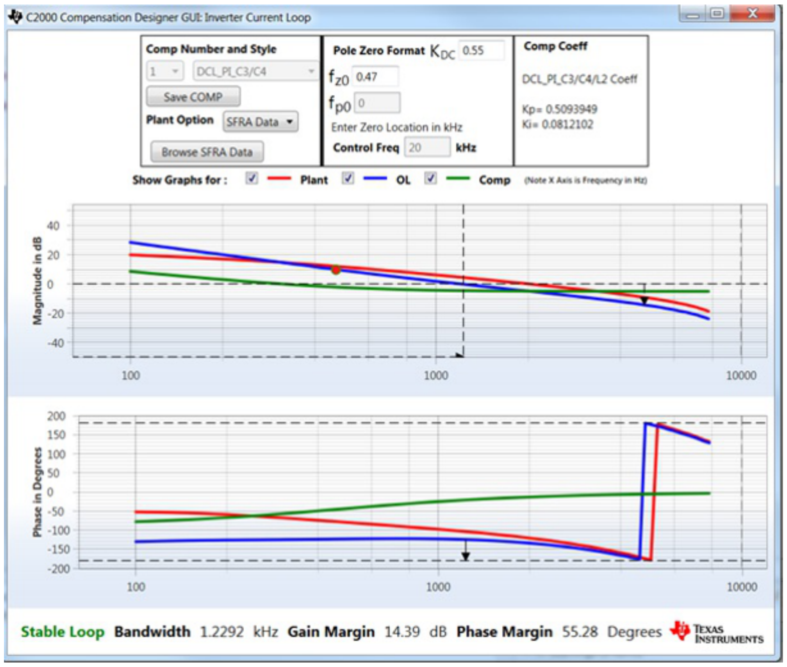
For the voltage source inverter, TI recommends to keep the crossover of the inner current loop at greater than ten times the AC frequency, which is met by this compensator, and no changes are needed in the design. If an adapted solution is not met, the compensator must be changed to ensure the crossover of the current loop meets this requirement.

24. Click on the *Compensation Designer* from the *SYSCFG page*.
25. Choose *SFRA Data* for the plant option on the GUI. This will use the measured plant information to design the compensator. This option must be used to fine tune the compensation.

NOTE: By default, the compensation designer will point to the latest SFRA run.

26. If a previous SFRA run plant information needs to be used, the user must select the *SFRAData.csv* file by browsing to it.
27. Click on *Browse SFRA Data*.
28. Close *Compensation Designer* to return to the *syscfg page*.

Figure 30 shows the compensation designer with measured plant frequency response data.


C2000 Compensation Designer GUI: Inverter Current Loop

Comp Number and Style: 1, DCL_PI_C3/C4
Plant Option: SFRA Data

Pole Zero Format K_{DC} 0.55
f_{z0} 0.47
f_{p0} 0
Enter Zero Location in kHz
Control Freq 20 kHz

Comp Coeff
DCL_PI_C3/C4/L2 Coeff
Kp= 0.5093949
Ki= 0.0812102

Show Graphs for : ✓ Plant ✓ OL ✓ Comp (Note X Axis is Frequency in Hz)

Stable Loop Bandwidth 1.2292 kHz Gain Margin 14.39 dB Phase Margin 55.28 Degrees

Figure 30. Compensation Designer With Measured Plant Frequency Response Data

29. The current compensator design has been verified.
30. Set the *invIiRef* to zero.
31. Reduce the DC bus to zero.
32. Disconnect the relay by setting *rlyConnect* to zero.
33. Fully halting the MCU when in real-time mode is a two step process.
34. First, halt the processor by clicking ⏸ on the toolbar, or by using *Target→Halt*.
35. Take the MCU out of real-time mode by clicking on .
36. Finally, reset the MCU by clicking on .
37. Close the CCS debug session by clicking on *Terminate Debug Session* , or by using *Target → Terminate all*.

3.1.2.3.4 Build 3—Closed Voltage Loop With Inner Current Loop

In this build, the outer voltage loop is closed with the inner current loop (designed in Section 3.1.2.3.2). Multiple resonant controllers compensate for the errors found in the fundamental harmonics, as shown in Figure 27. A lead lag compensator is added to improve the phase margin and is turned through the compensation designer. Figure 31 shows the build level 3 control diagram of the output voltage control with inner current loop.

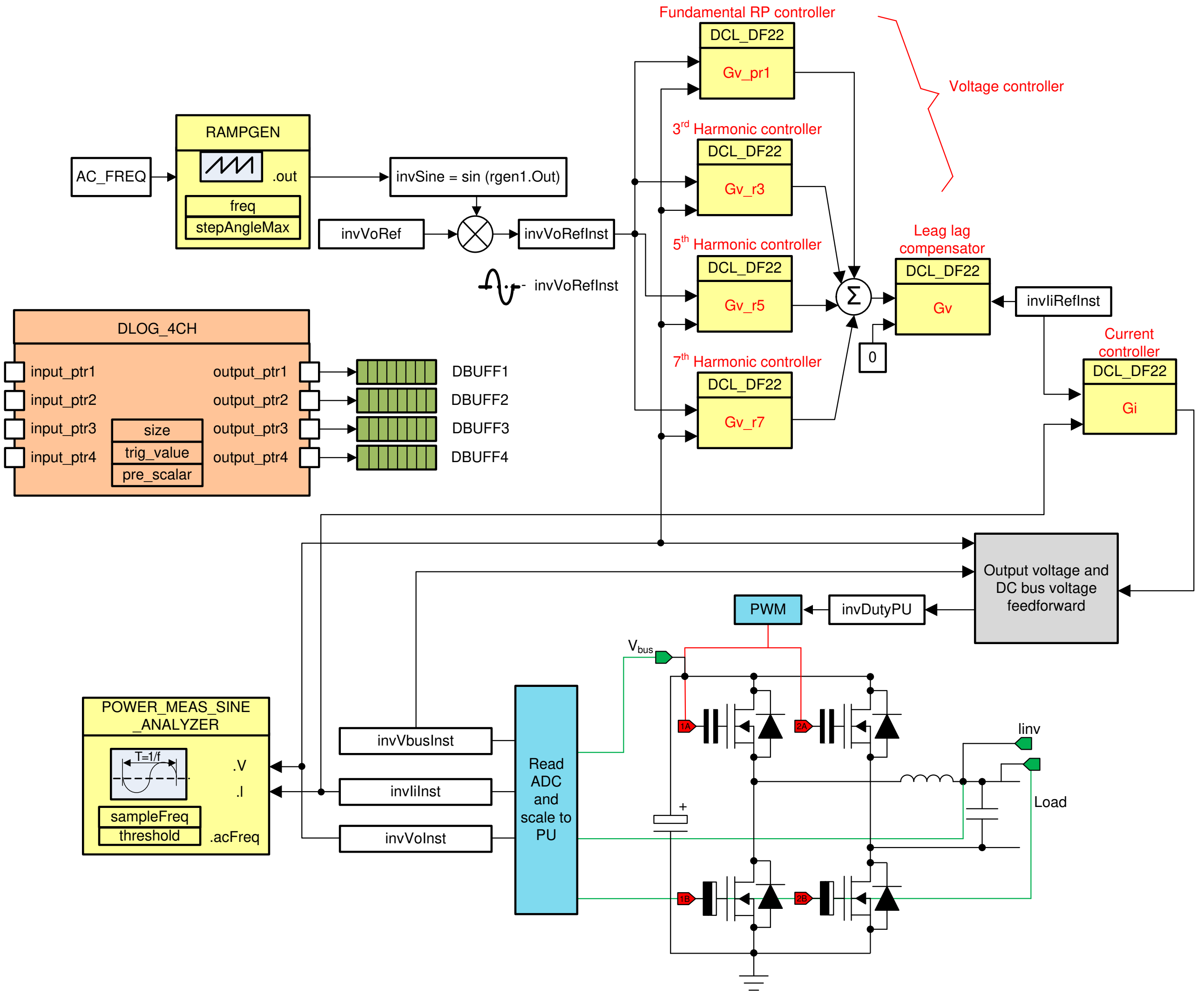


Figure 31. Build Level 3 Control Diagram: Output Voltage Control With Inner Current Loop

