TI Designs

C2000™ Non-Isolated Bi-Directional Buck-Boost User's Guide

TI Designs

TI Designs provide the foundation that you need including methodology, testing, and design files to quickly evaluate and customize the system. TI Designs help *you* accelerate your time to market.

Design Resources

TIDM-BUCKBOOST-BIDIR	Design Folder

TI E2E™ Community

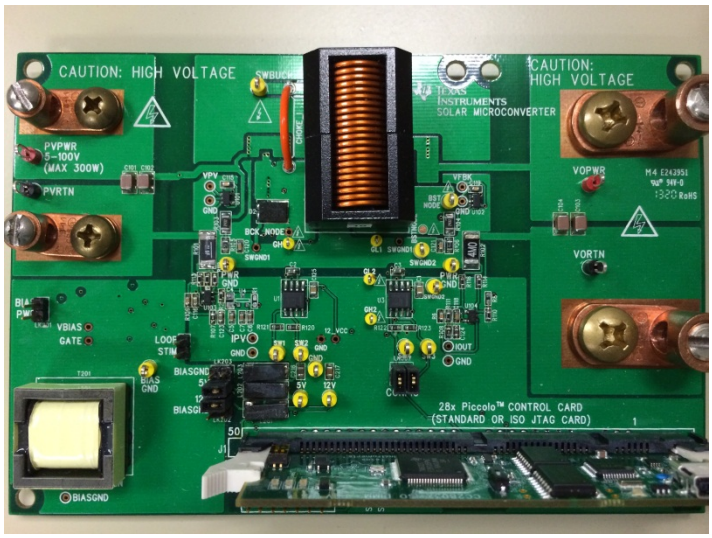
ASK Our E2E Experts
WEBENCH® Calculator Tools

Design Features

- Bi-Directional Buck-Boost Converter
- Fast PWM Switching, 250 Khz
- High Efficiency > 95% Including Bias Drop
- SFRA Support in Software

Featured Applications

- Renewable Energy
- DC-DC Converters
- Electric Vehicles



An IMPORTANT NOTICE at the end of this TI reference design addresses authorized use, intellectual property matters and other important disclaimers and information.

1 Introduction

The C2000™ Bi-Directional Non-Isolated Buck-Boost TI design (TIDM_BUCKBOOST_BIDIR) illustrates the use of a C2000 microcontroller (MCU) for controlling a bi-directional non-isolated buck-boost power stage. This power stage can also be used for an emerging concept called solar micro-converter. This document reviews the design of the board, hardware implementation, and software structure that is used to control the power stage.

WARNING

This EVM is meant to be operated in a lab environment only and is not considered by TI to be a finished end-product fit for general consumer use.

This EVM must be used only by qualified engineers and technicians familiar with risks associated with handling high-voltage electrical and mechanical components, systems, and subsystems.

This equipment operates at voltages and currents that can result in electrical shock, fire hazard, and/or personal injury if not properly handled or applied. Use equipment with necessary caution and employ appropriate safeguards to avoid personal injury or property damage.

It is the user's responsibility to confirm that the voltages and isolation requirements are identified and understood prior to energizing the board or simulation. When energized, the EVM or components connected to the EVM should not be touched.

2 Getting Familiar with the Design

This design follows a control card concept; therefore, any device from the C2000 family with a DIMM100 control card can be used. The software associated with this design uses the F28035 MCU control card with an isolated JTAG connection.

2.1 Design Overview

Figure 1 illustrates the DC-DC buck-boost topology implemented using the UCC27211 driver. The power stage comprises four FETs: SW1 and SW2 form the buck pair, and SW3 and SW4 form the boost pair when power is being transferred from V1 to V2. When the power flows in the reverse direction, SW4 and SW3 form the buck pair and SW2 and SW1 form the boost pair. The board contains an onboard bias power supply that can power the supporting control, power processing, and sensing circuit including the MCU. This bias is powered from the Vpv / V1 side.

Looking at the power stage in Figure 1, the ideal DC gain of the converter is given by Equation 1:

$$G = \frac{V_o}{V_i} = \frac{D_{bu}}{1 - D_{bo}} \quad (1)$$

Where D_{bu} is the duty cycle of the switch SW1 and D_{bo} is the duty cycle of the switch SW3 when power is transferred from V1 to V2. MOSFET usage and the symmetry of the power stage enable it to be used for bi-directional-buck-boost-type applications.

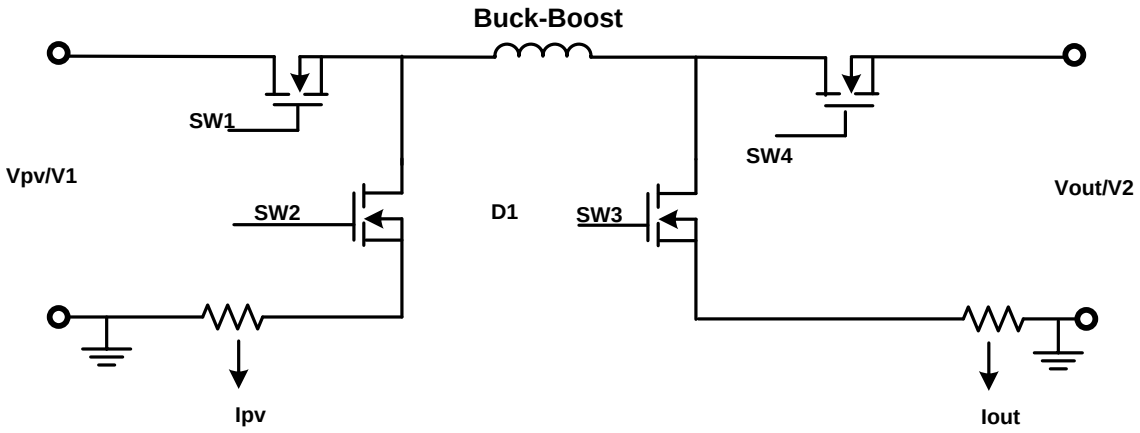


Figure 1. DC-DC Buck-Boost Power Stage

Figure 2 shows the positioning of key components on the kit base board.

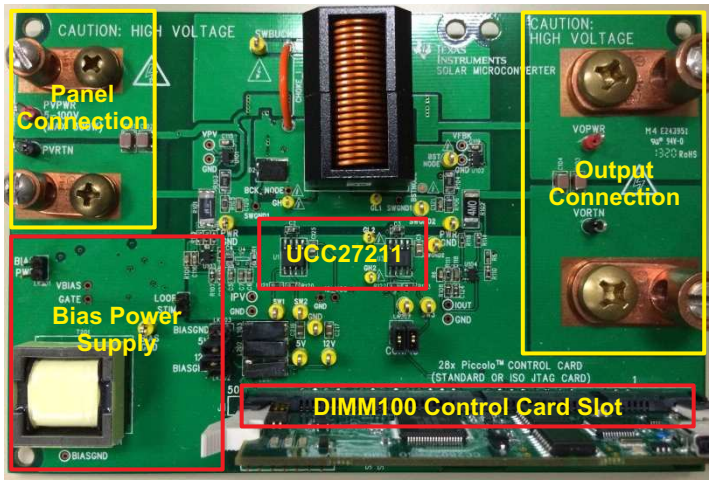


Figure 2. Bi-Directional Buck-Boost Board Key Sections Highlighted

Power stage parameters:

- Input voltage: 10 to 100 V (Panel Input / V1) DC max
- Input current: 0 to 8 A (Panel Input); input power must be < 300 W
- Output voltage: 5 to 100-V DC max
- Output current: 0 to 8 A
- Power rating: 300 W max
- Fsw: 250 Khz

2.2 Switching Scheme

The switch pairs, SW1 & SW2 and SW3 & SW4, are switched complimentary to each other, thus giving synchronous buck and boost operation. The switching scheme for the switches is illustrated in Figure 3.

