Design Guide: TIDM-02002

Bidirectional CLLLC Resonant Dual Active Bridge (DAB) Reference Design for HEV/EV Onboard Charger

Description

CLLLC resonant DAB with bidirectional power flow capability and soft switching characteristics is an ideal candidate for Hybrid Electric Vehicle/Electric Vehicle (HEV/EV) on-board chargers and energy storage applications. This design illustrates control of this power topology using a C2000™ MCU in closed voltage and closed current-loop mode. The hardware and software available with this design help accelerate your time to market.

Resources

TIDM-02002	Design Folder
TMS320F280049	Product Folder
UCC21530-Q1	Product Folder
AMC1311-Q1	Product Folder
AMC1302-Q1	Product Folder
LMV116	Product Folder
OPA320	Product Folder
C2000WARE-DIGITAL-POWERSDK	Software Folder
PMP21553	Design Folder
PMP21561	Design Folder
PMP21495	Design Folder
TMDSCNCD280049C	Tool Folder

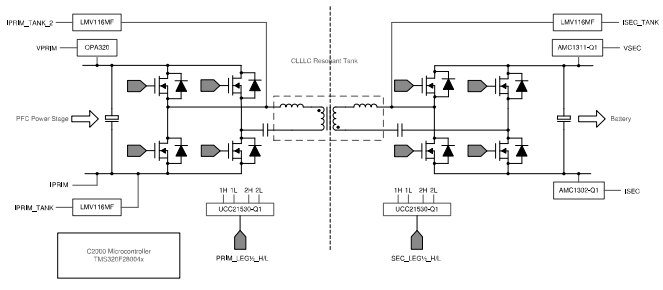
Search Our E2E™ support forums

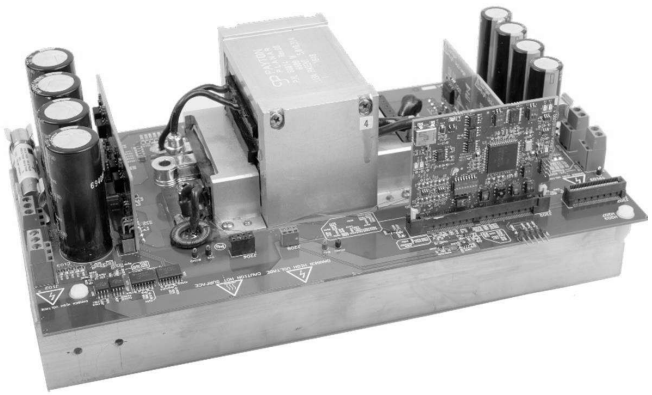
Features

- Vprim: 380–600 V DC; Vsec: 280–450 V DC
- Power Max: 6.6 kW, 98% peak efficiency
- CLLLC resonant tank with 500-kHz nominal PWM switching (300 kHz–700 kHz range) enables higher power density
- Soft switching with Zero Voltage Switching (ZVS) on the primary; Zero Current Switching (ZCS) and ZVS on the secondary enable higher efficiency
- Active synchronous rectification scheme implementation using Rogowski coil sensor enables higher efficiency
- Software Frequency Response Analyzer (SFRA) and Compensation Designer for ease of tuning of control loops
- Software support for TMS320F28004x device with the Control Law Accelerator (CLA), which enables integrated OBC design with AC-DC and DC-DC controlled using a single C2000 MCU

Applications

- On-board chargers for EV
- Off-board chargers
- Grid storage

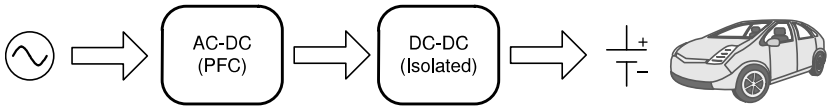




1 System Description

On-board chargers (OBCs) are an essential part of Electric Vehicles (EVs) and Hybrid Electric Vehicles (HEV). An OBC typically consists of an AC-DC [power factor correction (PFC) rectifier stage] and an isolated DC-DC converter, as shown in Figure 1. C2000 MCUs are designed to implement advanced digital power control that automotive applications demand; for more information, see C2000 Digital Power and C2000 EV.

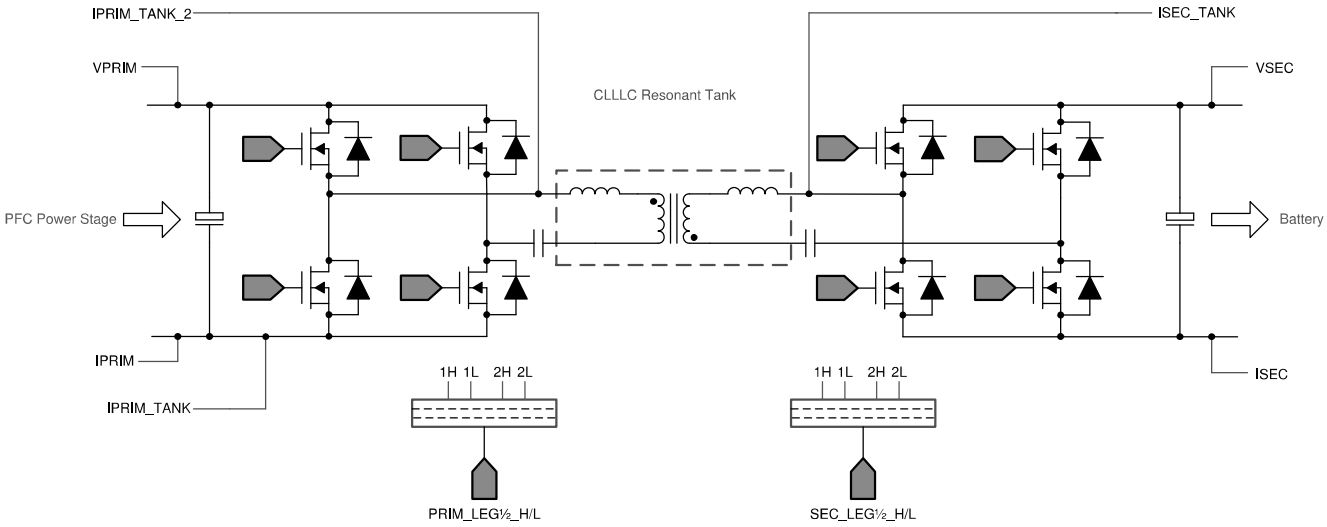
Figure 1. Typical OBC Architecture



The ability to charge the battery fully overnight is highly desired for most EV Level 1 and Level 2 chargers. With battery capacity increasing, the OBCs need to be designed for even higher power. With the increasing power capacity of the OBC, specifications such as power density and efficiency are even more important, due to limited space and cooling capacity in the car.

The CLLLC (Capacitor-Inductor-Inductor-Inductor-Capacitor)—with its symmetric tank, soft switching characteristics, and ability to switch at higher frequencies—is a good choice for these applications. In this design, control and implementation of a CLLLC topology, as shown in Figure 2, is illustrated.

Figure 2. CLLLC Topology for Isolated DC-DC Converter



The nomenclature for Figure 2 is as follows:

VPRIM	Primary side voltage (typically comes from a PFC converter)
IPRIM	Return current of the primary side, can be used for protection and monitoring.
IPRIM_TANK, IPRIM_TANK_2	Tank current on the primary side, two methods to sense using shunt current sense and other is rogowski's coil. Only one is needed, used to implement synchronous rectification in the reverse direction i.e. secondary to primary. Also used for protection.
VSEC	Secondary side voltage (typically, a battery)

ISEC	Return current of the secondary side, used to implement the battery current control loop.
ISEC_TANK	Tank current on the secondary side, used to implement the synchronous rectification for the foward direction power flow i.e. primary to secondary.
PRIM_LEG1/2_H/L	PWMs for the primary side full bridge
SEC_LEG1/2_H/L	PWMs for the secondary side full bridge

1.1 Key System Specifications

The CLLLC reference design power specifications are listed in Table 1.

Table 1. Key System Specifications

PARAMETER	SPECIFICATIONS
Prim Voltage (Vprim)	380 V–600 V DC (typically sourced from a PFC stage, max ripple 5%)
Sec Voltage (Vsec)	280 V–450 V DC (connected to the battery)
Power Rating	6.6 kW Max
Output Current (Iout)	18 A Max
Efficiency	Peak 98%
PWM Switching Frequency	500 kHz Nominal (300 kHz–700 kHz Range)

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 temprature and is not tested for operation under other ambient tempratures.

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

CAUTION

Do not leave the design powered when unattended.

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!

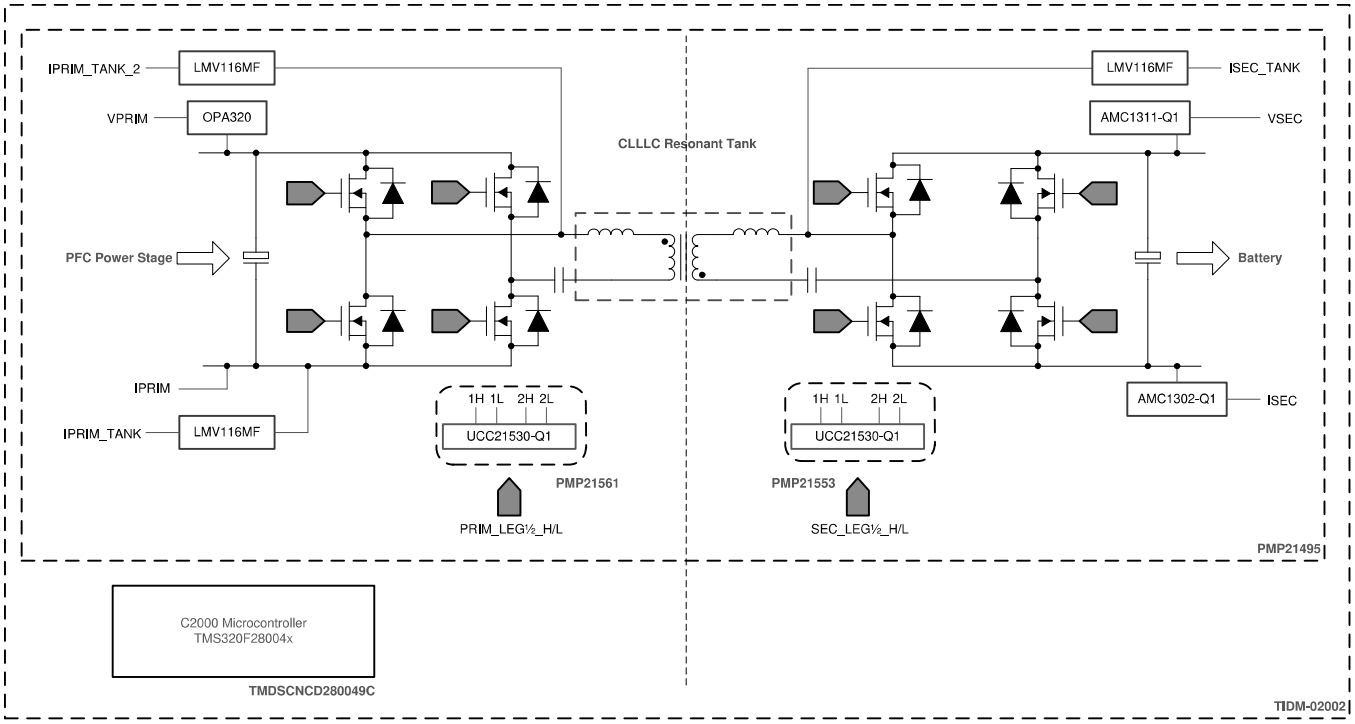
Some components may reach high temperatures >55°C when the board is powered on. The user must not touch the board at any point during operation or immediately after operating, as high temperatures may be present.

2 System Overview

2.1 Block Diagram

Figure 3 shows the block diagram of the CLLLC topology.

Figure 3. TIDM-02002 Block Diagram



This reference design uses the following EVMs and PMP designs to achieve operation as documented in this guide:

1. CLLLC Base Board: PMP21495
2. Safety isolated primary SiC MOSFET driver reference design: PMP21553
3. Safety isolated secondary SiC MOSFET driver reference design: PMP21561
4. F280049C controlCARD Evaluation Module: TMDSCNCD280049C

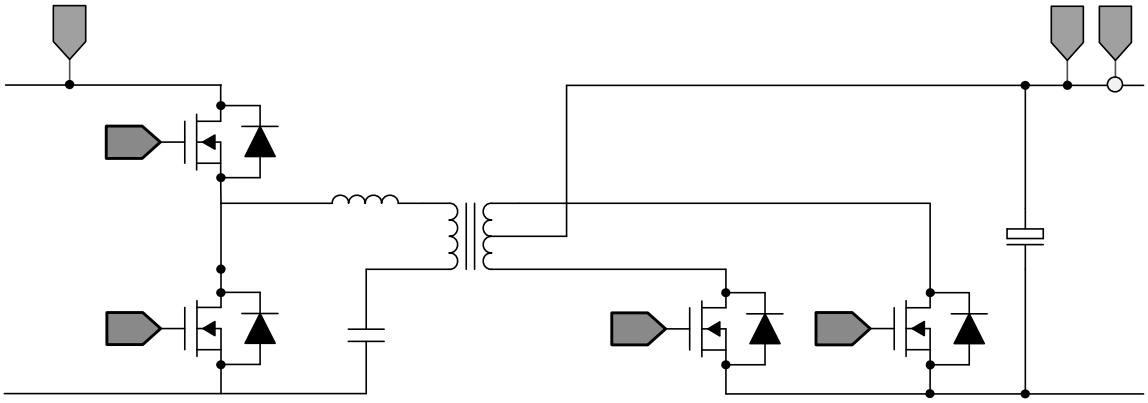
The following sections discuss details of the hardware, software and system design.

2.2 Design Considerations and System Design Theory

LLC converters are widely popular due to their ability to achieve ZVS at the primary side, and ZCS on the secondary side. However, LLC converters suffer when operating away from the resonant frequency point, as the resonant tank current increases and switching losses rise along with the loss of ZCS. However, with the use of silicon carbide (SiC) for higher power chargers, SiC power devices are typically rated for 900 V+“, which allows the output voltage of the PFC converter to be varied in a wide range. This can then be used to operate the LLC at resonance or close-to-resonance frequency, which can eliminate these drawbacks of LLC converters when operating under wide output voltage range.

A typical LLC Series Resonant Converter (SRC) is shown in Figure 4. The primary side of this converter is half bridge; thus, the transformer utilization from a Volt-sec perspective is half. In addition, the current rating for the switches is twice of what is needed, compared to when a full-bridge structure is used.

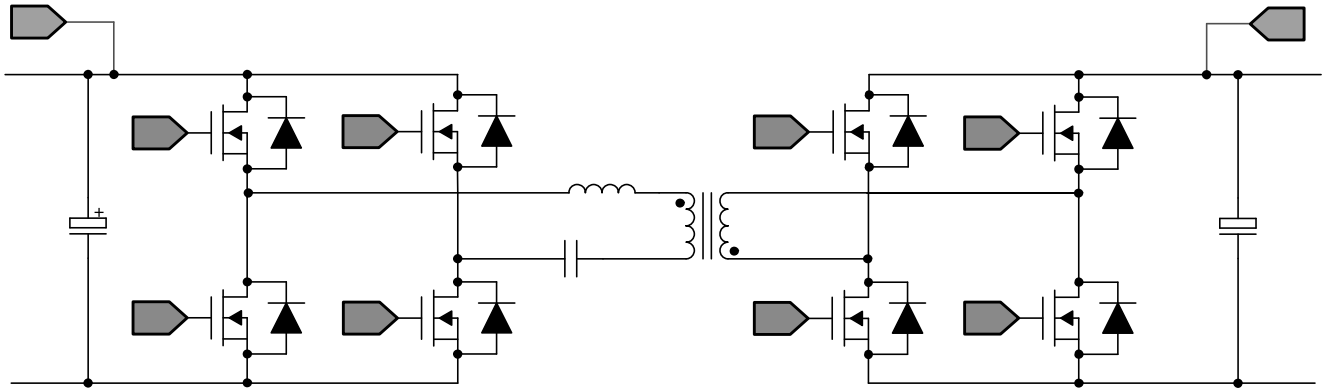
Figure 4. LLC Half-Bridge SRC



Although half-bridge LLC SRCs are attractive at lower power for cost reasons, for high-power and high-density applications, a full-bridge LLC SRC is desired for the following reasons:

1. A full-bridge LLC converter better utilizes the magnetic core of the transformer on both the secondary side and primary side; therefore, it is able to offer better power density.
2. A full-bridge LLC converter reduces current rating; and therefore, reduces the cost of copper in wires. It also enables higher power (compared to half-bridge SRCs) to be achieved with the same copper wires.

Figure 5. Full-Bridge LLC Converter



A full-bridge LLC converter, as shown in Figure 5, falls under the broad category of Dual Active Bridge (DAB) converters. Under DAB converters, the converter can be classified on the basis of model or operation:

1. A phase-shifted DAB converter is one of the most popular converters historically.
2. Resonant DAB converters have different variants on resonant tanks (LC, LLC, CLLC, CLLLC, and so forth).

Resonant DAB converters are of interest because high efficiency, high power, and high density are achievable with such converters. CLLLC, with its symmetric tank, is capable of bidirectional operation. The problem with using an LLC structure for bidirectional use is that the switching frequency, when operating in the reverse power flow mode, is governed by the transformer winding capacitance and the leakage inductance. This offers little or no control on the gain of the power stage and the switching frequency. Therefore, the CLLLC type of structure is preferred as it offers much better control on the switching frequency and an additional degree of freedom on the gain.

One of the biggest challenges with resonant converters is the operation across a wide input/output voltage range. This is particularly challenging with battery-charging type applications where the battery voltage varies widely from 280 V to 450 V. Techniques in literature have been presented to mitigate this with a variable DC link voltage from the PFC. This design relies on the ability to regulate the PFC voltage in a 380 V–6000 V range; a resonant tank is designed to cover that range of voltages.

2.2.1 Tank Design

In this section, the tank parameter selection for the CLLLC is discussed based on the voltage gain desired, soft switching characteristics, and an appropriate power profile is selected for the charger based on CLLLC.

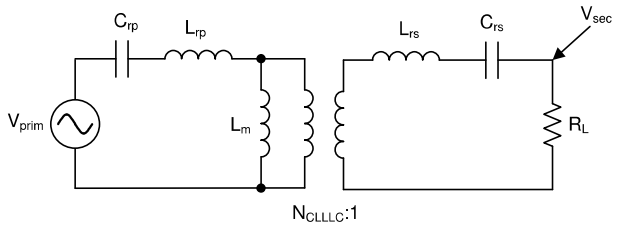
For additional calculations and information, refer to the following files located inside the software install package at C2000Ware_DigitalPower_SDK_<ver>/solution/tidm_02002/hardware/

1. CLLLC_calculations.xlsx (Excel file)
2. CLLLC_tankSimulation.m (MATLAB file)

2.2.1.1 Voltage Gain

To understand tank design, first the gains for both battery-charging mode and reverse-power-flow mode must be analyzed with First Harmonic Analysis (FHA) using first harmonic approximation. The simplified diagram of the resonant tank is given in Figure 6.

Figure 6. FHA Model for CLLLC Resonant Tank During Battery Charging Mode (BCM)



The nomenclature for Figure 6 is as follows:

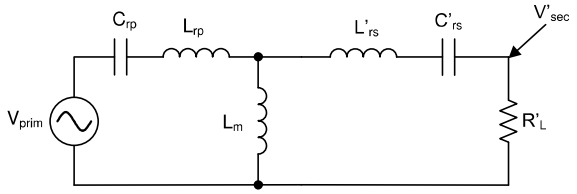
V_{prim}	Voltage input at primary side
L_{rp}	Primary side resonant inductor
C_{rp}	Primary side resonant capacitor
N_{CLLLC}	Turns ratio of the transformer
L_m	Magnetizing inductor
V_{sec}	Voltage output at secondary side
L_{rs}	Secondary side resonant inductor
C_{rs}	Secondary side resonant capacitor
R_L	Effective load seen with FHA on the secondary output

Note here the effective R_L is accounted for as $R_L = \left(\frac{8}{\pi^2}\right) R_{L\_dc}$ where R_{L_dc} is the DC resistive load at the output.

Referring to secondary side quantities on the primary side,

- L_{rs}' is equal to $L_{rs} * N_{CLLLC} * N_{CLLLC}$
- C_{rs}' is equal to $C_{rs} / (N_{CLLLC} * N_{CLLLC})$
- R_L' is equal to $R_L * (N_{CLLLC} * N_{CLLLC})$
- V_{rs}' is equal to $V_{rs} * N_{CLLLC}$

Figure 7. FHA CLLLC With Quantities Referred to Primary Side in BCM



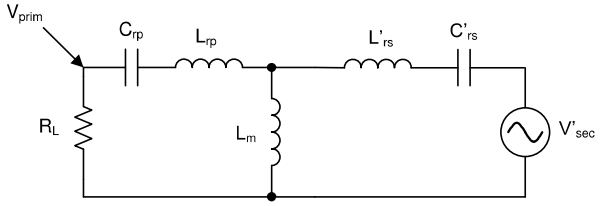
Using KCL and KVL, the gain equation can be written as Equation 1.

$$\frac{V_{sec}}{V_{prim}} = \frac{\left[Z_m \,||\left(Z_{rs}' + R_L'\right)\right]R_L'}{\left(Z_{rp} + \left[Z_m \,||\left(Z_{rs}' + R_L'\right)\right]\right)\left(Z_{rs}' + R_L'\right)N_{CLLLC}} \tag{1}$$

Similarly, for the reverse power flow, the circuit can be simplified as shown in Figure 8, and the gain can be written as Equation 2.

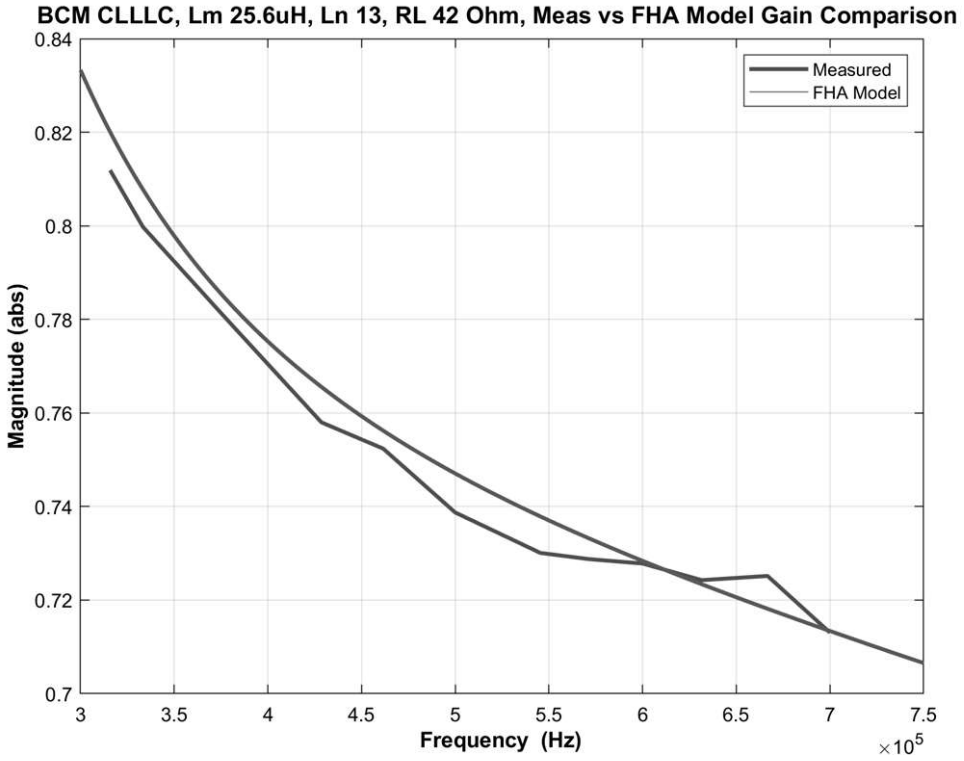
$$\frac{V_{prim}}{V_{sec}} = \frac{N_{CLLLC}\left[Z_m \,||\left(Z_{rp} + R_L\right)\right]R_L}{\left(Z_{rs}' + \left[Z_m \,||\left(Z_{rp} + R_L\right)\right]\right)\left(Z_{rp} + R_L\right)} \tag{2}$$

Figure 8. FHA Model for Gain Calculation in RCM



Equation 1 and Equation 2 are used in the following section to study the voltage gain based on the parameters selected for the design. The FHA model described above is verified by the measurement on the designed CLLLC, as shown in Figure 9.

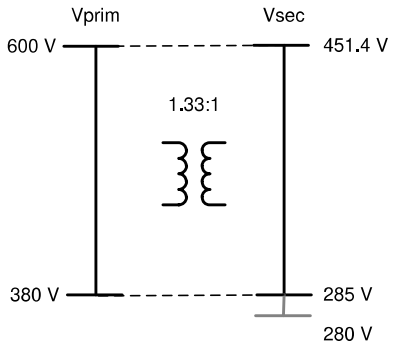
Figure 9. FHA Model Verification in BCM



2.2.1.2 *Transformer Gain Ratio Design (N_{CLLLC})*

The first part in tank design is to identify the voltage gain desired for the application. As the design is intended to work with a variable PFC bus voltage (380 V–600 V) to generate voltage from 280 V–450 V, a turns ratio of 1.33 is selected, as shown in Figure 10. This enables operation at resonance for the CLLLC for a wide range of operations.

Figure 10. Gain Requirements for Onboard Charger



2.2.1.3 Magnetizing Inductance Selection (L_m)

To ensure ZVS operation of the primary side FETs, we need to make sure the energy stored in the resonant tank is greater than the energy stored in the FET output capacitors. We can use Equation 3 to determine the needed L_m for full-bridge LLC SRCs.

$$L_m \le \frac{Tt_{dead}}{16 * C_{oss}} \tag{3}$$

where the intended switching frequency for the converter is 500 kHz, hence T = 1/(500 * 10^3), and based on the power device (that is, the SiC MOSFET). Selected parameters such as t_{dead} and C_{oss} can also be identified from the power device data sheet. Typically, the effective C_{oss} must be calculated using curve fitting (see the C_{oss} effective sheet in CLLLC_calculations.xlsx). On this design, based on the design parameters discussed, L_m must be less than 48 µH. In addition to what is accounted for in the above calculation, there is interwinding capacitance in a real transformer that needs to be discharged by the resonant tank current. Therefore, using simulation, a value of 25 µH was selected to ensure ZVS across the operating range of the converter; this value is used in the subsequent selection processes.

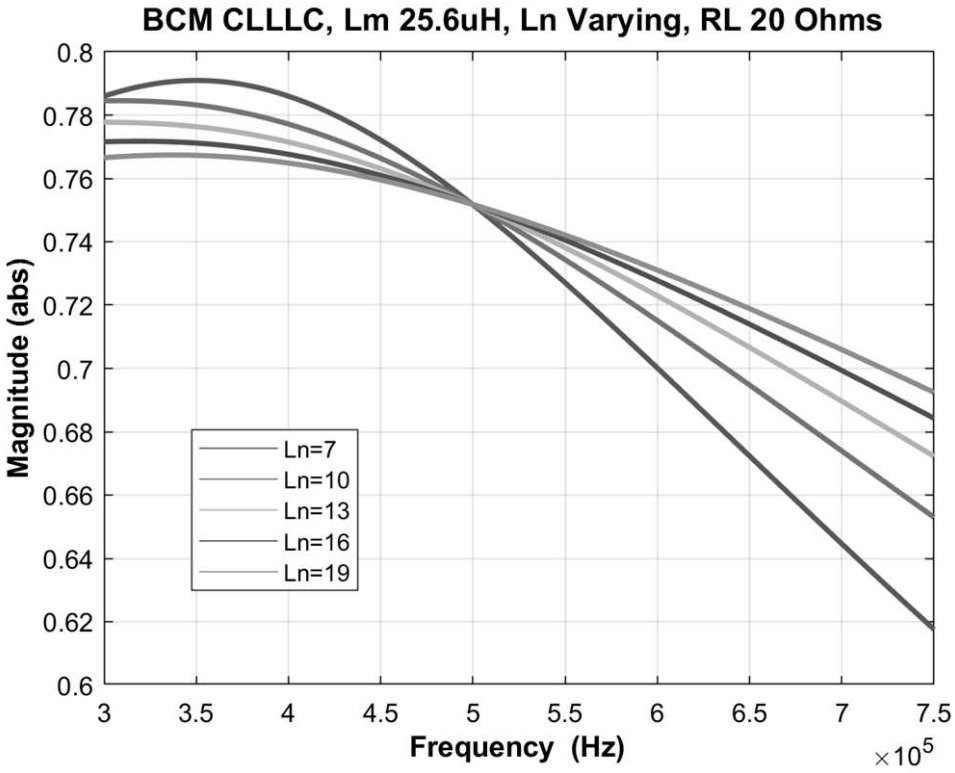
2.2.1.4 Resonant Inductor and Capacitor Selection (L_{rp} and C_{rp})

While selecting L_{rp}, the ratio of L_m to L_{rp} is widely used as a design parameter,

$$L_n = \frac{L_m}{L_{rp}} \tag{4}$$

The L_n value is selected such that it ensures the voltage gain in the resonant tank is enough across the operating range of the converter. In this design, as the input voltage comes from a PFC stage and will have a 10% ripple, a gain variation of at least 10% is needed. With this criteria in mind and the fact that L_n should be kept higher to reduce the inductor value, and hence the losses, L_n equal to 13 is selected for this design, based on the plot of the FHA with L_n varying (see Figure 11).

Figure 11. CLLLC Tank Gain Variation With L_n Varying

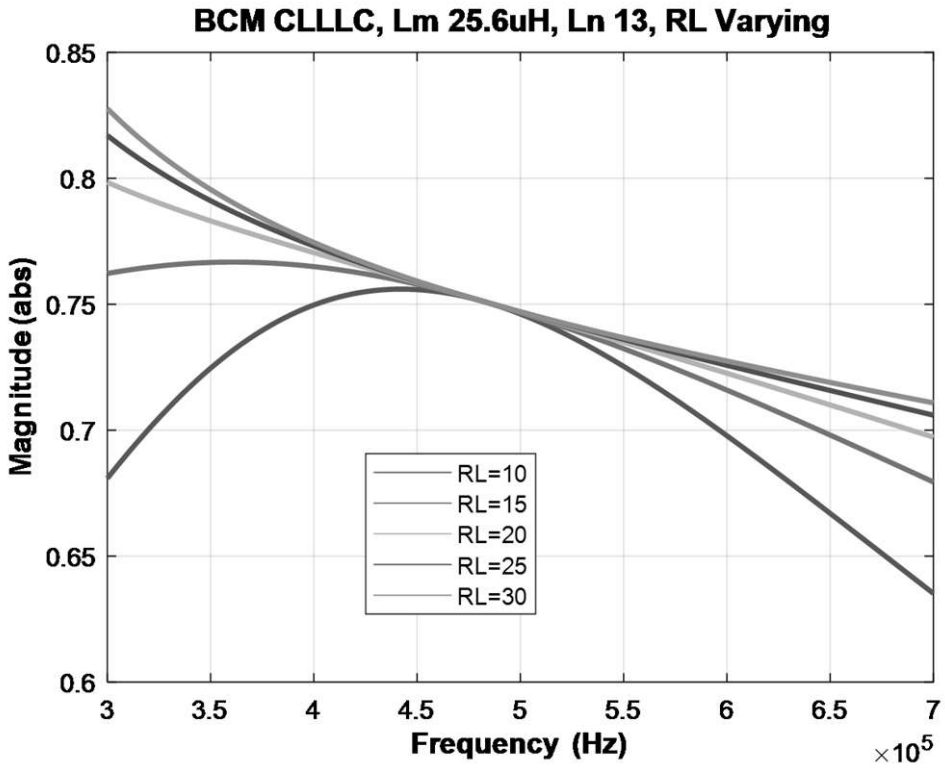


Now that the selection of L_n is made, L_{rp} can be calculated using Equation 4. L_{rp} and C_{rp} determine the series resonant frequency of the converter and they are related by Equation 5.

$$f_{res} = \frac{1}{2\pi\sqrt{L_{rp}C_{rp}}} \tag{5}$$

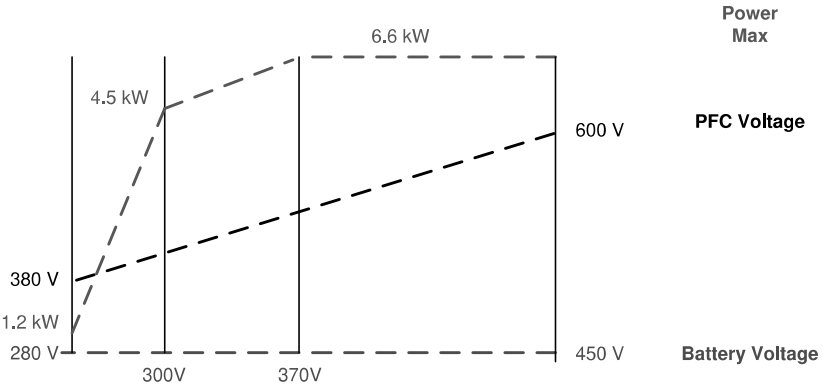
Equation 5 can then be used to calculate the C_{rp} needed on the design. However, due to component availability, the next closest value of C_{rp} is used on the design. With these component values, the BCM gain is shown in Figure 12.