3.1.2.3.4.1 Setting Software Options for Build 3

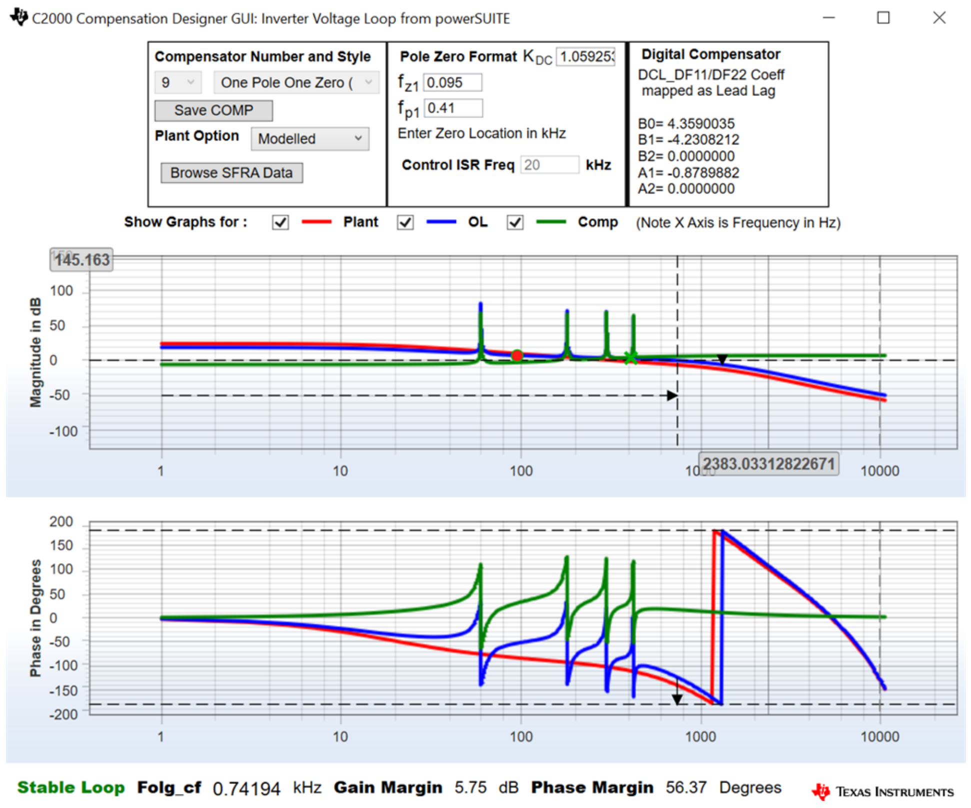
1. Make sure the hardware is set up, as shown in Figure 16, and no power except for the bias is energized.
2. If an adapted solution, modified parameters must be entered as outlined in Section 3.1.2.3.1 for the custom design.
3. Under *Project Options*, select *Closed Voltage and Current Loop*.
4. Select *AC* for output.
5. The user must select sensing to be what is available for the design.
6. Under *Control Loop Design*:
   a. Select *Tuning* as *Voltage Loop*.
   b. Style will be set to *Lead Lag*.
   c. Parameters for the resonant controller must be specified under the *Voltage Loop Resonant Controller* section.
   d. Save the page by entering *Ctrl+S*.
   e. Click on the *Compensation Designer* button .

3.1.2.3.4.2 Designing Voltage Loop Compensator

1. The compensation designer launches with the model of the voltage loop plant for the inverter with the current loop as shown in Figure 32.
2. Note in Build Level 2, the measured gain of the plant was lower than the modeled, hence the margins in this build are lower when using the modeled plant. With the measured response these margins will be larger and hence more stable.

NOTE: The compensation designer implements the PR and harmonic resonant controller, which are entered on the powerSUITE SYSCFG page. This allows tuning of the lead lag compensator to improve the phase margin. To tune the resonant controllers, modify the parameters on the powerSUITE page, and re-invoke the compensation designer.

A lead-lag compensator can be designed using the compensation designed to improve upon the gain and phase margins in the design.


C2000 Compensation Designer GUI: Inverter Voltage Loop from powerSUITE

Compensator Number and Style: 9, One Pole One Zero (

Plant Option: Modelled

Pole Zero Format K_{DC} 1.05925
f_{z1} 0.095
f_{p1} 0.41
Enter Zero Location in kHz
Control ISR Freq 20 kHz

Digital Compensator
DCL_DF11/DF22 Coeff mapped as Lead Lag

B0= 4.3590035
B1= -4.2308212
B2= 0.0000000
A1= -0.8789882
A2= 0.0000000

Show Graphs for : ✓ Plant ✓ OL ✓ Comp (Note X Axis is Frequency in Hz)

Stable Loop Folg_cf 0.74194 kHz Gain Margin 5.75 dB Phase Margin 56.37 Degrees

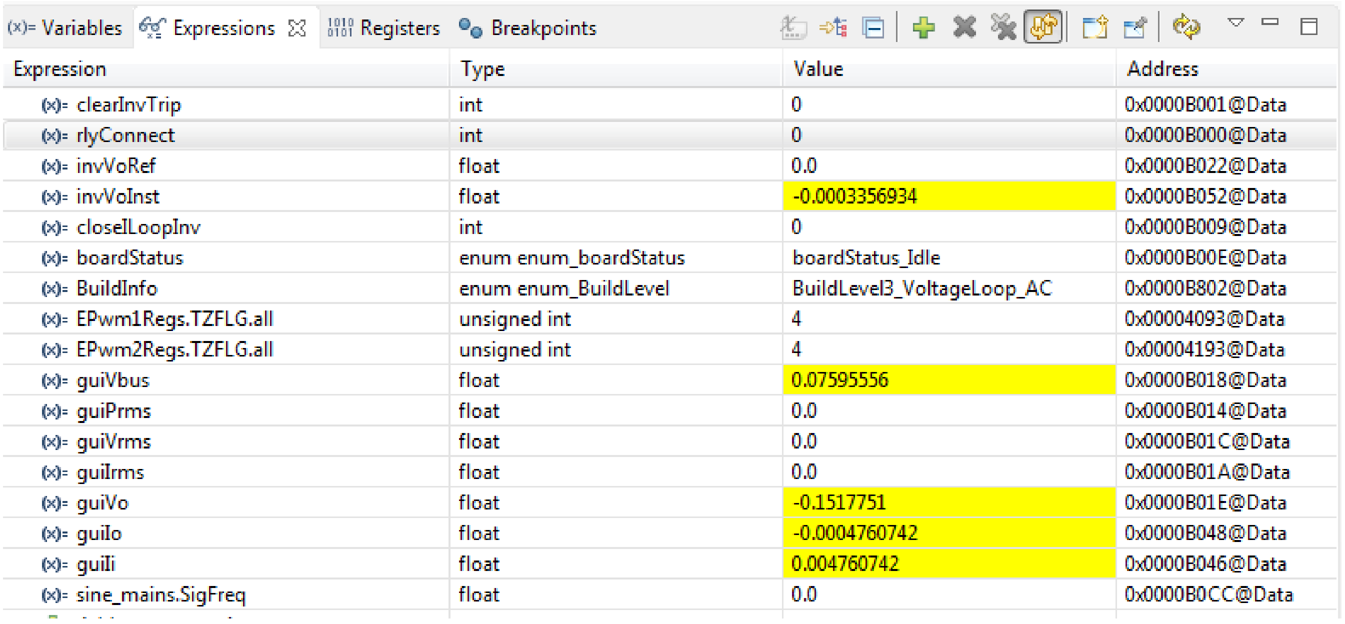
Figure 32. Voltage Loop Lag Compensation Tuning Using Compensation Designer

1. Once satisfied with the compensator design, click *Save Comp*.
2. The compensator values are now saved into the project.
3. Close the compensation designer.
4. Return to the powerSUITE page.
5. Save by typing in *Ctrl+S*.