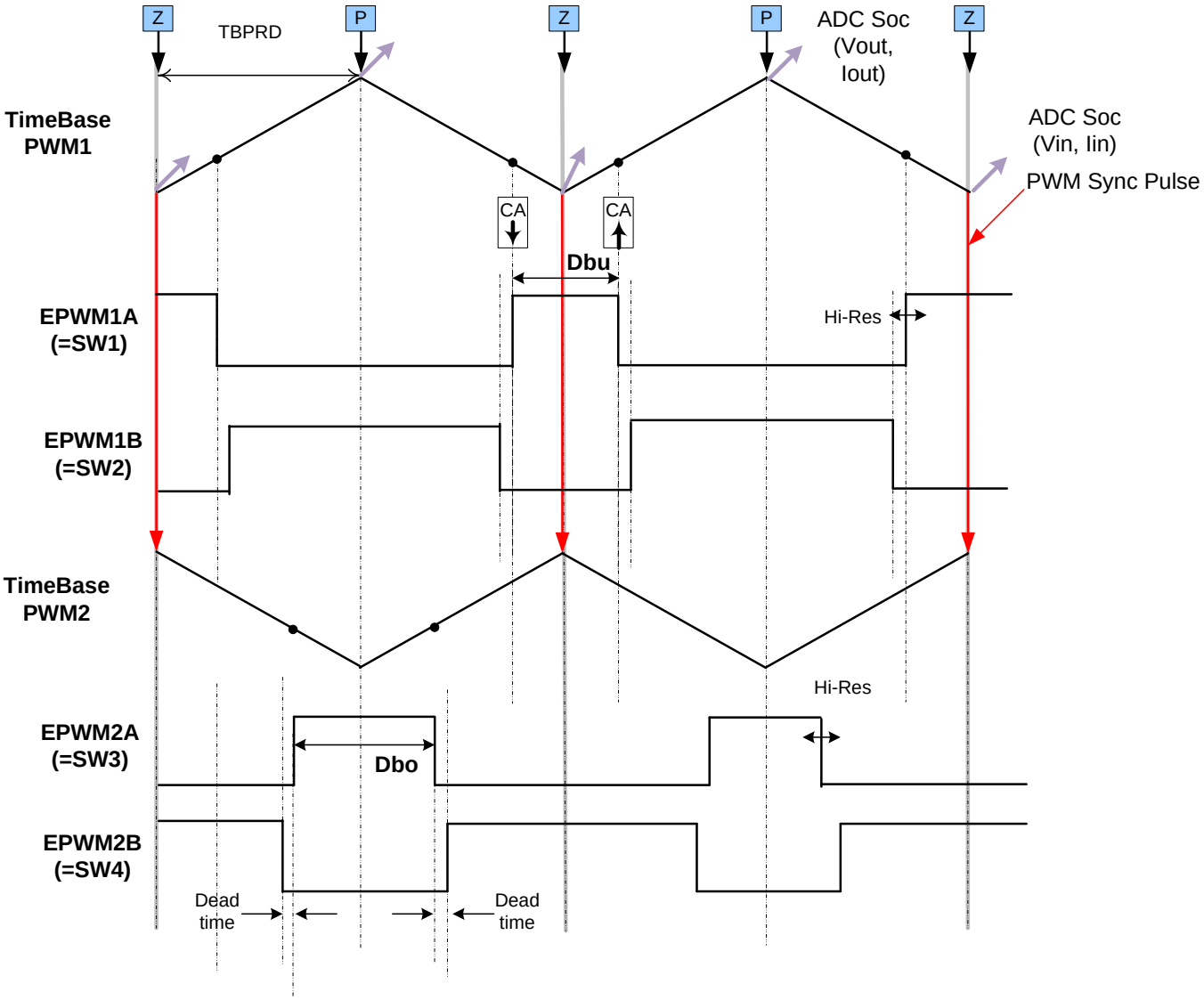


Figure 3. Switching Scheme for Synchronous Buck-Boost Power Stage

A novel switching scheme, as proposed in *High-Efficiency, Wide-Load Range Buck/Boost Photovoltaic Microconverter* [1], is used to switch the power stage and achieve good efficiency. The operation range of the stage is split into four regions, as seen in Table 1, and the duty value is computed for each region so that the gain increases monotonically between the different modes.

Table 1. Mode of Operation of Buck-Boost Stage and Duty Calculation

DUTY OR GAIN NEEDED	D_{bu}	D_{bo}	MODE
0 – BuckModeMaxGain	Gain	0	Buck
BuckModeMaxGain – ConstantBoostMaxGain	Gain – C1	ConstantBoostDuty	Constant boost, buck
ConstantBoostMaxGain – ConstantBuckMaxGain	ConstantBuckDuty	Gain – C2	Constant buck, boost
ConstantBuckMaxGain – up	1.0	Gain – C3	Boost

The boundary values for changing from one mode to another, such as BuckModeMaxGain, ConstantBoostMaxGain, and ConstantBuckMaxGain, are determined empirically to get the maximum efficiency. Also, the values to calculate the duty for the buck and the boost switches (such as C1, C2, C3, and so on) are determined empirically to make sure the gain increases monotonically. (Equation 1 alone cannot provide this information as it does not take into account the minimum on time for the switches.) The values can be empirically tuned to get the best efficiency as they are readily available as variables in the software. See the Excel file "Gain_Worksheet.xlsx" provided with the design for the values used with this design.

3 Hardware Details

3.1 Resource Allocation

Figure 4 shows the board in a block-diagram format and illustrates the major connections and feedback values that are being mapped to the C2000 MCU. Table 2 lists these resources. Note that the table only lists the resources used for power stage and that are mapped to the DIMM100 connector on the board.