Figure 12. Voltage Gain Curve vs Frequency for Battery Charging Mode With R_L Varying, and Ln = 13 (Run matlab scriptCLLLC_tankSimulation.m)



In Figure 12, as the load increases (that is, R_{L_dc} goes lower), the gain curve becomes non-monotonic in the region below series resonant frequency. This can lead to the loss of ZVS on the primary FETs and, more critically, the loss of control. Therefore, the load is limited or clamped to R_{L_dc} = 20 Ω, for which the gain is monotonic (see Figure 12). By limiting the load, and with the max power being clamped at 6.6 kW, the power rating of the converter in BCM can be determined and is graphed in Figure 13.

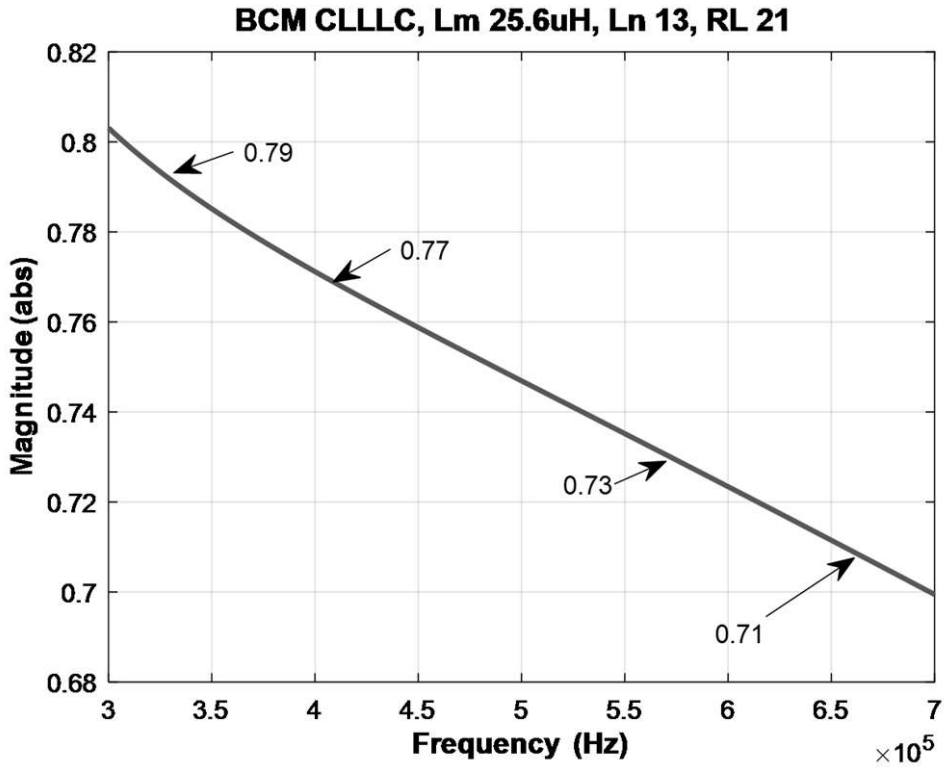
Figure 13. Power Rating in Battery Charging Mode



In BCM, the PFC bus voltage is set to be a ratio of 1.33 of the battery voltage. Therefore, for V_{batt} = 300 V, the PFC bus voltage needs to be set at 400 V. Now, looking at the variation due to PFC bus ripple, with a 10% ripple on the PFC bus, the voltage will vary from 380 V to 420 V, for which the gain needed from the resonant tank will vary from 0.79 to 0.71. Based on the zoomed-in gain profile, Figure 14, it can be determined that the frequency will vary from 330 kHz to 670 kHz.

Similarly, it can be seen that by reducing PFC ripple, the frequency variation can be reduced; for example, for a 5% bus ripple, the range is reduced to 410 kHz–570 kHz. This criteria must be taken into consideration when sizing the PFC bus capacitor.

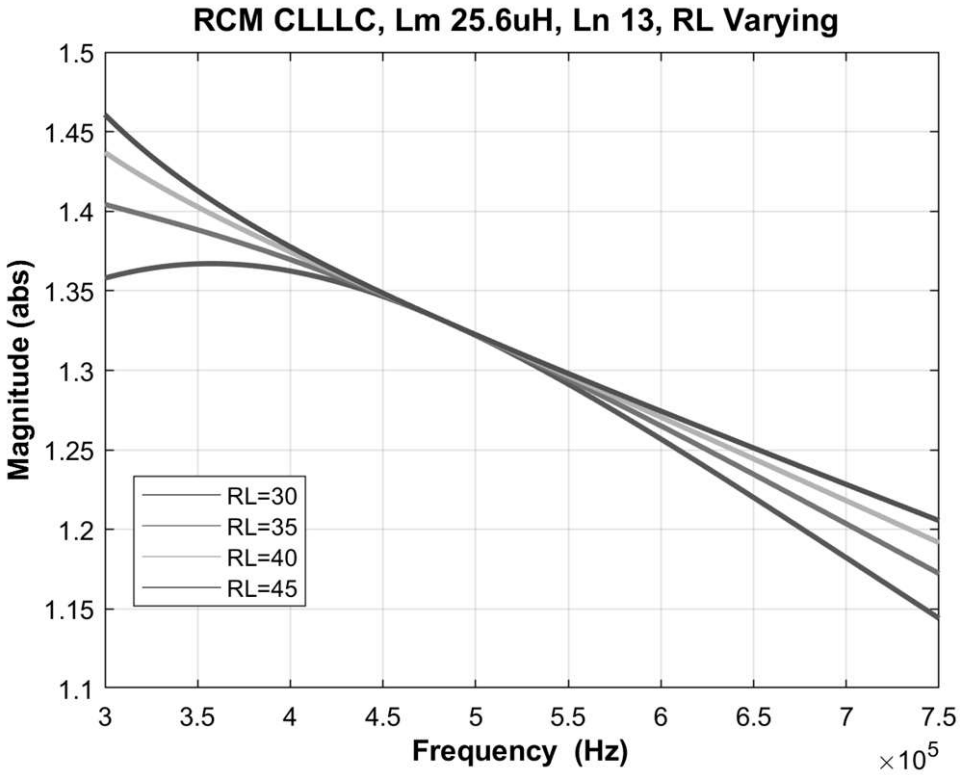
Figure 14. CLLLC Tank Gain Profile, Zoomed In, for Battery Charging Mode at a Given Load



2.2.1.5 Discharging Mode Gain Estimate

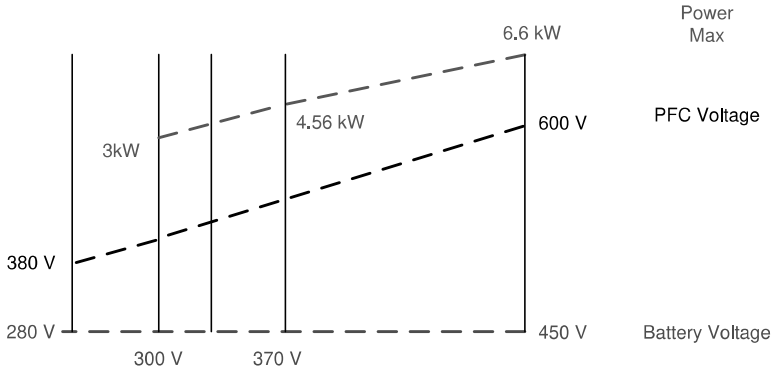
Using the equations in Section 2.2.1.1 and the tank parameters selected, the voltage gain for discharge mode can be plotted as shown in Figure 15.

Figure 15. CLLLC Tank Voltage Gain in Reverse Power Flow Mode



With the CLLLC in discharge mode, the PFC bus voltage is set to be a ratio of 1.33 of the battery, with which at a battery of 400 V, the bus is set to be 600 V. The max power is limited by the load that is seen by the battery, which is clamped to be 30 Ω to avoid the non-monotonic region in the CLLLC gain curve. With the above clamp for the reverse power flow load, the power rating is determined as shown in Figure 16.

Figure 16. Power Rating in Reverse Power Flow Mode



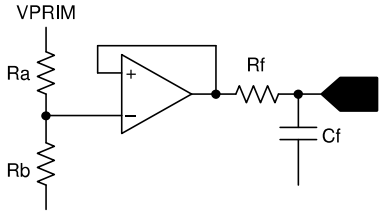
2.2.2 Current and Voltage Sensing

In the following sections, the sensing scheme for different currents and voltages on the design are discussed. On the design, multiple schemes are implemented so that users can select the appropriate one for their application needs. The calculations for the sense range are provided in the CLLLC_calculations.xlsx .

2.2.2.1 VPRIM Voltage Sensing

The C2000 MCU is biased on the primary side; hence, the primary voltage is sensed by a resistor divider to the ground of the board. As oversampling is used, an opamp in voltage follower arrangement is used to buffer the signal for the ADC as shown in Figure 17. The buffer helps reduce impedance as seen by the ADC, and hence, a faster sampling rate can be used. Otherwise, the sampling will be limited by the time constant of the resistor divider resistance, which is typically high, and hence, only slow sampling can be done.

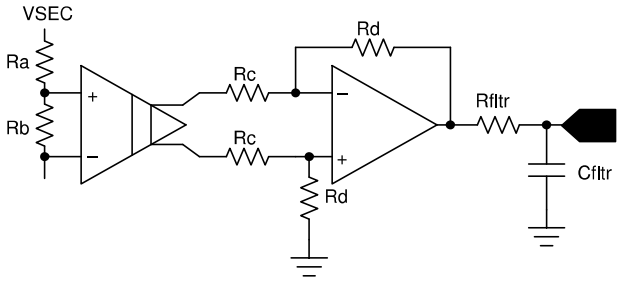
Figure 17. VPRIM Voltage Sensing Circuit



2.2.2.2 VSEC Voltage Sensing

The secondary side voltage is sensed in an isolated manner using the AMC1311, as shown in Figure 18.

Figure 18. VSEC Voltage Sensing Circuit

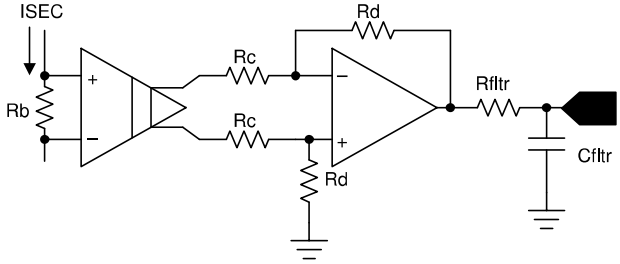


2.2.2.3 ISEC Current Sensing

The secondary side output current is also sensed in an isolated manner using the AMC1302, as shown in Figure 19.

Also, see the TINA simulation file located under the software install directory <solution>/hardware/AMC1302_CurrentSenseSec.TSC.

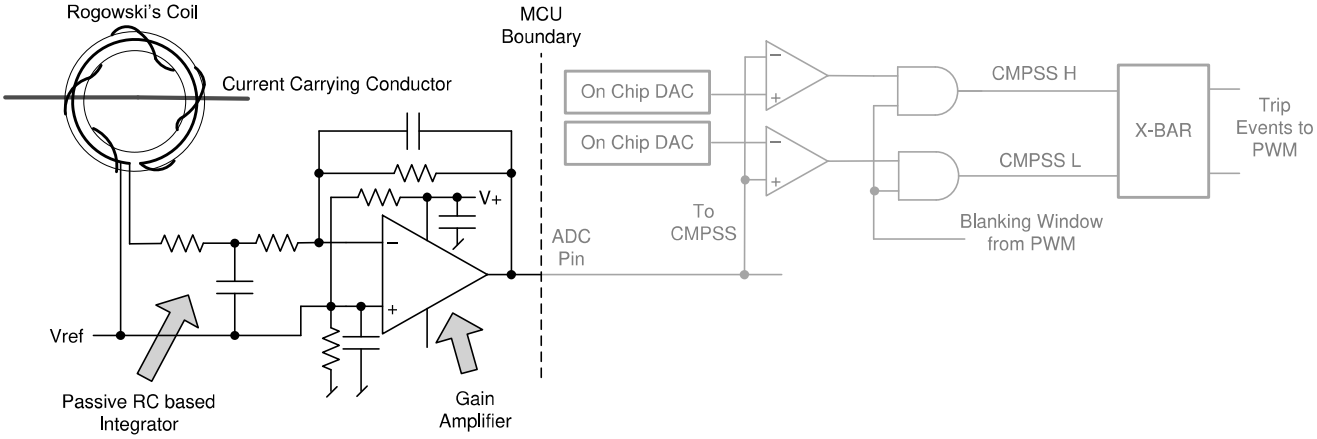
Figure 19. ISEC Current Sensing Circuit



2.2.2.4 ISEC TANK and IPRIM TANK_2

A Rogowski coil-based sensing mechanism is chosen to sense the high-frequency current in the tank on the primary side and the secondary side in an isolated manner, as shown in Figure 20. The ADC pin is internally connected to the Comparator Subsystem (CMPSS), which can generate the correct pulses that go through the X-Bar to the PWM to get the action required for synchronous rectification.

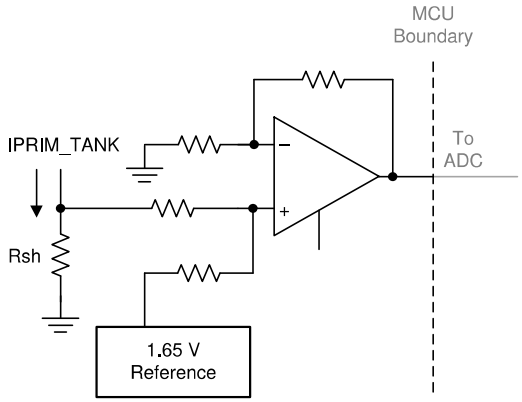
Figure 20. ISEC Tank Current Sensing Using Rogowski's Coil



2.2.2.5 IPRIM_TANK

For the primary side, as the MCU is on the same ground plane, a non-isolated current scheme can also be implemented using a shunt-based sensing method, as shown in Figure 21.

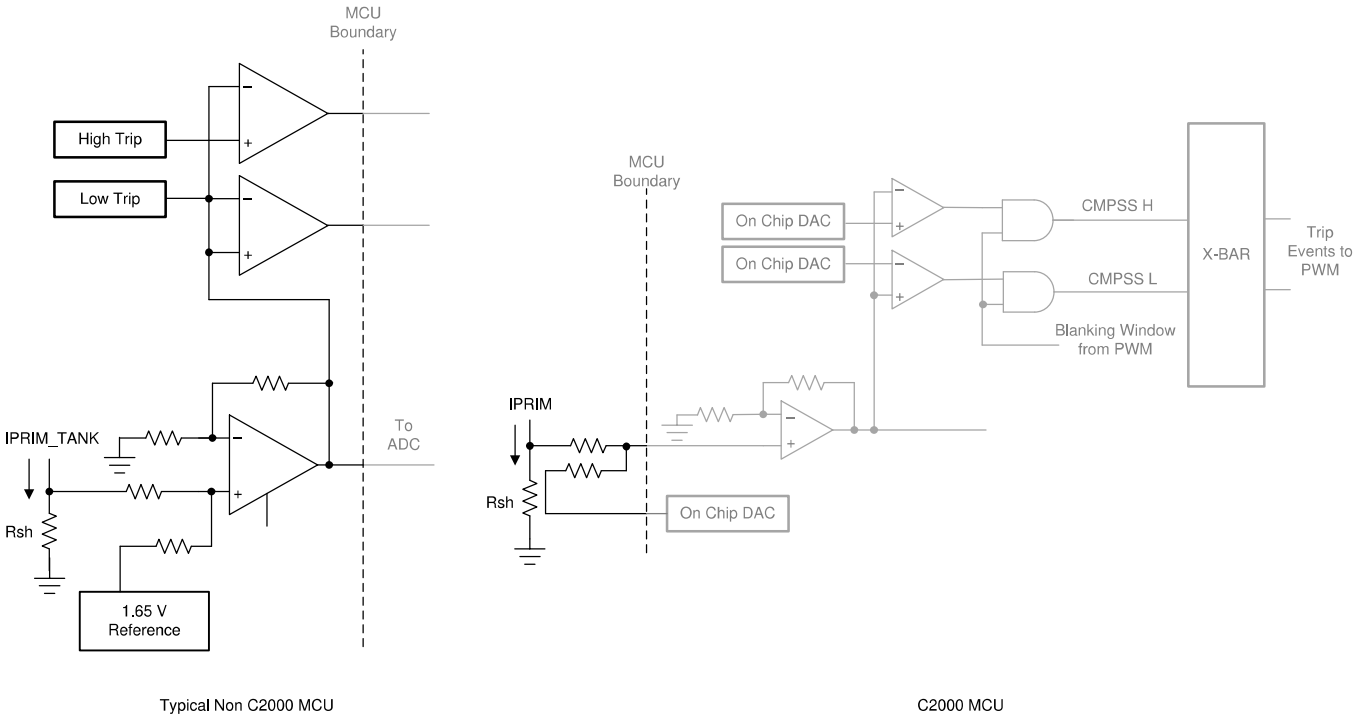
Figure 21. IPRIM Tank Current Sensing Circuit



2.2.2.6 IPRIM Current Sensing

The primary return current is sensed using the on-chip Programmable Gain Amplifier (PGA). Compared to IPRIM_TANK sensing, the on-chip resources are used to sense this current, thus enabling cost and board-space savings (see Figure 22).

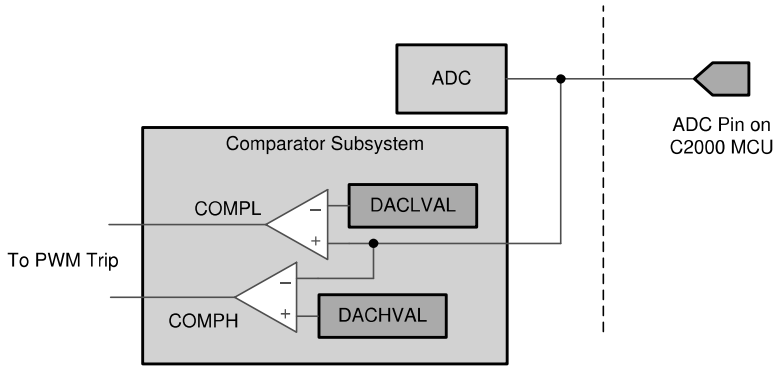
Figure 22. IPRIM Current Sensing Circuit, Comparison With Typical MCU vs C2000 MCU



2.2.2.7 Protection (CMPSS and X-Bar)

Most power electronics converters need protection from overcurrent events. For this design, multiple comparators are needed, and references for the trip points need to be generated. Using C2000 MCUs—such as TMS320F280049, which has an on-chip windowed comparator as part of the Comparator Subsystem (CMPSS) along with 12-bit DACs for trip set points—that are internally connected to the PWM module enables fast tripping of the PWM without the need of external hardware. This saves board space and cost in the end application as extra components can be avoided by using on-chip resources such as DAC, comparators, and ADC. All of these resources can be used together and at the same time, without any extra external connections. Furthermore, the CMPSS-generated signals go to the X-Bar, where they can be combined in different and unique fashions to flag unique trip events from multiple sources.

Figure 23. Comparator Subsystem (CMPSS) Used for Overcurrent Protection



2.2.3 PWM Modulation

Figure 24 shows the PWM waveform configuration used on this design.

High-resolution PWM is used for the primary legs and the secondary legs. Up-down count mode is used to generate the PWMs. To use the high-resolution PWMs, the PRIM_LEG1_H PWM pulse is centered on the period event and the time base is configured to be up-down count. A complementary pulse with high-resolution dead time is then generated for the complementary switch. Between LEG1 and LEG2, there is a 180-degree phase shift for a full-bridge operation. This is achieved by using the feature on the PWM module to swap the xA and xB output. (Alternatively, a phase shift can also be implemented, but is not needed on this design.)

The PWM pulse to the secondary side goes through an isolator, which adds additional propagation delay. To account for this propagation delay, a small advance of the PWM is required. This is implemented in form of a phase-shift delay with respect to the primary active PWM pulse's falling edge. The phase shift of the secondary side is a combination of the period and the delay needed for the isolator, as shown in Figure 24. As active synchronous rectification scheme is used, the rising edge is controlled by the primary side PWM switch timing. As the switching event can be noisy, a blanking window is used. The current in the secondary tank can be discontinuous depending on the operating frequency and load. Hence, the falling edge is controlled by the trip action that is triggered as soon as the secondary current reaches zero. The trip is then latched until the next zero or period event to avoid any spurious turnon of the secondary side switches because of noise. The blanking pulse is generated by the PWM time base but the trip latch and the blanking actions happen as part of the CMPSS. Depending on whether it is the positive half or the negative half of the tank current, two different trip signals are generated and sent to the PWM module through the X-Bar. The Type-4 PWM on the C2000 MCU can uniquely use these events to trip the xA pulse during the up count and xB during the down count. For details, refer to the code in the function CLLLC_HAL_setupSynchronousRectificationAction(), which is the HAL file for the solution, see Section 3.1.2.

The global link mechanism on the Type-4 PWM is used to reduce the number of cycles needed to update the registers and enables high-frequency operation. For example, the following code in the CLLLC_HAL_setupPWM() function links the TBPRD registers for all the PWM Legs. Using this linkage, a single write to the PRIM_LEG1 TBPRD register will write the value to PRIM_LEF2, SEC_LEG1, and SEC_LEG2.

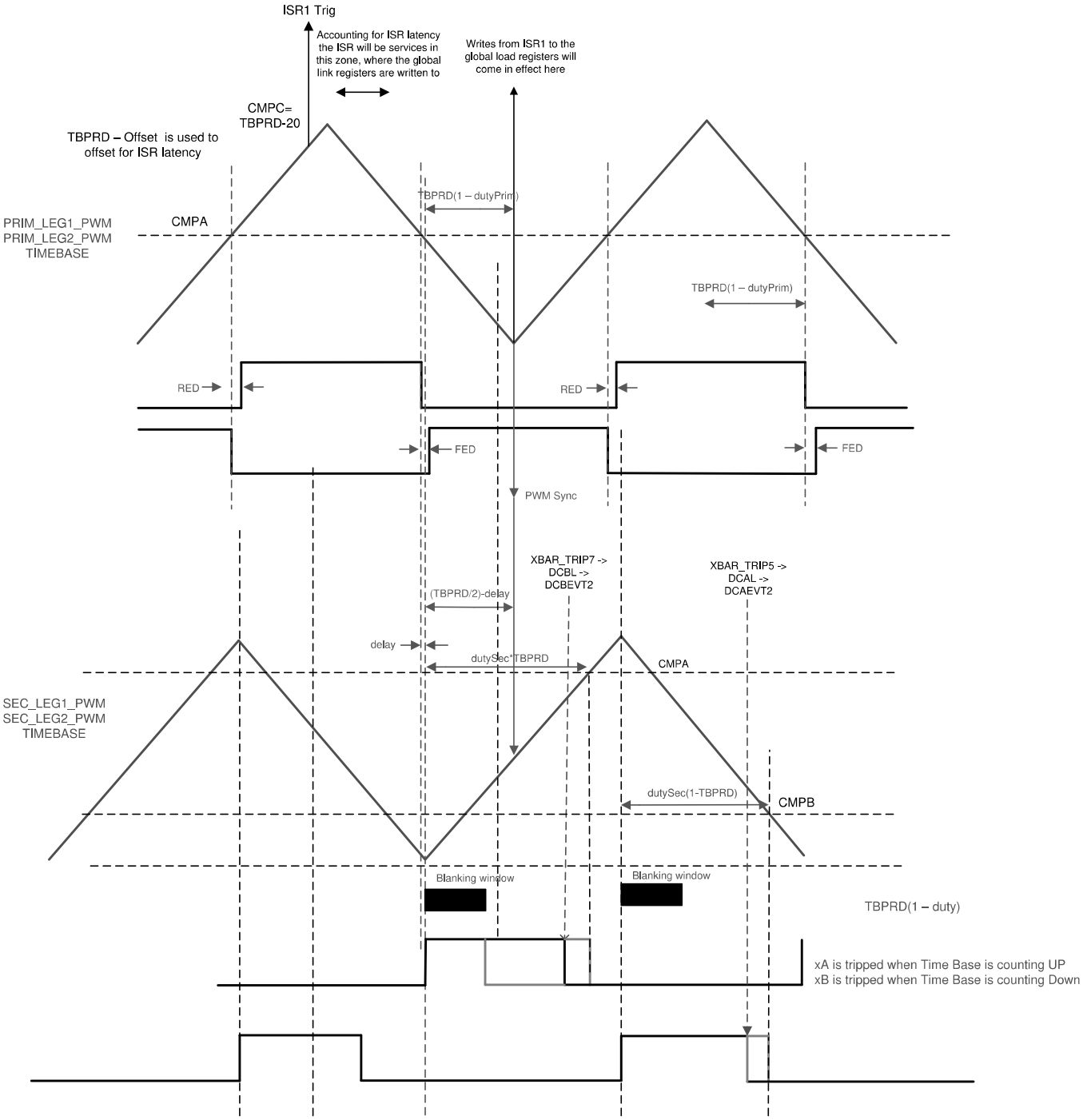
```
EPWM_setupEPWMLinks(CLLLC_PRIM_LEG2_PWM_BASE,
                    EPWM_LINK_WITH_EPWM_1,
                    EPWM_LINK_TBPRD);

EPWM_setupEPWMLinks(CLLLC_SEC_LEG1_PWM_BASE,
                    EPWM_LINK_WITH_EPWM_1,
                    EPWM_LINK_TBPRD);

EPWM_setupEPWMLinks(CLLLC_SEC_LEG2_PWM_BASE,
                    EPWM_LINK_WITH_EPWM_1,
                    EPWM_LINK_TBPRD);
```

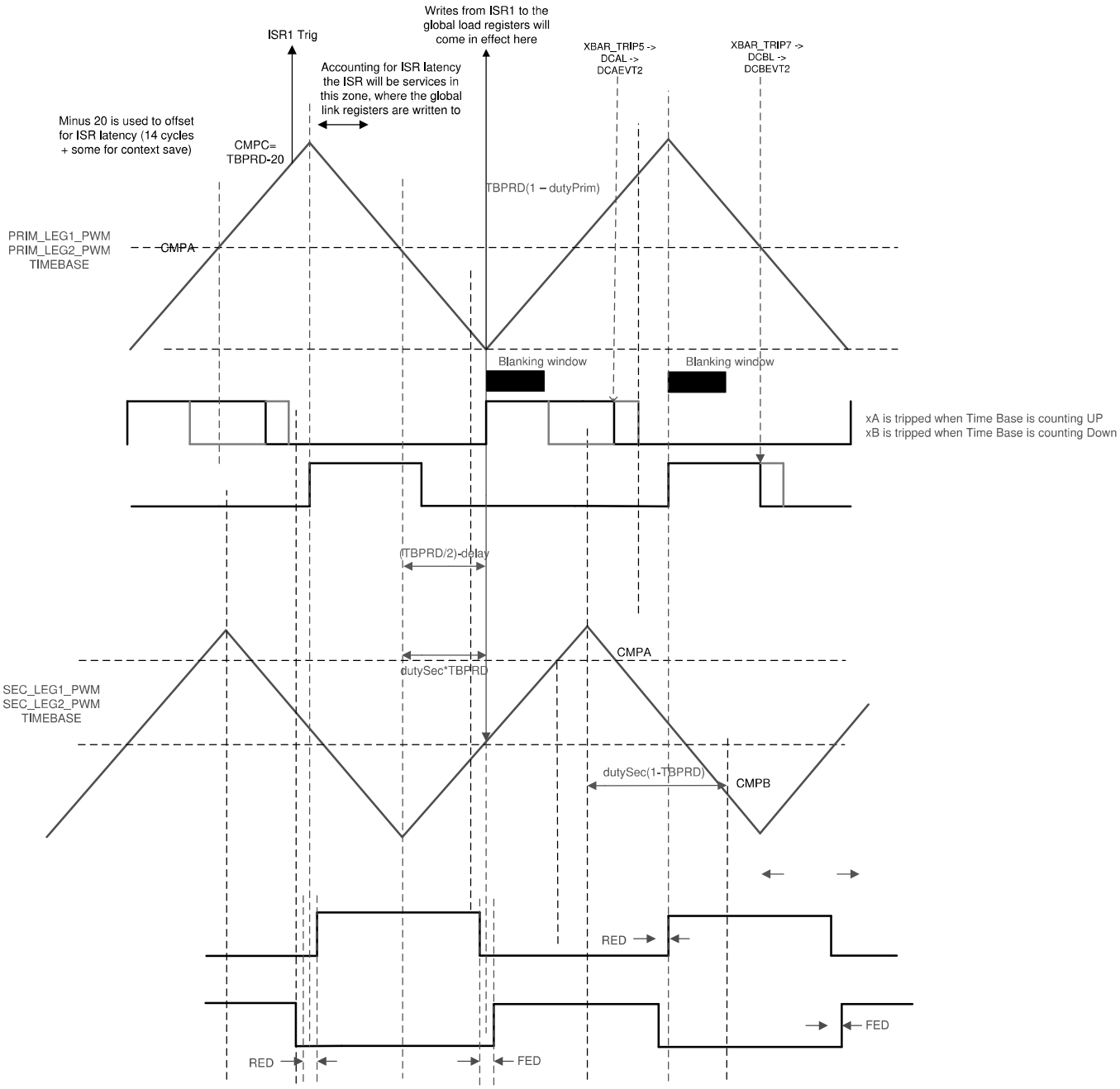
High-resolution PWM relies on carrying forward remainder calculation from the previous cycle into the next; hence, a periodic sync should not be used between the primary and secondary side PWMs to maintain the phase relation. Whenever a frequency change or duty change is detected, a one-time sync is issued using a fast interrupt service routine (ISR1, see Section 3.1.2.2).

Figure 24. PWM Scheme Used on CLLLC Design With Active Synchronous Rectification with Power Flow Primary to Secondary



Similarly for the reverse power flow direction, the PWM configuration used is shown in Figure 25

Figure 25. PWM Scheme Used on CLLLC Design With Active Synchronous Rectification with Power Flow Secondary to Primary



2.3 Highlighted Products

2.3.1 C2000 MCU TMS320F28004x

C2000™ 32-bit microcontrollers are optimized for processing, sensing, and actuation to improve closed-loop performance in real-time control applications such as industrial motor drives; solar inverters and digital power; electrical vehicles and transportation; motor control; and sensing and signal processing.

The TMS320F28004x (F28004x) is a powerful 32-bit floating-point microcontroller unit (MCU) that lets designers incorporate crucial control peripherals, differentiated analog, and nonvolatile memory on a single device.

The CLA allows significant offloading of common tasks from the main C28x CPU. The CLA is an independent 32-bit floating-point math accelerator that executes in parallel with the CPU. Additionally, the CLA has its own dedicated memory resources and it can directly access the key peripherals that are required in a typical control system. Support of a subset of ANSI C is standard, as are key features like hardware breakpoints and hardware task-switching.

High-performance analog blocks are integrated on the F28004x MCU to further enable system consolidation. Three separate 12-bit ADCs provide precise and efficient management of multiple analog signals, which ultimately boosts system throughput. Seven PGAs on the analog front end enable on-chip voltage scaling before conversion. Seven analog comparator modules provide continuous monitoring of input voltage levels for trip conditions.

The TMS320C2000™ devices contain industry-leading control peripherals with frequency-independent Enhanced Pulse Width Modulator/High-Resolution Pulse Width Modulator (ePWM/HRPWM) and Enhanced Capture (eCAP) module allow for a best-in-class level of control to the system. The built-in 4-channel Sigma-Delta Filter Module (SDFM) allows for seamless integration of an oversampling sigma-delta modulator across an isolation barrier.