3.1.2.3.4.3 Building and Loading the Project, and Setting Up Debug

1. Right-click on the project name.
2. Click *Rebuild Project*.
3. The project will build successfully.
4. Click *Run* → *Debug*.
5. A debugging session is launched.
6. A window may appear to select the CPU. Perform a debug for dual CPU devices.
7. Select CPU1.
8. The project loads on the device, and the CCS debug view is active.
9. The code halts at the start of the main routine.
10. To add the variables in the watch and expressions window, click *View* → *Scripting Console* to open the scripting console dialog box.
11. On the upper right corner, click open to browse the *setupdebugenv_build3.js* script file, located inside the project folder.
12. The watch window populates with the appropriate variables needed to debug the system.
13. Click on the *Continuous Refresh* button on the watch window to enable a continuous update if values from the controller.

14. Figure 33 shows how the watch window will appear.



Variables | Expressions | Registers | Breakpoints

Expression	Type	Value	Address
clearInvTrip	int	0	0x0000B001@Data
rlyConnect	int	0	0x0000B000@Data
invVoRef	float	0.0	0x0000B022@Data
invVoInst	float	-0.0003356934	0x0000B052@Data
closeILoopInv	int	0	0x0000B009@Data
boardStatus	enum enum_boardStatus	boardStatus_Idle	0x0000B00E@Data
BuildInfo	enum enum_BuildLevel	BuildLevel3_VoltageLoop_AC	0x0000B802@Data
EPwm1Regs.TZFLG.all	unsigned int	4	0x00004093@Data
EPwm2Regs.TZFLG.all	unsigned int	4	0x00004193@Data
guiVbus	float	0.07595556	0x0000B018@Data
guiPrms	float	0.0	0x0000B014@Data
guiVrms	float	0.0	0x0000B01C@Data
guiIrms	float	0.0	0x0000B01A@Data
guiVo	float	-0.1517751	0x0000B01E@Data
guiIo	float	-0.0004760742	0x0000B048@Data
guiIi	float	0.004760742	0x0000B046@Data
sine_mains.SigFreq	float	0.0	0x0000B0CC@Data

Figure 33. Build Level 1 Expressions View

15. Enable real-time mode by hovering the mouse on the buttons on the horizontal toolbar, and clicking the *Enable Silicon Real-time Mode (service critical interrupts when halted, allow debugger accesses while running)* button.

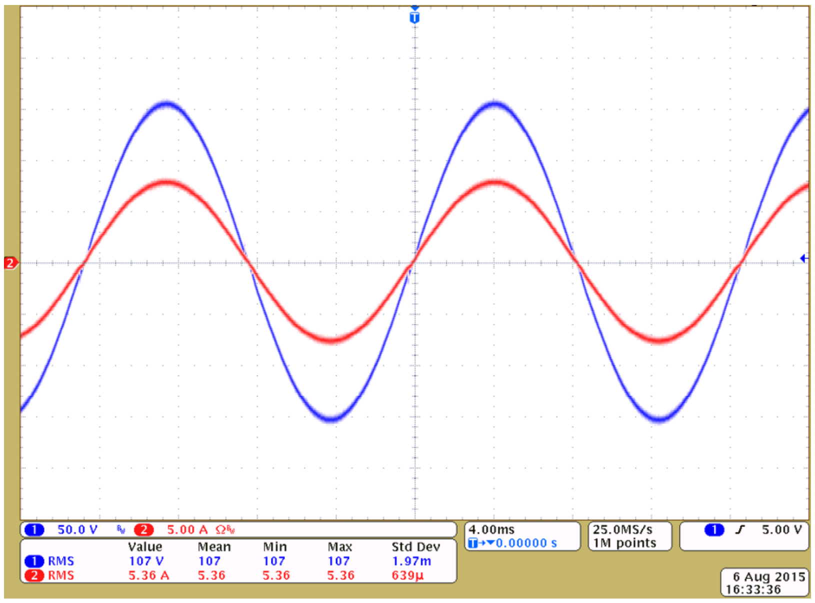
3.1.2.3.4.4 Running the Code

1. Run the project by clicking on .
2. Clear the inverter trip by setting the *clearInvTrip* variable to 1.
3. Set *rlyConnect* to 1 to connect the relay.
4. Set a small current command of 0.02 for *invVoRef*.
5. Slowly raise the DC bus.
6. Observe that the *invVoRef* will appear square in shape to begin with, and as the DC bus is raised, it will take a clean AC waveform shape. Raise the DC bus slowly up to 50 V.
7. If the clean AC waveform shape does not appear, revisit the *compensation design* for the *voltage loop*.

   A quick verification of the closed loop operation can be done by increasing the DC Bus further, but the output voltage AC will regulate at a fixed voltage set at 0.02 by invVoRef. Because this is an AC Build, the expressions window values will constantly change and hence cannot be used to infer on a closed loop operation.
8. Once the closed loop operation is verified, raise the DC bus to the operating voltage to approximately 380 V.
9. Increase the invVoRef to 0.25 gradually in steps of 0.05.
10. Observe a regulated voltage at the output, which will be similar to as shown in Figure 34.

Note in Figure 34 the output is being regulated at full power for this design, and this is not be the exact power level one will reach following the previous steps. Figure 34 is shown as an illustration of full power operation which is at maximum power level.

VAC = 110 V_{RMS}, P_{OUT} = 589 W, THD = 0.36%, and efficiency = 96.9% (blue is voltage, red is current).



	Value	Mean	Min	Max	Std Dev
1 RMS	107 V	107	107	107	1.97m
2 RMS	5.36 A	5.36	5.36	5.36	639µ

Figure 34. Output Voltage Regulated With Linear Load in Build 3

11. The voltage compensator design is verified.
12. Set the *invVoRef* to zero.
13. Reduce the DC bus to zero.
14. Set *rylConnect* to zero to disconnect the relay.
15. Fully halting the MCU when in real-time mode is a two-step process.
16. Halt the processor and clicking the halt button on the toolbar, or by using *Target* → *Halt*.
17. Take the MCU out of real-time mode by clicking the clock button .
18. Reset the MCU by clicking .
19. Close the CCS debug session by clicking *Terminate Debug Session* , or by using *Target* → *Terminate all*.
20. The steps above may be repeated by connecting a non-linear load to the output of the inverter.