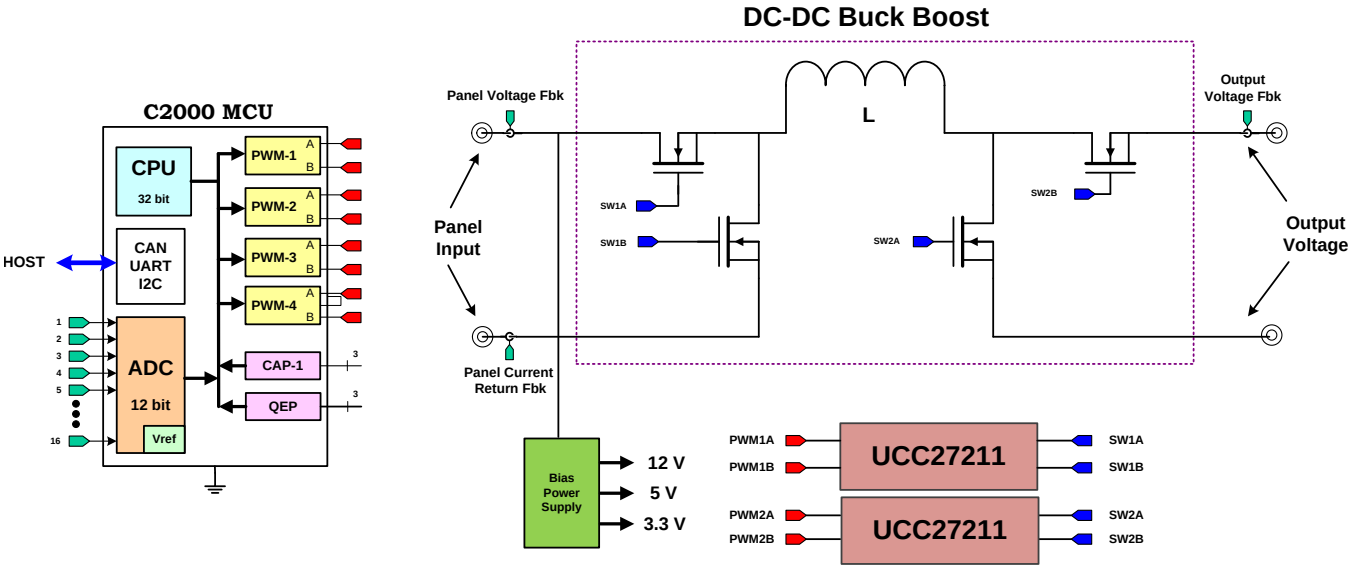


Figure 4. Kit Block Diagram with C2000 MCU [LK301 Jumpers in Vertical (Normal) Configuration]

Table 2. Resource Mapping: PWM, ADC [LK301 and LK302 Jumpers in Vertical (Normal) Configuration]

SIGNAL NAME	PWM CHANNEL / ADC CHANNEL NO / RESOURCE MAPPING F2803x	FUNCTION
INL1	EPWM-1B	PWM input, Buck low side
INH1	EPWM-1A	PWM input, Buck high side
INL2	EPWM-2A	PWM input, Boost low side
INH2	EPWM-2B	PWM input, Boost high side
VPV	ADCIN-B2	V1, Panel voltage feedback
IPV	ADCIN-A2	I1, Panel current feedback
VFBK	ADCIN-B4	V2, Output voltage feedback
IOUT	ADCIN-A4	I2, Output current feedback
12_VCC	ADCIN-B6	12-V board supply feedback
5_VCC	ADCIN-A6	5-V board supply feedback
3V3	ADCIN-A7	3.3-V board supply feedback

NOTE: Connectivity peripherals may differ from one device to the other including Ethernet, USB, CAN, SPI, and so on.

3.2 Jumpers and Connectors

Table 3 lists the jumpers on the board.

Table 3. Jumpers and Connectors on the Kit Base Board

JUMPER OR CONNECTOR	DESCRIPTION
LK201	Jumper – BIAS PWR. Connecting this jumper enables the internal bias power generation on board (also requires LK202 and LK203 to be connected). When enabled, the bias power is drawn from the panel voltage (internal bias powers are enabled when PVIN rises above approximately 9 V, and disabled when falls below approximately 8 V) If this jumper is not enabled, then the 12-, 5-, and 3.3-V supplies must be externally connected to the 5- and 12-V test points (3.3 V is generated from the 5-V supply) Default: Connected
T201	Transformer for generating the internal bias powers when LK201, LK202, and LK203 are connected
LOOP STIIM	When connected, the internally generated bias supplies are slightly increased Default: Not connected
LK202, LK203	Connects the transformer (T201) output voltages to the onboard 12 V (LK202) and 5 V (LK203). 3.3 V is generated from the 5-V supply To enable the voltages, connect the jumpers horizontally at the 5- and 12-V labels (requires LK201 to be enabled) Default: Connected horizontally at 5- and 12-V labels
LK301	Jumpers for alternative connections Vertical configuration (normal): • EPWM-2A controls Boost low side MOSFET • EPWM-2B controls Boost high side MOSFET Horizontal configuration (alternative): • EPWM-2A controls Boost high side MOSFET • EPWM-2B controls Boost low side MOSFET Default: Vertical connection

4 Software

4.1 Project Framework

Power stage control requires a real-time ISR for the closed loop control of the DC-DC stage. The project makes use of the "C-background/C-ISR/ASM-ISR" framework. The fast ISR, controlling DC-DC Boost stage, runs in assembly environment using the digital power library. The project uses C-code as the main supporting program for the application, and is responsible for all system management tasks, decision making, intelligence, and host interaction.

The project framework also integrates the Software Frequency Response Analyzer (SFRA), which the user can use to measure the frequency response of the power supply and verify if the control design has enough margin of stability. For more details on the SFRA, refer to http://www.ti.com/tool/sfra.

The key framework C files used in the project are:

BuckBoostBiDir-Main.c— This file is used to use, run, and manage the application. This file contains the interrupt service routine that is used to run the control loop of the power stage.

BuckBoostBiDir-DevInit_F2803x.c— This file contains all the initialization routines and configuration of I/Os and peripherals for this application. This file also includes functions such as setting up the clocks, PLL, Watchdog, and so on.

BuckBoostBiDir-Settings.h— This file contains of setting such as incremental build option and various defines for PWM frequency and ISR triggers that are used in the project framework.

BuckBoostBiDir-Includes.h— This file contains of all the header files used by the project.

BuckBoostBiDir-DPL.asm— This file contains time critical “control type” code. This file has an initialization section (one-time execute) and a run-time section that executes at control loop execution rate. This routine is called from the C-based ISR in Main.c file.

Figure 5 shows the structure of the software with the main background loop and the DC-DC ISR.