Connectivity is supported through various industry-standard communication ports [such as Serial Peripheral Interface (SPI); Serial Communication Interface (SCI); Inter-integrated Circuit (I2C); Local Interconnect Network (LIN); and Controller Area Network (CAN)] and offers multiple muxing options for optimal signal placement in a variety of applications. New to the C2000™ platform is the fully compliant Power-Management Bus (PMBus). Additionally, in an industry first, the Fast Serial Interface (FSI) enables high-speed, robust communication to complement the rich set of peripherals that are embedded in the device.

A specially enabled device variant, TMS320F28004xC, allows access to the Configurable Logic Block (CLB) for additional interfacing features and allows access to the secure ROM, which includes a library to enable InstaSPIN-FOC™. See the Device Comparison table in the TMS320F28004x microcontrollers data manual for more information.

The Embedded Real-Time Analysis and Diagnostic (ERAD) module enhances the debug and system analysis capabilities of the device by providing additional hardware breakpoints and counters for profiling.

Following is the subset of features of the C2000 MCU that are highlighted on this design to enable control of high-frequency CLLLC topology:

1. **High-resolution PWM**: With picosecond resolutions possible, the ePWM module on the C2000 MCU can generate high-frequency PWM with accuracy. With the Type-4 PWM high-resolution period control, high-resolution duty control, high-resolution dead-band control, along with high-resolution phase shift control, is possible. This enables generation of balanced pulses for the resonant tank excitation and is an enabling feature for high-frequency power converters.
2. **Comparator Subsystem (CMPSS)** with ePWM for active synchronous rectification: Active synchronous rectification enables higher efficiency and for high-frequency resonant converters that operate both below and above the resonant point, it is a necessary feature for the topology. The C2000 MCUs' integrated CMPSS enables generation of the active synchronous rectification pulses by using the integrated comparators and the integrated Digital-to-Analog Converters (DACs). (See Section 2.2.2.4.)
3. **Blanking window**: Due to noise, which is unavoidable in switching converters, the blanking window feature is used to suppress the CMPSS output during noisy switching events. This blanking window is provided by the ePWM time base and can be applied at different times during the PWM cycle. (See Section 2.2.3.)

4. **X-Bar**: Multiple trip sources are possible in a power converter. The X-bar enables combining different trip actions from either different CMPSS or from GPIO to generate trip behavior desired in the ePWM without need of external logic.
5. **Control Law Accelerator (CLA)** enables integration of control of multiple topologies on a single controller. The software provided with this design provides the option to run the control loop on the CLA or the C28x. (See Section 3.1.2.3.)
6. **Global Link feature in PWM module** enables update to multiple PWMs through a single write, which reduces the CPU burden and enables higher-frequency converters to be controlled easily.
7. **On-chip PGA**: The on-chip PGA Is used to sense current for protection purpose on the primary side, thus enabling more cost-saving while increasing protection capability. See Section 2.2.2.6.

2.3.2 UCC21530-Q1

The UCC221530-Q1 is an automotive-grade, isolated dual-channel gate driver with 4-A source and 6-A sink peak current capability and a 3.3-mm channel-to-channel spacing. It is designed to drive silicon carbide (SiC) MOSFETs and insulated-gate bipolar transistors (IGBTs). Its input side is isolated from the two output drivers by a 5.7-kVRMS and 8-kV peak-reinforced isolated barrier, with a minimum of 100 V/ns Common-Mode Transient Immunity (CMTI). The internal functional isolation between the two secondary side drivers allows a working voltage of up to 850-V DC. The gate driver is also certified according to VDE, CSA, UL, CQC, and various isolation standards. Each driver accepts VDD supply voltages up to 25 V and a wide input VCCI range from 3 V to 18 V. The UCC21530-Q1 also has programmable dead-time (DT) control. An enable pin enables both outputs simultaneously when it is set high.

2.3.3 AMC1311-Q1

The AMC1311 is a precision, isolated amplifier with an output separated from the input circuitry by an isolation barrier that is highly resistant to magnetic interference. This barrier is certified to provide reinforced galvanic isolation of up to 7 kVPEAK according to VDE V 0884-11 and UL1577. Used in conjunction with isolated power supplies, this isolated amplifier separates parts of the system that operate on different common-mode voltage levels and protects lower-voltage parts from damage.

The high-impedance input of the AMC1311 is optimized for connection to high-voltage resistive dividers or other voltage signal sources with high output resistance. The excellent performance of the device supports accurate, low temperature drift voltage or temperature sensing and control in closed-loop systems. The integrated missing high-side supply voltage detection feature simplifies system-level design and diagnostics.

2.3.4 AMC1302-Q1

The AMC1302 is a precision isolated amplifier with a capacitive isolation barrier that has high immunity to magnetic interference. This barrier provides reinforced isolation of 5 kVRMS (maximum) with a very long lifetime and low power dissipation. When used with isolated power supplies, this device isolates components that operate on different common-mode voltage levels. Furthermore, the AMC1302 also protects lower-voltage devices from damage.

The input of the AMC1302 is optimized for direct connection to shunt resistors or other low voltage-level signal sources. The ±50-mV input voltage range allows significant reduction of the power dissipation through the shunt. Additionally, the low high-side supply current and voltage of the AMC1302 allow use of low-cost isolated power supply solutions. The performance of the device supports accurate current control, resulting in system-level power savings and low-torque ripple that is particularly important in motor control applications.

2.3.5 LMV116

The LMV116 (single) rail-to-rail output voltage feedback amplifiers offer high-speed (45 MHz) and low-voltage (2.7 V) operation in addition to micropower shutdown capability (LMV118).

Output voltage range extends to within 20 mV of either supply rail, allowing wide dynamic range especially in low-voltage applications. Even with low supply current of 600 μA, output current capability is kept at a respectable ±20 mA for driving heavier loads. Important device parameters such as bandwidth, slew rate, and output current are kept relatively independent of the operating supply voltage by a combination of process enhancements and design architecture.

For portable applications, the LMV118 provides shutdown capability while keeping the turnoff current to 15 μA. Both turnon and turnoff characteristics are well-behaved with minimal output fluctuations during transitions; thus, the device can be used in power-saving mode, as well as multiplexing applications. Miniature packages (5-pin and 6-pin SOT-23) are further means to ease the adoption of these low-power, high-speed devices in applications where board area is at a premium.

3 Hardware, Software, Testing Requirements, and Test Results

3.1 Required Hardware and Software

This section details the hardware and explains the different sections on the board and how to set them up for the experiments as outlined in this design guide.

3.1.1 Hardware Settings

The design follows a High-Speed Edge Card (HSEC) control card concept, and any device for which a HSEC control card is available from the C2000 MCU product family can be potentially used on this design. The key resources used for controlling the power stage on the microcontroller are listed in Table 2. Figure 26 shows the key power stage and connectors on the reference design. Table 3 lists the key connectors and their functions. To get started:

1. Make sure no power source is connected to the board.
2. Insert the control card in the J202-J207 slot.
3. Connect a power source (*but do not power up*) for the primary and secondary side (+5 V, 2 A) at the test points shown in Figure 26.
4. Now, switch the power source on for both the primary and secondary side. A green LED on the control card will light up. This indicates the C2000 MCU device is powered. Note: The bias for the microcontroller is separated from the power stage; this enables a safe bringup of the system in this set of instructions.
5. To connect JTAG, use a USB cable from the control card and connect it to a host computer.
6. A DC power supply must be connected (*but not powered up*) for the primary side voltage (VPRIM) across J100 and J103. Typical power rating for this supply is: VDC 350 V–600 V, Power 6.6 kW. However, a lower-rating supply can be used in the case where only low-power tests are being conducted.
7. A second DC power source is needed to emulate a battery voltage lab, and can be connected (*but not powered up*) to J101-J102 to provide the secondary side voltage (VSEC).
8. Mechanism to load the secondary side is needed for an appropriate power level. For example, for a 300-V, 4-kW test, a resistive load of ~22.5 Ω is needed.
9. Current and voltage probes can be connected to observe the tank current at primary and secondary. Optionally, a power meter can be connected to measure the efficiency.

Figure 26. Board Overview

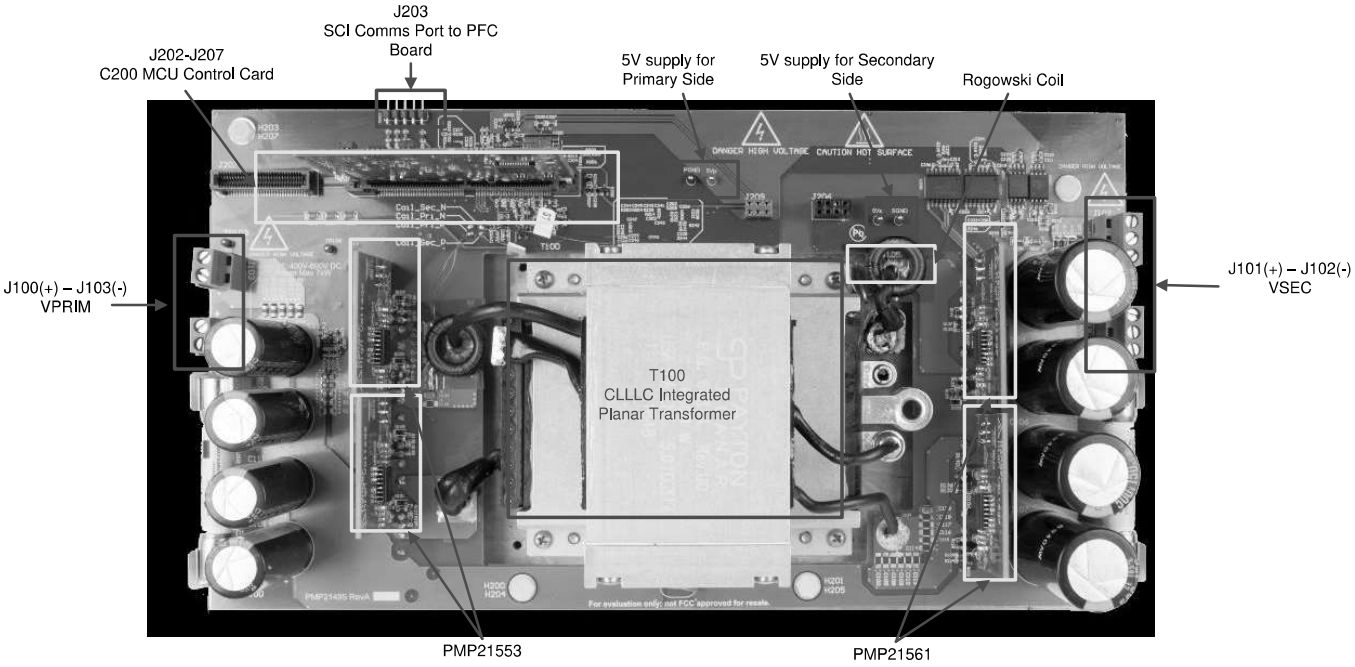


Table 2. Key Controller Peripherals Used for Control of the Power Stage on the Board

SIGNAL NAME	HSEC PIN NUMBER	FUNCTION
PWM-1A	49	PWM : PRIM_LEG1_H (Primary side leg 1, high side switch)
PWM-1B	51	PWM : PRIM_LEG1_L ((Primary side leg 1,low side switch)
PWM-2A	53	PWM : PRIM_LEG2_H (Primary side leg 2, high side switch)
PWM-2B	55	PWM : PRIM_LEG2_L ((Primary side leg 2,low side switch)
PWM-3A	50	PWM : SEC_LEG1_H (Secondary side leg 1, high side switch)
PWM-3B	52	PWM : SEC_LEG1_L (Secondary side leg 1, low side switch)
PWM-4A	54	PWM : SEC_LEG2_H (Secondary side leg 2, high side switch)
PWM-4B	56	PWM : SEC_LEG2_L (Secondary side leg 2, low side switch)
VDAC_PGA	9	DAC: 1.65 reference generated for on chip PGA bases sensing of IPRIM
VSEC	14	ADC
IPRIM	15	PGA +
VPRIM	18	ADC
IPRIM_TANK	21	ADC with CMPSS (dual mapped)
IPRIM_TANK_2	23	ADC with CMPSS (dual mapped)
ISEC_TANK	24	ADC with CMPSS (dual mapped)
IPRIM_TANK_2	25	ADC with CMPSS (dual mapped)
ISEC	27	ADC with CMPSS
IPRIM_TANK	28	ADC with CMPSS (dual mapped)
ISEC_TANK	34	ADC with CMPSS (dual mapped)
PRI_FAULT	74	GPIO : IO flag for the primary side
SCI_RX	76	GPIO : Communication to PFC controller
SCI_TX	78	GPIO : Communication to PFC controller

Table 3. Key Connectors and Their Function

CONNECTOR NAME	FUNCTION
J100-J103	Primary power connection
J101-J102	Secondary power connection
J203	Connection for communication to the PFC controller
J202-J207	HSEC control card connector slot

Specific to the F28004x device, the following IOs are used in the code:

1. Profiling code, ISR1, ISR2, ISR3 → GPIO40 (HSEC pin 68), GPIO11 (HSEC pin 70), GPIO16 (HSEC pin 67)
2. Debugging active synchronous rectification, the CMPSS output is brought out on XBAR1 and XBAR2 → GPIO58 (HSEC pin 99), GPIO59 (HSEC pin 101)

3.1.1.1 Control Card Settings

Certain settings on the device control card are required to communicate over JTAG and use the isolated UART port. The user must also provide a correct ADC reference voltage. The following are the required settings for revision A of the F280049M control card. The user can also refer to the information sheet located inside C2000Ware at \c2000ware\boards\controlcards\TMDSCNCD280049C or alternatively get it from the *F280049 controlCARD Information Guide*.

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

3.1.2 Software

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

3.1.2.1 Opening the Project Inside Code Composer Studio

To start:

1. Install Code Composer Studio from the Code Composer Studio (CCS) Integrated Development Environment (IDE) tools folder. Version 9.3 or above is recommended.
2.
3. Install C2000WARE-DIGITAL-POWER-SDK in one of two ways:
   - Through the C2000Ware Digital Power SDK tools folder
   - Go to CCS and under View → Resource Explorer. Under the TI Resource Explorer, go to C2000WARE-DIGITAL-POWER-SDK, and click on the install button.
4. Once installation completes, close CCS, and open a new workspace. CCS automatically detects powerSUITE. A restart of CCS may be required for the change to be effective.

NOTE: powerSUITE is installed with the SDK by default.

The firmware project can now be imported as follows:

Through resource explorer:

1. In the Resource Explorer, under C2000WARE-DIGITAL-POWER-SDK, click on powerSUITE → Solution Adapter Tool ( ).
2. Select CLLLC from the list of solutions presented under DC-DC section.
3. The development kit page will be displayed
4.

1. The icon to run the project will appear in the top bar. Click Run Project.
2. This action imports the project into the workspace environment, and a configuration page page with a GUI similar to Figure 27 appears.
3. If this GUI page does not appear, refer to the FAQ section under powerSUITE in the C2000WARE-DIGITAL-POWER-SDK resource explorer.

Direct import from the solution folder:

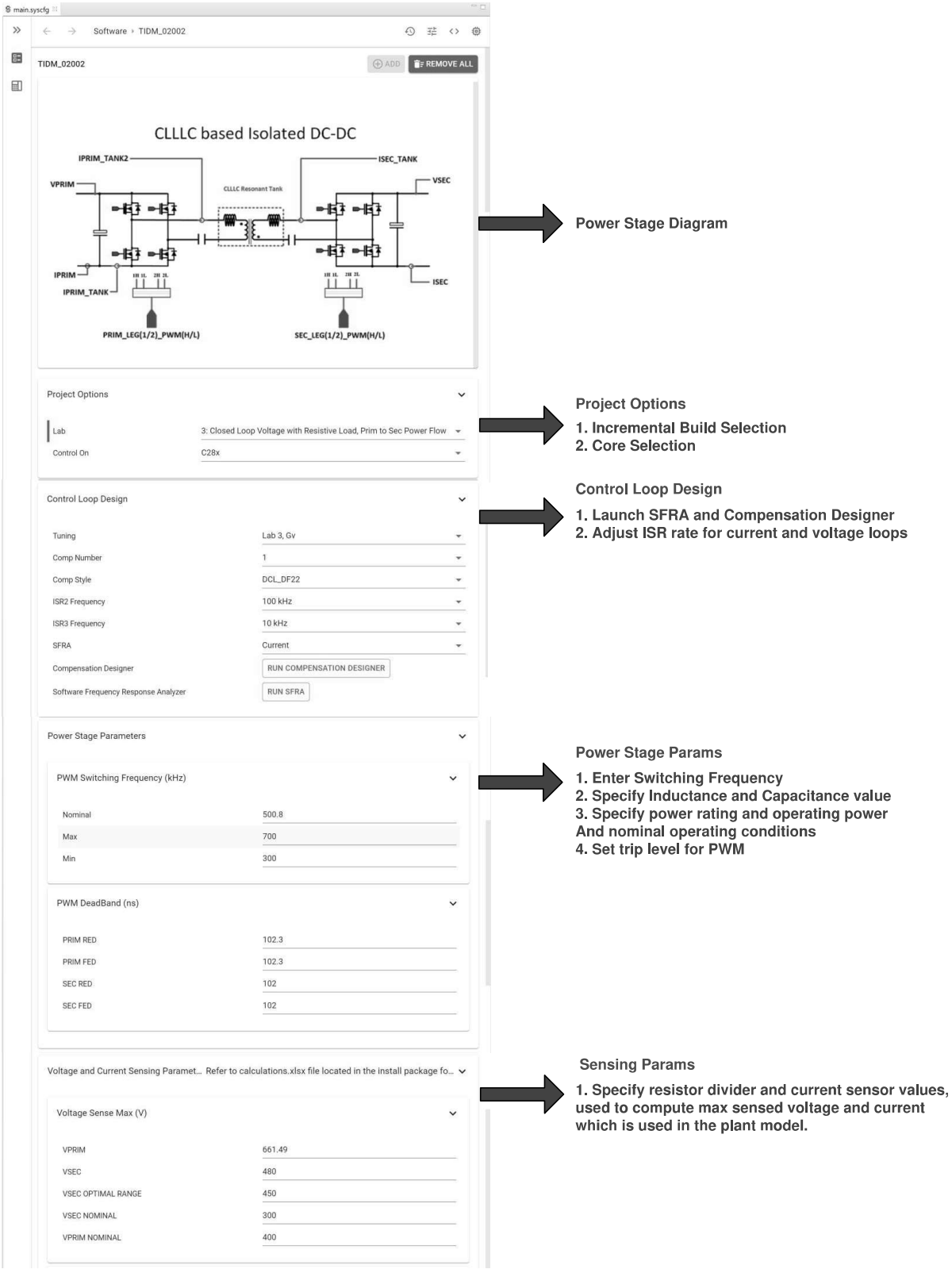
The user can also directly import the project, by going inside CCS to Click Project → Import CCS Projects and browsing to the solution folder located at <SDK>/solutions/tidm_02002/f28004x/ccs.

Two project specs will appear: one of the project with powerSUITE, and the other without powerSUITE. Clicking on either will create a self-contained folder of the project with all the dependencies inside it.

The non-powerSUITE project is provided for customers who find the powerSUITE GUI limiting or want to remove powerSUITE for production code.

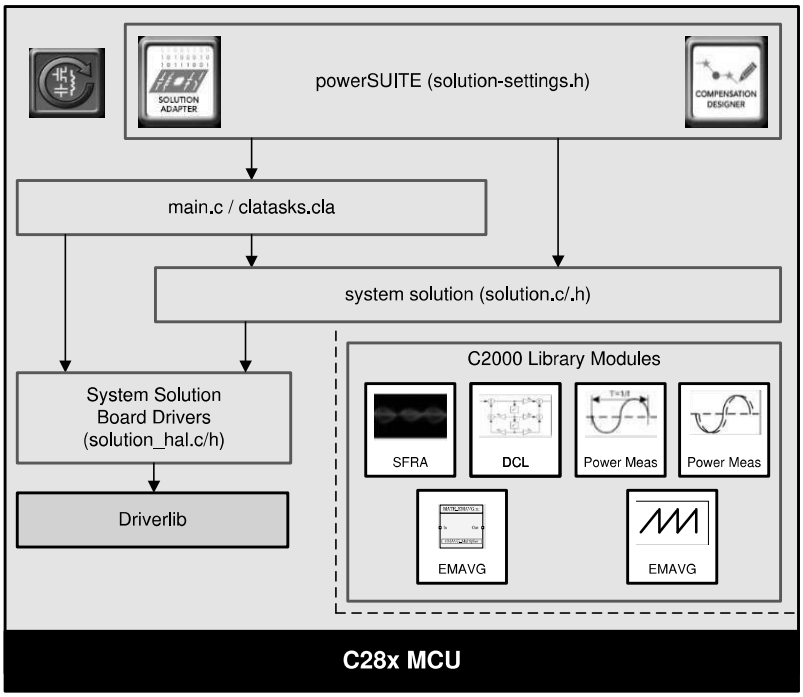
This document guides the user through the powerSUITE project, but all the steps can be repeated with the non-powerSUITE project with modification to the relevant #defines in the powerSUITE settings.h file, which are documented in this design guide.

Figure 27. PowerSUITE Page for the Design



3.1.2.2 Project Structure

Figure 28. Project Structure Overview



The general structure of the project is shown in Figure 28. Once the project is imported, the Project Explorer will appear inside CCS as shown in Figure 29. Note: Figure 29 shows the project for F28004x; however, if a different device is chosen from the powerSUITE page, the structure will be similar.

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

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

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

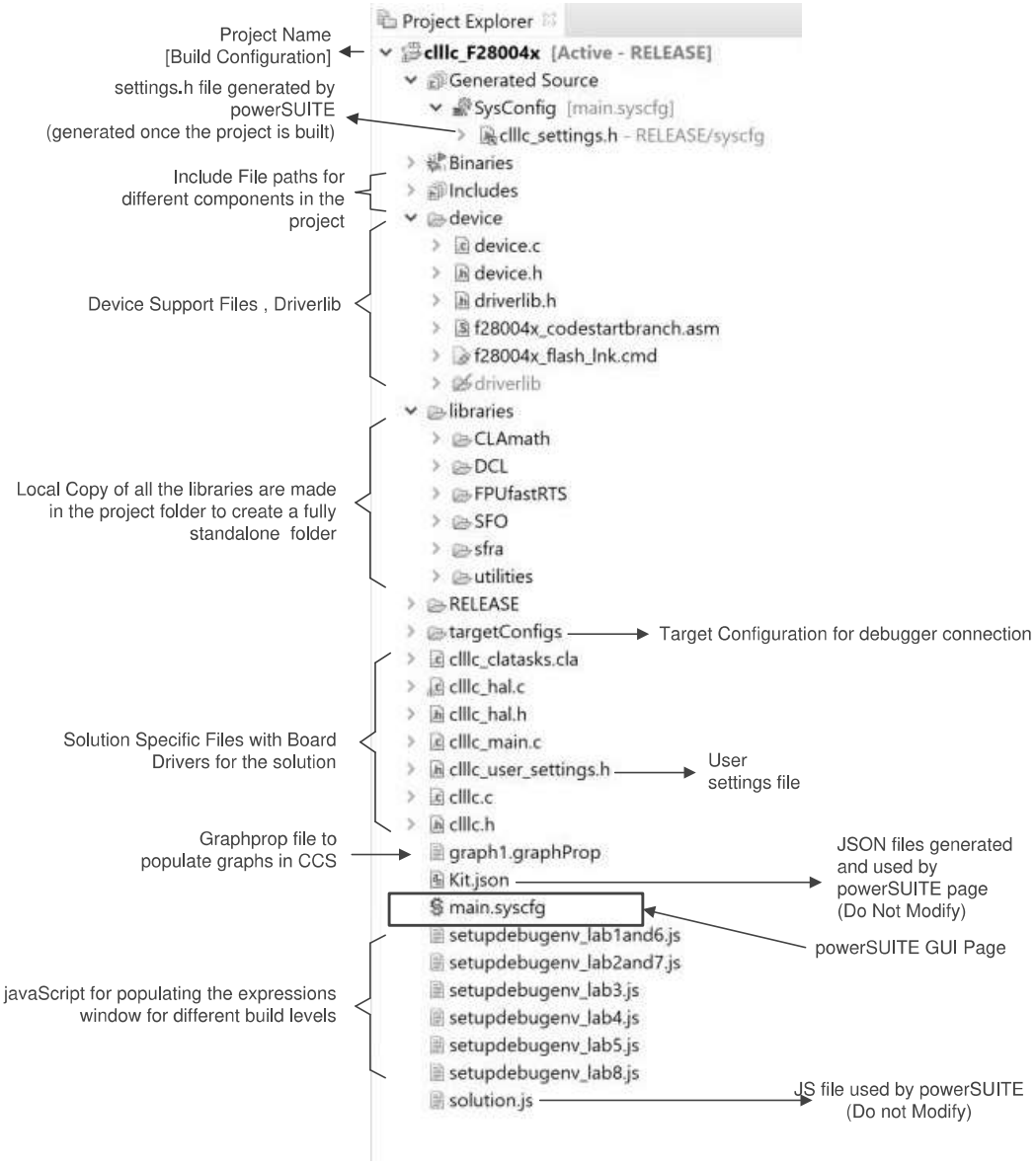
For this design, <solution> is "clllc".

The powerSUITE page can be opened by clicking on the "*main.syscfg*" file, listed under the Project Explorer. The powerSUITE page generates the "<*solution*>*_settings.h*" file. This file is the only C based file used in the compile of the project that is generated by the powerSUITE page. The user must not modify this file manually, as the changes will be overwritten by powerSUITE each time the project is saved. "<*solution*>*_user_settings.h*" is included by the "<*solution*>*_settings.h*"and can be used to keep any settings that are outside the scope of powerSUITE tools such as #defines for ADC mapping, GPIOs etc.

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

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

Figure 29. Project Explorer View of the CLLLC Project



The CLLLC project consists of three **ISRs** (ISR1, ISR2, and ISR3) with ISR1 being the fastest and non-nestable ISR.

ISR1 is reserved for the PWM update ISR. ISR1 is triggered by the PRIM_LEG1_PWM_BASE → EPWM_INT_TBCTR_U_CMPC event. In general, this interrupt is disabled by writing a value to CMPC, which is greater than all possible TBPRD register values for PRIM_LEG1_PWM_BASE. This is done through the CLLLC_HAL_setupISR1Trigger function. Following are the defines that are related to this ISR.

```
#define CLLLC_ISR1_PERIPHERAL_TRIG_BASE CLLLC_PRIM_LEG1_PWM_BASE
#define CLLLC_ISR1_TRIG INT_EPWM1
#define CLLLC_ISR1_PIE_GROUP INTERRUPT_ACK_GROUP3

#define CLLLC_ISR1_TRIG_CLA CLA_TRIGGER_EPWM1INT
```

ISR2 is responsible for writing a valid value to trigger ISR1 by a write to CMPC when ISR1 is desired. (Note: The CMPC is not tied to the global load mechanism to enable this. Also, shadow load of CMPC is disabled.) The CMPC value can be adjusted to get the desired timing from the ISR1. Each time ISR1 is enabled, it triggers two times. In the first ISR1, PWM registers are updated and a sync is enabled. In the second ISR1, the PWM sync is disabled and CMPC is set to a value such that ISR1 does not trigger again. For simplicity the software diagrams and structure only show ISR1 that is triggered the first time.

ISR2 is periodically triggered at ISR2_FREQUENCY. A spare CAP module is used to generate the time base and trigger the interrupt. A spare ePWM module is also configured with the same time base and is used to trigger the ADC conversions. ISR2 is responsible for running the control law and calculating the clock ticks required for the PWM. Once the writes to the shadow registers are complete, the ISR2 enables the ISR1 trigger by writing a valid value (that is, one that is less than the current TBPRD register) to the CMPC register. ISR2 has two variants ISR2_primToSecPowerFlow and ISR2_secToPrimPowerFlow, one for the power flow from primary to secondary side and other for secondary to primary. This is done to to optimize CPU cycles when controlling power flow in different directions. For simplicity of explaination either of them are just referred to as ISR2 in respective labs. Depending on the timing, the ISR1 may nest ISR2 for the update writes, which are very timing-critical. Following are the defines that are related to this ISR.

```
#define CLLLC_ISR2_ECAP_BASE ECAP1_BASE
#define CLLLC_ISR2_PWM_BASE EPWM5_BASE
#define CLLLC_ISR2_TRIG INT_ECAP1
#define CLLLC_ISR2_PIE_GROUP INTERRUPT_ACK_GROUP4

#define CLLLC_ISR2_TRIG_CLA CLA_TRIGGER_ADCA2
```

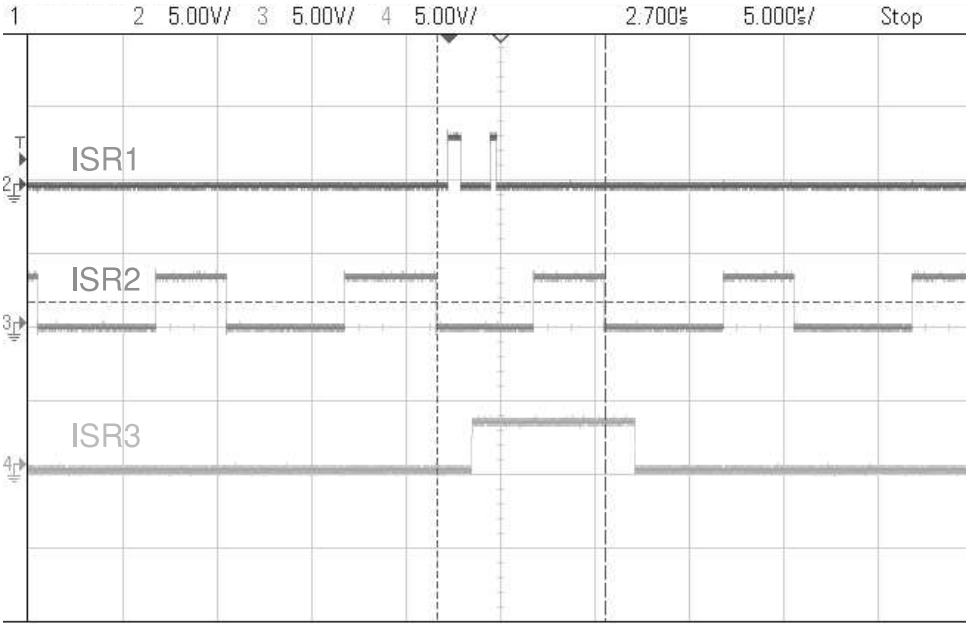
ISR3 is triggered by ADCINT2, which is initiated by a conversion that is started using a CPU timer. It is used to run housekeeping functions such as doing a running average on the currents and voltage signals to remove noise, and running the slew rate function for commanded references.

```
#define CLLLC_ISR3_TIMEBASE CLLLC_TASKC_CPUTIMER_BASE
#define CLLLC_ISR3_PERIPHERAL_TRIG_BASE ADCC_BASE
#define CLLLC_ISR3_TRIG INT_ADCC2

#define CLLLC_ISR3_PIE_GROUP INTERRUPT_ACK_GROUP10
```

With this, the interrupts can be nested easily. Figure 30 shows the nesting of the three interrupts occurring. The image is taken for the system in open loop and when a period change is initiated through the watch window, hence only one time ISR1 trigger is observed. For a closed-loop system, the period will only have minor changes from one control ISR cycle to the other and hence the ISR1 will be triggered repeatedly.

Figure 30. Three-Level Nested ISRs



Additionally, CPU timers are used to trigger slow background tasks (these are not interrupt-driven but polled).

"A" tasks are triggered at TASKA_FREQ, which is 100 Hz. The SFRA GUI must be called at this rate. One task, A1, is executed at this rate.

"B" tasks are triggered at TASKB_FREQ, which is 10 Hz. These are used for some basic LED toggles and state machine items that are not timing-critical. Three tasks—B1, B2, B3—are serviced by this; hence, each has an execution rate of 3.33 Hz.

```
#define TASKA_FREQ 100
#define TASKB_FREQ 10
```

The software of this reference design is organized in eight labs, with incremental builds (INCR_BUILD) and project options (CLLLC_CONTROL_MODE, CLLLC_SEC_CONNECTED_IN_BATTERY_EMULATION_MODE). These tests simplify the system bringup and design.

Lab 1: Primary to Secondary Power Flow, Open Loop Check PWM driver, no high power applied to the board. See Section 3.2.2.1

Lab 2: Primary to Secondary Power Flow, Open Loop Check PWM driver and ADC with protection, resistive load connected on secondary. See Section 3.2.2.2.