3.1.2.3.5 Build 3 Demo Mode

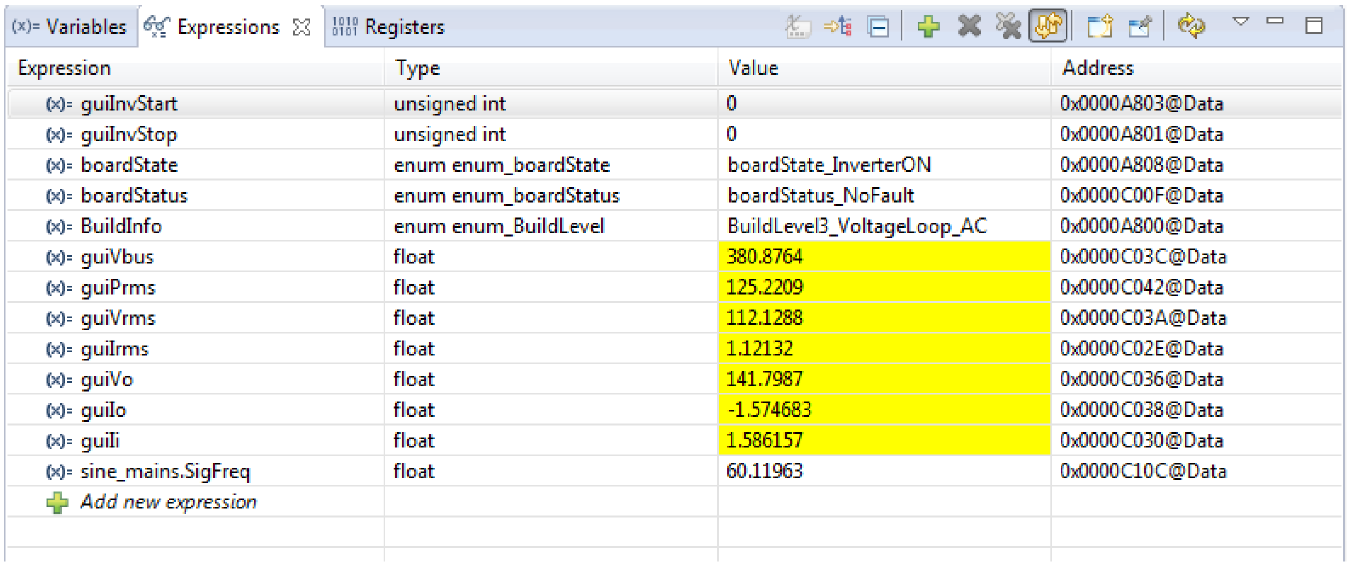
In this build, once the control loops have been verified, start and stop the inverter with variables in the expressions window. Demo mode is also used to run the design out of the box if the exact same design is available. If any modifications have been done, the control loops need to be re-verified as described in Section 3.1.2.3.1, Section 3.1.2.3.2, and Section 3.1.2.3.4. Open the project inside CCS, as outlined in Section 3.1.2.1. Make the following changes and verify the following settings.

3.1.2.3.5.1 Setting Software Options for Build 3 Demo Mode

1. Set up the hardware as outlined in Figure 16.
2. On the powerSUITE page, select *Closed Voltage and Current Loop* under *Project Options*.
3. Select *AC* for *output*.
4. Select *SDFM* for sensing if available on the design.
5. Enter 60 Hz for *frequency* for the AC waveform. This will be the frequency of the inverter output.
6. Under *Inverter Power Stage Parameters*, enter 110 V_{RMS} for the *output voltage*. This will be the value that the AC output will regulate to.
7. Type *Ctrl+S* to save the page.

3.1.2.3.5.2 Building and Loading the Project and Setting up Debug

1. Right-click on the project name.
2. Select *Rebuild Project*.
3. The project will build successfully.
4. Click *Run* → *Debug*, and the debugging session will launch.
5. Select CPU1 if a window appears asking to select a CPU. A debug must be performed in the cause of dual CPU devices.
6. The project will load on the device, and CCS debug view becomes active.
7. The code halts at the start of the main routine.
8. Click *View* → *Scripting Console* to add variables to the watch and expressions window.
9. The scripting console dialog box opens.
10. Click open in the upper right corner of the console to browse the *setupdebugenv_demo.js* script file located inside the project folder.
11. The watch window populates with the appropriate variables required to debug the system.
12. Click the *Continuous Refresh* button on the watch window to enable continuous update of values from the controller.
13. Figure 35 shows how the watch window will appear.



Expression	Type	Value	Address
guiInvStart	unsigned int	0	0x0000A803@Data
guiInvStop	unsigned int	0	0x0000A801@Data
boardState	enum enum_boardState	boardState_InverterON	0x0000A808@Data
boardStatus	enum enum_boardStatus	boardStatus_NoFault	0x0000C00F@Data
BuildInfo	enum enum_BuildLevel	BuildLevel3_VoltageLoop_AC	0x0000A800@Data
guiVbus	float	380.8764	0x0000C03C@Data
guiPrms	float	125.2209	0x0000C042@Data
guiVrms	float	112.1288	0x0000C03A@Data
guiIrms	float	1.12132	0x0000C02E@Data
guiVo	float	141.7987	0x0000C036@Data
guiIo	float	-1.574683	0x0000C038@Data
guiIi	float	1.586157	0x0000C030@Data
sine_mains.SigFreq	float	60.11963	0x0000C10C@Data
Add new expression			

Figure 35. Build Level 3 Demo Mode Expressions View

14. Enable real-time mode by hovering your mouse on the button on the horizontal toolbar, and click the *Enable Silicon Real-time Mode (service critical interrupts when halted, allow debugger accesses while running)* button.

3.1.2.3.5.3 Running the Code

1. Click the green arrow button to run the project.
2. Set *guiInvStart* to 1 to start the inverter (boardStatus will change to checkDCBus).
3. Now slowly raise the DC Bus at the input to be 10% higher than the peak value of the output voltage desired. For example, if 110 V_{RMS} was specified under the Inverter Power Stage Parameters in the previous section, the peak voltage will be 110 × 1.414 = 155.54 and the DC Bus voltage must be set to be 10% higher than this, which is 155.54 × 1.1 = 171 V.
4. The inverter starts as soon as the DC bus voltage is present at a greater level than 10% of the AC maximum.
5. Observe the controlled AC voltage waveform on the output.

NOTE: The frequency and the amplitude of the AC voltage is determined by the values on the powerSUITE page of the solution.

6. If any changes are required, stop the inverter.
7. Set *guiInvStop* to 1.
8. Reduce the input DC bus to zero V.
9. Halt the processor.
10. Disable real-time mode.
11. Return to the powerSUITE page.
12. Make the required changes to the page.
13. Save.
14. Rebuild.
15. Reload and run the project.
16. Set *guiInvStart* to 1 to start the inverter operation.
17. Raise the DC bus to be greater than the maximum voltage of the AC nominal.
18. The inverter now operates with the new setting condition.
19. Never exceed the VA rating of the inverter at any operating point.

NOTE: Fully halting the MCU in real-time mode is a two-step process.

20. Click the halt button on the toolbar , or use *Target* → *Halt* to halt the processor.
21. Click on the clock button to take the MCU out of real-time mode.
22. Rest the MCU by clicking on .
23. Click the *Terminate Debug Session* button , or use *Target* → *Terminate all* to close the CCS debug session.

3.2 Testing and Results

3.2.1 Test Results With Linear Loads

Figure 36 shows that the tests for the linear load are carried out by inserting a resistance at the output of the inverter.

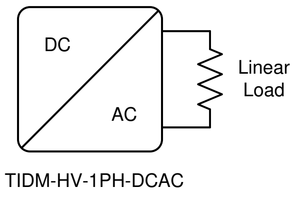


Figure 36. Test Setup for Linear Load

3.2.1.1 Power Stage Efficiency With Linear Load (Same for SDFM or ADC)

Figure 37 shows the efficiency of operating this reference design across power range and with 110 V_{RMS} or 220 V_{RMS} of output.