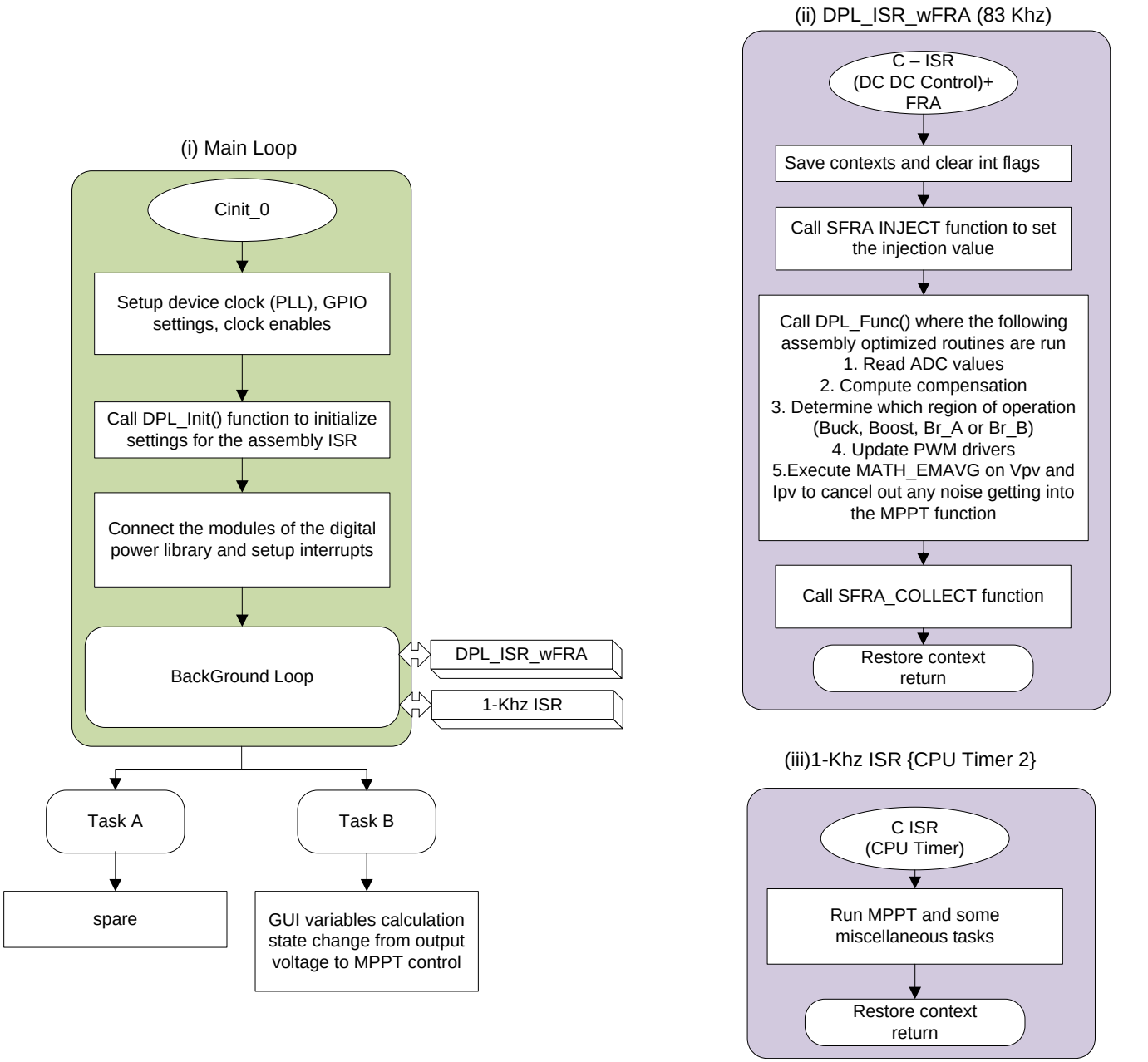


Figure 5. Software Structure: (i) Main Loop, (ii) DC-DC ISR, (iii) 1-Khz ISR

4.2 Project Dependencies and Resources

- Hardware kit: TIDM_BUCKBOOSTBIDR [Rev 1D]
- Control card: TMDSCNCD28035ISO
- Software IDE: CCSv6 or later
- Control suite dependencies:
  - Device support (F28035 header files): controlSUITE\device_support\f2803x\v125
  - IQMath library: controlSUITE\libs\math\IQmath\v160
  - Digital power library: controlSUITE\app_libs\digital_power\f2803x_v3.4
  - Solar library: controlSUITE\app_libs\solar\v1.0\IQ
  - SFRA library: controlSUITE\app_libs\SFRA\v1_00_00_00\IQ

4.3 Control Description

This design provides information for sensing the input and output voltage and input and output current. Several control schemes can be implemented. For simplicity, the associated software implements the following three modes:

1. Output voltage control (Figure 6)
2. Maximum power point tracking (MPPT) control, input current control (Figure 9)
3. Voltage control in the reverse direction (Figure 7)

4.3.1 Output Voltage Control

For controlling the output voltage, the control scheme shown in Figure 6 is used. The output voltage is sensed (V_{out}) and compared with the reference value (V_{out_ref}), and the 2p2z compensator calculates the effort needed. The voltage controller is executed at a rate of 83 kHz (one-third of the PWM switching frequency).

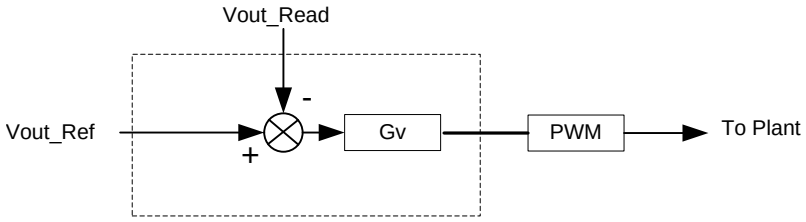


Figure 6. Output Voltage Control

The switching frequency of the DC-DC stage is fairly high at 250 Khz. A section of the control ISR for the DC-DC is implemented in an optimized assembly function, which is called the C ISR. In the project, the DC-DC ISR (DPL_ISR_wFRA()) is invoked every third switching cycle. Figure 7 gives the software diagram for the DC-DC stage using the optimized blocks from the digital power library and SFRA routine in C.

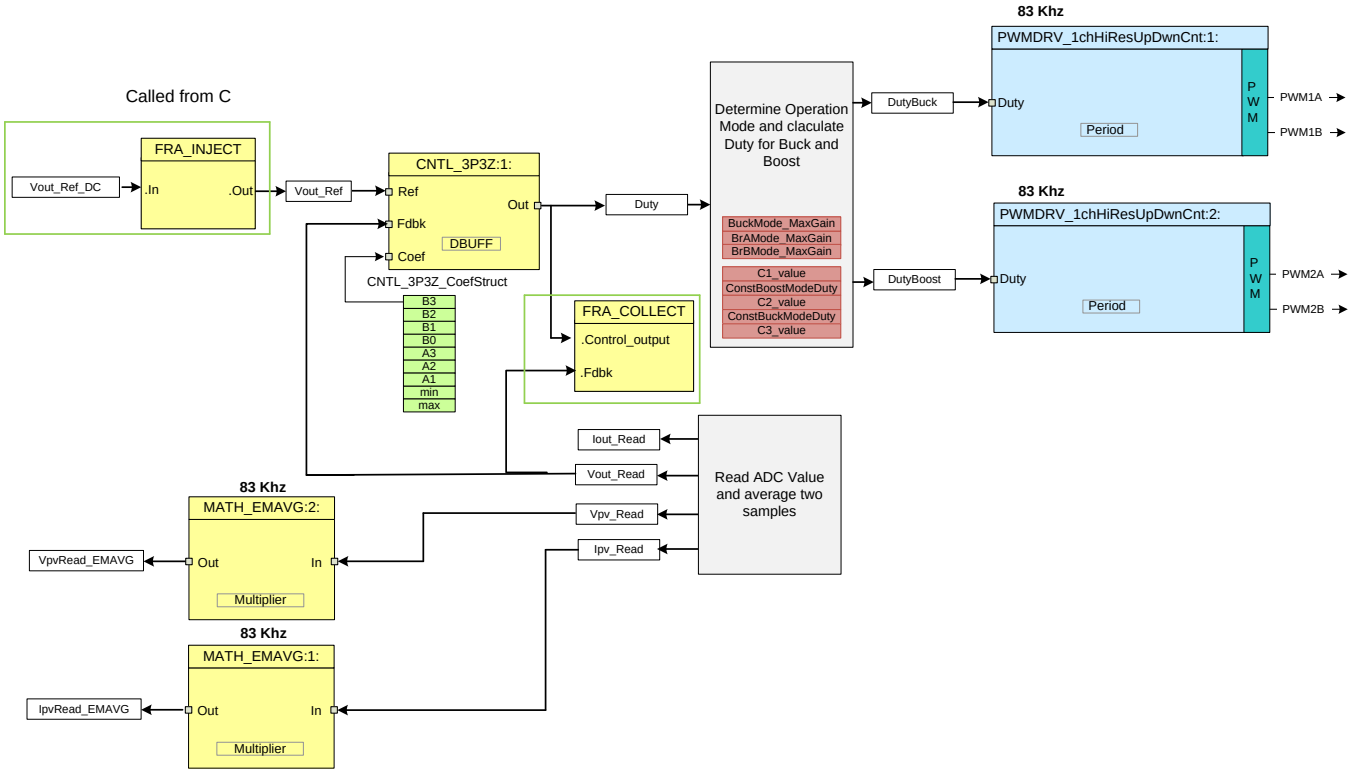


Figure 7. DC-DC Output Voltage Control

The ADC result registers are read, converted to normalized values, and stored in variables I_{pvRead}, V_{pvread}, $I_{outread}$, and $V_{outread}$. A 2-pole 2-zero controller (CNTL_2P2Z) closes the DC-DC voltage loop. The output of the 2p2z is limited to 0 to 1. This output is then multiplied by 2, and the duty for the buck side switches and boost side switches is determined in an assembly routine according to Table 1. The routine also imparts a refresh cycle on the high side drivers if need be.