Lab 3: Primary to Secondary Power Flow, Closed Voltage Loop Check, with resistive load connected on secondary. See Section 3.2.2.3.

Lab 4: Primary to Secondary Power Flow, Closed Current Loop Check, with resistive load connected on secondary. See Section 3.2.2.4.

Lab 5: Primary to Secondary Power Flow, Closed Current Loop Check, with resistive load connected on secondary in parallel to a voltage source to emulate a battery connection on secondary side. See Section 3.2.2.5.

Lab 6: Secondary to Primary Power Flow, Open Loop Check PWM driver, no high power applied to the board. See Section 3.2.2.6

Lab 7: Secondary to Primary Power Flow, Open Loop Check PWM driver and ADC with protection, resistive load connected on primary. See Section 3.2.2.7

Lab 8: Secondary to Primary Power Flow, Closed Voltage Loop Check, with resistive load connected on primary. See Section 3.2.2.8

These defines are in the "*settings.h*" file, and can be changed through the powerSUITE settings page.

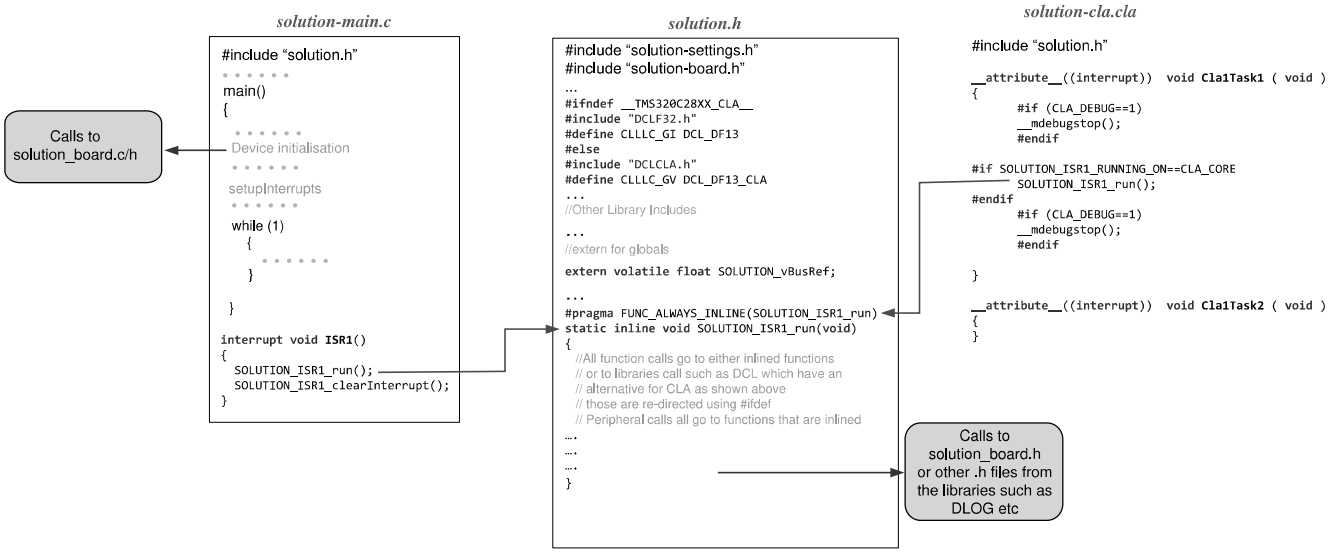
3.1.2.3 Using CLA on C2000 MCU to Alleviate CPU Burden

The control law accelerator (CLA) is a coprocessor available on the C2000 MCU family of devices. This coprocessor enables offloading the control-ISR functions from the main C28x CPU core. To run the control ISR on the CLA for solutions supported in powerSUITE, selection is achieved through a drop-down menu on the powerSUITE SYSCFG page.

The software structure of the powerSUITE solution is designed such that offloading the task to the CLA is simply a drop-down menu selection. The code is not duplicated and a single source for the solution algorithm is maintained even when code is run on the CLA or the C28x, see Figure 31. To enable single source development, the code is kept inside the "*solution.h*" file. This enables the CLA task to see the source and compile it for the CLA.

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

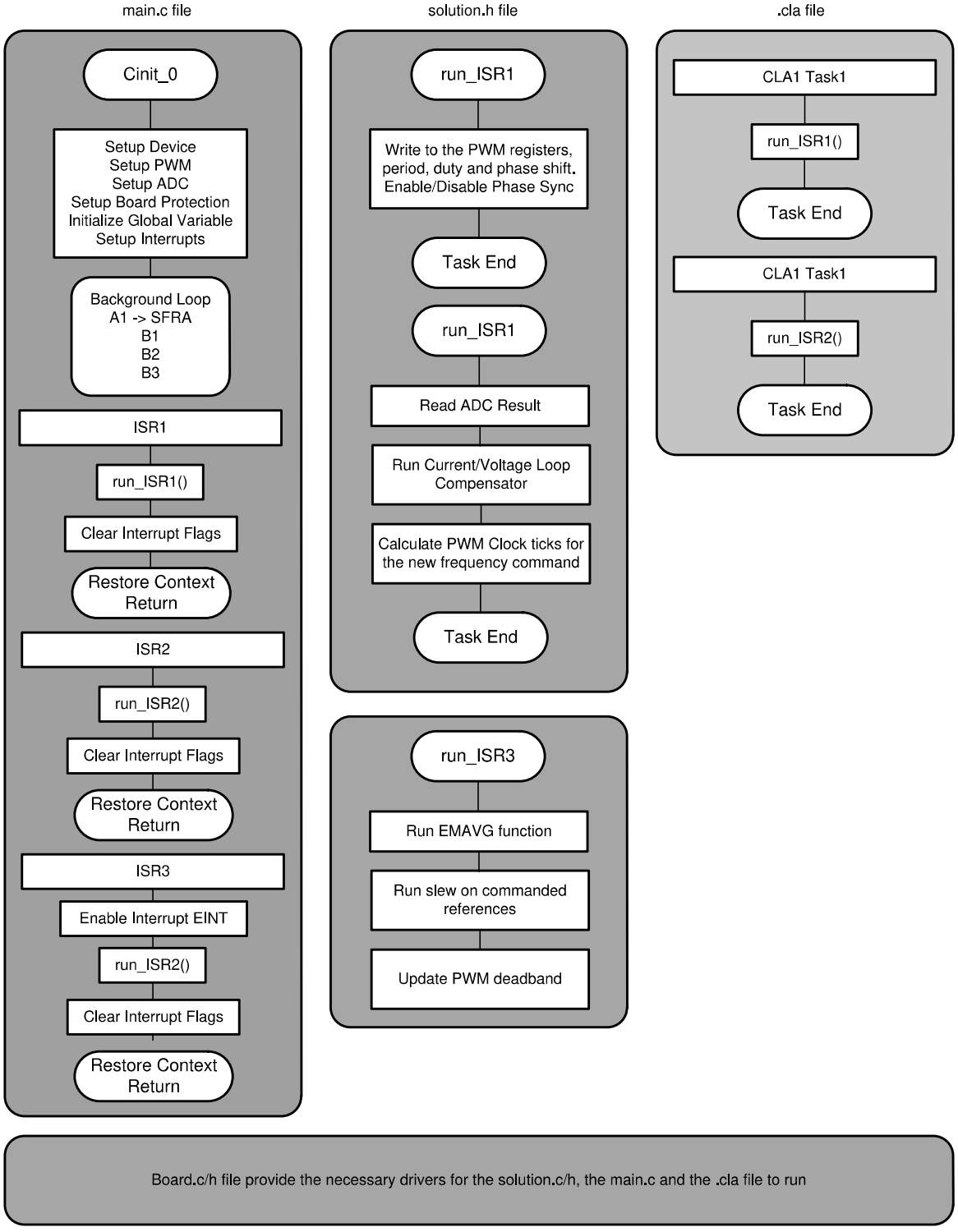
Figure 31. Software Architecture to Support CLA



On this design, ISR1 and ISR2 are offloaded to the CLA when the selection to run the system on CLA is made. On the F28004x, the CPU use is approximately 42% for the 100-kHz loop (not including advanced options such as phase-shedding, adaptive dead time, SFRA running, and so forth) and 11% for the 10-kHz loop that runs the voltage loop and instrumentation functions. Thus, the total CPU use is approximately 53%. With the CLA option, the CPU burden is reduced to 0% when both ISRs are offloaded to the CLA. For more information on the CLA, visit the Control Law Accelerator (CLA) Hands-On Workshop website and refer to the respective device technical reference manuals.

To enable single source development, the code for the ISR1 and ISR2 run is kept inside the "*solution.h*" file. This enables the CLA task to see the source and compile it for the CLA. (As ISR3 is not run on the CLA, code for ISR3 is kept inside the "*solution.c*" file.) This is shown in the project software diagram, Figure 32.

Figure 32. Project Software Diagram Explained

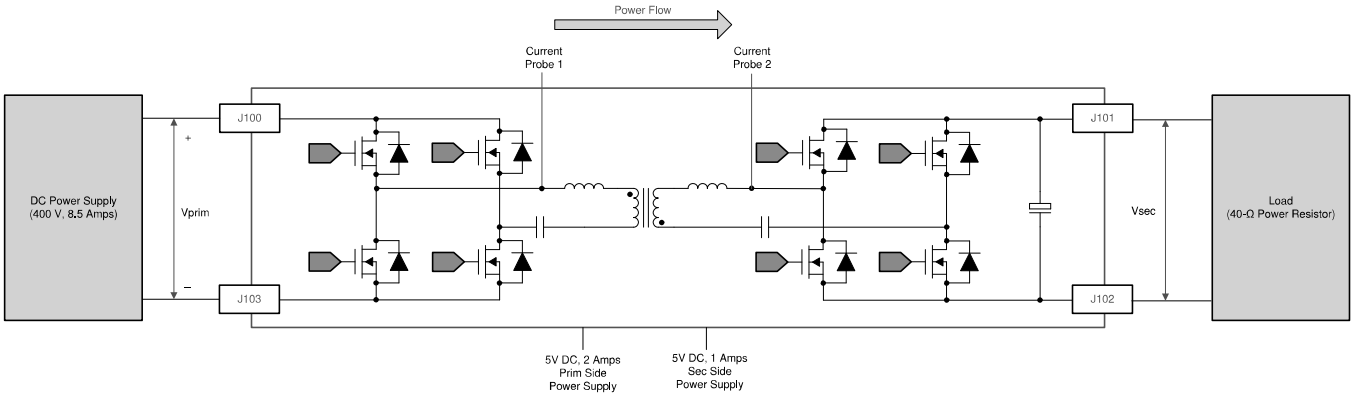


3.2 Testing and Results

3.2.1 Test Setup (Initial)

Hardware test setup is shown in Figure 33 to begin testing this design. Detailed hardware setup is described in Section 3.1.1.

Figure 33. Hardware Setup to Run Software



3.2.2 Test Procedure

3.2.2.1 Lab 1. Primary to Secondary Power Flow, Open Loop Check PWM Driver

This lab option is primarily provided as a focused test just for the PWM from a software perspective so that it can be ran independent of the hardware connections of the reference design. With this lab the user can run the code on a C2000 control card or launchpad just to observe the PWM waveforms.

Steps used to load and run are similar to Section 3.2.2.2 .

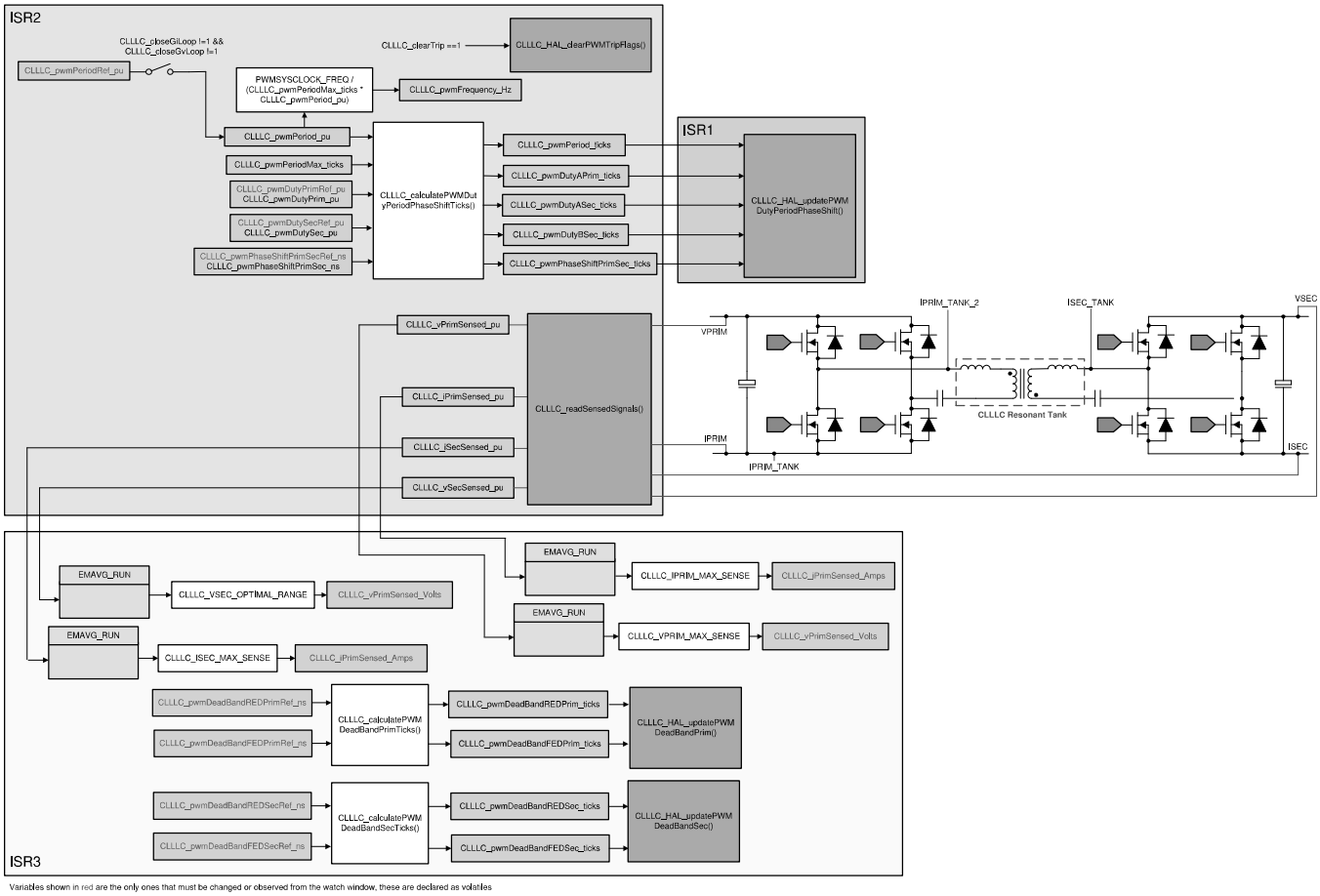
This lab can be easily skipped and the user can go to Lab 2 directlty if no changes to the PWM driver are anticipated. Hence this lab procedure is not documented as it is primarlily for PWM driver development and debug purpose.

3.2.2.2 Lab 2. Primary to Secondary Power Flow, Open Loop CheckPWM Driver and ADC with protection, Resistive Load Connected on Secondary

In this build, the board is excited in open-loop fashion with a specified frequency that can be changed through the watch window. The frequency is controlled with the CLLLC_pwmPeriodRef_pu variable.

This build verifies the sensing of feedback values from the power stage and also operation of the PWM gate driver, and ensures there are no hardware issues. Additionally, calibration of input and output voltage sensing can be performed in this build. Figure 34 shows the software structure for this build.

Figure 34. Lab 1 and 2 Software Diagram



3.2.2.2.1 Setting Software Options for Lab 2

1. To get started, open the CCS project, as outlined in Section 3.1.2.1. If using the powerSUITE page, go to step 2, otherwise jump to step 3.
2. Open the SYSCFG page and select under the *"Project Options"* section:
   - *"Lab 2: Open Loop PWM with Protection, Prim to Sec Power Flow"* for the Lab
   - Under "Tuning", select SFRA to run on the voltage to measure the voltage loop plant.

   Save the page.
3. The following defines are set in the "*settings.h*" file for this build. These will be set by the powerSUITE page automatically.

```
#if CLLLC_LAB == 2
#define CLLLC_CONTROL_RUNNING_ON C28x_CORE
#define CLLLC_POWER_FLOW CLLLC_POWER_FLOW_PRIM_SEC
#define CLLLC_INCR_BUILD CLLLC_OPEN_LOOP_BUILD
#define CLLLC_TEST_SETUP CLLLC_TEST_SETUP_RES_LOAD
#define CLLLC_PROTECTION CLLLC_PROTECTION_ENABLED
#define CLLLC_SFRA_TYPE  CLLLC_SFRA_VOLTAGE
#define CLLLC_SFRA_AMPLITUDE (float32_t)CLLLC_SFRA_INJECTION_AMPLITUDE_LEVEL2
#endif
```

3.2.2.2.2 Building and Loading the Project and Setting up Debug Environment

1. Right-click on the project name and click “Rebuild Project”.
2. The project will build successfully.
3. In the Project Explorer, make sure the correct target configuration file is set as Active under targetConfigs (see Figure 29).
4. Then, click “Run → Debug” to launch a debugging session. In case of dual-CPU devices, a window may appear for the user to select the CPU on which the debug is to be performed. In this case, select CPU1.
5. The project will then load on the device and the CCS debug view will become active. The code will halt at the start of the main routine.
6. To add the variables in the watch/expressions window, click View → Scripting Console to open the scripting console dialog box. On the upper right corner of this console, click on "open" and then browse to the “*setupdebugenv_lab2and7.js*” script file located inside the project folder. This will populate the watch window with the appropriate variables needed to debug the system.
7. Click on the Continuous Refresh button  on the watch window to enable continuous update of values from the controller. The watch window will appear as shown in Figure 35.

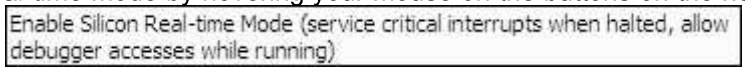
Figure 35. Lab 2 Expression Window

Variables | Expressions | Registers | Breakpoints

Expression	Type	Value	Address
CLLLC_buildLevel.CLLLC_BuildLevel_Enum	enum <unnamed>	openLoopCheck	0x0000805E@Data
CLLLC_pwmSwState.CLLLC_PwmSwState_Enum	enum <unnamed>	pwmSwState_primPWMActive_SecPWMActiveSynchRectification	0x00008058@Data
CLLLC_powerFlowState.CLLLC_PowerFlowState_Enum	enum <unnamed>	powerFlow_BatteryCharging	0x00008060@Data
CLLLC_tripFlag.CLLLC_TripFlag_Enum	enum <unnamed>	noTrip	0x0000805C@Data
CLLLC_clearTrip	long	0	0x00008050@Data
CLLLC_vPrimSensed_Volts	float	0.149676159	0x00008076@Data
CLLLC_iPrimSensed_Amps	float	0.051383622	0x0000804C@Data
CLLLC_vSecSensed_Volts	float	0.450056016	0x00008020@Data
CLLLC_iSecSensed_Amps	float	0.0225450285	0x0000801C@Data
CLLLC_pwmPeriodRef_pu	float	0.599041522	0x00008040@Data
CLLLC_pwmPhaseShiftPrimSecRef_ns	float	81.0	0x00008032@Data
CLLLC_pwmDeadBandREDPrimRef_ns	float	102.300003	0x0000807E@Data
CLLLC_pwmDeadBandFEDPrimRef_ns	float	102.599998	0x00008086@Data
CLLLC_pwmFrequency_Hz	float	500800.0	0x0000803C@Data
EPwm1Regs.TBPRD	unsigned int	99	0x00004063@Data
EPwm2Regs.TBPRD	unsigned int	99	0x00004163@Data
EPwm3Regs.TBPRD	unsigned int	99	0x00004263@Data
EPwm4Regs.TBPRD	unsigned int	99	0x00004363@Data
EPwm1Regs.TZFLG	union TZFLG_REG	{all=4,bit={INT=0,CBC=0,OST=1,DCAEVT1=0,DCAEVT2=0...}}	0x00004093@Data
Add new expression			

3.2.2.2.3 Using Real-time Emulation

Real-time emulation is a special emulation feature that allows windows within Code Composer Studio 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 your mouse on the buttons on the horizontal toolbar and clicking the  

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

3.2.2.2.4 Running the Code

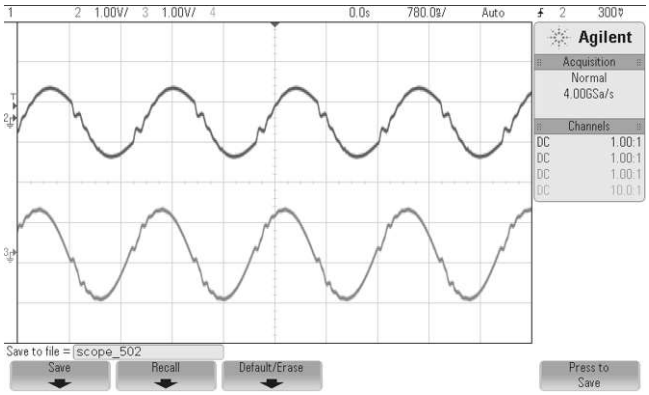
1. Run the project by clicking  .
2. Now, clear the trip by writing “1” to the CLLLC_clearTrip variable.
3. In the watch view, check if the CLLLC_vPrimSensed_Volts, CLLLC_iPrimSensed_Amps, CLLLC_vSecSensed_Volts, and CLLLC_iSecSensed_Amps variables are updating periodically. (Note: As no power is applied right now, these will be close to zero.)
4. Now, slowly increase the input VPRIM DC voltage from 0 V to 400 V. Make sure CLLLC_vPrimSensed_Volts displays the correct values.
5. By default, the CLLLC_pwmPeriodRef_pu variable is set to 0.599, as shown in Figure 36, which is 500.8 kHz. This is close to the series resonant frequency of the converter; however, due to variation in the components on the actual hardware, it can be lower or higher than the series resonant frequency. For example, in Figure 37, we see frequency slightly lower than series resonant frequency.

Figure 36. Lab 2 Expression Window, at Resonance

Variables | Expressions | Registers | Breakpoints

Expression	Type	Value	Address
CLLLC_buildLevel.CLLLC_BuildLevel_Enum	enum <unnamed>	openLoopCheck	0x0000805E@Data
CLLLC_pwmSwState.CLLLC_PwmSwState_Enum	enum <unnamed>	pwmSwState_primPWMActive_SecPWMActiveSynchRectification	0x00008058@Data
CLLLC_powerFlowState.CLLLC_PowerFlowState_Enum	enum <unnamed>	powerFlow_BatteryCharging	0x00008060@Data
CLLLC_tripFlag.CLLLC_TripFlag_Enum	enum <unnamed>	noTrip	0x0000805C@Data
CLLLC_clearTrip	long	0	0x00008050@Data
CLLLC_vPrimSensed_Volts	float	403.606567	0x00008076@Data
CLLLC_iPrimSensed_Amps	float	4.82742071	0x0000804C@Data
CLLLC_vSecSensed_Volts	float	296.258301	0x00008020@Data
CLLLC_iSecSensed_Amps	float	6.27206373	0x0000801C@Data
CLLLC_pwmPeriodRef_pu	float	0.599041522	0x00008040@Data
CLLLC_pwmPhaseShiftPrimSecRef_ns	float	81.0	0x00008032@Data
CLLLC_pwmDeadBandREDPrimRef_ns	float	102.300003	0x0000807E@Data
CLLLC_pwmDeadBandFEDPrimRef_ns	float	102.599998	0x00008086@Data
CLLLC_pwmFrequency_Hz	float	500800.0	0x0000803C@Data
EPwm1Regs.TBPRD	unsigned int	99	0x00004063@Data
EPwm2Regs.TBPRD	unsigned int	99	0x00004163@Data
EPwm3Regs.TBPRD	unsigned int	99	0x00004263@Data
EPwm4Regs.TBPRD	unsigned int	99	0x00004363@Data
EPwm1Regs.TZFLG	union TZFLG_REG	{all=0,bit={INT=0,CBC=0,OST=0,DCAEVT1=0,DCAEVT2=0...}}	0x00004093@Data
Add new expression			

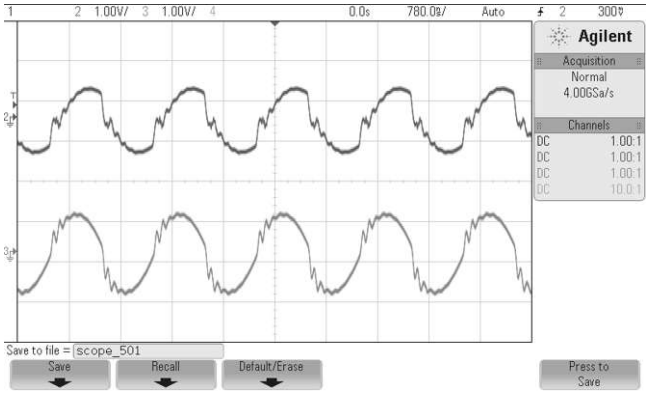
Figure 37. Lab 2, Primary (ch2) and Secondary (ch3) Currents at Resonance



6. The VSEC variable will show a voltage of close to 300 V per the tank gain designed. Verify that CLLLC_vSecSensed_Volts shows the correct voltage. This verifies the voltage sensing on the board.
7. With the load specified in the test conditions, the current from the PRIM and SEC side will be close to 4.8 A for CLLLC_iPrimSensed_Amps, and 6.8 A for CLLLC_iSecSensed_Amps.

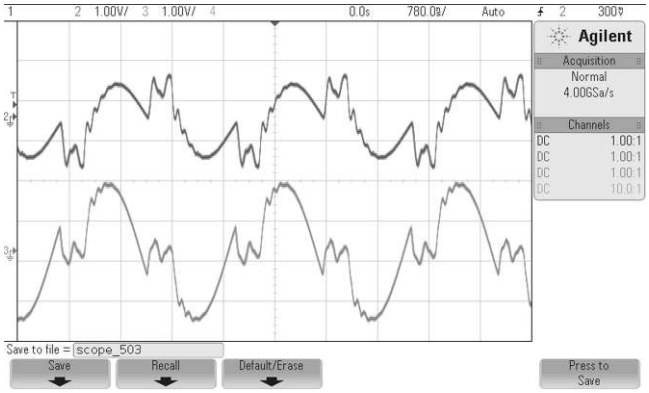
8. Next, to see operation under different frequencies (that is, above resonance, below resonance), change the CLLLC_pwmPeriodRef_pu variable to be 0.47 which will correspond to a frequency of 639 kHz. The waveform under this condition is shown in Figure 38.

Figure 38. Lab 2, Primary (ch2) and Secondary (ch3) Currents Above Series Resonance Frequency



9. Next, test behavior with lower than series resonant frequency by entering 0.8 as the CLLLC_pwmPeriodRef_pu, which will make generate frequency of 374kHz. In this case, the primary current will become discontinuous and the secondary side duty cycle will modulate to achieve diode emulation shown in Figure 39.

Figure 39. Lab 2, Primary (ch2) and Secondary (ch3) Currents Below Series Resonance Frequency

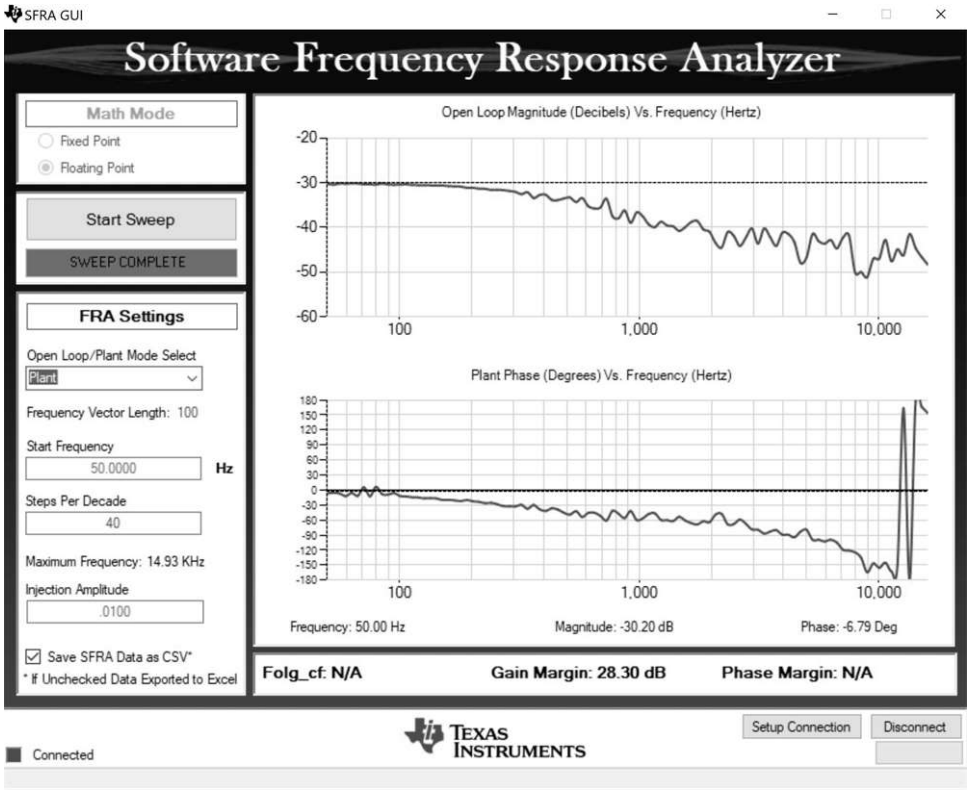


10. This verifies at a basic level the PWM driver and connection of hardware.

3.2.2.2.5 Measure SFRA Plant for Voltage Loop

1. The SFRA is integrated in the software of this build to measure the plant response which can then be used to design a compensator. To run the SFRA, keep the project running, and from the SYSCFG page, click on the SFRA icon. The SFRA GUI will pop up.
2. Select the options for the device on the SFRA GUI; for example, for F280049, select floating point. Click on setup connection. In the pop-up window, uncheck the boot-on-connect option and select an appropriate COM port. Click Ok. Return to the SFRA GUI and click Connect.
3. The SFRA GUI will connect to the device. A SFRA sweep can now be started by clicking “Start Sweep”. The complete SFRA sweep will take a few minutes to finish. Activity can be monitored by seeing the progress bar on the SFRA GUI and also by checking the flashing of blue LED on the back of the control card, which indicates UART activity. Once complete, a graph with the measurement will appear, as shown in Figure 40. (Note that the open-loop measurement is not valid in the lab as the loop is not closed. The user must only refer to the plant measurement.)

Figure 40. SFRA Open Loop Plot for the Closed Voltage Loop (Vprim 400 V, Vsec 300 V, Power 1.972 kW, Fsw 500 kHz)



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. SFRA can be run at different frequency set points to cover the range of operation of the system. A compensator will be designed using these measured plots in the next lab; therefore, remember this timestamp, or rename the "*SFRA.csv*" file to a convenient name that is easy to identify.

Repeat the analysis at different frequency points, the plant gain will be different at different frequency points, see Figure 41for gain measured at 333kHz and see Figure 42for gain measured at 680kHz. Hence a compensator needs to be chosen that will be stable across the frequency range of the converter. All the runs will be saved in CSV file and can then be imported into compensation designer to check stability across the range of operation.