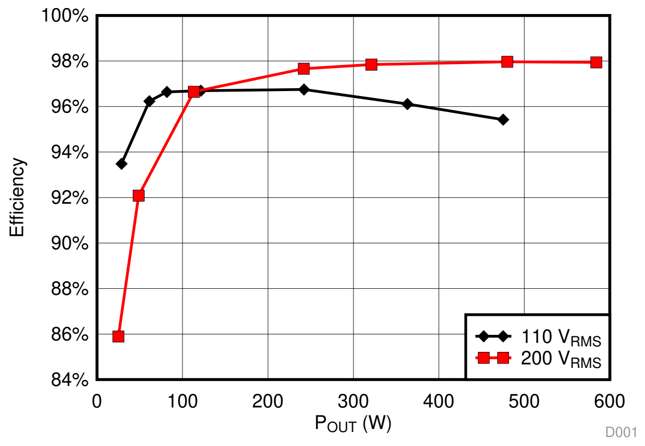
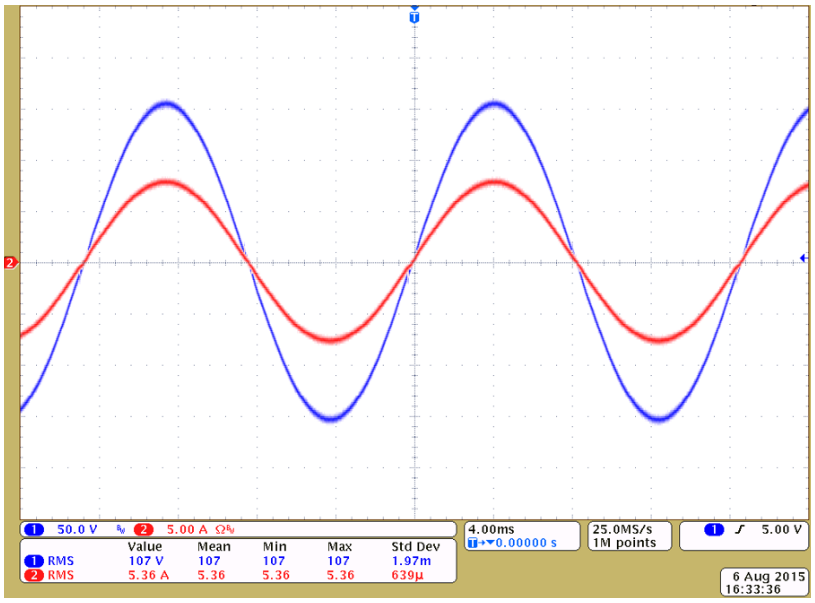


Figure 37. Power Stage Efficiency at 110 V_{RMS} and 220 V_{RMS} of Output

3.2.1.2 Steady State Waveform

Figure 38 and Figure 39 show the current and voltage waveform under a steady state condition at 100 V_{RMS} of output and 220 V_{RMS} of output, respectively.

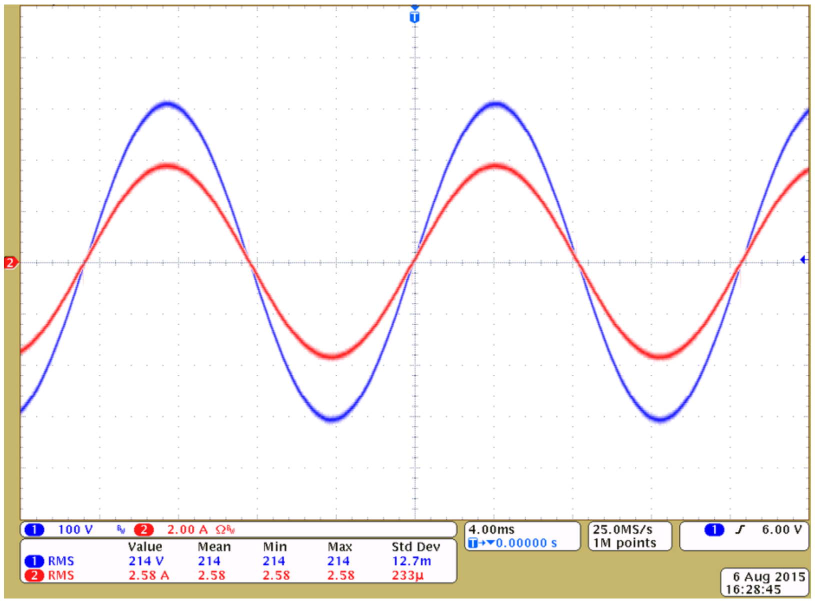
VAC = 110 V_{RMS}, P_{OUT} = 589 W, THD = 0.36%, and efficiency = 96.9% (blue is voltage, red is current).



	Value	Mean	Min	Max	Std Dev
1 RMS	107 V	107	107	107	1.97m
2 RMS	5.36 A	5.36	5.36	5.36	639µ

Figure 38. Steady State Waveforms With the Linear Load Operating at 110 V_{RMS}

VAC = 220 V_{RMS}, P_{OUT} = 552 W, THD = 0.35%, and efficiency = 98% (blue is voltage, red is current).

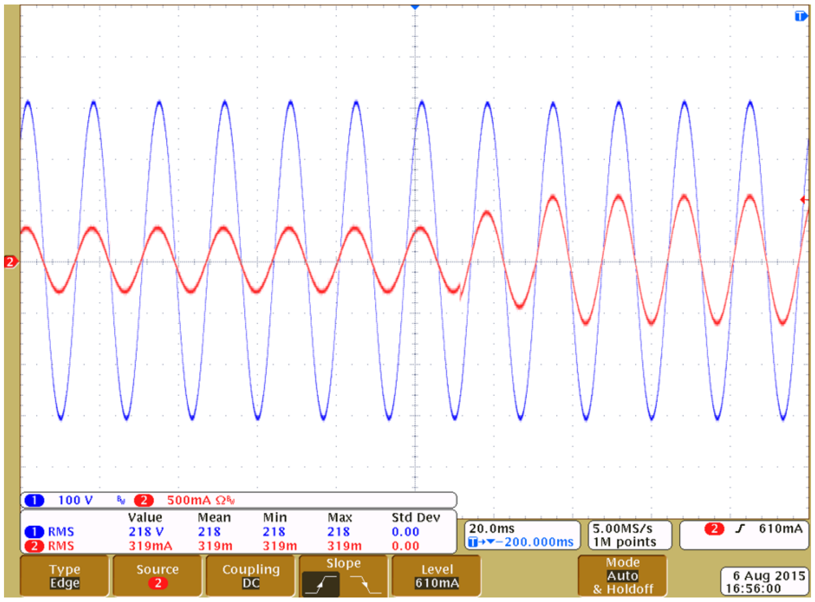


	Value	Mean	Min	Max	Std Dev
1 RMS	214 V	214	214	214	12.7m
2 RMS	2.58 A	2.58	2.58	2.58	233µ

Figure 39. Steady State Waveforms With the Linear Load Operating at 220 V_{RMS}

3.2.1.3 Transient Waveform With Step Change in Load

Figure 40 shows the voltage waveform when a step change in load is applied, illustrating good output voltage regulation (blue is voltage, red is current).



	Value	Mean	Min	Max	Std Dev
1 RMS	218 V	218	218	218	0.00
2 RMS	319mA	319m	319m	319m	0.00

Figure 40. Voltage Waveform With Transient Change in Load

3.2.1.4 Total Harmonic Distortion With Linear Loads Using SDFM-Based Sensing

Figure 41 shows total harmonic distortion with the linear load at the output, and the output voltage controlled at 110 V_{RMS} and 220 V_{RMS} at different power levels with SDF sensing used for control.