NOTE: As the control is executed every third PWM cycle, three refresh pulses will be generated. This situation is not the most ideal, yet it saves any additional software overhead and is seen to work appropriately.

The PWMDRV_1ch_UpDwnCnt, PWMDRV_1ch_UpDwnCntCompl block drives the DC-DC stage buck and boost side switches. Note the original digital power library block does not configure the channel B to be switched complementary to channel A. This process is taken care of in the Main.c where the deadband for the PWM is configured.

Notice the color coding for the software blocks in Figure 7. The blocks in dark blue represent the hardware modules on the C2000 controller. The blocks in blue are the software drivers for these modules. Blocks in yellow are the controller blocks for the control loop.

The following code snippet shows the I/O connections between the different blocks used from the digital power library to implement the DC-DC MPPT control software. This code can directly be related to the control diagram shown in Figure 7.

```
// Connect the PWM Driver
PWMDRV_1chHiResUpDwnCnt_Duty1 = &DutyBuck;
PWMDRV_1chHiResUpDwnCnt_Duty2 = &DutyBoost;

ADCDRV_4ch_RltPtrA=&Ipv_Read;
ADCDRV_4ch_RltPtrB=&Vpv_Read;
ADCDRV_4ch_RltPtrC=&Vout_Read;
ADCDRV_4ch_RltPtrD=&Iout_Read;

//2p2z connections for outpur voltage loop
CNTL_3P3Z_Ref1 = &Vout_Ref;
CNTL_3P3Z_Out1 = &Duty;
CNTL_3P3Z_Fdbk1= &Vout_Read;

CNTL_3P3Z_Coef1 = &CNTL_2P2Z_CoefStruct1.b2;
```

The run time ISR calls the FRA functions and then DPL_Func(), which consists of just calling the run time macros from the digital power library.

```
interrupt void DPL_ISR_wFRA()
{
..
        Vout_Ref=SFRA_IQ_INJECT(Vout_Ref_DC);
        DPL_Func();
        SFRA_IQ_COLLECT(&Duty,&Vout_Read);
...
}
```

4.3.2 MPPT Mode (Solar Micro-Converter Mode)

The solar panel (or PhotoVoltaic [PV] panel, as it is more commonly called) is a DC source with a nonlinear V-versus-I characteristic. The key challenges in PV system design are to extract maximum power from the panel by operating the panel at the maximum power point (MPP) of this nonlinear V-versus-I curve, and to convert the power such that it can be used to charge batteries, run DC loads, run AC loads, or feed power into the electrical grid. A typical PV-grid-tied inverter consists of a string of PV panels tied together to a single inverter stage; these are called string inverters. An emerging system architect that supplements the string-inverter paradigm involves DC-DC converters (referred to here as Micro-Converters) dedicated to individual PV modules (Figure 8). The localized MPPT at each panel improves the performance of the system under partial shading and unmatched panels conditions.

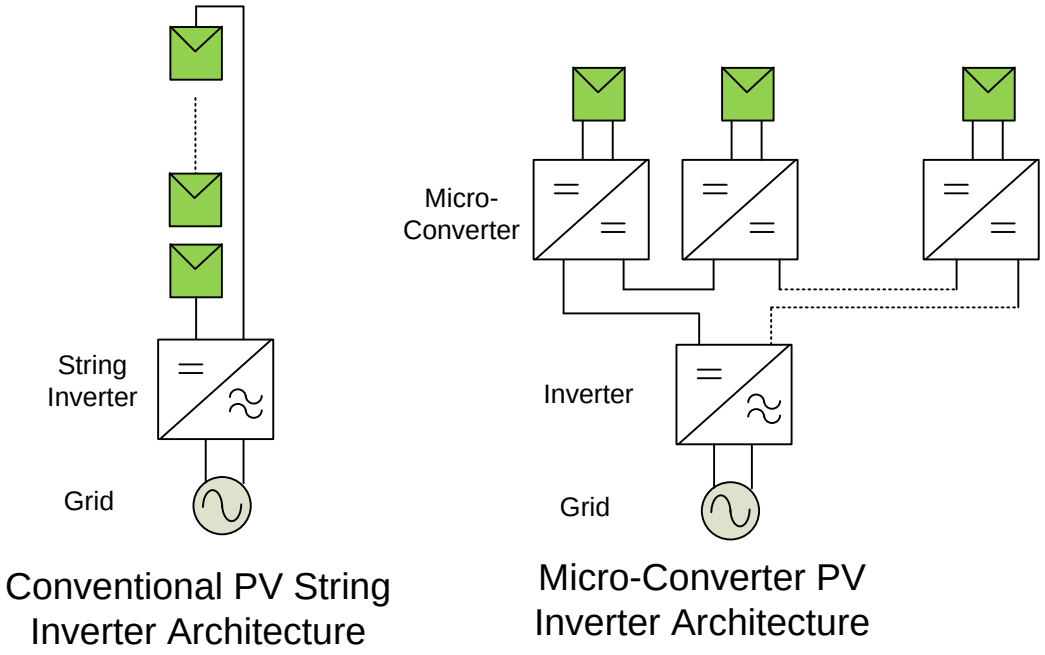


Figure 8. PV Inverter Architectures

To control the power stage in this mode, the control scheme shown in Figure 9 is used.

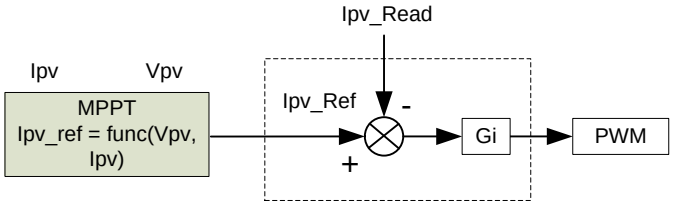


Figure 9. MPPT Control

MPP, however, is not fixed due to the nonlinear nature of the PV cell and changes with temperature, light intensity, and so on. Therefore, different techniques are used to track the MPP of the panel's incremental conductance algorithms like Perturb and Observe. These techniques try to track the MPP of the panel under given operating conditions and are thus referred to as MPPT techniques and algorithms. The kit has a DC-DC buck-boost stage that can take input voltage from the solar panel and provide the maximum power possible at the output.

To track the MPP, input voltage (V_{pv}) and input current (I_{pv}) are sensed. The MPPT is realized using an inner current loop that regulates I_{pv}. When a higher input voltage is set as a reference, more current or power is drawn from the panel, and the panel voltage drops. The current and voltage controllers are executed at a rate of 83 kHz (one-third of the PWM switching frequency) while the MPPT controller is executed at a much slower rate at approximately 10 Hz. The DC-DC stage output voltage is not being controlled through software; therefore, an appropriate load must be connected to the output so that the output voltage does not rise above its specified limits.