Figure 41. SFRA Open Loop Plot for the Closed Voltage Loop (Vprim 400 V, Vsec 320 V, Power 2.174 kW, Fsw 333 kHz)

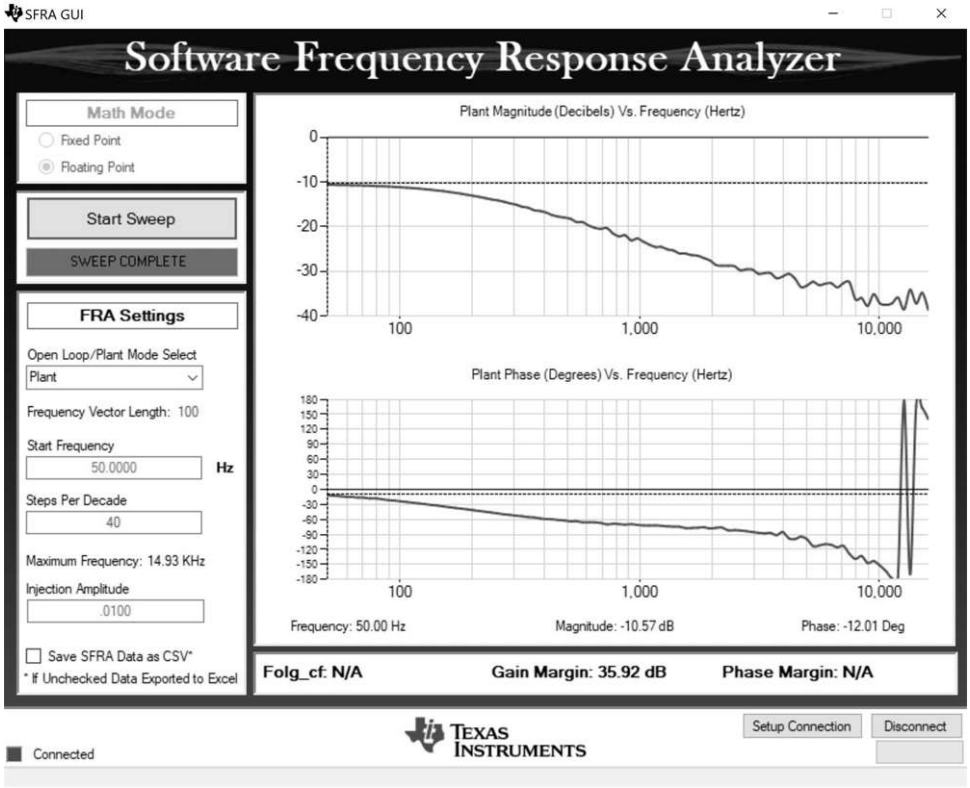
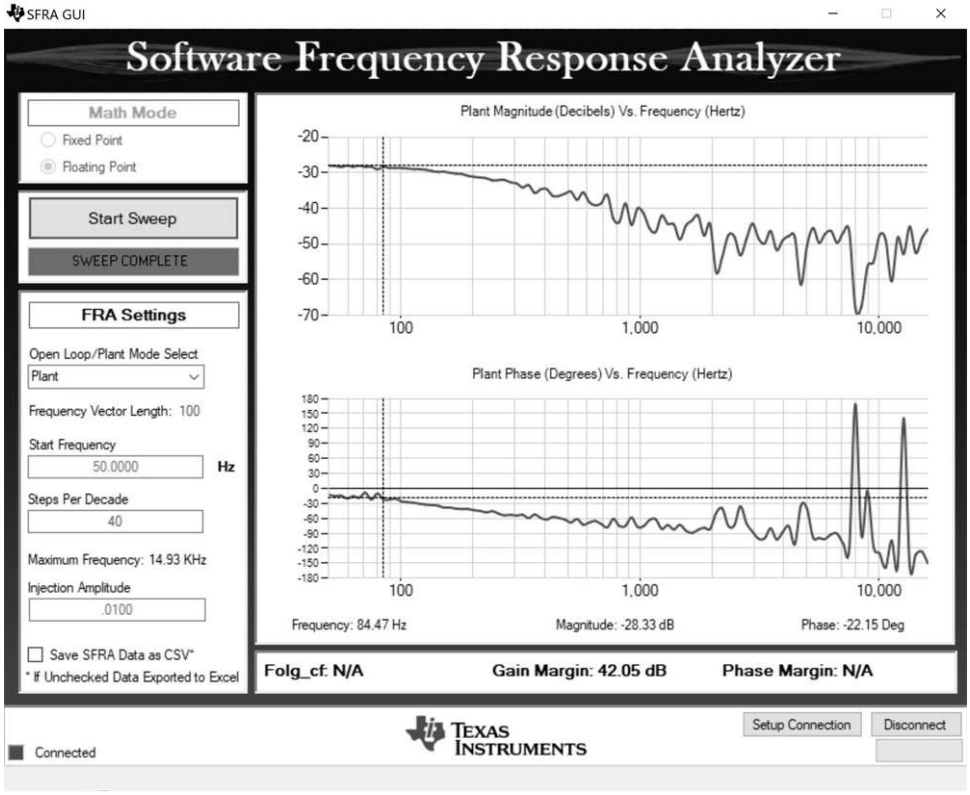


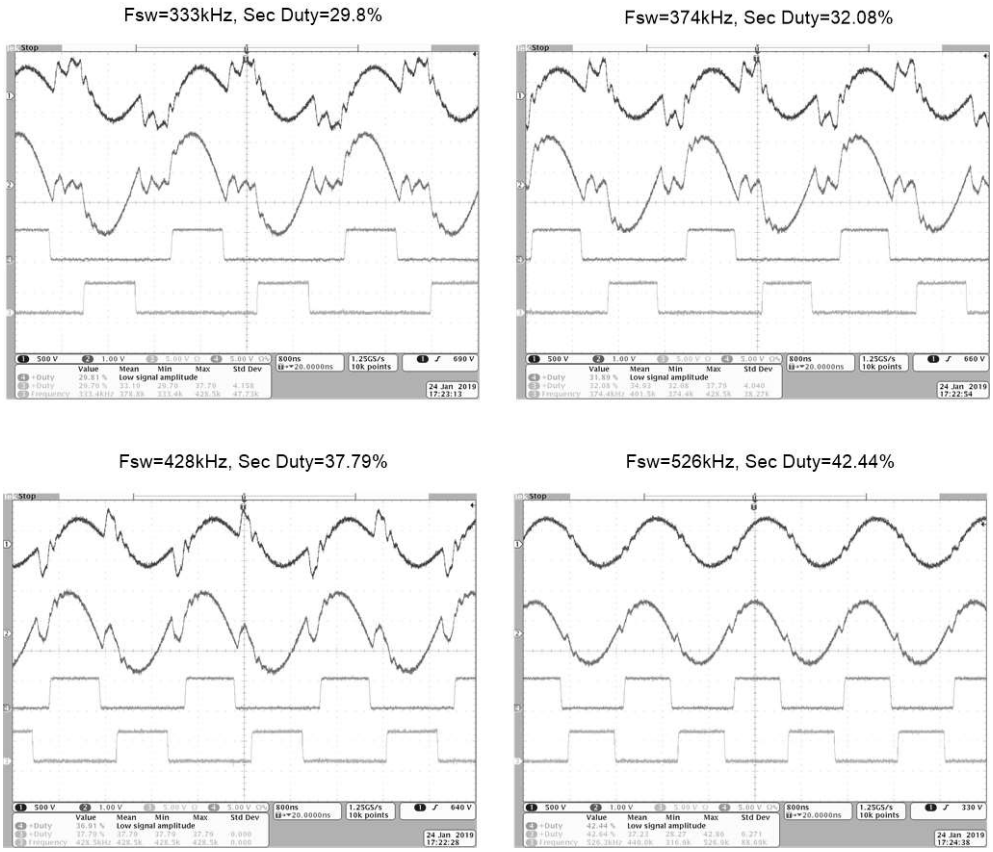
Figure 42. SFRA Open Loop Plot for the Closed Voltage Loop (Vprim 400 V, Vsec 293 V, Power 1.828 kW, Fsw 680 kHz)



3.2.2.2.6 Verify Active Synchronous Rectification

1. Optionally, to verify active synchronous rectification, the user can also probe the PWM signals and see the change in duty cycle. To connect the probe for it, the user must first stop the power stage as outlined below and de-energize all circuits under test.

Figure 43. Active Synchronous Rectification Check, ch1 → IPRIM_TANK, ch2 → ISEC_TANK, ch3/ch4 → SEC_LEG1_PWMH/L

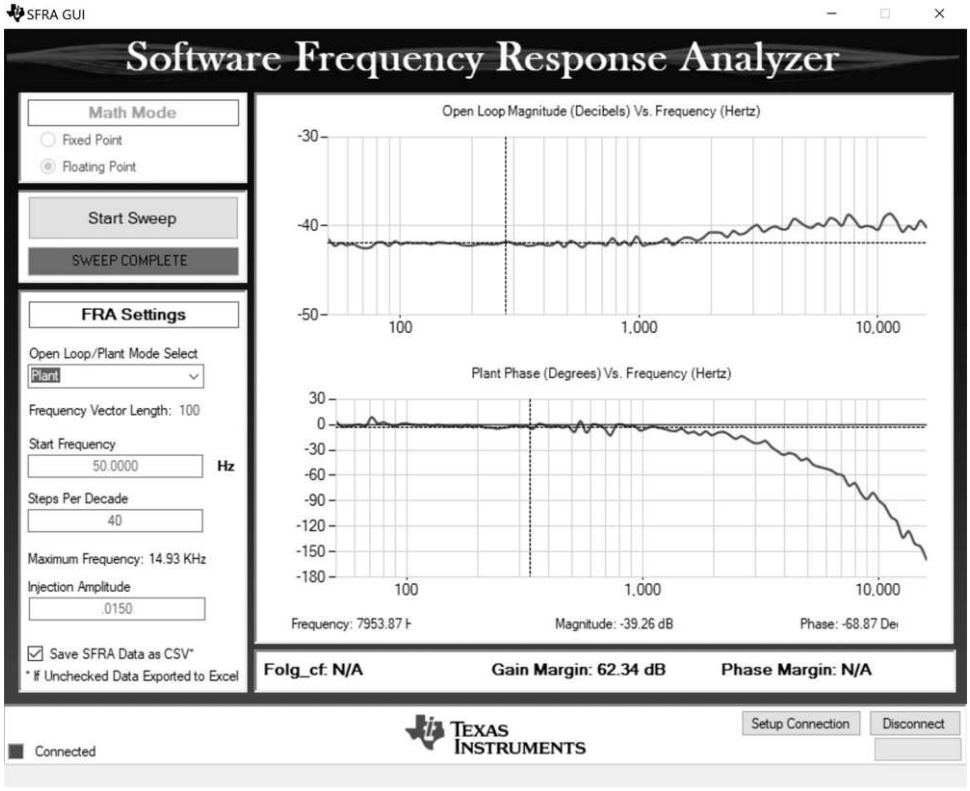


2. Once finished, reduce the input voltage, VPRIM, to zero. Watch for the voltages in the watch window to reduce down to zero.
3. Fully halting the MCU when in real-time mode is a two-step process. First, halt the processor by using the Halt button on the toolbar , or by using Target → Halt. Then, take the MCU out of real-time mode by clicking on . Finally, reset the MCU .
4. Close the CCS debug session by clicking on Terminate Debug Session (Target → Terminate all).

3.2.2.2.7 Measure SFRA Plant for Current Loop

1. Now, return to the SYSCFG page, and select current in the SFRA options to measure the current loop plant.
2. Rebuild the project, and reload the project. Repeat step 3 (in Section 3.2.2.2.1) to step 2 (in Section 3.2.2.2.5). This time, the SFRA sweep will measure the plant for the current loop. Save this CSV file for use later in the Lab 3. The user can measure this at multiple points to ensure all operating conditions are covered, Figure 44shows the measurement of the current loop plant at 500kHz.

Figure 44. SFRA Plant Measurement for the Current Loop at Vprim 400V, Vsec 295V, Switching Frequency at 500kHz and 1887W



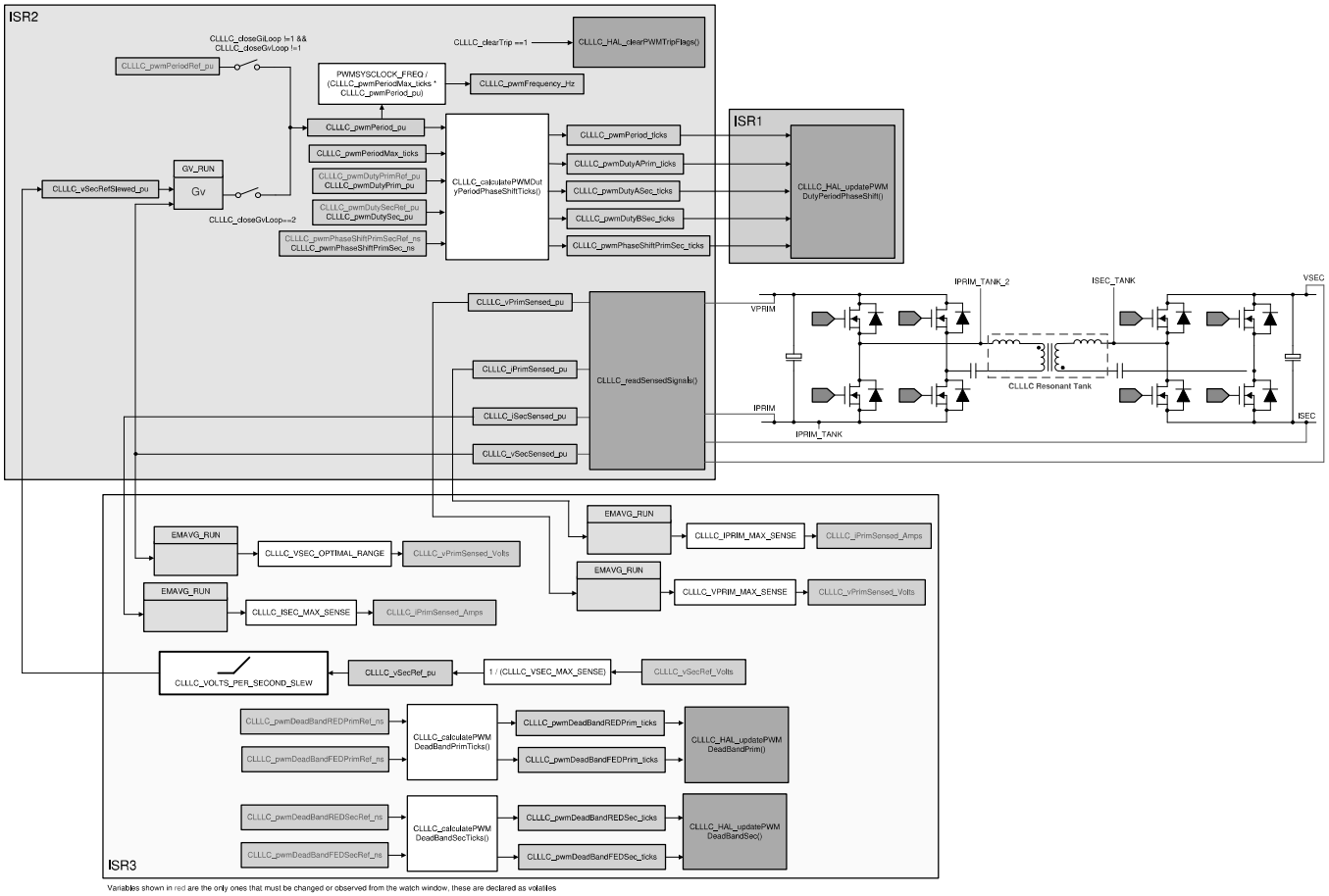
3. This completes the check for this build, the following items are verified on successful completion of this build:
   a. Sensing of voltages and currents and scaling to be correct
   b. Interrupt generation and execution of the build 1 code in ISR1, ISR2, and ISR3
   c. PWM driver and switching
   d. Plant measurement for current and voltage loop

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

3.2.2.3 *Lab 3. Primary to Secondary Power Flow, Closed Voltage Loop Check, With Resistive Load Connected on Secondary*

In this lab, the voltage loop, G_v, is closed with a resistive load at the output. Figure 45 shows the complete software diagram for this build. Hardware is assumed to be set up as shown in Figure 33.

Figure 45. Software Diagram: Closed Voltage Loop Primary to Secondary Power Flow



3.2.2.3.1 *Setting Software Options for Lab 3*

1. To run this lab, make sure the hardware is set up as outlined in the previous sections, Figure 33. Do not supply any high-voltage power to the board yet. If using the powerSUITE page, go to step 2; otherwise, jump to step 8.
2. ***powerSUITE Settings***: On the powerSUITE page,
   a. Select under the *"Project Options"* section *"Lab 3 : Closed Loop Voltage with Resistive Load, Prim to Sec Power Flow"* for the Lab.
   b. Under "Tuning", select SFRA to run on the voltage loop using the drop-down box. Save the page.
3. Under Control Loop Design, options for the current loop tuning will automatically be selected,
   Tuning → Comp Number 1 → DF22.
4. Save the page by "Ctrl-S" and click on the Compensation Designer button  ,
5. The compensation designer will then launch and prompt the user to select a valid SFRA data file. Import the SFRA data from the run in Lab 1 into the compensation designer to design a two-pole, two-zero compensator. It is good to keep more margins during this iteration of the design to ensure that when the loop is closed, the system is stable. Plant data from different runs of the SFRA can be checked to get a stable system under all conditions, for example two runs at 500kHz and 300kHz with the designed compensator are shown to be stable in Figure 46 and Figure 47.

Figure 46. Compensator Design With SFRA Based Plant Measurement for the Voltage Loop When Resistive Load is Connected at the Output, With Measurement Data at 500kHz

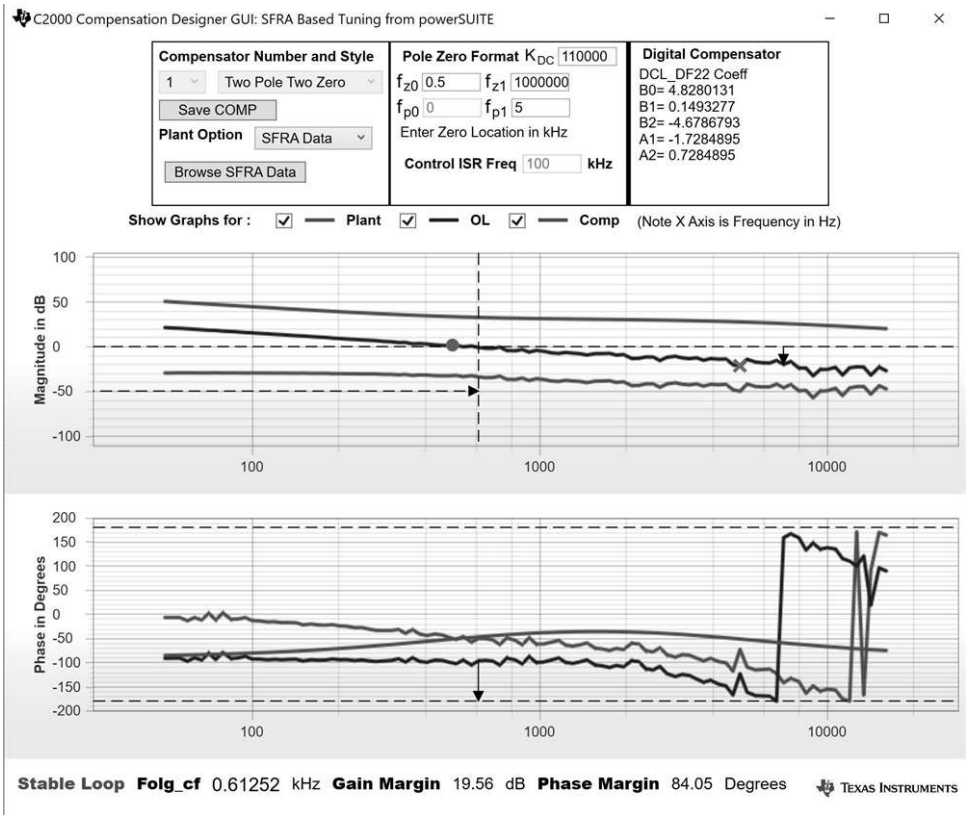
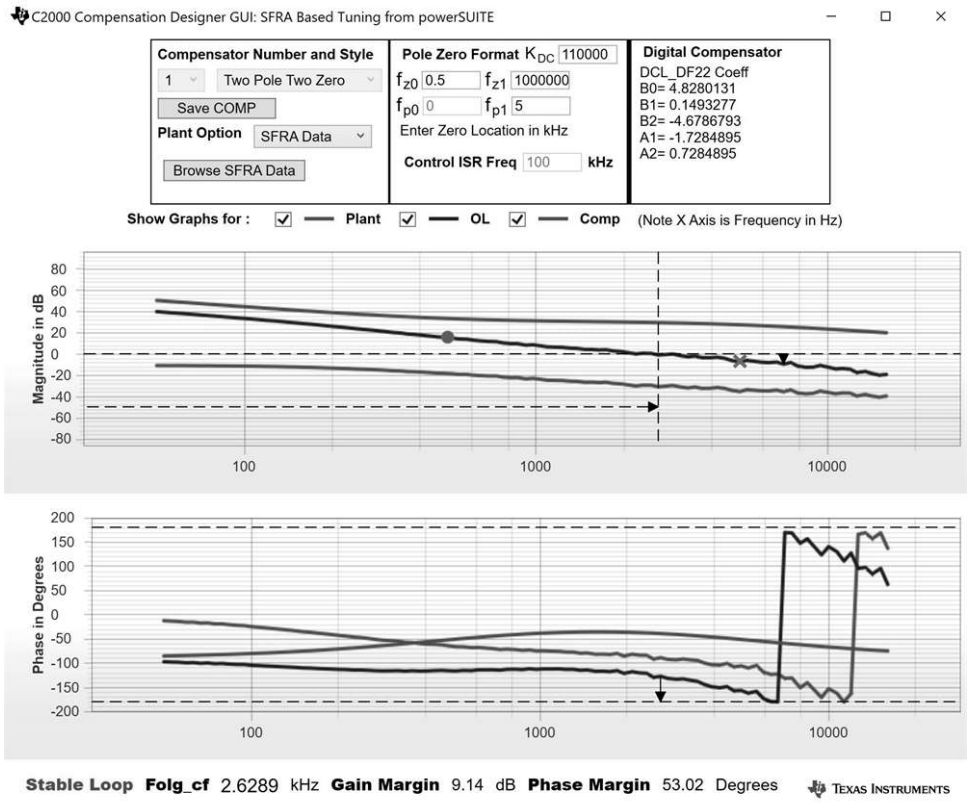


Figure 47. Compensator Design With SFRA Based Plant Measurement for the Voltage Loop When Resistive Load is Connected at the Output, With Measurement Data at 333kHz



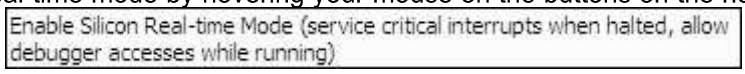
NOTE: The tuning is carried out in DF22 fashion; however, we run the DF13 in the software. This is done because soft-starting the DF13 is easier, whereas not possible with the DF22 structure. The coefficients in both cases remain the same. At the writing of this document, the DF12 structure is not available in DCL.

6. Once satisfied with the compensator design, click on “Save COMP”. This will save the compensator values into the project.
7. Close the compensation designer, return to the powerSUITE page, and save (“Ctrl-S”). This will write the new compensator values to the "*settings.h*" file.

8. The following defines are set in the "*settings.h*" file for this build. These will be set by the powerSUITE page automatically; otherwise, edit the page manually.

```
#if CLLLC_LAB == 3
#define CLLLC_CONTROL_RUNNING_ON C28x_CORE
#define CLLLC_POWER_FLOW CLLLC_POWER_FLOW_PRIM_SEC
#define CLLLC_INCR_BUILD CLLLC_CLOSED_LOOP_BUILD
#define CLLLC_CONTROL_MODE CLLLC_VOLTAGE_MODE
#define CLLLC_TEST_SETUP CLLLC_TEST_SETUP_RES_LOAD
#define CLLLC_PROTECTION CLLLC_PROTECTION_ENABLED
#define CLLLC_SFRA_TYPE  CLLLC_SFRA_VOLTAGE
#define CLLLC_SFRA_AMPLITUDE (float32_t)CLLLC_SFRA_INJECTION_AMPLITUDE_LEVEL1
#endif
```

3.2.2.3.2 Building and Loading the Project and Setting up Debug Environment

1. Now, 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. In case of dual-CPU devices, a window may appear for the user to select the CPU on which the debug is to be performed. In this case, select CPU1.
4. The project will then load on the device and the CCS debug view will become active. The code will halt at the start of the main routine.
5. To add the variables in the watch/expressions window, click View → Scripting Console to open the scripting console dialog box. On the upper right corner of this console, click on "open" to browse to the “*setupdebugenv_lab3.js*” script file located inside the project folder. This will populate the watch window with the appropriate variables needed to debug the system.
6. Click on the Continuous Refresh button  on the watch window to enable continuous update of values from the controller.
7. Enable real-time mode by hovering your mouse on the buttons on the horizontal toolbar and clicking the  

button.

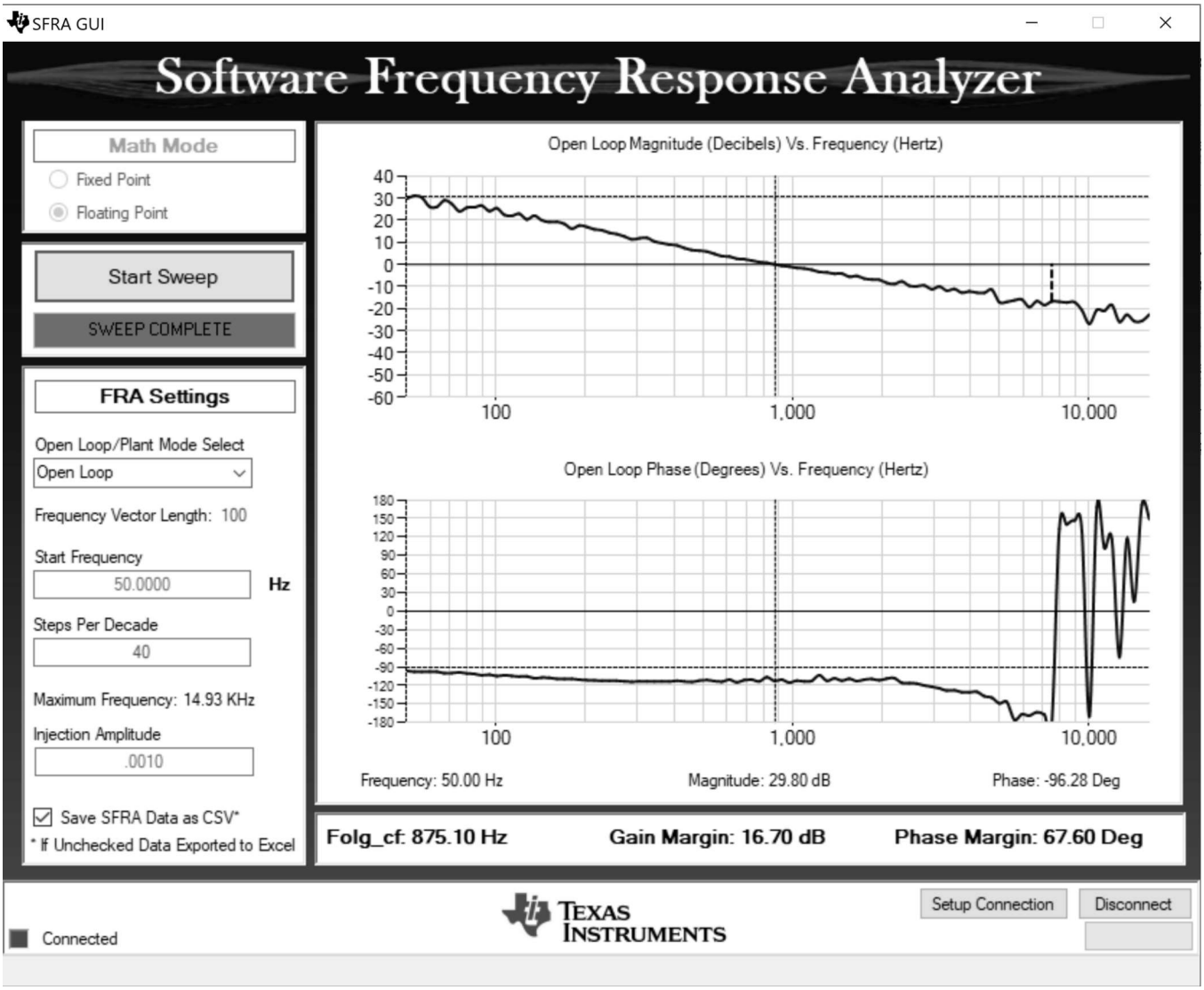
3.2.2.3.3 Running the Code

1. Run the project by clicking 
2. Now, clear the trip by writing “1” to the CLLLC_clearTrip variable. The converter will operate in open loop as the CLLLC_closeGvLoop variable is not yet set to “0”. As there is no soft start implemented in the firmware, first soft-start the voltages on the primary and secondary sides manually.
3. In the watch view, check if the CLLLC_vPrimSensed_Volts, CLLLC_iPrimSensed_Amps, CLLLC_vSecSensed_Volts, and CLLLC_iSecSensed_Amps variables are updating periodically. (Note: As no power is applied right now, these will be close to zero.)
4. Now, slowly increase the input PRIM DC voltage from 0 V to 400 V to soft-start the converter. Make sure CLLLC_vPrimSensed_Volts displays the correct values for VPRIM (that is, close to 400 V).
5. By default, the CLLLC_pwmPeriodRef_pu variable is set to 0.599, which is 500.8 kHz. This is close to the series resonant frequency of the converter; however, due to variation in the components on the actual hardware, it can be lower or higher than the series resonant frequency.
6. For the 400-V primary input, with turns ratio being 1.33, the CLLLC_vSecSensed_Volts variable will be close to 300 V. Set the CLLLC_vSecRef_Volts variable to be 300 V.
7. Now, set the CLLLC_closeGvLoop variable to "1". This will close the voltage loop and the controller will now try to regulate the voltage.
8. Test the closed-loop operation by varying CLLC_vSecRef_Volts from 295 V to 320 V. The user will observe that the CLLLC_vSecSensed_Volts will track this command reference. The converter will operate below series resonant, at resonance, and above resonance. Now, change the voltage back to 300 V to run the SFRA.

3.2.2.3.4 Measure SFRA for Closed Voltage Loop

1. The SFRA is integrated in the software of this build to verify that the designed compensator provides enough gain and phase margin by measuring on hardware. To run the SFRA, keep the project running, and from the SYSCFG page, click on the SFRA icon. The SFRA GUI will pop up.
2. Select the options for the device on the SFRA GUI; for example, for F280049, select floating point. Click on setup connection. In the pop-up window, uncheck the boot-on-connect option and select an appropriate COM port. Click Ok. Return to the SFRA GUI and click Connect.
3. The SFRA GUI will connect to the device. A SFRA sweep can now be started by clicking “Start Sweep”. The complete SFRA sweep will take a few minutes to finish. Activity can be monitored by seeing the progress bar on the SFRA GUI and also by checking the flashing of blue LED on the back of the control card, which indicates UART activity. Once complete, a graph with the open loop plot will appear, as shown in Figure 48.

Figure 48. SFRA Open Loop Plot for the Closed Voltage Loop (Vprim 400 V, Vsec 300 V, Power 1.972 kW, with Resistive Load at the Output)



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 data matches closely to the designed compensator, but it is reasonable to expect deviations because the measurement in open loop is susceptible to error, due to small signal injection, which can drift the DC point of the converter.