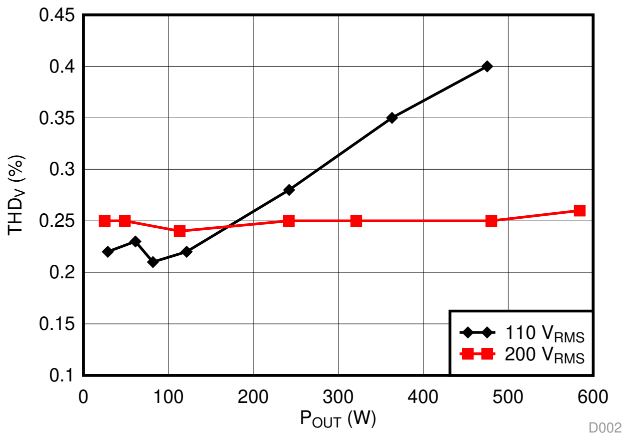


Figure 41. Total Harmonic Distortion With Linear Loads Using SDFM Sensing

3.2.1.5 Total Harmonic Distortion Using ADC-Based Sensing

Figure 42 shows total harmonic distortion with the linear load at the output, and the output controlled at 110 V_{RMS} and 220 V_{RMS} at different power levels with ADC sensing used for control.

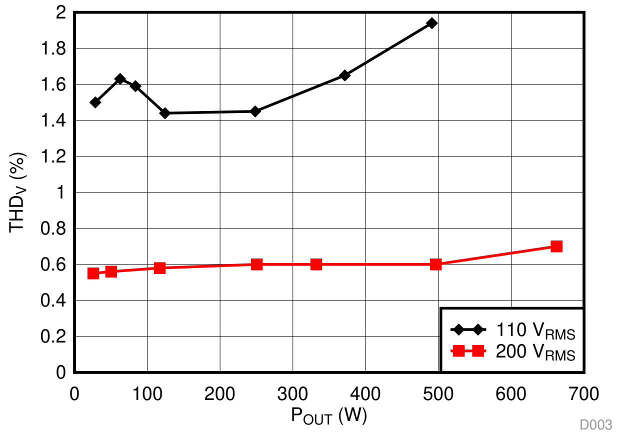


Figure 42. Total Harmonic Distortion Using ADC-Based Sensing

3.2.1.6 Test Data Table With SDFM-Based Sensing

Table 5 and Table 6 shows the complete test data at 110 V_{RMS} of output voltage and 220 V_{RMS} of output voltage, respectively.

Table 5. Test Data at 110-V_{RMS} V_{OUT} and SDFM-Based Sensing

V_{IN}	I_{IN}	V_{OUT}	I_{OUT}	P_{IN}	P_{OUT}	THDv	EFFICIENCY
382.8	0.08	109.93	0.2604	30.624	28.628	0.22	0.934822362
382.7	0.166	109.89	0.5563	63.5282	61.135	0.23	0.962328541
382.6	0.221	109.83	0.7439	84.5546	81.714	0.21	0.966405139
382.8	0.328	109.75	1.106	125.5584	121.41	0.22	0.966960395
382.6	0.654	109.47	2.2112	250.2204	242.08	0.28	0.967467081
382.8	0.987	109.49	3.3156	377.8236	363.12	0.35	0.961083426
382.8	1.301	108.86	4.3628	498.0228	475.23	0.4	0.954233421

Table 6. Test Data at 220-V_{RMS} V_{OUT} and SDFM-Based Sensing

V_{IN}	I_{IN}	V_{OUT}	I_{OUT}	P_{IN}	P_{OUT}	THDv	EFFICIENCY
382.6	0.076	218.74	0.1209	29.0776	24.976	0.25	0.858942966
382.8	0.138	218.71	0.2246	52.8264	48.646	0.25	0.920865325
382.8	0.306	218.61	0.5178	117.1368	113.21	0.24	0.966476803
382.7	0.647	218.5	1.1066	247.6069	241.82	0.25	0.97662868
382.8	0.857	218.38	1.4695	328.0596	320.99	0.25	0.978450257
382.8	1.28	218.16	2.2003	489.984	480.03	0.25	0.979685051
382.8	1.558	219.44	2.6615	596.4024	584.12	0.26	0.979405851

3.2.1.7 Test Data Table With ADC

Table 7 and Table 8 show complete test data at 110 V_{RMS} of output voltage and 220 V_{RMS} of output voltage, respectively.

Table 7. Test Data at 110-V_{RMS} V_{OUT} and ADC-Based Sensing

V_{IN}	I_{IN}	V_{OUT}	I_{OUT}	P_{IN}	P_{OUT}	THDv	EFFICIENCY
382.8	0.08	109.93	0.2604	30.624	28.628	1.5	0.934822362
382.8	0.1705	111.2	0.563	65.2674	62.621	1.63	0.959452958
382.6	0.226	111.13	0.7528	86.4676	83.669	1.59	0.96763412
382.8	0.335	111.04	1.1195	128.238	124.3	1.44	0.969291474
382.6	0.67	111.8	2.2393	256.342	248.15	1.45	0.968042693
382.8	1.007	110.69	3.3543	385.4796	371.38	1.65	0.963423227
382.8	1.34	110.64	4.4367	512.952	490.96	1.94	0.957126593

Table 8. Test Data at 220-V_{RMS} V_{OUT} and ADC-Based Sensing

V_{IN}	I_{IN}	V_{OUT}	I_{OUT}	P_{IN}	P_{OUT}	THDv	EFFICIENCY
382.6	0.079	222.46	0.1229	30.2254	25.832	0.55	0.854645431
382.8	0.143	222.42	0.2286	54.7404	50.339	0.56	0.919595034
382.8	0.318	222.35	0.5268	121.7304	117.14	0.58	0.962290439
382.8	0.67	222.27	1.1262	256.476	250.38	0.6	0.976231694
382.8	0.886	222.13	1.4944	339.1608	332.04	0.6	0.979004649
382.8	1.323	221.9	2.237	506.4444	496.42	0.6	0.980206317
382.8	1.767	221.7	2.9867	676.4076	662.19	0.7	0.978980721

3.2.2 Test Results With Non-Linear Loads

Figure 43 shows the test setup for the measurement of performance under non-linear loads.

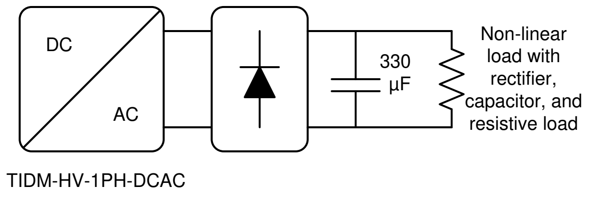
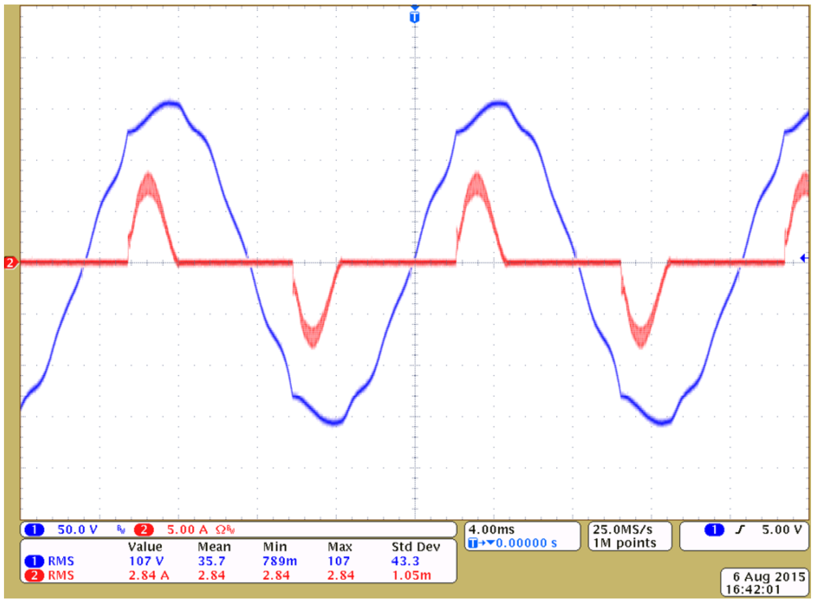

DC
AC
330 µF
Non-linear load with rectifier, capacitor, and resistive load
TIDM-HV-1PH-DCAC

Figure 43. Non-Linear Load Test Setup

3.2.2.1 Waveform With SDFM

Figure 44 shows the output voltage and the current of the inverter when a non-linear load featuring a rectifier and a 330-µF capacitor and the resistive load is connected at the output and the SDFM is used for control.