Because the switching rate of the DC-DC stage is fairly high at 250 Khz, the control ISR for the DC-DC is implemented in an optimized assembly ISR (ASM – ISR), which uses components from the digital power library. In the project, the DC-DC ISR is invoked every third switching cycle because PV panel output does not change very fast. Figure 10 shows the software diagram for the DC-DC stage using the optimized blocks from the digital power library.

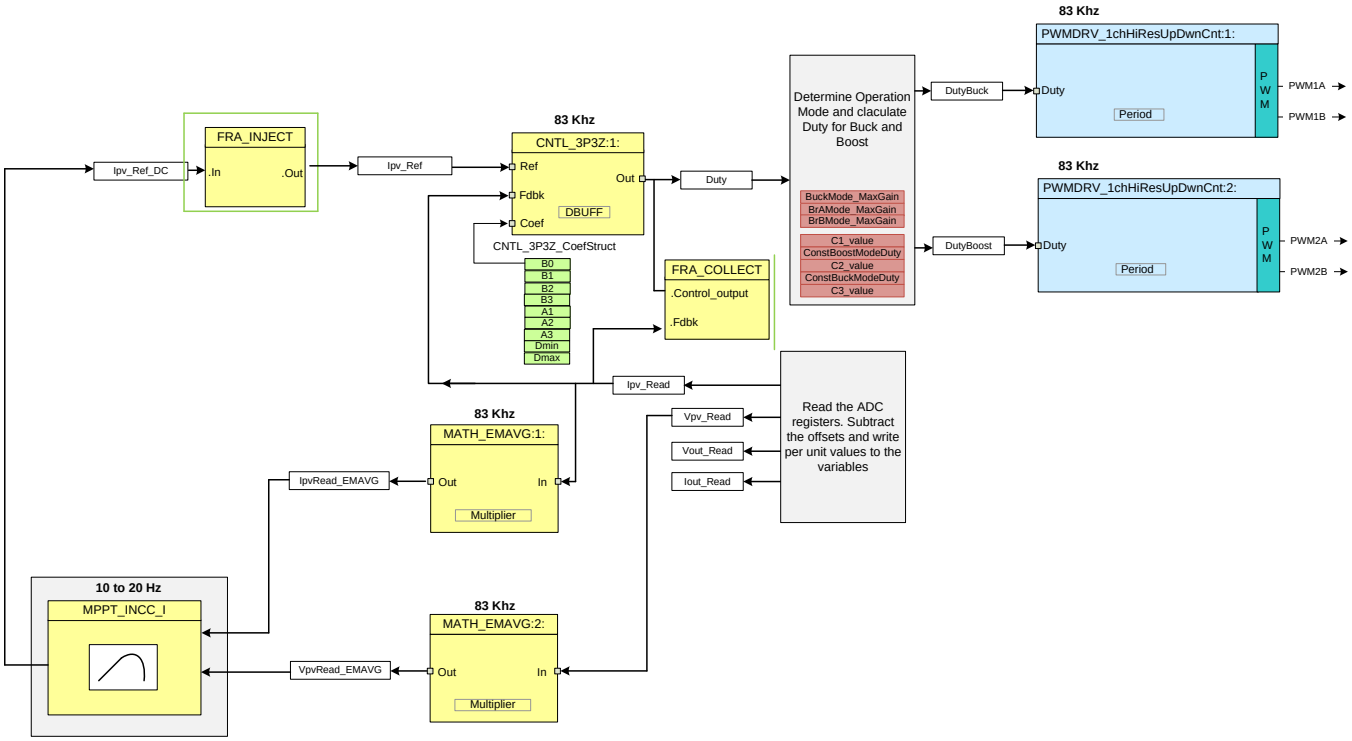


Figure 10. DC-DC With MPPT Software Diagram

The ADC result registers are read and normalized values stored in variables I_{pvRead}, V_{pvread}, $I_{outread}$, and $V_{outread}$. A CNTL_2P2Z is used to close the current loop. MPPT algorithm provides reference input current that is used by the compensator as reference. The output of the current loop controller decides the amount of gain needed in the system. The output of the 2p2z is limited to 0 to 1. This output is then multiplied by 2, and the duty for the buck side switched and boost side switched is determined in an assembly routine according to Table 1. The routine also imparts a refresh cycle on the high side drivers if need be.

NOTE: As the control is executed every third PWM cycle, three refresh pulses will be generated. This is not the ideal situation, yet this saves any additional software overhead and is seen to work appropriately.

The PWMDRV_1ch_UpDwnCnt, PWMDRV_1ch_UpDwnCntCompl block is used to drive the DC-DC stage buck and boost side switches. Note the original digital power library block does not configure channel B to be switched complimentary of channel A. This switch is taken care of in Main.c where the deadband for the PWM is configured. The panel current and voltage are filtered using the MATH_EMAVG block to remove any noise on the panel current and voltage sensing that may confuse the MPPT algorithm.

Notice the color coding for the software blocks in Figure 10. The blocks in dark blue represent the hardware modules on the C2000 controller. The blocks in blue are the software drivers for these modules. Blocks in yellow are the controller blocks for the control loop. Although a 2p2z controller is used here, the controller could very well be a PI/PID, a 3p3z, or any other controller that can be implemented for this application. Similarly for MPPT, users can choose to use a different algorithm.

The following code snippet shows the I/O connections between the different blocks used from the digital power library to implement the DC-DC MPPT control software. This snippet can directly be related to the control diagram shown in Figure 10.

```
// Connect the PWM Driver
PWMDRV_1chHiResUpDwnCnt_Duty1 = &DutyBuck;
PWMDRV_1chHiResUpDwnCnt_Duty2 = &DutyBoost;

ADCDRV_4ch_RltPtrA=&Ipv_Read;
ADCDRV_4ch_RltPtrB=&Vpv_Read;
ADCDRV_4ch_RltPtrC=&Vout_Read;
ADCDRV_4ch_RltPtrD=&Iout_Read;

//3p3z connections for current loop
CNTL_3P3Z_Ref1 = &Ipv_Ref;
CNTL_3P3Z_Out1 = &DutyZero;
CNTL_3P3Z_Fdbk1= &Ipv_Read;
CNTL_3P3Z_Coef1 = &CNTL_2P2Z_CoefStruct1.b2;

// MATH_EMAVG1 block connections
MATH_EMAVG_In1=&Ipv_Read;
MATH_EMAVG_Out1=&Ipv_Read_EMAVG;
MATH_EMAVG_Multiplier1=_IQ30(0.001);

// MATH_EMAVG2 block connections
MATH_EMAVG_In2=&Ipv_Read;
MATH_EMAVG_Out2=&Ipv_Read_EMAVG;
MATH_EMAVG_Multiplier2=_IQ30(0.001);
```

The run time ISR calls the FRA functions and then DPL_Func(), which consists of calling the run time macros from the digital power library. The MPPT algorithm is called from a background task in the background C framework.

4.3.3 Reverse Power Flow

The DC-DC buck-boost stage can be used in bi-directional mode by a simple switch in the software. The control scheme used is illustrated in Figure 11. To demonstrate the reverse flow, the input voltage is controlled.

NOTE: In a typical bi-directional application, the current control scheme is used. This mode is just to illustrate the hardware capability in reversing the power flow.