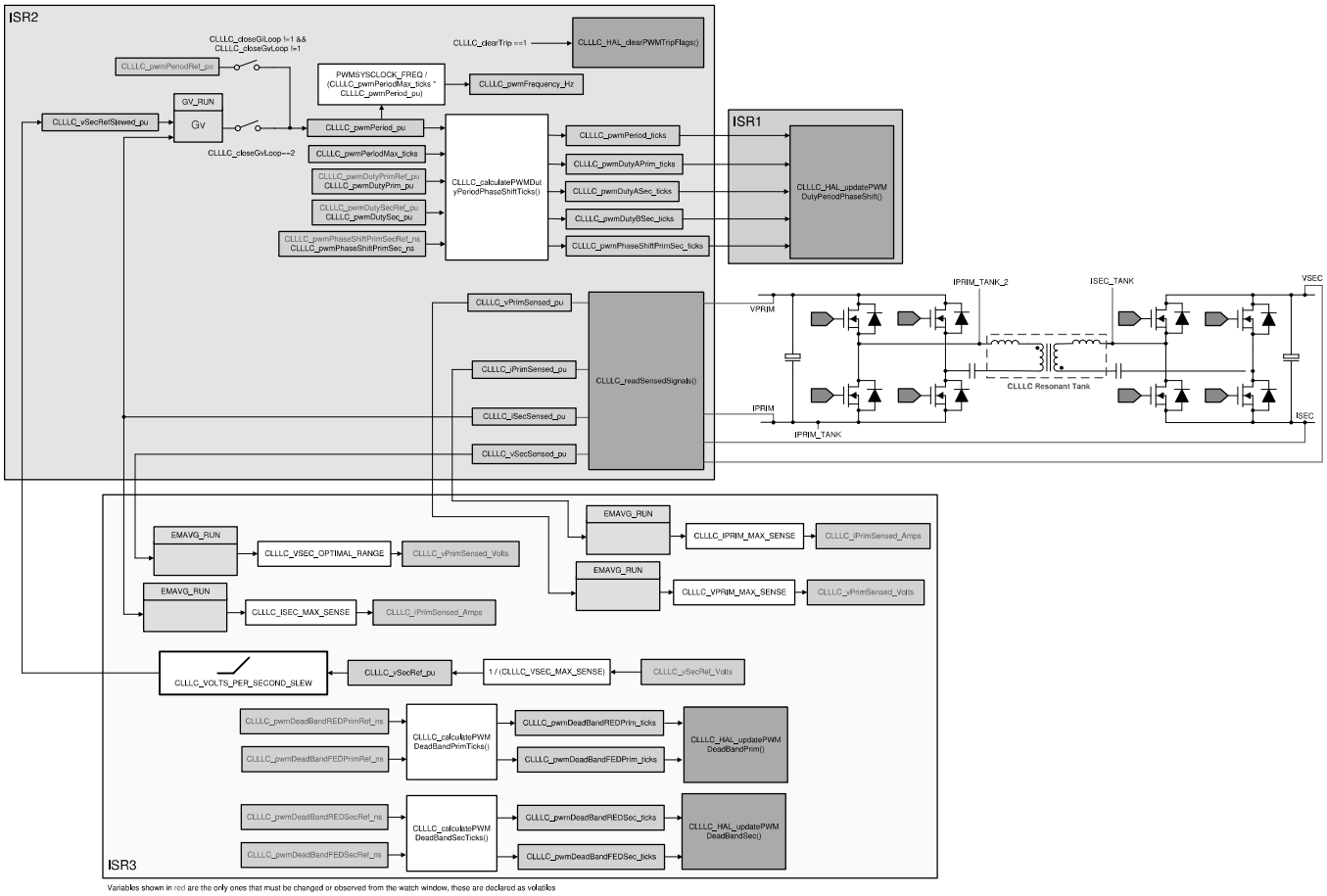
Test the SFRA at different voltages to verify that the system is stable across the operable range.

4. This verifies the voltage loop design.
5. To bring the system to a safe stop, bring the input VPRIM voltage down to zero. Observe the voltages and currents on the watch window go down to zero.
6. Fully halting the MCU when in real-time mode is a two-step process. First, halt the processor by using the Halt button on the toolbar  , or by using Target → Halt. Then, take the MCU out of real-time mode by clicking on  . Finally, reset the MCU  .
7. Close the CCS debug session by clicking on Terminate Debug Session  (Target → Terminate all).

3.2.2.4 Lab 4. Primary to Secondary Power Flow, Closed Current Loop Check, With Resistive Load Connected on Secondary

In this lab, the output current control loop is closed. Figure 49 shows the complete software diagram for this build. Hardware is assumed to be set up as shown in Figure 33.

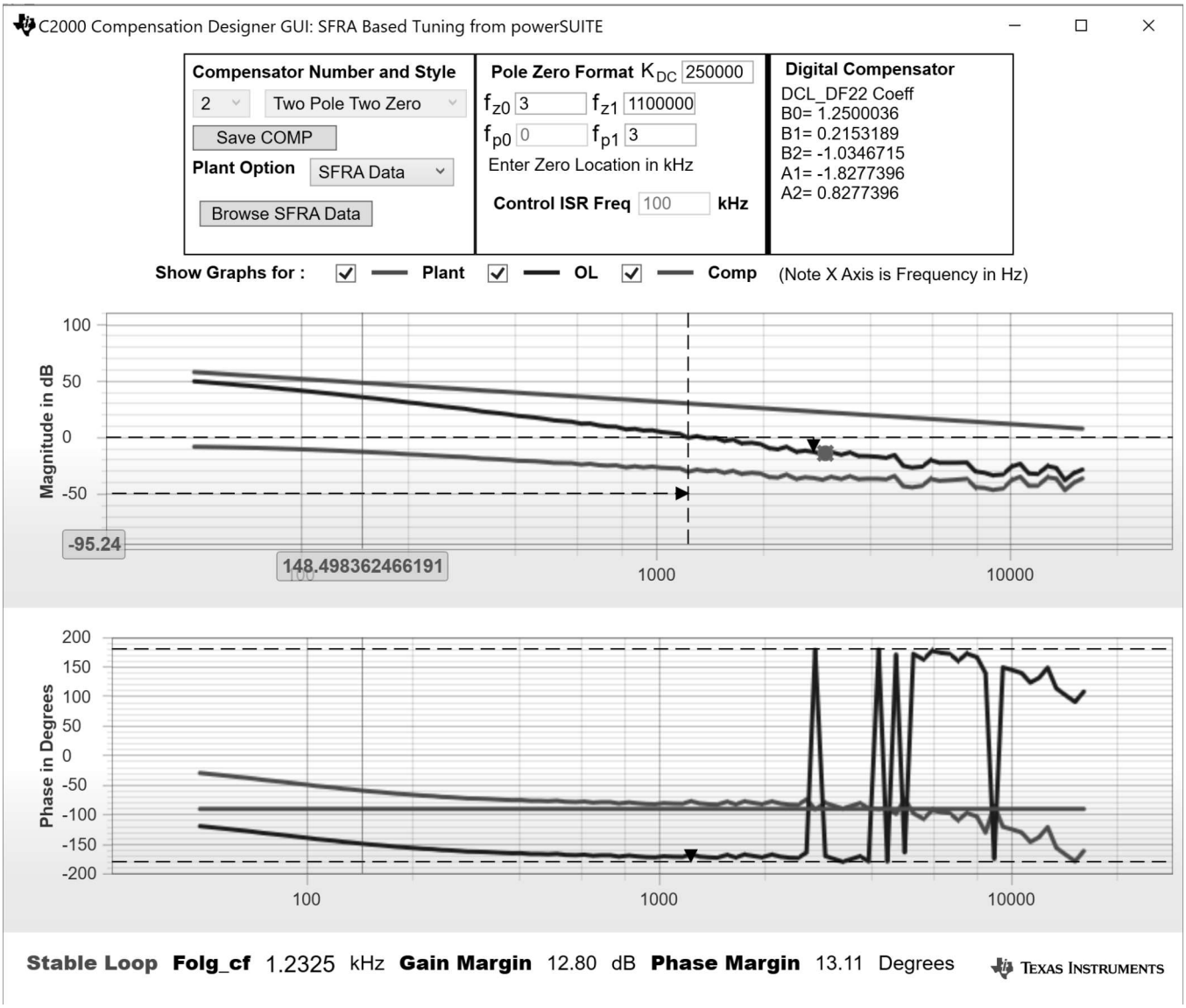
Figure 49. Lab 4 Control Software Diagram: Closed Current Loop



3.2.2.4.1 Setting Software Options for Lab 4

1. If using the powerSUITE page, go to step 2; otherwise, jump to step 8.
2. ***powerSUITE Settings***: On the powerSUITE page,
   a. Select under the *"Project Options"* section *"Lab 3 : Closed Loop: Current"* for the Lab.
   b. Under "Tuning", select SFRA to run on the current loop using the drop-down box. Save the page.
3. Under Control Loop Design, options for the current loop tuning will automatically be selected, Tuning → Comp Number 2 → DF22.
4. Save the page by "Ctrl-S" and click on the Compensation Designer button  .
5. The compensation designer will then launch and prompt the user to select a valid SFRA data file. Import the SFRA data from the run in Lab 1 for the current loop, into the compensation designer to design a two-pole, two-zero compensator. It is good to keep more margins during this iteration of the design to ensure that when the loop is closed, the system is stable. Plant data from different runs of the SFRA can be checked to get a stable system under all conditions.

Figure 50. Compensator Design With SFRA Based Plant Measurement for the Current Loop, Lab 3



NOTE: The tuning is carried out in DF22 fashion; however, we run the DF13 in the software. This is done because soft-starting the DF13 is easier, whereas not possible with the DF22 structure. The coefficients in both cases remain the same. At the writing of this document, the DF12 structure is not available in DCL.

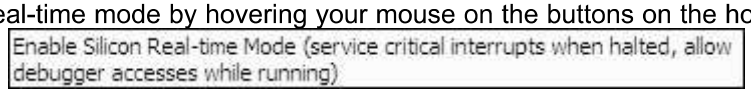
6. Once satisfied with the compensator design, click on “Save COMP”. This will save the compensator values into the project.
7. Close the compensation designer, return to the powerSUITE page, and save (“Ctrl-S”). This will write the new compensator values to the "*settings.h*" file.

8. The following defines are set in the "*settings.h*" file for this build. These will be set by the powerSUITE page automatically; otherwise, edit the page manually.

```
#if CLLLC_LAB_NO == 3
#define CLLLC_INCR_BUILD 2
#define CLLLC_CONTROL_MODE CLLLC_CURRENT_MODE
// 0 means check with resistive load
#define CLLLC_SEC_CONNECTED_IN_BATTERY_EMULATION_MODE 0
#endif

#define CLLLC_SFRA_TYPE       CLLLC_SFRA_CURRENT
```

3.2.2.4.2 Building and Loading the Project and Setting up Debug

1. Now, 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. In case of dual-CPU devices, a window may appear for the user to select the CPU on which the debug is to be performed. In this case, select CPU1.
4. The project will then load on the device and the CCS debug view will become active. The code will halt at the start of the main routine.
5. To add the variables in the watch/expressions window, click View → Scripting Console to open the scripting console dialog box. On the upper right corner of this console, click on "open" to browse to the “*setupdebugenv_build3c.js*” script file located inside the project folder. This will populate the watch window with the appropriate variables needed to debug the system.
6. Click on the Continuous Refresh button  on the watch window to enable continuous update of values from the controller.
7. Enable real-time mode by hovering your mouse on the buttons on the horizontal toolbar and clicking the  

button.

3.2.2.4.3 Running the Code

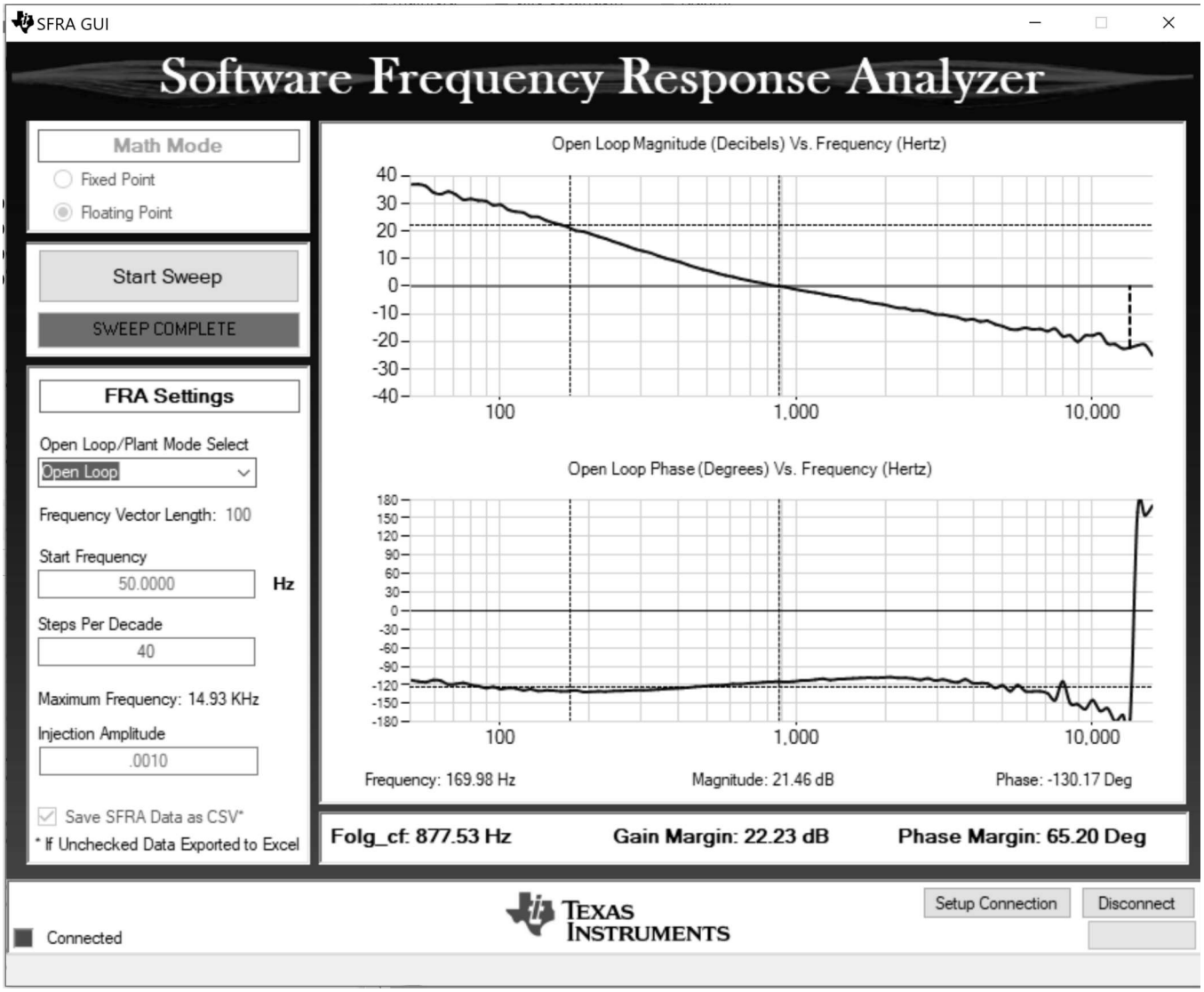
1. Run the project by clicking 
2. Now, clear the trip by writing “1” to the CLLLC_clearTrip variable. The converter will operate in open loop as the CLLLC_closeGvLoop variable is not yet set to “0”. As there is no soft start implemented in the firmware, first soft-start the voltages on the primary and secondary sides manually.
3. In the watch view, check if the CLLLC_vPrimSensed_Volts, CLLLC_iPrimSensed_Amps, CLLLC_vSecSensed_Volts, and CLLLC_iSecSensed_Amps variables are updating periodically. (Note: As no power is applied right now, these will be close to zero.)
4. Now, slowly increase the input PRIM DC voltage from 0 V to 400 V to soft-start the converter. Make sure CLLLC_vPrimSensed_Volts displays the correct values for VPRIM (that is, close to 400 V).
5. By default, the CLLLC_pwmPeriodRef_pu variable is set to 0.6, which is 500.8 kHz. This is close to the series resonant frequency of the converter; however, due to variation in the components on the actual hardware, it can be lower or higher than the series resonant frequency.
6. For the 400-V primary input, with turns ratio being 1.33, the CLLLC_vSecSensed_Volts variable will be close to 300 V. Also, for the load specified in the test conditions, the load will be close to 6.5 A. Set the CLLLC_iSecRef_Amps variable to 6.5 A. If, for some reason, the measured current is different from the 6.5 A, set the Ref close to the measured value. As there is no soft start in the software, it is critical to keep this reference close to the operating point.
7. Now, set the CLLLC_closeGiLoop variable to "1". This will close the current loop and the controller will now try to regulate the current.
8. Test the closed-loop operation by varying CLLC_iSecRef_Amps from 6.3 A to 6.8 A. The user cannot vary the current too much as a resistive load is connected at the output whose voltage will change with current much more than a battery. This rapid increase in voltage can quickly put the converter in a

range that exceeds the controllable range for the fixed VPRIM. Within the small range, the user can see the tracking of the current.

3.2.2.4.4 Measure SFRA for Closed Current Loop

1. The SFRA is integrated in the software of this build to verify that the designed compensator provides enough gain and phase margin by measuring on hardware. To run the SFRA, keep the project running, and from the SYSCFG page, click on the SFRA icon. The SFRA GUI will pop up.
2. Select the options for the device on the SFRA GUI; for example, for F280049, select floating point. Click on setup connection. In the pop-up window, uncheck the boot-on-connect option, select an appropriate COM port, and click Ok. Return to the SFRA GUI and click Connect.
3. The SFRA GUI will connect to the device. A SFRA sweep can now be started by clicking “Start Sweep”. The complete SFRA sweep will take a few minutes to finish. Activity can be monitored by seeing the progress bar on the SFRA GUI; and also by checking the flashing of blue LED on the back of the control card, which indicates UART activity. Once complete, a graph with the open loop plot will appear, as shown in Figure 51.

Figure 51. SFRA Open Loop Plot for the Closed Current Loop (Vprim 400 V, Vsec 300 V, Power 1.972 kW, Lab 3)



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 data matches closely to the designed compensator, but it is reasonable to expect deviations because the measurement in open loop is susceptible to error due to small signal injection which can drift the DC point of the converter.

Test the SFRA at different current set points, making sure the period is not clamped, to verify that the system is stable across the operable range.

4. This verifies the Lab 3 current loop design.
5. To bring the system to a safe stop, bring the input VPRIM voltage down to zero. Observe the voltages and currents on the watch window go down to zero.
6. Fully halting the MCU when in real-time mode is a two-step process. First, halt the processor by using the Halt button on the toolbar  , or by using Target → Halt. Then, take the MCU out of real-time mode by clicking on  . Finally, reset the MCU  .
7. Close the CCS debug session by clicking on Terminate Debug Session  (Target → Terminate all).

3.2.2.5 *Lab 5. Primary to Secondary Power Flow, Closed Current Loop Check, With Resistive Load Connected on Secondary in Parallel to a Voltage Source to Emulate a Battery Connection on Secondary Side*

In this lab, the output current control loop is closed, with a resistive load connected on the secondary in parallel with a voltage source, to emulate a battery connection. Hardware is assumed to be set up as shown in Figure 52. Figure 53 shows the complete software diagram for this build.

Figure 52. Hardware Setup for Lab 5

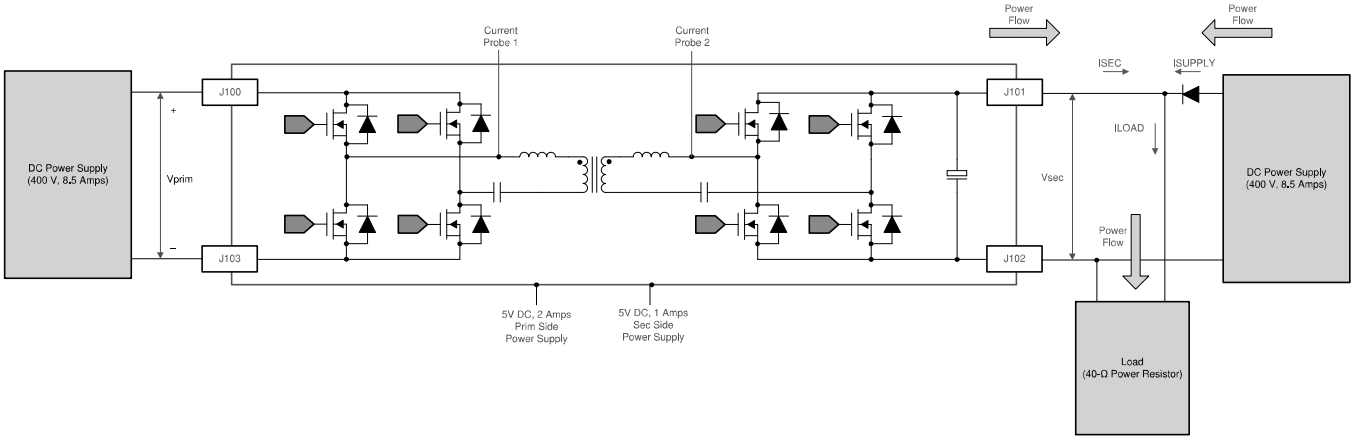
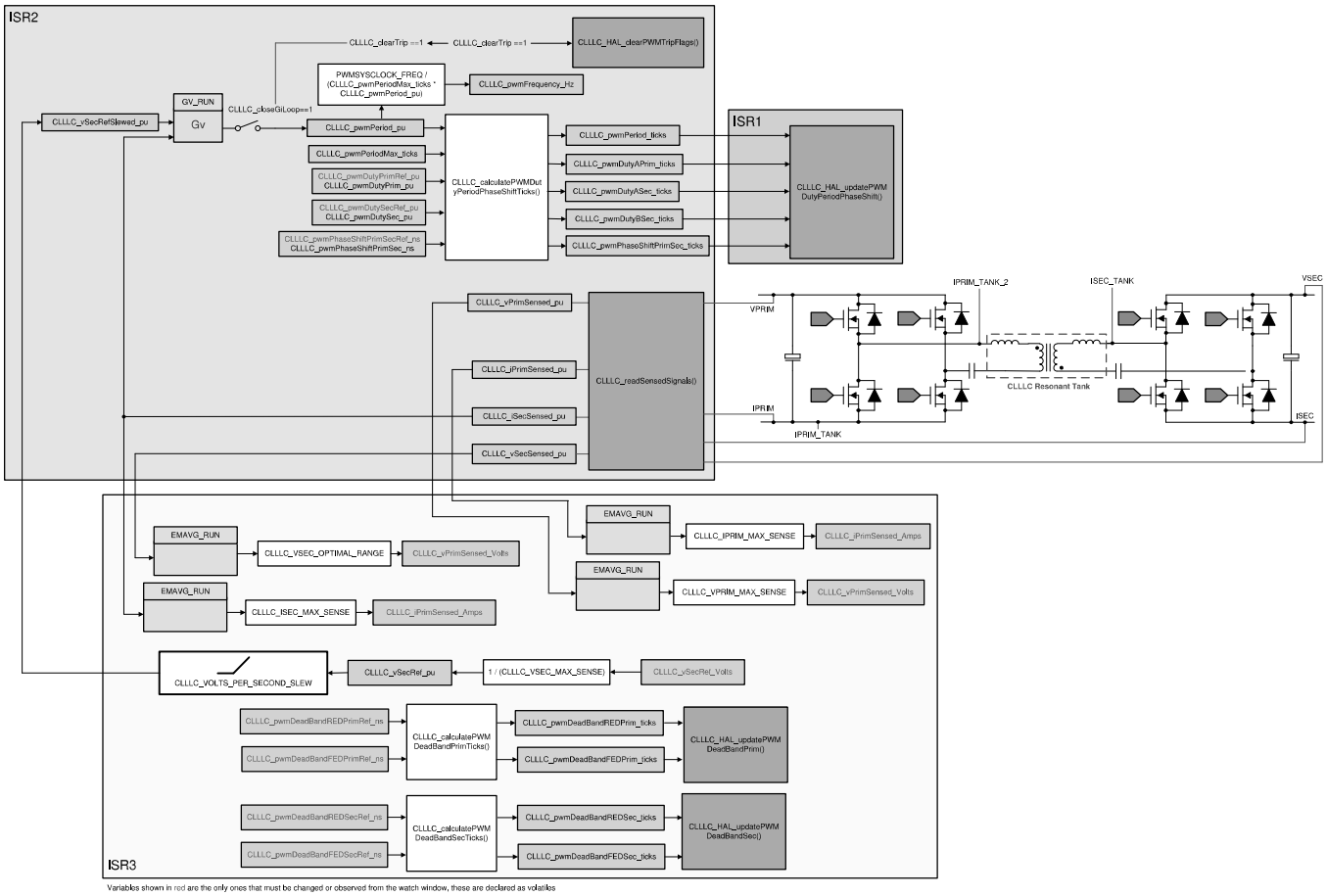


Figure 53. Lab 5 Software Diagram



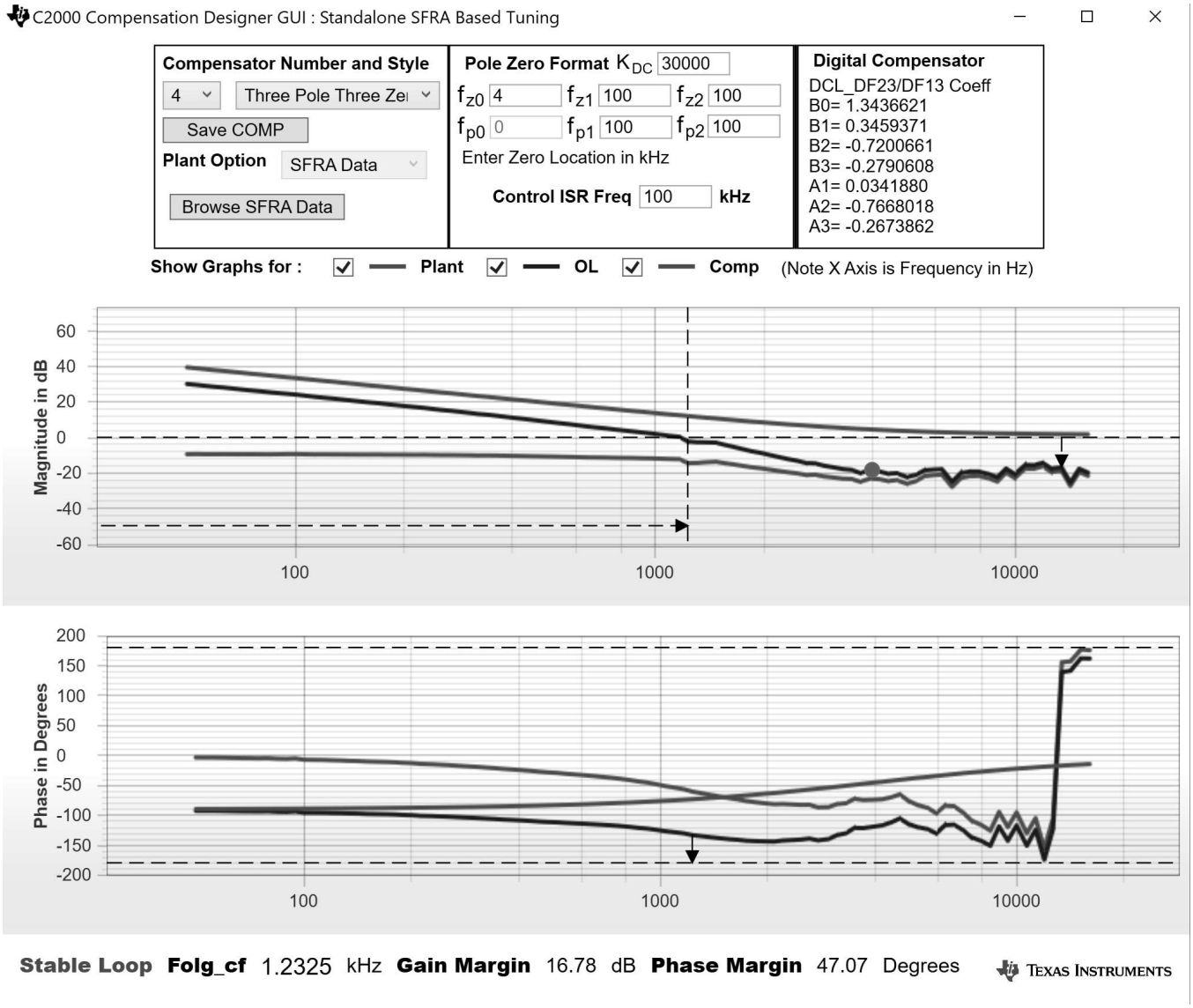
3.2.2.5.1 Setting Software Options for Lab 5

1. If using the powerSUITE page, go to step 2; otherwise, jump to step 4 of Section 3.2.2.5.2.
2. ***powerSUITE Settings***: On the powerSUITE page,
   a. Select under the *"Project Options"* section *"Lab 5 : Closed Loop, Current Battery Emulated"* for the Lab.
   b. Under "Tuning", select SFRA to run on the current loop using the drop-down box. Save the page.
3. Under Control Loop Design, options for the current loop tuning will automatically be selected,
   Tuning → Comp Number 3 → DF22.
4. Save the page by "Ctrl-S" and click on the Compensation Designer button 

3.2.2.5.2 Designing Current Loop Compensator

1. The compensation designer will then launch. Currently, a mathematical model is not available; hence, using tuning done on this board, the following compensation is designed. The plant in the emulated battery mode will have more gain, and hence, the coefficients will need to be reduced to accommodate that. Figure 54 shows the coefficients used on this design for this lab.

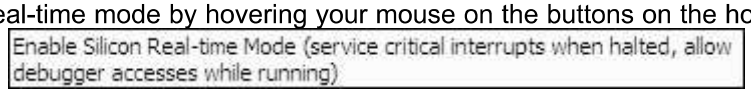
Figure 54. Lab 5, Compensation Designer



2. Once satisfied with the compensator design, click on “Save COMP”. This will save the compensator values into the project. It is best to keep the coefficients conservative and much lower than the one used in Lab 3.
3. Close the compensation designer, return to the powerSUITE page, and save (“Ctrl-S”).
4. The following defines are set in the "*settings.h*" file for this build. These will be set by the powerSUITE page automatically; otherwise, edit the page manually.

```
#define #if CLLLC_LAB == 5
#define CLLLC_CONTROL_RUNNING_ON C28x_CORE
#define CLLLC_POWER_FLOW CLLLC_POWER_FLOW_PRIM_SEC
#define CLLLC_INCR_BUILD CLLLC_CLOSED_LOOP_BUILD
#define CLLLC_CONTROL_MODE CLLLC_CURRENT_MODE
#define CLLLC_TEST_SETUP CLLLC_TEST_SETUP_EMULATED_BATTERY
#define CLLLC_PROTECTION CLLLC_PROTECTION_ENABLED
#define CLLLC_SFRA_TYPE  CLLLC_SFRA_CURRENT
#define CLLLC_SFRA_AMPLITUDE (float32_t)CLLLC_SFRA_INJECTION_AMPLITUDE_LEVEL1
#endif
```

3.2.2.5.3 Building and Loading the Project and Setting up Debug

1. Now, 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. In case of dual-CPU devices, a window may appear for the user to select the CPU on which the debug is to be performed. In this case, select CPU1.
4. The project will then load on the device and the CCS debug view will become active. The code will halt at the start of the main routine.
5. To add the variables in the watch/expressions window, click View → Scripting Console to open the scripting console dialog box. On the upper right corner of this console, click on "open" to browse to the “*setupdebugenv_build4.js*” script file located inside the project folder. This will populate the watch window with the appropriate variables needed to debug the system.
6. Click on the Continuous Refresh button  on the watch window to enable continuous update of values from the controller.
7. Enable real-time mode by hovering your mouse on the buttons on the horizontal toolbar and clicking the  

button.