VAC = 110 V_{RMS}, P_{OUT} = 312 W, THD = 2.9%, efficiency = 94%, and PF of load = 0.64% (blue is voltage, red is current).

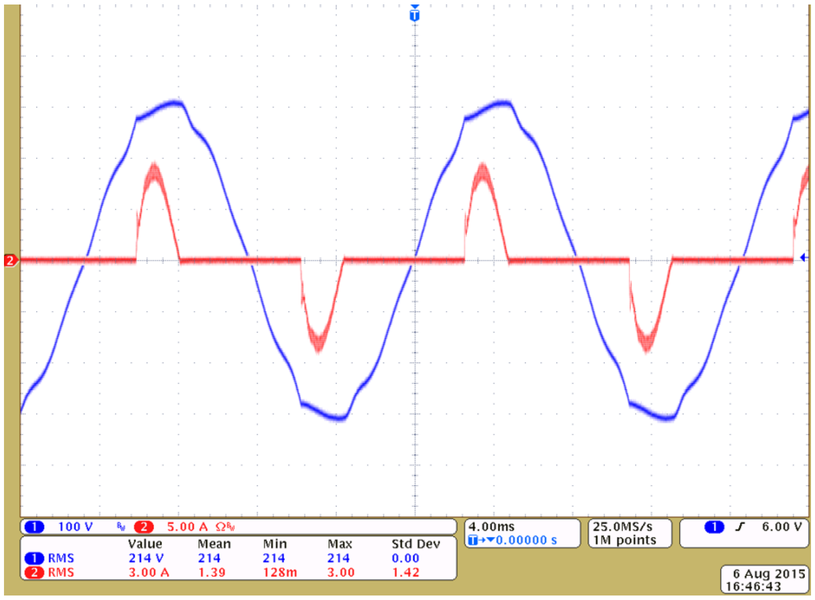


	Value	Mean	Min	Max	Std Dev
1 RMS	107 V	35.7	789m	107	43.3
2 RMS	2.84 A	2.84	2.84	2.84	1.05m

Figure 44. Current and Voltage Waveform Under a Non-Linear Load With SDFM Sensing at 110 V_{RMS}

Figure 45 shows the output voltage and current of the inverter when a non-linear load featuring a rectifier and a 300-µF capacitor and the resistive load is connected at the output, and the ADC is used for control.

VAC = 220 V_{RMS}, P_{OUT} = 575 W, THD = 2.6%, efficiency = 94.5%, and PF of load = 0.64 (blue is voltage, red is current).



	Value	Mean	Min	Max	Std Dev
1 RMS	214 V	214	214	214	0.00
2 RMS	3.00 A	1.39	128m	3.00	1.42

Figure 45. Current and Voltage Waveform Under a Non-Linear Load With SDFM Sensing at 220 V_{RMS}

3.2.2.2 Waveform With ADC

Figure 46 shows the output voltage and current of the inverter when a non-linear load featuring a rectifier and a 330-µF capacitor and the resistive load is connected at the output, and the ADC is used for control at 110 V_{RMS} of output.

VAC = 100 V_{RMS}, P_{OUT} = 209 W, THD = 3.856%, efficiency = 94.7%, and PF of load = 0.64 (blue is voltage, red is current).

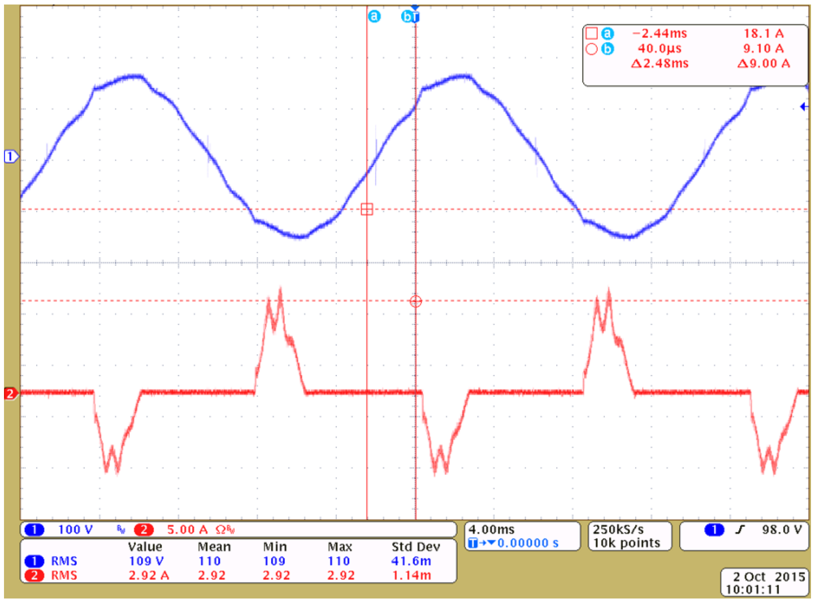


Figure 46. Current and Voltage Waveform Under a Non-Linear Load With ADC Sensing at 110 V_{RMS}

Figure 47 shows the output voltage and current of the inverter when a non-linear load featuring a rectifier and a 330-µF capacitor and the resistive load is connected at the output and ADC is used for control at 220 V_{RMS} of output.

VAC = 220 V_{RMS}, P_{OUT} = 390 W, THD = 3.1%, efficiency = 95.9%, and PF of load = 0.64 (blue is voltage, red is current).

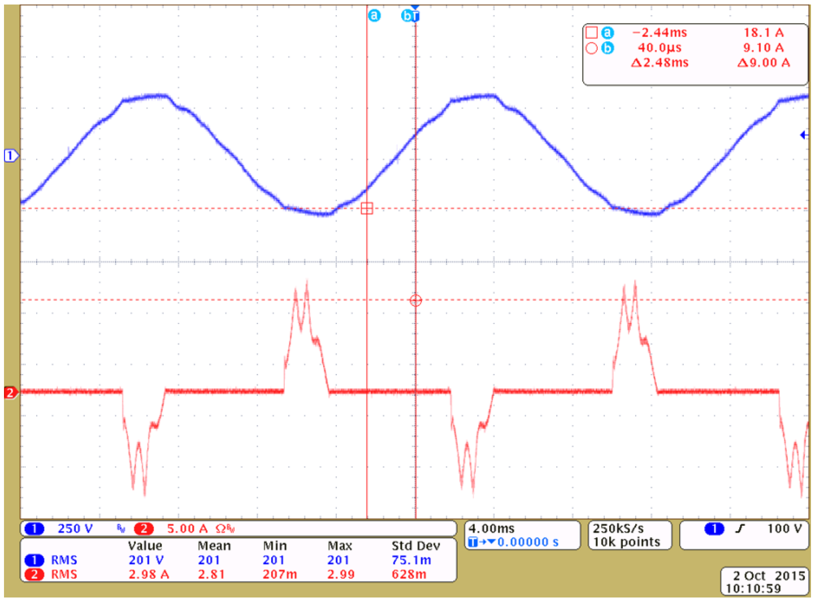


Figure 47. Current and Voltage Waveform Under a Non-Linear Load With ADC Sensing at 220 V_{RMS}

4 Design Files

For schematics, bill of materials (BOM), altium project, and Gerber files, see the design files at http://www.ti.com/tool/TIDM-HV-1PH-DCAC or under the DigitalPower SDK package at *C2000Ware_DigitalPower_SDK_<version>/solutions/tidm_hv_1ph_dcac*.