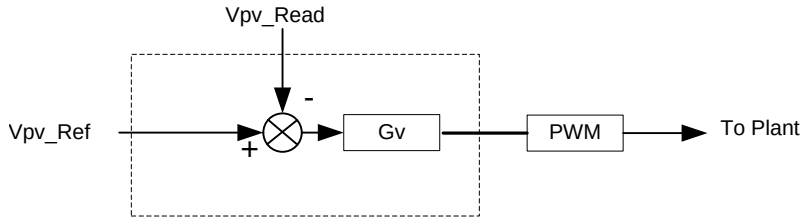


Figure 11. Reverse Power Flow

The control of the stage is described in Figure 11. To enable the reverse flow, change the DIRECTION definition in the "Settings.h" file.

```
#define DIRECTION 2 // 1 for FORWARD // 2 for Backward
```

Notice that with the reverse direction, the PWM switches are swapped. The piccolo PWM module is very versatile and offers the option to swap the high-side and low-side switches when operating in buck-boost mode.

```
#elif (DIRECTION==2)
    PWMDRV_1chHiResUpDwnCnt_Duty1 = &DutyBoost;
    PWMDRV_1chHiResUpDwnCnt_Duty2 = &DutyBuck;
    EALLOW;
    EPwm1Regs.HRCNFG.bit.SWAPAB=1;
    EPwm2Regs.HRCNFG.bit.SWAPAB=1;
    EDIS;
#endif
```

Additionally the refresh pulse also needs to be applied in the opposite way by refreshing PWM1 when in buck mode and refreshing PWM2 when in boost mode. This is also taken care of in the 1-Khz ISR routine.

The control loop net terminals are also changed as follows:

```
#elif (DIRECTION==2)
    CNTL_2P2Z_Ref1 = &Vpv_Ref;

    #if (OPEN_LOOP == 1)
    CNTL_2P2Z_Out1 = &DutyZero;
    #else
    CNTL_2P2Z_Out1 = &Duty;

    CNTL_2P2Z_Fdbk1= &Vpv_Read;
    CNTL_2P2Z_Coef1 = &CNTL_2P2Z_CoefStruct1.b2;

#endif
```

The output voltage (here, panel input voltage) is sensed (V_{pv}) and compared with the reference value (V_{pv_ref}), and the 2p2z compensator calculates the effort needed. The voltage controller is executed at a rate of 83 kHz (one-third of the PWM switching frequency).

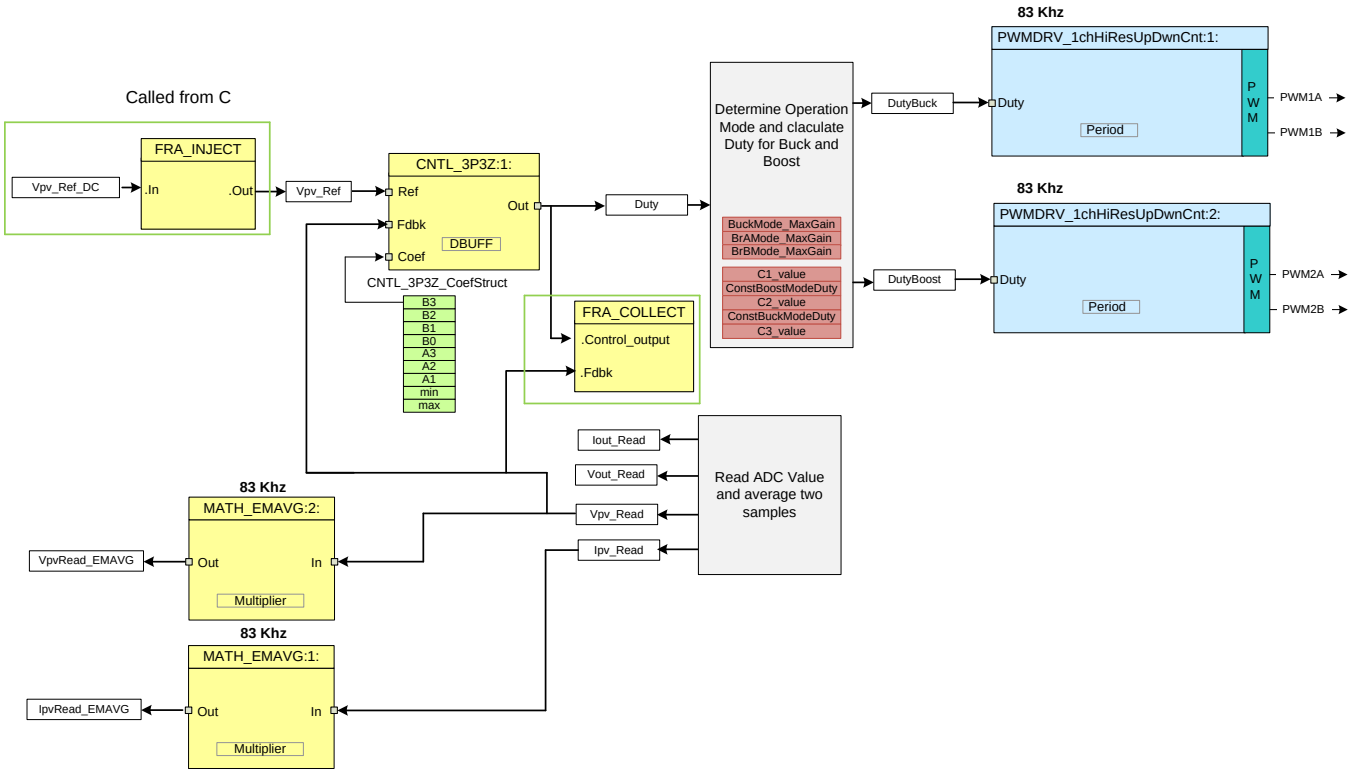


Figure 12. DC-DC Output Voltage Control in Reverse Direction

5 Running the Software using CCS

5.1 Hardware Setup Instructions

NOTE: Do not power up the board before you have verified these settings.

Before starting the labs, make sure the following settings are correct.

1. Make sure nothing is connected to the board, and no power is being supplied to the board.
2. Insert the controlCARD into the controlCARD connector if it is not already installed (F28035 ISO DIMM CARD).
3. Set the following switch settings on the controlCARD:
   - Control Card SW3 is in the ON position (JTAG Connection)
   - Control Card SW2, Position 1 = ON, Position 2 = ON
4. Connect a DC load between VOPWR and VORTN (make sure the output load is of the appropriate value needed for the test).
5. Connect a USB cable (mini-to-A cable) from the control card to the host computer. LD4 on the control card will light up to indicate a successful USB connection.
6. Make sure the following jumpers are connected or disconnected:
   (a) LK201: Connected (enables on-board bias power)
      (i) If desired to supply 5 V and 12 V externally, then leave this jumper open, and supply 5 V and 12 V externally to their test points.
   (b) LOOP STIM: Leave the jumper open.
   (c) LK202, LK203: Connect jumpers horizontally at 5- and 12-V labels.
      (i) If 5 V and 12 V are supplied externally, leave these jumpers open.
   (d) LK301, LK302: Connect jumpers vertically for normal configuration.
7. Verify a different mode of operation with a slightly different setup:
   (a) Output voltage control

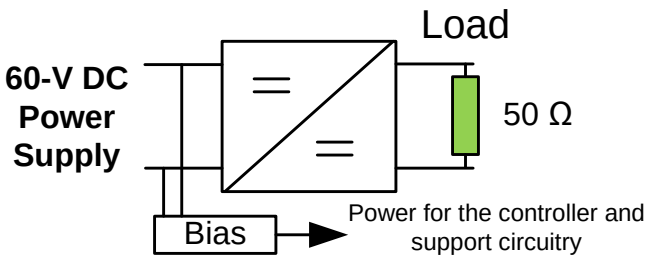


Figure 13. Output Voltage Control Setup

(b) MPPT mode
Connect a solar panel or panel emulator between PVPWR and PVRTN. If the solar panel is producing enough energy, the bias supply will power up the controller and the green LED, LD1 on the control card will light up to indicate power. The bias supply will kick in if the input voltage exceeds approximately 9 V.