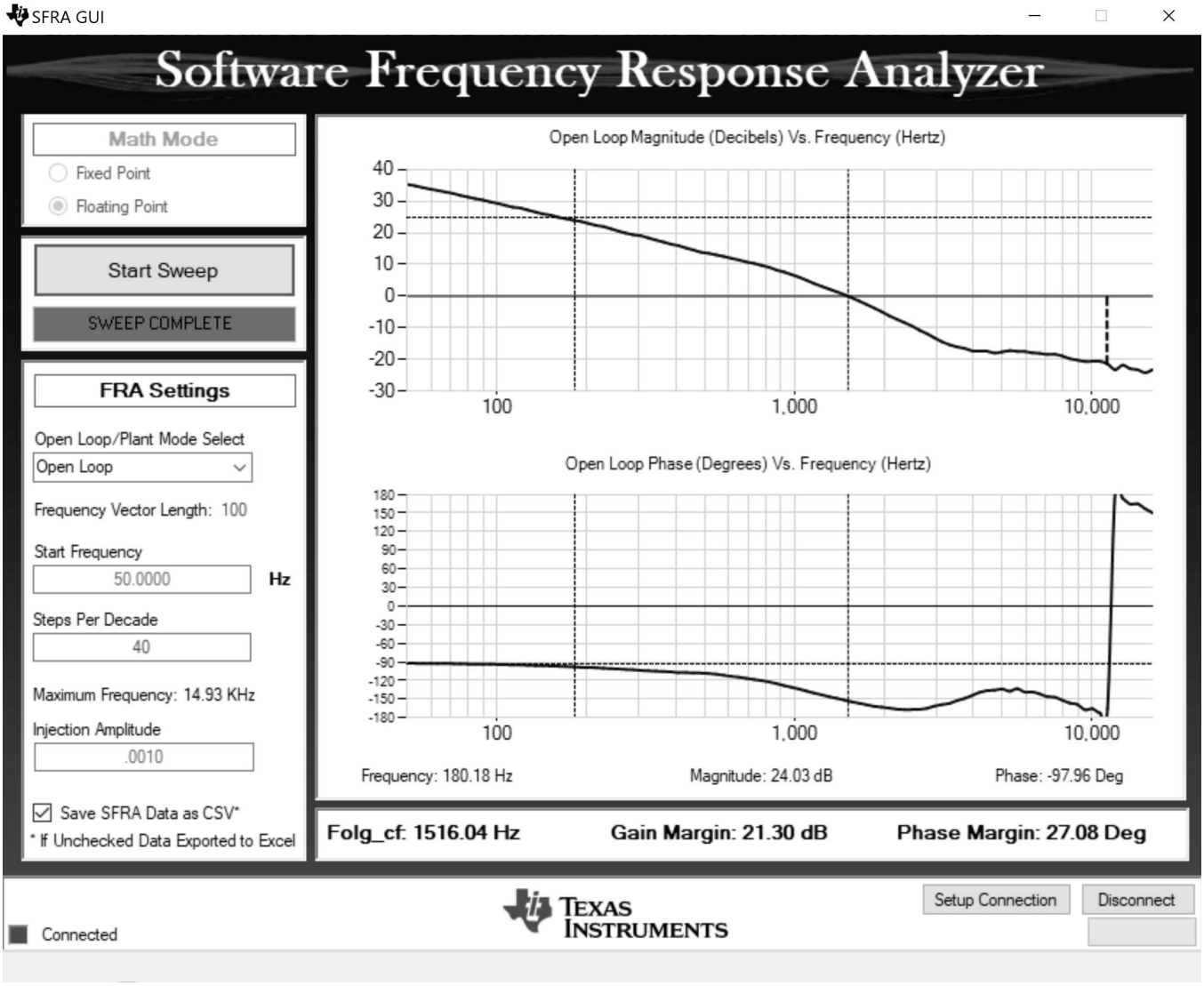
3.2.2.5.4 Running the Code

1. Run the project by clicking  .
2. Now, slowly increase the input PRIM DC voltage from 0 V to 400 V. Make sure CLLLC_vPrimSensed_Volts displays the correct values for VPRIM (that is, close to 400 V). At this point, the PWMs are tripped; therefore, no current will be drawn from the primary side.
3. Next, increase the VSEC to 300 V. Load will draw all the current from the secondary connected power supply, which will be close to 6.5 A.
4. Now, set the CLLLC_iSecRef_Amps variable to 0.1 A.
5. Clear the trip by writing “1” to the CLLLC_clearTrip variable. The software in this lab will automatically set the CLLLC_closeGiLoop variable to “1”.
6. Due to the narrow range of voltage at a fixed primary side voltage, the converter will saturate to the highest frequency, and the current drawn at the ISEC will be higher than 0.1 A. The user can observe this by monitoring the CLLLC_pwmFrequency_Hz variable, which will be close to 700 kHz during the upper saturation limit and 300 kHz during the lower saturation limit.
7. Slowly raise the current to be 2–3 A. Now, the current will be shared by the secondary connected voltage source and the design under test (DUT).

3.2.2.5.5 Measure SFRA for Closed Current Loop in Battery Emulated Mode

1. The SFRA is integrated in the software of this build to verify that the designed compensator provides enough gain and phase margin by measuring on hardware. To run the SFRA, keep the project running, and from the SYSCFG page, click on the SFRA icon. The SFRA GUI will pop up.
2. Select the options for the device on the SFRA GUI; for example, for F280049, select floating point. Click on setup connection. In the pop-up window, uncheck the boot-on-connect option, select an appropriate COM port, and click Ok. Return to the SFRA GUI and click Connect.
3. The SFRA GUI will connect to the device. A SFRA sweep can now be started by clicking “Start Sweep”. The complete SFRA sweep will take a few minutes to finish. Activity can be monitored by seeing the progress bar on the SFRA GUI; and also by checking the flashing of blue LED on the back of the control card, which indicates UART activity. Once complete, a graph with the open loop plot will appear, as shown in Figure 55.

Figure 55. SFRA Open Loop Plot for the Closed Current Loop With Battery Connection Emulated (Vprim 400 V, Vsec 300 V, Power 1.972 kW, Lab 4)



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.

Test the SFRA at different current set points, making sure the period is not clamped, to verify that the system is stable across the operable range.

4. This verifies the Lab 4 current loop design.
5. To bring the system to a safe stop, bring the input VPRIM voltage down to zero. Observe the voltages and currents on the watch window go down to zero.
6. Fully halting the MCU when in real-time mode is a two-step process. First, halt the processor by using the Halt button on the toolbar , or by using Target → Halt. Then, take the MCU out of real-time mode by clicking on . Finally, reset the MCU .
7. Close the CCS debug session by clicking on Terminate Debug Session (Target → Terminate all).

3.2.2.6 Lab 6, Secondary to Primary Power Flow, Open Loop Check PWM driver.

This lab option is primarily provided as a focussed test just for the PWM from a software perspective, so that it can be ran independent of the hardware connections of the reference design. With this lab the user can run the code on a C2000 control card or launchpad just to observe the PWM waveforms.

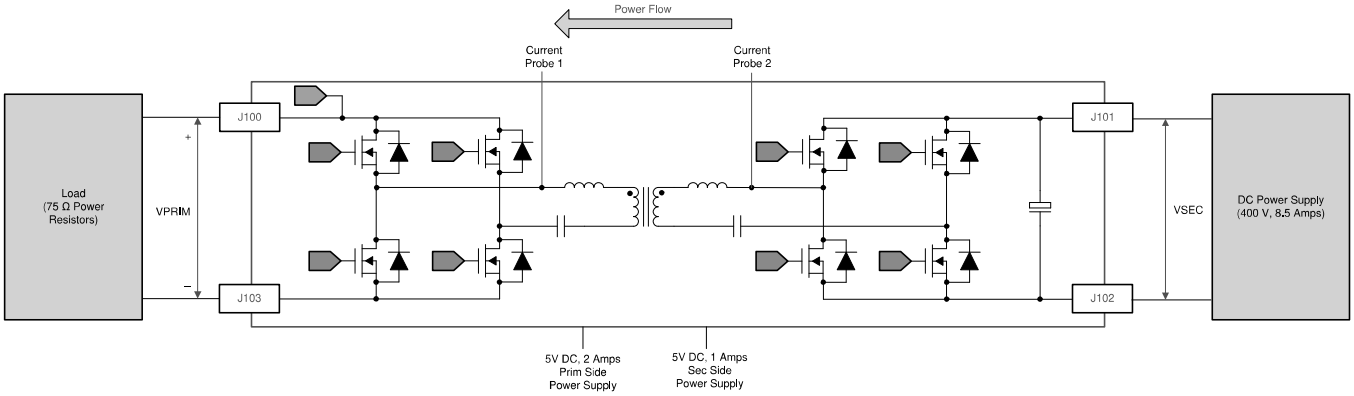
Steps used to load and run are similar to Section 3.2.2.7.

This lab can be easily skipped and the user can go to Lab 7 directlty if no changes to the PWM driver are anticipated. Hence this lab procedure is not documented as it is primarlily for PWM driver development and debug purpose.

3.2.2.7 Lab 7, Secondary to Primary Power Flow, Open Loop Check PWM driver and ADC with protection, resistive load connected on primary

In this build, the board is excited in open-loop fashion with a specified frequency that can be changed through the watch window. The frequency is controlled with the CLLLC_pwmPeriodRef_pu variable. The power flow is from secondary side to the primary side. The hardware setup for Lab 7 is shown is figure

Figure 56. Hardware Setup for Sec to Prim Power Flow



This build verifies the sensing of feedback values from the power stage and also operation of the PWM gate driver in the reverse power direction, and ensures there are no hardware issues. Additionally, calibration of input and output voltage sensing can be performed in this build. Figure 34 shows the software structure for this build.

3.2.2.7.1 Setting Software Options for Lab 7

1. To get started, open the CCS project, as outlined in Section 3.1.2.1. If using the powerSUITE page, go to step 2, otherwise jump to step 3.
2. Open the SYSCFG page and select under the *"Project Options"* section:
   - *"Lab 7: Open Loop PWM with Protection, Sec to Prim Power Flow"* for the Lab
   - Under "Tuning", select SFRA to run on the voltage to measure the voltage loop plant.

   Save the page.
3. The following defines are set in the "*settings.h*" file for this build. These will be set by the powerSUITE page automatically.

```
#if CLLLC_LAB == 7
#define CLLLC_CONTROL_RUNNING_ON C28x_CORE
#define CLLLC_POWER_FLOW CLLLC_POWER_FLOW_SEC_PRIM
#define CLLLC_INCR_BUILD CLLLC_OPEN_LOOP_BUILD
#define CLLLC_CONTROL_MODE CLLLC_VOLTAGE_MODE
#define CLLLC_TEST_SETUP CLLLC_TEST_SETUP_RES_LOAD
#define CLLLC_PROTECTION CLLLC_PROTECTION_ENABLED
#define CLLLC_SFRA_TYPE  CLLLC_SFRA_VOLTAGE
#define CLLLC_SFRA_AMPLITUDE (float32_t)CLLLC_SFRA_INJECTION_AMPLITUDE_LEVEL2
#endif
```

3.2.2.7.2 Building and Loading the Project and Setting up Debug Environment

1. Now, 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 under targetConfigs (see Figure 29).
4. Then, click “Run → Debug” to launch a debugging session. In case of dual-CPU devices, a window may appear for the user to select the CPU on which the debug is to be performed. In this case, select CPU1.
5. The project will then load on the device and the CCS debug view will become active. The code will halt at the start of the main routine.
6. To add the variables in the watch/expressions window, click View → Scripting Console to open the scripting console dialog box. On the upper right corner of this console, click on "open" and then browse to the “*setupdebugenv_lab2and7.js*” script file located inside the project folder. This will populate the watch window with the appropriate variables needed to debug the system.
7. Click on the Continuous Refresh button  on the watch window to enable continuous update of values from the controller. The watch window will appear as shown in Figure 35.

3.2.2.7.3 Running the Code

1. Run the project by clicking  .
2. Now, clear the trip by writing “1” to the CLLLC_clearTrip variable.
3. In the watch view, check if the CLLLC_vPrimSensed_Volts, CLLLC_iPrimSensed_Amps, CLLLC_vSecSensed_Volts, and CLLLC_iSecSensed_Amps variables are updating periodically. (Note: As no power is applied right now, these will be close to zero.)
4. Now, slowly increase the input VSEC DC voltage from 0 V to 300 V. Make sure CLLLC_vSecSensed_Volts displays the correct values. As default, the CLLLC_pwmPeriodRef_pu variable is set to 0.599, which is 500.8 kHz. This is close to the series resonant frequency of the converter; however, due to variation in the components on the actual hardware, it can be lower or higher than the series resonant frequency.
5. The VPRIM variable will show a voltage of close to 400 V per the tank gain designed. Verify that CLLLC_vPrimSensed_Volts shows the correct voltage. This verifies the reverse power flow working.
6. With the load specified in the test conditions, the current from the PRIM and SEC side will be close to 4.8 A for CLLLC_iPrimSensed_Amps, and 6.8 A for CLLLC_iSecSensed_Amps.

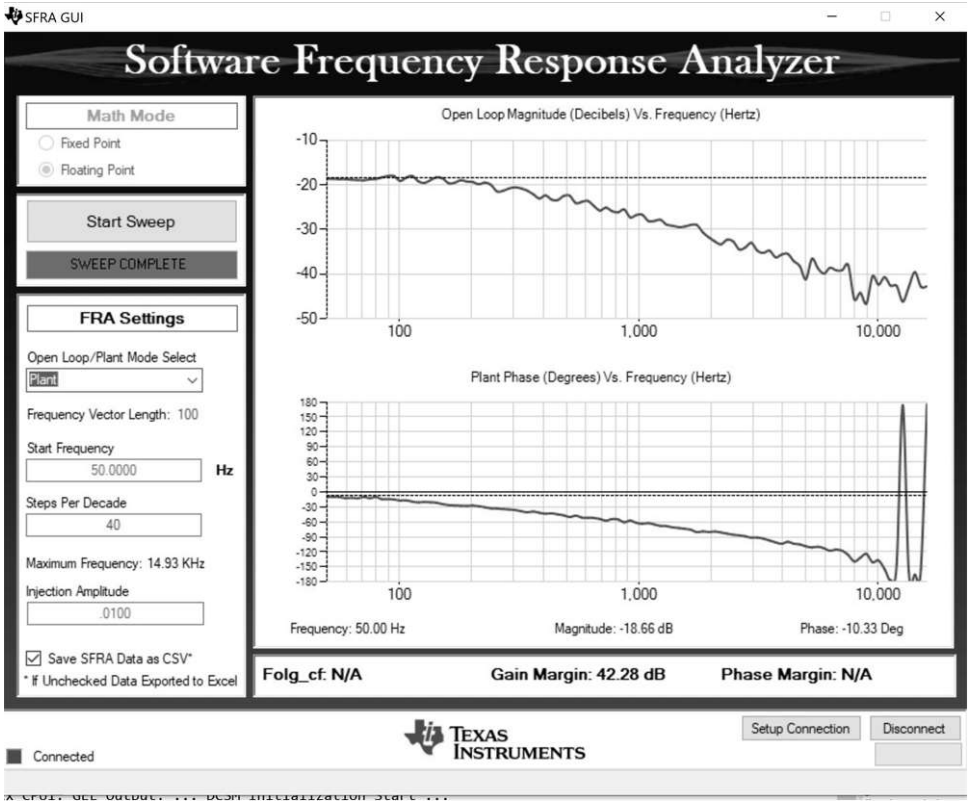
7. Next, to see operation under different frequencies (that is, above resonance, below resonance), change the CLLLC_pwmPeriodRef_pu variable to be 0.47 which will correspond to a frequency of 639 kHz. Then by entering 0.8 as the CLLLC_pwmPeriodRef_pu, which will make generated frequency of 374kHz. In this case, the primary current will become discontinuous and the secondary side duty cycle will modulate to achieve diode emulation.

8. The PWM duty cycle for the primamry side can be probed as well to check working of the active synchronous rectification scheme in the reverse direction.
9. This verifies at a basic level the PWM driver and connection of hardware for the reverse power flow.

3.2.2.7.4 Measure SFRA Plant for Voltage Loop

1. The SFRA is integrated in the software of this build to measure the plant response which can then be used to design a compensator. To run the SFRA, keep the project running, and from the SYSCFG page, click on the SFRA icon. The SFRA GUI will pop up.
2. Select the options for the device on the SFRA GUI; for example, for F280049, select floating point. Click on setup connection. In the pop-up window, uncheck the boot-on-connect option and select an appropriate COM port. Click Ok. Return to the SFRA GUI and click Connect.
3. The SFRA GUI will connect to the device. A SFRA sweep can now be started by clicking “Start Sweep”. The complete SFRA sweep will take a few minutes to finish. Activity can be monitored by seeing the progress bar on the SFRA GUI and also by checking the flashing of blue LED on the back of the control card, which indicates UART activity. Once complete, a graph with the measurement will appear, as shown in Figure 40. (Note that the open-loop measurement is not valid in the lab as the loop is not closed. The user must only refer to the plant measurement.)

Figure 57. SFRA Open Loop Plot for the Closed Voltage Loop, Sec to Prim Power Flow (Vprim 400 V, Vsec 300 V, Power 1.972 kW, Fsw 500 kHz)



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. SFRA can be run at different frequency set points to cover the range of operation of the system. A compensator will be designed using these measured plots in the next lab; therefore, remember this timestamp, or rename the "*SFRA.csv*" file to a convenient name that is easy to identify.

4. This verifies this Lab 4 current loop design.

5. To bring the system to a safe stop, bring the input VPRIM voltage down to zero. Observe the voltages and currents on the watch window go down to zero.

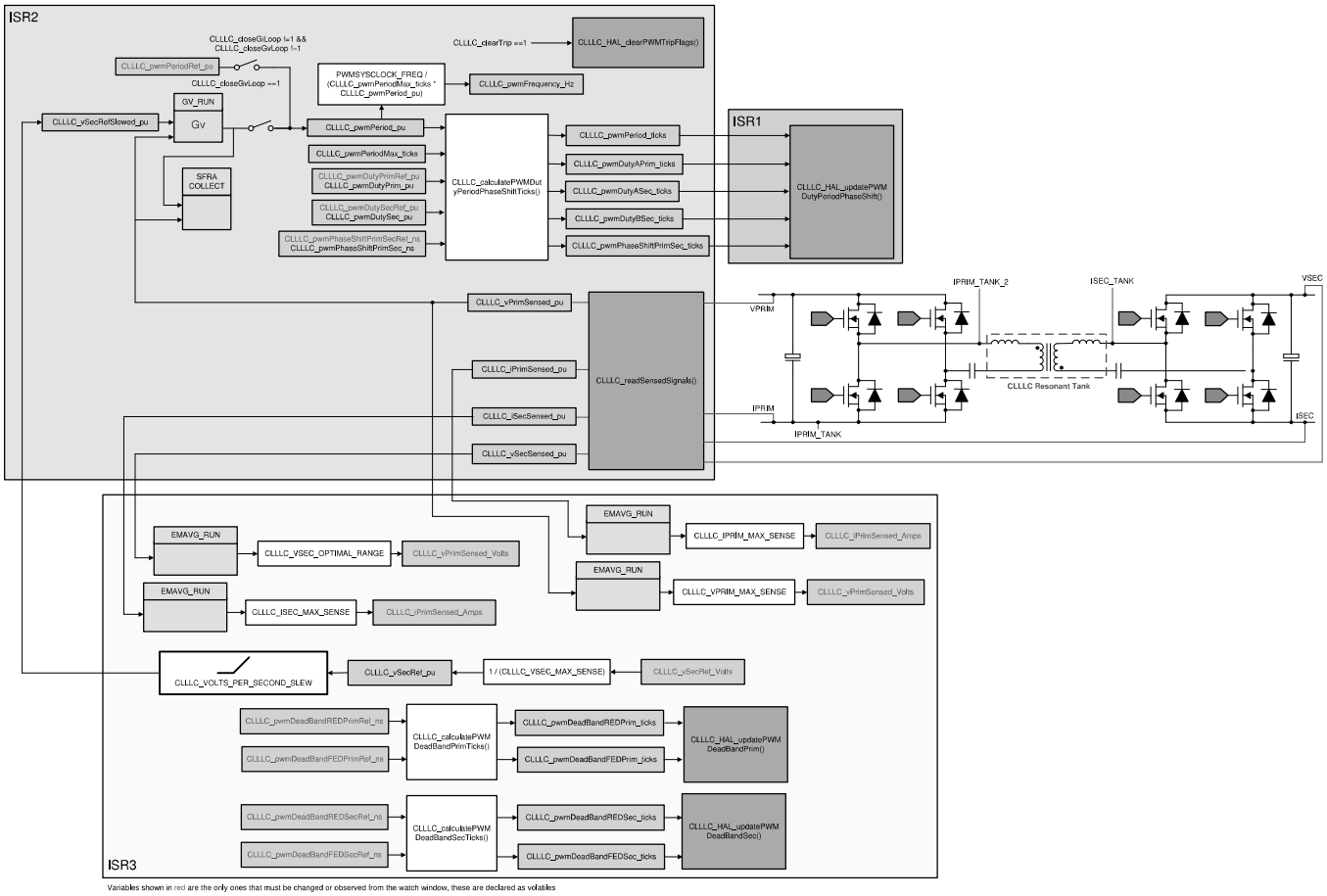
6. Fully halting the MCU when in real-time mode is a two-step process. First, halt the processor by

using the Halt button on the toolbar  , or by using Target → Halt. Then, take the MCU out of real-time mode by clicking on  . Finally, reset the MCU  7. Close the CCS debug session by clicking on Terminate Debug Session  (Target → Terminate all).

3.2.2.8 Lab 8, Secondary to Primary Power Flow, Closed Voltage Loop Check, with Resistive Load Connected on Primary

In this lab, the voltage loop, G_V, is closed with a resistive load at the output on the secondary side. Figure 58 shows the complete software diagram for this build. Hardware is assumed to be set up as shown in Figure 33.

Figure 58. Software Diagram, Closed Voltage Loop, Secondary to Primary Power Flow

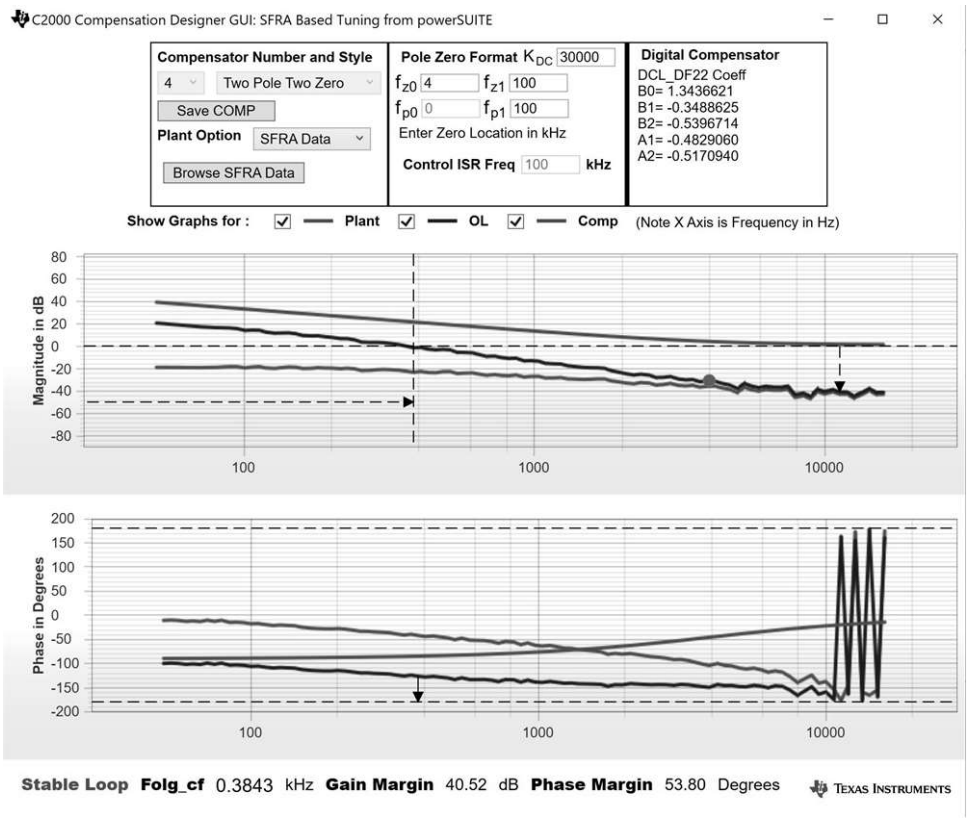


3.2.2.8.1 Setting Software Options

1. To run this lab, make sure the hardware is set up as outlined in the previous section, Figure 56. Do not supply any high-voltage power to the board yet. If using the powerSUITE page, go to step 2; otherwise, jump to step 8.
2. ***powerSUITE Settings***: On the powerSUITE page,
   a. Select under the *"Project Options"* section *"Lab 8 : Closed Loop Voltage with Resistive Load, Sec to Prim Power Flow"* for the Lab.
   b. Under "Tuning", select SFRA to run on the voltage loop using the drop-down box. Save the page.
3. Under Control Loop Design, options for the current loop tuning will automatically be selected, Tuning → Comp Number 1 → DF22.

4. Save the page by “Ctrl-S” and click on the Compensation Designer button  ,
5. The compensation designer will then launch and prompt the user to select a valid SFRA data file. Import the SFRA data from the run in Lab 1 into the compensation designer to design a two-pole, two-zero compensator. It is good to keep more margins during this iteration of the design to ensure that when the loop is closed, the system is stable. Plant data from different runs of the SFRA can be checked to get a stable system under all conditions.

Figure 59. Compensator Design With SFRA Based Plant Measurement for the Voltage Loop, in the reverse power flow direction with resistive load at the prim side output



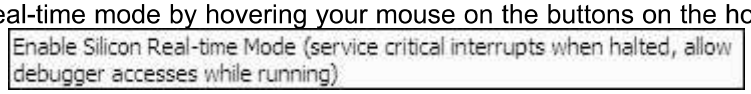
NOTE: The tuning is carried out in DF22 fashion; however, we run the DF13 in the software. This is done because soft-starting the DF13 is easier, whereas not possible with the DF22 structure. The coefficients in both cases remain the same. At the writing of this document, the DF12 structure is not available in DCL.

6. Once satisfied with the compensator design, click on “Save COMP”. This will save the compensator values into the project.
7. Close the compensation designer, return to the powerSUITE page, and save (“Ctrl-S”). This will write the new compensator values to the "*settings.h*" file.

8. The following defines are set in the "*settings.h*" file for this build. These will be set by the powerSUITE page automatically; otherwise, edit the page manually.

```
#if CLLLC_LAB == 8
#define CLLLC_CONTROL_RUNNING_ON C28x_CORE
#define CLLLC_POWER_FLOW CLLLC_POWER_FLOW_SEC_PRIM
#define CLLLC_INCR_BUILD CLLLC_CLOSED_LOOP_BUILD
#define CLLLC_CONTROL_MODE CLLLC_VOLTAGE_MODE
#define CLLLC_TEST_SETUP CLLLC_TEST_SETUP_RES_LOAD
#define CLLLC_PROTECTION CLLLC_PROTECTION_ENABLED
#define CLLLC_SFRA_TYPE  CLLLC_SFRA_VOLTAGE
#define CLLLC_SFRA_AMPLITUDE (float32_t)CLLLC_SFRA_INJECTION_AMPLITUDE_LEVEL1
#endif
```

3.2.2.8.2 Building and Loading the Project and Setting up Debug Environment

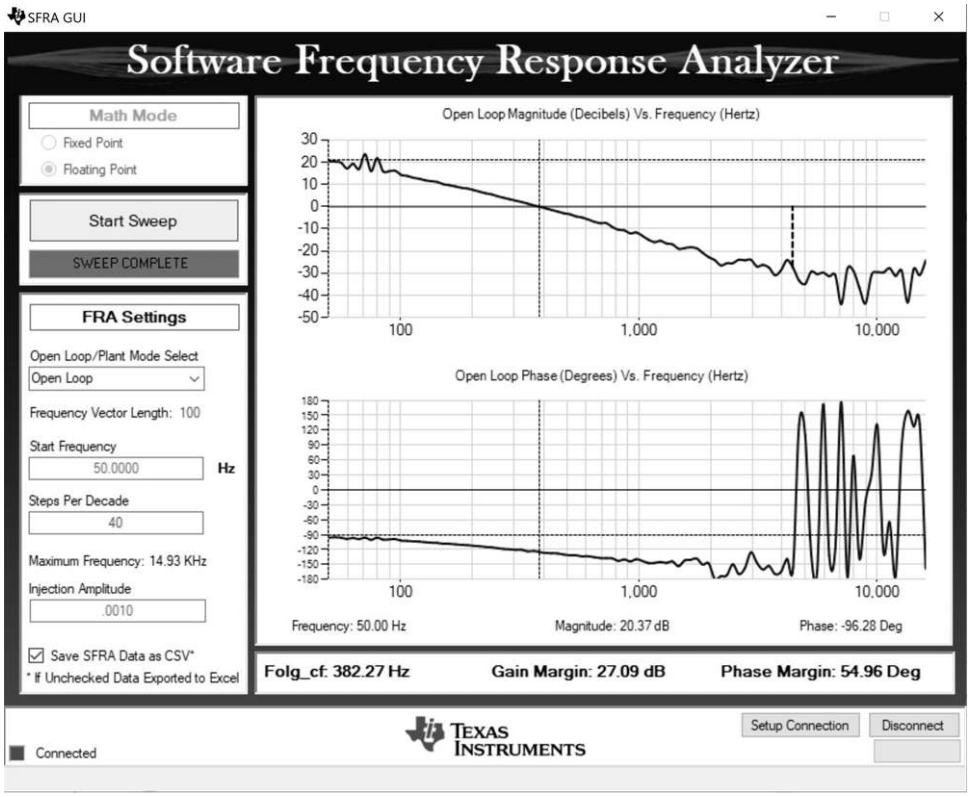
1. Right-click the project name and click "Rebuild Project".
2. The project will build successfully.
3. Click "Run → Debug" to launch a debugging session. In case of dual-CPU devices, a window may appear for the user to select the CPU on which the debug is to be performed. In this case, select CPU1.
4. The project will then load on the device and the CCS debug view will become active. The code will halt at the start of the main routine.
5. To add the variables in the watch/expressions window, click View → Scripting Console to open the scripting console dialog box. On the upper right corner of this console, click on "open" to browse to the "*setupdebugenv_lab3.js*" script file located inside the project folder. This will populate the watch window with the appropriate variables needed to debug the system.
6. Click on the Continuous Refresh button  on the watch window to enable continuous update of values from the controller.
7. Enable real-time mode by hovering your mouse on the buttons on the horizontal toolbar and clicking the  

button.
8. Run the project by clicking 
9. Now, clear the trip by writing "1" to the CLLLC_clearTrip variable. The converter will operate in open loop as the CLLLC_closeGvLoop variable is not yet set to "0". As there is no soft start implemented in the firmware, first soft-start the voltages on the primary and secondary sides manually.
10. In the watch view, check if the CLLLC_vPrimSensed_Volts, CLLLC_iPrimSensed_Amps, CLLLC_vSecSensed_Volts, and CLLLC_iSecSensed_Amps variables are updating periodically. (Note: As no power is applied right now, these will be close to zero.)
11. Now, slowly increase the input PRIM DC voltage from 0 V to 400 V to soft-start the converter. Make sure CLLLC_vPrimSensed_Volts displays the correct values for VPRIM (that is, close to 400 V).
12. By default, the CLLLC_pwmPeriodRef_pu variable is set to 0.599, which is 500.8 kHz. This is close to the series resonant frequency of the converter; however, due to variation in the components on the actual hardware, it can be lower or higher than the series resonant frequency.
13. For the 400-V primary input, with turns ratio being 1.33, the CLLLC_vSecSensed_Volts variable will be close to 300 V. Set the CLLLC_vSecRef_Volts variable to be 300 V.
14. Now, set the CLLLC_closeGvLoop variable to "1". This will close the voltage loop and the controller will now try to regulate the voltage.
15. Test the closed-loop operation by varying CLLC_vSecRef_Volts from 295 V to 320 V. The user will observe that the CLLLC_vSecSensed_Volts will track this command reference. The converter will operate below series resonant, at resonance, and above resonance. Now, change the voltage back to 300 V to run the SFRA.

3.2.2.8.2.1 Measure SFRA for Closed Voltage Loop

1. The SFRA is integrated in the software of this build to verify that the designed compensator provides enough gain and phase margin by measuring on hardware. To run the SFRA, keep the project running, and from the SYSCFG page, click on the SFRA icon. The SFRA GUI will pop up.
2. Select the options for the device on the SFRA GUI; for example, for F280049, select floating point. Click on setup connection. In the pop-up window, uncheck the boot-on-connect option and select an appropriate COM port. Click Ok. Return to the SFRA GUI and click Connect.
3. The SFRA GUI will connect to the device. A SFRA sweep can now be started by clicking “Start Sweep”. The complete SFRA sweep will take a few minutes to finish. Activity can be monitored by seeing the progress bar on the SFRA GUI and also by checking the flashing of blue LED on the back of the control card, which indicates UART activity. Once complete, a graph with the open loop plot will appear, as shown in Figure 48.

Figure 60. SFRA Open Loop Plot for the Closed Voltage Loop, Secondary to Primary Power Flow (Vprim 400 V, Vsec 300 V, Power 1.972 kW)



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 data matches closely to the designed compensator, but it is reasonable to expect deviations because the measurement in open loop is susceptible to error, due to small signal injection, which can drift the DC point of the converter.

Test the SFRA at different voltages to verify that the system is stable across the operable range.

4. This verifies the voltage loop design. Note at this point the bandwidth in the reverse power flow is not fully tuned and more adjustments to the compensator can enable better bandwidth out of the system.
5. To bring the system to a safe stop, bring the input VPRIM voltage down to zero. Observe the voltages and currents on the watch window go down to zero.
6. Fully halting the MCU when in real-time mode is a two-step process. First, halt the processor by using the Halt button on the toolbar  , or by using Target → Halt. Then, take the MCU out of real-time mode by clicking on  . Finally, reset the MCU  .

7. Close the CCS debug session by clicking on Terminate Debug Session  (Target → Terminate all).

3.2.2.9 Running on CLA

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

The following #define is changed to make this selection in the "*settings.h*" file. When using powerSUITE, this selection is done through the SYSCFG page for the solution.

```
#define C28x_CORE 1
#define CLA_CORE 2
#define CLLLC_ISR1_RUNNING_ON 1

...
#if CLLLC_ISR1_RUNNING_ON == CLA_CORE
#define CLLLC_ISR2_RUNNING_ON CLA_CORE
#else
#define CLLLC_ISR2_RUNNING_ON C28x_CORE
#endif
```

NOTE: The SFRA library does not support CLA; therefore, the SFRA cannot be run when using the CLA. DLOG is also not used when using CLA; therefore, the data logging graphs will not work when using CLA.