5 Software Files

To download the software files for this reference design, see the link at http://www.ti.com/tool/C2000WARE-DIGITALPOWER-SDK. This reference design can be found at http://www.ti.com/tool/TIDM-HV-1PH-DCAC.

6 Related Documentation

1. C. Loh, M. Newman, D. Zmood and D. Holmes, "A comparative analysis of multiloop voltage regulation strategies for single and three-phase UPS systems", *IEEE Transactions on Power Electronics*, vol. 18, no. 5, pp. 1176—1185, 2003
2. J. Guerrero, L. Garcia De Vicuna, J. Matas, M. Castilla and J. Miret, "Output Impedance Design of Parallel-Connected UPS", *IEEE Transaction on Industrial Electronics*, vol. 52, no. 4, pp. 1126—1135, 2005
3. Texas Instruments, *C2000™ Software Frequency Response Analyzer (SFRA) Library and Compensation Designer User's Guide*
4. Texas Instruments, *TMS320F2837xD Dual-Core Delfino™ Microcontrollers Data Sheet*
5. Texas Instruments, *TMS320F28004x Piccolo™ Microcontrollers Data Sheet*

6.1 Trademarks

7 About the Author

MANISH BHARDWAJ is a systems application engineer with the 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 and Bachelor of Engineering from Netaji Subhash Institute of Technology, University of Delhi, India.

Revision History

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

Changes from D Revision (May 2019) to E Revision	Page
• Changed powerSUITE Page for Voltage Source Inverter Solution image	19
• Changed Project Explorer View of the Solution Project image	20
• Changed Current Loop Design Using Compensation Designer image	28
• Changed Voltage Loop Lag Compensation Tuning Using Compensation Designer image	35

Changes from C Revision (July 2018) to D Revision	Page
• Changed part number from TMS3420F280049M to TMS320F280049C	1
• Changed http://www.ti.com/product/amc1304 hyperlink with http://www.ti.com/product/AMC1304M25	1
• Changed part number from TMS320F280049M to TMS320F280049C	1
• Changed TMS320F280049M to TMS320F28377D	5
• Changed TMS320F280049M to TMDSCNCD280049C	17
• Changed TMS320F280049M to TMDSCNCD280049C	17
• Changed TMS320F280049M to TMDSCNCD280049C	17
• Changed TMS320F280049M to TMDSCNCD280049C	18

Changes from B Revision (April 2017) to C Revision	Page
• Updated formatting to fit current design guide template	1
• Added C2000Ware DigitalPower SDK to *Resources*	1
• Added "TMS320F28377D and TMS320F280049M" to *Features*	1
• Changed location of Excel sheet to sdk_install_path_<version>\solutions\tidm_hv_1ph_dcac\hardware\baseboard\calculation.xlsx	6
• Updated the equation featured in Figure 4: *AC Output Voltage Differential Sensing Using Resistor Divider and Op Amp*	7
• Added note in Section 3.1.2: *Firmware: powerSUITE and Incremental Build Software*	18
• Changed Step 5 in Section 3.1.2.1: *Opening the Project Inside Code Composer Studio™*	18
• Added *guiIo* to Step 2 in Section 3.1.2.3.1.5: *Running the Code*	25
• Changed the destination for the Compensation Designer icon in Step 3 of Section 3.1.2.3.2.1: *Setting Software Options for Build 2*	28
• Changed image of Figure 30: *Compensation Designer With Measured Plant Frequency Response Data*	32
• Changed location of DigitalPower SDK package in Section 4: *Design Files*	47
• Changed the link to the software files	47
• Added the *TMS320F28004x Piccolo™ Microcontrollers Data Sheet* to Section 6: *Related Documentation*	47

Changes from A Revision (September 2016) to B Revision	Page
• Deleted third paragraph in Section 3.2: *Voltage and Current Sensing*	6
• Changed Figure 13: *Control Diagram for Voltage Source Inverter*	12

Changes from Original (November 2015) to A Revision	Page
• Changed output voltage in Table 1: *Key System Specifications* to "110 V_{RMS} or 200 V_{RMS}"	4
• Added efficiency for 110 V_{RMS} in Table 1: *Key System Specifications*	4
• Deleted "These can also be modified from the powerSUITE page for an adapted solution." from Section 3.2: *Voltage and Current Sensing*	6
• Added introduction to link in Section 3.4: *Inductor Design*	13

- Added TIEVM-HV-1PH-DCAC note in Section 4: *Getting Started With Hardware* ... 14
- Added L2N slot in Step 2 in Section 4.1: *Base Board Settings* ... 14
- Changed control card input from H5 to J15-J16 in Step 4 in Section 4.1: *Base Board Settings* ... 14
- Changed jumper insert spot from JP2 to J10 in Step 5 in Section 4.1: *Base Board Settings* ... 14
- Deleted flipping the switch S1 step ... 14
- Added "Insert a jumper at J4 if not already populated." to Step 7 in Section 4.1: *Base Board Settings* ... 14
- Changed DC source connection from CON1 to J17 in Step 9 in Section 4.1: *Base Board Settings* ... 14
- Changed resistive load value from 200 Ω to 100 Ω in Step 10 in Section 4.1: *Base Board Settings* ... 14
- Changed connector name from CON1 to J17 in Table 4: *Key Connectors and Their Function* ... 16
- Changed connector name from S1 to J4 in Table 4: *Key Connectors and Their Function* ... 16
- Changed function from "Switch to connect and disconnect the DC bias of the board" to "Can be used to disconnect bias power" in Table 4: *Key Connectors and Their Function* ... 16
- Changed connector name from H5 to J15-J16 in Table 4: *Key Connectors and Their Function* ... 16
- Changed connector name from JP2 to J10 in Table 4: *Key Connectors and Their Function* ... 16
- Added Section 4.3: *Tips to Connect JTAG USB Cable* ... 17
- Changed Step 14 in Section 5.3.1: *Build Level 1, Open Loop* ... 25
- Added paragraph before Figure 25: *Build Level 1 Graph.1.GraphProp File Showing Measured Per-Unit Voltage and Current Values* ... 26
- Added ending ordered list to previous ordered list in Section 5.3.1: *Build Level 1, Open Loop* ... 26
- Added note at the end of Section 5.3.1: *Build Level 1, Open Loop* ... 26
- Added Equation 12 ... 27
- Added setting software options for Build 2 to Section 5.3.2: *BUILD Level 2, Close Current Loop* ... 28
- Changed from invliRef to invIiInst in Step 6 in Section 5.3.2: *BUILD Level 2, Close Current Loop* ... 30
- Added "Increasing the DC Bus further will not result in increase in the *invIiInst*, which verifies closed loop operation." to Step 7 in Section 5.3.2: *BUILD Level 2, Close Current Loop* ... 30
- Added "at 380V DC input with 0.08-pu current invIiRef set" and "in case of SDFM monitored using guili variable" to Step 2 in Section 5.3.2: *BUILD Level 2, Close Current Loop* ... 30
- Added paragraph to Step 14 in Section 5.3.2: *BUILD Level 2, Close Current Loop* ... 30
- Deleted "By default, the compensation designer will point to the latest SFRA run." from note ... 31
- Deleted "Inst" from *invliRefInst* ... 32
- Changed Step 2 of *Designing Voltage Loop Compensator* ... 34
- Added "Raise the DC Bus slowly up to 50 V. " to Step 6 in *Running the Code* ... 36
- Added paragraph to Step 6 in *Running the Code* ... 36
- Changed VAC from 100 V_{RMS} to 110 V_{RMS} ... 37
- Changed Step 3 of *Running the Code* ... 39