Once the option is changed, the SYSCFG file must be saved and the project recompiled. Once recompiled, follow the steps as outlined in the specific incremental build level documentation.

3.2.3 Test Results

3.2.3.1 Efficiency Across load and Voltage

The efficiency data at 500 kHz nominal switching is measured and tabulated in Table 4. The efficiency is also graphed in Figure 61. With a fixed frequency and different loads applied, the converter will transverse through the Discontinuous Conduction Mode (DCM), Boundary Conduction Mode, and Continuous Conduction Mode (CCM) regions; and hence, the dip on the efficiency plot.

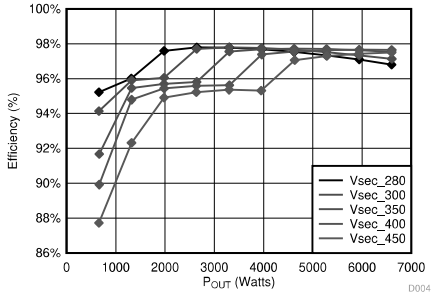
Table 4. Efficiency Under Varying Load and Voltage Conditions, With Frequency Close to Resonant Frequency, Kept Fixed at 500 kHz

VPRIM (V)	VSEC (V)	Pout (W)	EFFICIENCY (%)
362.68	280.04	657.60	95.22%
367.26	280.02	1312.60	96.01%
375.41	279.98	1981.30	97.59%
376.79	280.00	2645.00	97.79%
378.08	280.06	3304.40	97.78%
379.34	280.08	3962.10	97.68%
380.52	280.01	4619.10	97.53%
382.00	280.12	5279.00	97.33%
383.38	280.06	5941.00	97.10%
385.04	280.05	6600.00	96.80%
384.33	300.01	659.70	94.13%
393.06	300.01	1317.90	95.90%
395.24	300.07	1980.70	96.05%
403.11	300.08	2639.70	97.71%

Table 4. Efficiency Under Varying Load and Voltage Conditions, With Frequency Close to Resonant Frequency, Kept Fixed at 500 kHz (continued)

VPRIM (V)	VSEC (V)	Pout (W)	EFFICIENCY (%)
404.18	300.05	3299.60	97.79%
405.27	300.01	3959.00	97.76%
406.51	300.07	4619.30	97.65%
407.69	300.04	5280.10	97.52%
408.96	300.02	5939.80	97.33%
410.45	300.09	6604.00	97.13%
443.77	350.30	662.40	91.68%
457.47	350.02	1320.30	95.46%
459.49	350.06	1979.90	95.70%
461.72	350.11	2644.10	95.82%
469.83	350.05	3302.50	97.55%
470.80	350.05	3959.40	97.68%
471.71	350.02	4620.20	97.72%
472.72	350.03	5280.90	97.69%
473.74	350.04	5942.20	97.63%
474.81	350.05	6600.00	97.53%
504.11	400.00	662.30	89.92%
518.77	400.03	1319.20	94.79%
523.79	400.01	1981.50	95.44%
525.53	400.01	2634.30	95.59%
527.77	400.07	3305.20	95.62%
536.68	400.08	3960.80	97.38%
537.54	400.10	4620.20	97.55%
538.25	400.01	5282.80	97.63%
539.16	400.06	5943.20	97.65%
540.07	400.08	6601.60	97.64%
561.72	450.06	658.80	87.72%
574.38	450.06	1318.30	92.32%
588.02	450.08	1980.40	94.91%
590.14	450.10	2633.50	95.23%
591.88	450.12	3299.30	95.36%
594.03	450.05	3953.00	95.32%
603.37	450.08	4632.20	97.05%
604.21	450.04	5282.30	97.30%
605.01	450.13	5938.10	97.44%
605.67	450.07	6601.10	97.50%

Figure 61. Efficiency at Fsw = 500 kHz, Under Varying Load and Voltage Conditions

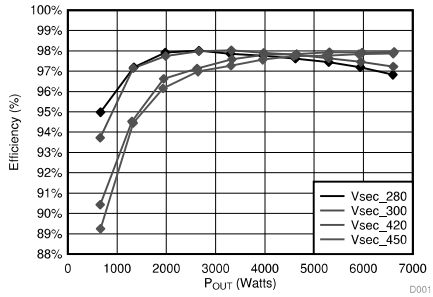


Ensuring that the converter operates at BCM or CCM by adjusting the switching frequency according to the load condition can create an optimal efficiency plane with which the efficiency is tabulated in Table 5 and is plotted on Figure 62.

Table 5. Optimal Efficiency in Open Loop Across Load and Voltage With Varying Frequency for CCM Mode Operation

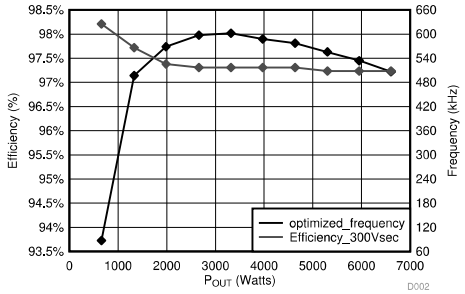
VPRIM (V)	VSEC (V)	Pout (W)	EFFICIENCY (%)	FREQUENCY (kHz)
376.66	280.09	662.6	94.98%	600
377.55	280.01	1337.1	97.19%	566
376.23	280.03	1978.9	97.91%	517
377.81	280.06	2660.9	98.01%	517
379.78	280.02	3301.1	97.85%	517
381.05	280.45	3967	97.76%	508
382.01	280.14	4620.5	97.62%	517
383.56	280.86	5296.7	97.44%	508
383.94	280.05	5937.9	97.19%	508
385.94	280.07	6597.3	96.82%	508
403.69	299.97	656.8	93.73%	625
404.21	300.5	1328.3	97.14%	566
404.11	300.17	1985.8	97.74%	526
404.01	300.14	2658.7	97.98%	517
405.07	300.3	3319.3	98.02%	517
406.78	300.03	3973.6	97.90%	517
408.02	300.29	4636.8	97.81%	517
408.96	300.38	5298.7	97.63%	508
409.65	300.05	5949.4	97.45%	508
411.13	300.05	6608	97.22%	508
544.35	420.03	658.8	90.43%	566
566.86	420.12	1302.6	94.53%	638
564.42	420.08	1944.2	96.62%	566
565.84	420.02	2624.4	97.12%	566
564.77	420.13	3329.9	97.59%	536
565.2	420.04	3986.2	97.81%	536
566.49	420.01	4643.8	97.83%	526
567.02	420.12	5306.8	97.93%	526
567.85	420.27	5971.3	97.91%	517
567.34	420.1	6626.1	97.94%	517
583.07	450.15	664.8	89.25%	577
605.77	450	1327	94.45%	638
604.4	450.02	1933.8	96.15%	566
603.53	450.08	2641.7	96.98%	546
604.77	450.03	3310.6	97.27%	526
605	450.04	3958.2	97.56%	526
605.55	450.05	4586.5	97.74%	526
607.38	450.08	5238.5	97.75%	526
607.29	450.04	5910.6	97.84%	517
608.08	450.06	6607.3	97.86%	517

Figure 62. Efficiency With Optimized Switching Frequency at Varying Load and Voltage



Graphing the optimal switching frequency and efficiency for the secondary voltage equal to 300 V shows the trend in efficiency (see Figure 63).

Figure 63. Efficiency and Optimized Frequency vs Power at 300 VSEC

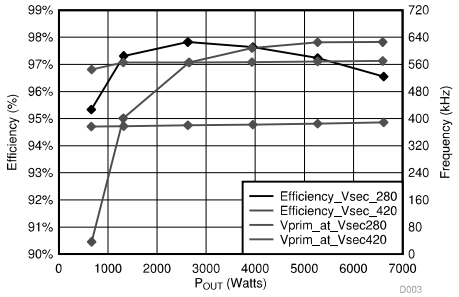


Running the system with the voltage loop closed, the gathered data is tabulated in Table 6. The data from this table is graphed in Figure 64.

Table 6. Closed Loop Efficiency With Optimized Vprim

VPRIM (V)	VSEC (V)	Pout (W)	EFFICIENCY (%)
388.67	281.12	6615.3	96.55%
385.03	281.11	5271.2	97.23%
382.42	281.12	3950.5	97.64%
380.18	281.09	2630.4	97.82%
377.69	281.1	1320.3	97.31%
376.69	281.12	659.5	95.33%
569.9	420.88	6593.6	97.82%
568.83	420.89	5271.2	97.81%
566.75	420.88	3933.3	97.60%
565.41	420.87	2649.7	97.07%
565.67	420.89	1311.6	95.01%
544.52	420.87	667.8	90.46%

Figure 64. Efficiency Under Closed Loop With Varying Load and Optimized VPRIM



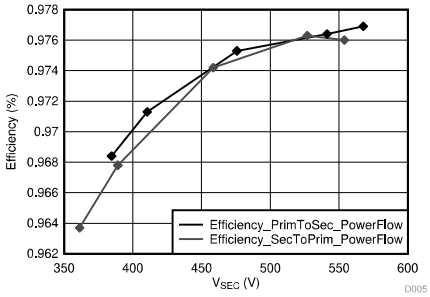
For the reverse power flow the efficiency is similar to the forward direction, table x lists the efficiency measured in the reverse power flow direction.

Table 7. Efficiency at 6.6kW Under Varying Vprim Voltage Conditions, With Frequency Close to Resonant Frequency, Kept Fixed at 508 kHz

VPRIM(V)	VSEC(V)	POUT(W)	EFFICIENCY(%)
553.99	420.01	6587.20	97.60
526.95	400.08	6600.00	97.63
458.44	350.15	6599.10	97.42
389.16	300.03	6585.10	96.78
361.30	279.99	6590.30	96.37

compares the efficiency in the primary to secondary power flow direction versus the secondary to primary power flow direction.

Figure 65. Efficiency comparaison at 6.6kW across different Vprim voltages in secondary to primary (discharging) power flow and primary to secondary (charging) power flow



3.2.3.2 Thermal

Forced air, 3 × 12-V fans (Delta FFB0412EN-00Y2E, operated at 12 V), is used on the bottom heat sink, and with it, tests are carried out up to 6.6 kW across the rated load and voltage for the design. Figure 66 shows the thermal under 300 V Vsec at 6.6 kW; as the current is high, the SiC power devices run up to 55°C. Whereas when running at Vsec 450 V at 6.6 kW, the transformer is hotter and the power devices cooler as the current is much lower due to the increased voltages, as shown in Figure 67.

Figure 66. Thermal Image of the Board Running at 6.6 kW, Vsec 300 V, Vprim 400 V, Fsw 500 kHz

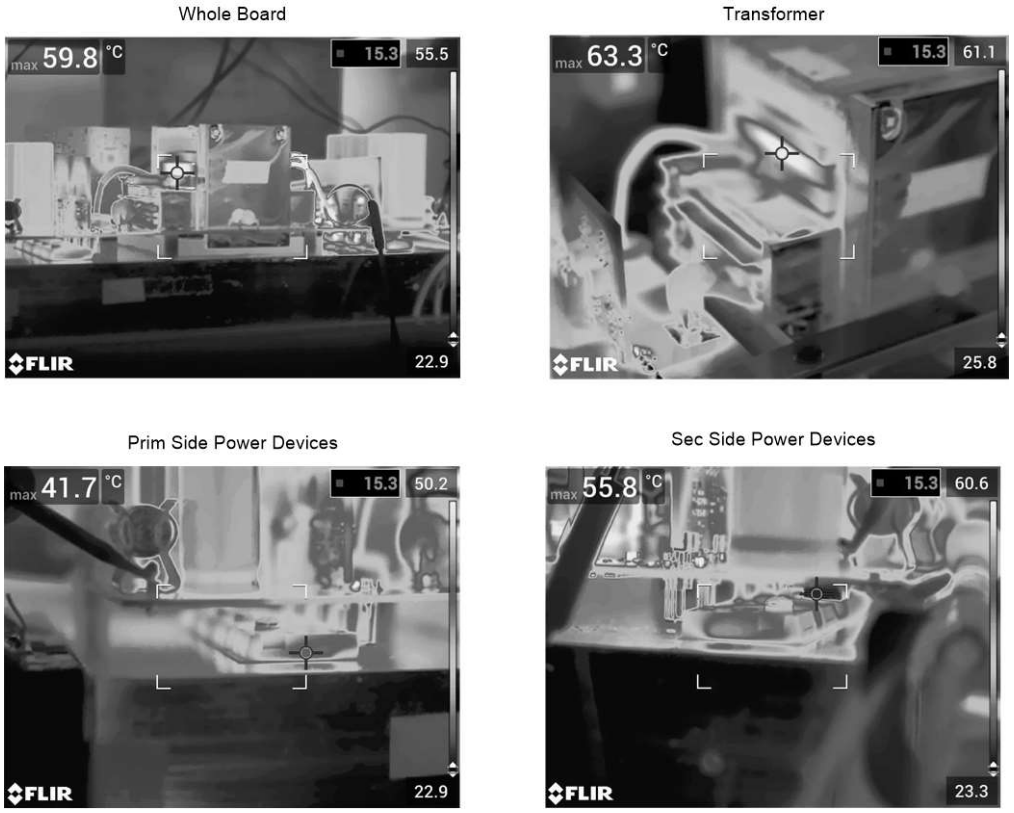
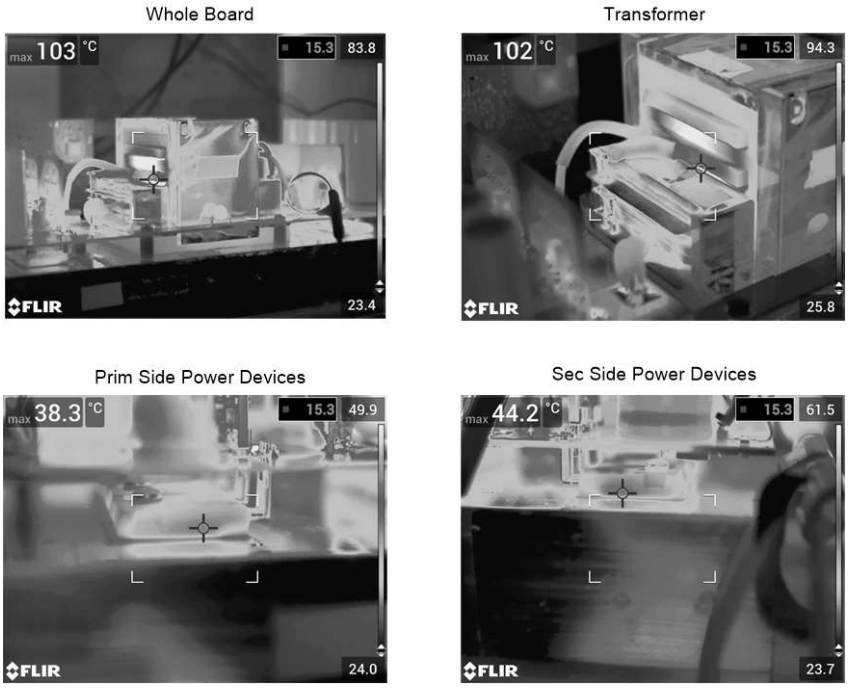


Figure 67. Thermal Image of the Board Running at 6.6 kW, Vsec 450 V, Vprim 600 V, Fsw 500 kHz



3.2.3.3 Soft Switching Waveforms

One key reason to use a resonant converter is that soft-switching or even ZVS can be achieved to reduce or eliminate turnon switching losses. Figure 68 and Figure 69 show the key soft-switching waveforms on the primary side and secondary side during different test conditions when sourcing from Vprim, respectively.

Figure 68. Soft-switching Waveforms
(C1: Q101_VGS; C2: Ipri Measured With Rogowski Coil with 100 mV/A Scale;
C3: Q105_VGS; C4: Q105_VDS):
(A) 375Vin, 280Vout/1315W; (B) 388Vin, 280Vout/6593W;
(C) 567Vin, 420Vout/1324W, and (D) 569Vin, 420Vout/6614W.

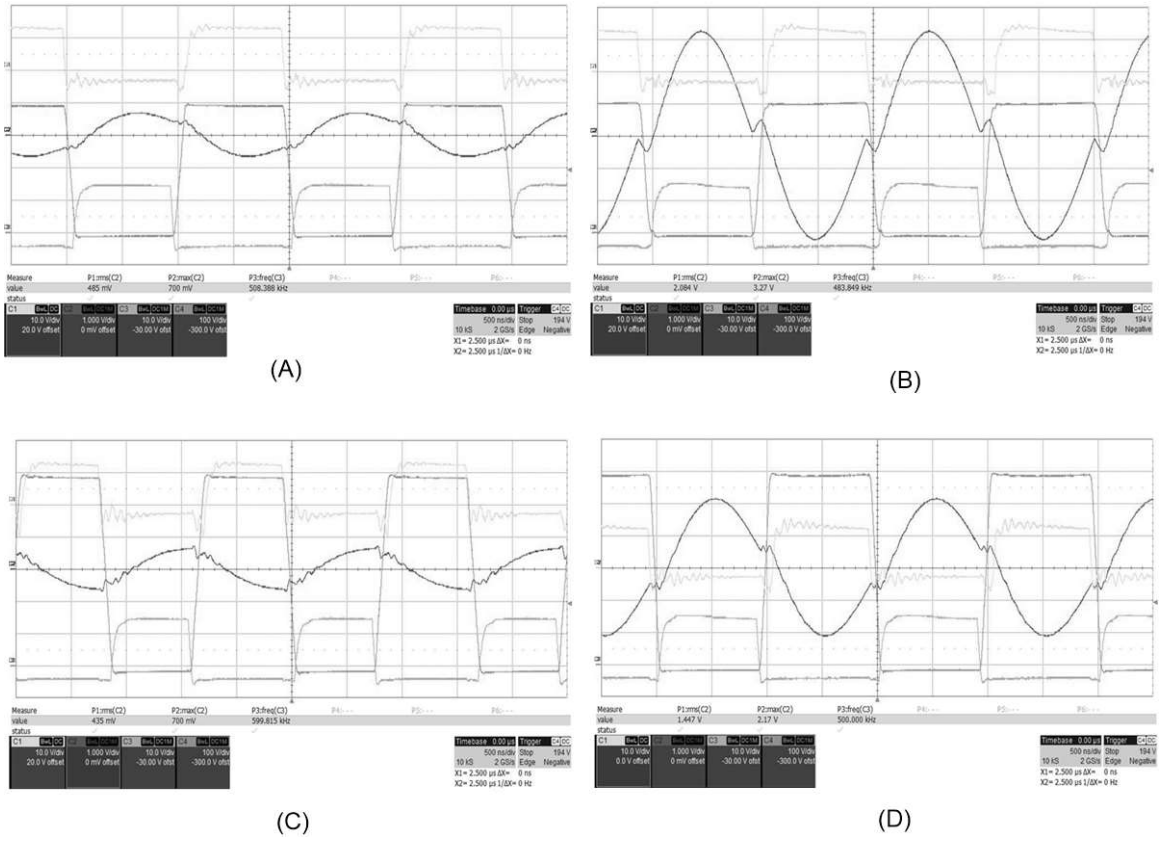
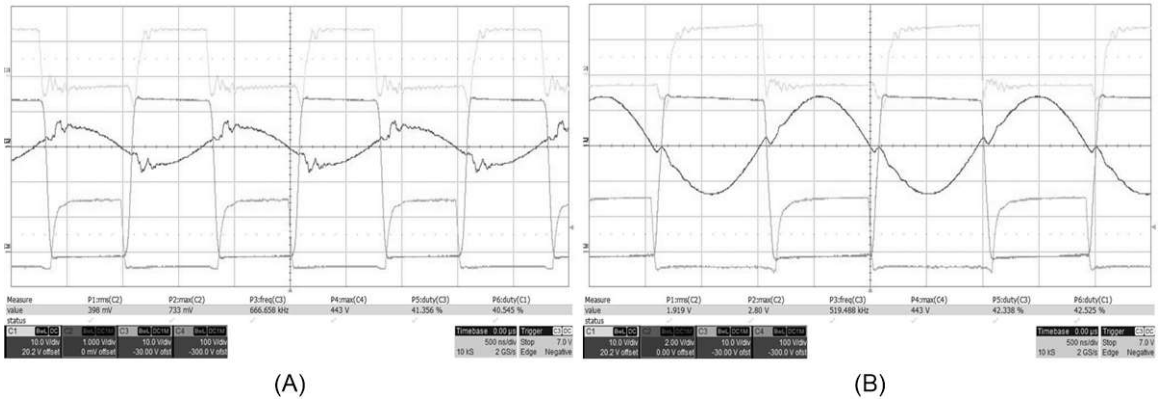


Figure 69. Soft-switching Waveforms
(C1: Q107_VGS; C2: Isec Measured With Rogowski Coil With 100mV/A Scale;
C3: Q103_VGS; C4: Q103_VDS.):
(A) 567Vin, 420 Vout/1333W and (B) 568Vin, 420Vout/6584W.



3.2.3.4 Active Synchronous Rectification

See Section 3.2.2.2.6, Verify Active Synchronous Rectification.

3.2.3.5 Closed Loop Bandwidth

See Section 3.2.2.5.5, Measure SFRA for Closed Current Loop in Battery Emulated Mode.

4 Design Files

4.1 Schematics

To download the schematics, see the design files at TIDM-02002.

4.2 Bill of Materials

To download the bill of materials (BOM), see the design files at TIDM-02002.

4.3 Altium Project

To download the Altium Designer® project files, see the design files at TIDM-02002.

4.4 Gerber Files

To download the Gerber files, see the design files at TIDM-02002.

5 Software Files

To download the software for this reference design, go to the DigitalPower Software Development Kit (SDK) for C2000 MCUs site.

6 Related Documentation

1. Texas Instruments, *TMS320F28004x microcontrollers data manual*
2. Texas Instruments, *C2000™ software frequency response analyzer (SFRA) library and compensation designer in SDK framework*
3. Zaka Ullah Zahid, Zakariya M. Dalala, Rui Chen, Baifeng Chen, and Jih-Sheng Lai, "Design of Bidirectional DC–DC Resonant Converter for Vehicle-to-Grid (V2G) Applications, " *IEEE Transactions on Transportation Electrification*, Vol. 1, No. 3, October 2015, pp. 232–244.
4. Zakariya M. Dalala, Zaka Ullah Zahid, Osama S. Saadeh, Jih-Sheng Lai, "Modeling and Controller Design of a Bidirectional Resonant Converter Battery Charger," *IEEE Access*, Vol. 6, April 2018, pp. 23338–23350.
5. Biao Zhao, Qiang Song, Wenhua Liu, and Yandong Sun, "Overview of Dual-Active-Bridge Isolated Bidirectional DC–DC Converter for High-Frequency-Link Power-Conversion System," *IEEE Transactions on Power Electronics*, Vol. 29, No. 8, August 2014, pp. 4091–4106.

6.1 Trademarks

7 Terminology

ACRONYM	DEFINITION
BCM	Battery Charging Mode
BW	Bandwidth
CAN	Controller Area Network
CCM	Continuous Conduction Mode
CCS	Code Composer Studio
CLA	Control Law Accelerator
CLB	Configurable Logic Block
CLLLC	Capacitor Inductor Inductor Inductor Capacitor
CMPSS	Comparator Subsystem
CMTI	Common-Mode Transient Immunity
DAB	Dual Active Bridge
DAC	Digital-to-Analog Converter
DCM	Discontinuous Conduction Mode
DLOG	Data Logger
DT	Dead time
DUT	Design under test
eCAP	Enhanced Capture
ePWM	Enhanced Pulse Width Modulator
ERAD	Embedded Real-Time Analysis and Diagnostic
EV	Electric Vehicle
FET	Field Effect Transistor
FHA	First Harmonic Analysis
FSI	Fast Serial Interface
HEV	Hybrid Electric Vehicle
HRPWM	High-Resolution Pulse Width Modulator
HSEC	High-Speed Edge Card
I2C	Inter-integrated Circuit
IDE	Integrated Development Environment
IGBT	Insulated-Gate Bipolar Transistor
ISR	Interrupt Service Routine
KCL	Kirchhoff's Current Law
KVL	Kirchhoff's Voltage Law
LIN	Local Interconnect Network
MOSFET	Metal Oxide Semiconductor Field Effect Transistor
OBC	On-Board Charger
PFC	Power Factor Correction
PGA	Programmable Gain Amplifier
PMBus	Power-Management Bus
SCI	Serial Communication Interface
SDFM	Sigma-Delta Filter Module
SFRA	Software Frequency Response Analyzer
SiC	Silicon Carbide
SPI	Serial Peripheral Interface
SRC	Series Resonant Converter

ZCS	Zero Current Switching
ZVS	Zero Voltage Switching

8 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. Manish received his PhD in Electrical Engineering from the University of Texas at Dallas in 2018; Masters of Science in Electrical and Computer Engineering from the Georgia Institute of Technology, Atlanta, in 2009; and his Bachelor of Engineering from the Netaji Subhash Institute of Technology, University of Delhi, India, in 2007. His research interests are control theory, power electronics, and signal processing.

Sheng-Yang Yu is an Application Engineer at Texas Instruments, where he is responsible for developing power supply reference design solutions for strategic customers. Sheng-Yang earned his PhD in Electrical Engineering from the University of Texas at Austin in 2012.

Revision History

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

Changes from B Revision (July 2019) to C Revision **Page**

- Changed *8.2* to *9.3* 26
- Deleted Open CCS. Go to View → CCS App Center. Under Code Composer Studio Add-ons, make sure GUI Composer Runtime v1.0 is installed. If it is not installed, install GUI Composer Runtime v1.0. 26
- Changed *Solution Adapter* to *Through resource explorer* 27
- Added under DC-DC section 27
- Changed *Select the device this solution must run on the next page* to *The development kit page will be displayed* 27
- Deleted Through the development kits page: In the Resource Explorer, under C2000WARE DIGITAL POWER SDK, select Development Kits > CLLLC TIDM-02002. This page can be used to browse all the information on the design, including this design guide, test reports, hardware design files, and so forth. 27
- Added configuration page 27
- Changed PowerSUITE Page for the Design image 28
- Changed *main.cfg* to *main.syscfg* throughout document 29
- Added C based 29
- Added "<*solution*>*_user_settings.h*" is included by the "<*solution*>*_settings.h*"and can be used to keep any settings that are outside the scope of powerSUITE tools such as #defines for ADC mapping, GPIOs etc. 29
- Changed *Kit.xm* to *Kit.json* 29
- Changed Product Explorer View of the CLLLC Project image 30
- Changed *powerSUITE CFG* TO *powerSUITE SYSCFG* 34
- Changed *cfg page* to *SYSCFG page* 40

Changes from A Revision (June 2019) to B Revision **Page**

- Changed Figure 24 19
- Changed Figure 25 20

Changes from Original (March 2019) to A Revision **Page**

- Added can be used for protection and monitoring 2
- Added two methods to sense using shunt current sense and other is rogowski's coil. Only one is needed, used to implement synchronous rectification in the reverse direction i.e. secondary to primary. Also used for protection. 2
- Added used to implement the battery current control loop. 3
- Added used to implement the synchronous rectification for the foward direction power flow i.e. primary to secondary. 3
- Changed hal to HAL 18
- Added with Power Flow Secondary to Primary to image title 20
- Deleted Once the icon is clicked, a pop-up window appears asking for a location to create the project. The user can also save the project inside the workspace itself. Once the location is specified, a project is created, and a GUI page appears with modifiable options for the solution (). This GUI can be used to change the parameters, select lab level, and change sensing circuit parameters. If this GUI page does not appear, refer to the FAQ section under powerSUITE in the C2000WARE-DIGITAL-POWER-SDK resource explorer. 27
- Added For simplicity the software diagrams and structure only show ISR1 that is triggered the first time. 31
- Added ISR2 has two variants ISR2_primToSecPowerFlow and ISR2_secToPrimPowerFlow, one for the power flow from primary to secondary side and other for secondary to primary. This is done to to optimize CPU cycles when controlling power flow in different directions. For simplicity of explaination either of them are just referred to as ISR2 in respective labs. 31
- Added The image is taken for the system in open loop and when a period change is initiated through the watch window, hence only one time ISR1 trigger is observed. For a closed-loop system, the period will only have minor changes from one control ISR cycle to the other and hence the ISR1 will be triggered repeatedly. 31
- Changed four to eight 32
- Changed "Open Loop Check" to "Primary to Secondary Power Flow, Open Look Check PWM driver," 32

- Changed Lab 2 to 3; changed "Closed Voltage Loop Check, with resistive load connected" to "Primary to Secondary Power Flow, Closed Voltage Loop Check, with resistive load connected on secondary." 32
- Changed Lab 3 to 4; changed "Closed Current Loop Check, with resistive load connected on secondary." to "Primary to Secondary Power Flow, Closed Current Loop Check, with resistive load connected on secondary. "..... 32
- Changed Lab 4 to 5 32
- Added Primary to Secondary Power Flow, ... 32
- Added Lab 6, 7, and 8 32
- Added section title "Lab1. Primary to Secondary Power Flow, Open Loop Check PWM Driver 36
- Changed Lab 1 Software Diagram to Lab 1 and 2 Software Diagram 37
- Changed Lab 1 to Lab 2 37
- Changed Lab 1 to Lab 2; added PWM with Protection, Prim to Sec Power Flow 37
- Changed code 37
- Changed setupdebugenv_build1.js to setupdebugenv_lab2and7.js 38
- Changed Lab 1 to Lab 2 38
- Changed Lab 1 to Lab 2 39
- Changed Lab 1 to Lab 2 39
- Added which will correspond to a frequency of 639 40
- Changed Lab 1 to Lab 2 40
- Added which will make generate frequency of 374kHz 40
- Changed Lab 1 to Lab 2 40
- Added cross-reference and added "shows the measurement of the current loop plant at 500kHz." 44
- Changed title of image 45
- Changed Lab 2 to Lab 3; added "Primary toSecondary Power Flow," 45
- Changed title from "Lab 2 Control Software Diagram: Closed Current Loop" to "Software Diagram: Closed Voltage Loop Primary to Secondary Power Flow" 46
- Changed Lab 2 to Lab 3 46
- Changed Lab 2 to Lab 3; added "...with Resistive Load Prim to Sec Power Flow" 46
- Added , for example two runs at 500kHz and 300kHz with the designed compensator are shown to be stable in Figure 46 and 46
- Added image title 47
- Added "...When Resistive Load is Connected at the Output, With Measurement Data at 333kHz" 48
- Changed code 49
- Changed setupdebugenv_build2.js to setupdebugenv_lab3.js" 49
- Deleted The watch window will appear as shown in... 49
- Added ...with Resistive Load at the Output 51
- Changed Lab 3 to Lab 4; Added "Primary to Secondary Power Flow,..." 52
- Changed Lab 3 to Lab 4 52
- Changed Lab 3 to Lab 4 52
- Deleted (see 3) 54
- Changed Lab 4 to Lab 4; added "Primary to Secondary Power Flow,..." 56
- Changed Lab 4 to Lab 5 56
- Changed Lab 4 to Lab 5 57
- Changed Lab 4 to Lab 5 58
- Changed Lab 4 to Lab 5 58
- Changed Lab 4 to Lab 5 58
- Changed code 59
- Added new subsection 62
- Added new subsection: Lab 8, Secondary to Primary Power Flow, Closed Voltage Loop Check, with Resistive Load Connected on Primary and image 67
- Added subjection Setting Software Options 67
- Added subjection title Building and Loading the Project and Setting up Debug Environment 69
- Added step "Right-click the project name and click Rebuild Project 69
- Added subsection Measure SFRA for Closed Voltage Loop 70
- Added TMS320F28004x microcontrollers data manual link 78

- Deleted Texas Instruments, TMS320F28004x microcontrollers data manual 78

IMPORTANT NOTICE AND DISCLAIMER

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

These resources are intended for skilled developers designing with TI products. You are solely responsible for (1) selecting the appropriate TI products for your application, (2) designing, validating and testing your application, and (3) ensuring your application meets applicable standards, and any other safety, security, or other requirements. These resources are subject to change without notice. TI grants you permission to use these resources only for development of an application that uses the TI products described in the resource. Other reproduction and display of these resources is prohibited. No license is granted to any other TI intellectual property right or to any third party intellectual property right. TI disclaims responsibility for, and you will fully indemnify TI and its representatives against, any claims, damages, costs, losses, and liabilities arising out of your use of these resources.

TI's products are provided subject to TI's Terms of Sale (www.ti.com/legal/termsofsale.html) or other applicable terms available either on ti.com or provided in conjunction with such TI products. TI's provision of these resources does not expand or otherwise alter TI's applicable warranties or warranty disclaimers for TI products.

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