Alternatively, a current limited DC supply can be used, but the user must be careful when operating in the MPPT mode. A resistor must be connected in series to create a nonlinear V versus I curve (Figure 14). For example, a voltage of 30 V and a series resistance of 15 Ω will provide the curve shown in Figure 15.

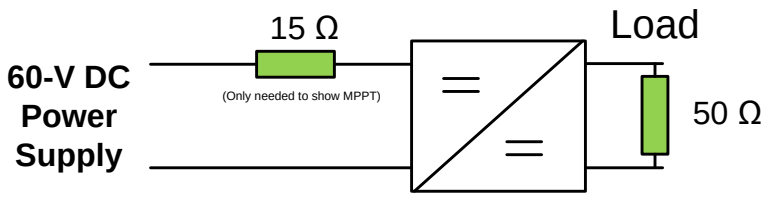


Figure 14. PV Emulator Using a DC Power Supply to Test MPPT Mode

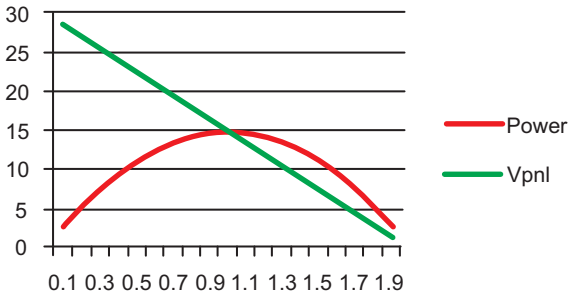


Figure 15. PV Curve Using DC Power Supply, 30 V, and 15-Ω Resistance

(c) Reverse power flow
For power flowed in the opposite direction, the connections are shown in Figure 16. An external 9-V supply is needed to connect to the bias power supply input for this mode.

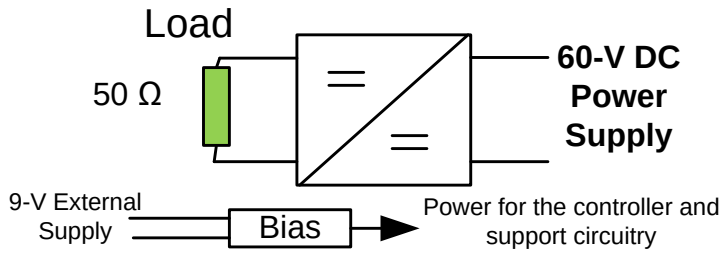


Figure 16. Setup to Test Reverse Power Flow

5.2 Software Setup

5.2.1 Installing Code Composer and controlSUITE™

1. If not already installed, install Code Composer Studio v6 from http://processors.wiki.ti.com/index.php/Category:Code_Composer_Studio_v6.
2. Go to http://www.ti.com/controlsuite and run the controlSUITE installer. Select to install the “Bidirectional Buck-Boost TI Design”.

5.2.2 Setup Code Composer Studio to Work With Kit

1. Open Code Composer Studio v6.
2. Once Code Composer Studio opens, the workspace launcher may appear asking to select a workspace location.
   (a) Click the *Browse…* button.
   (b) Create the following path by making new folders as necessary:
   - C:\MyWorkspaces\workspace1

   (c) Uncheck the box that says "Use this as the default and do not ask again".
   (d) Click *OK*.

NOTE: A workspace is a location on the hard drive where all the user settings for the IDE—which projects are open, what configuration is selected, and so on—are saved. This location can be anywhere on the disk; the location mentioned here is just for reference. Also, note that if this is not your first time running Code Composer, this dialog may not appear.

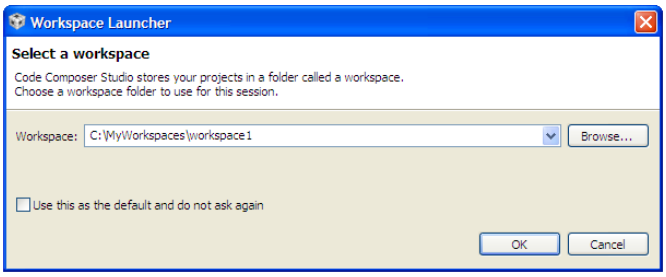


Figure 17. Open New Workspace

3. Add the project into your current workspace by clicking *Project → Import Existing CCS/CCE Eclipse Project*.

   (a) Select the root directory:
   "\controlSUITE\development_kits\TIDM_BUCKBOOST_BIDIR\version\BuckBoostBiDir_F2803x\".

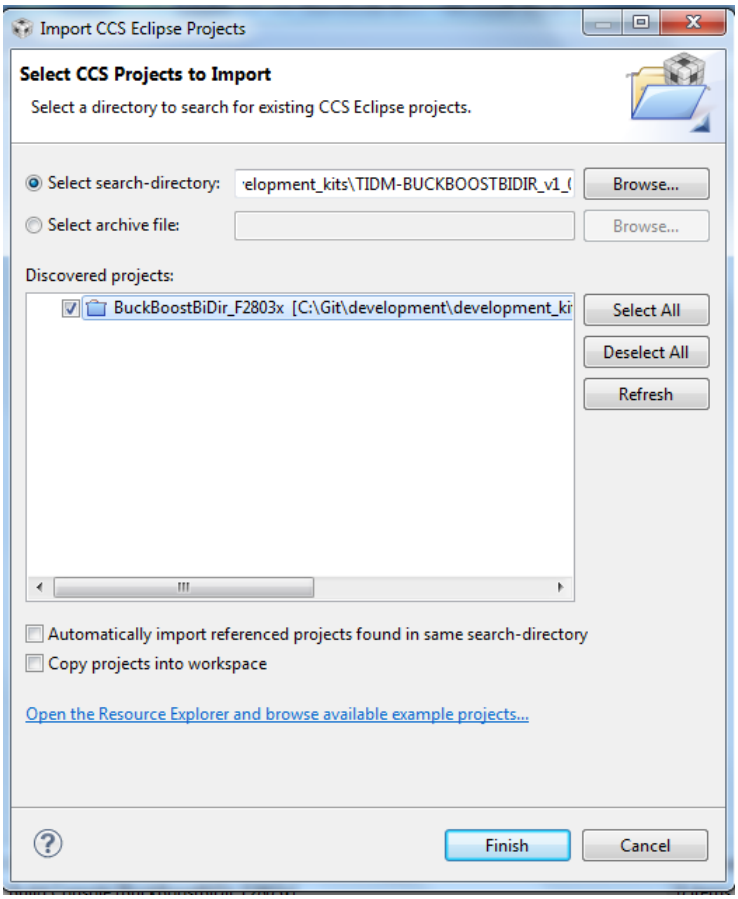


Figure 18. Adding F28035 Project to Workspace

   (b) Click *Finish*. This would copy all the projects relevant for the kit into the workspace. If you want only a particular project to be copied, uncheck the box next to the other project names.

5.2.3 Configuring a Project

1. Expand the file structure of the project you would like to run from the *C/C++ Projects* tab. Right-click on this project's name and select *Set as Active Project*, if this is not already the case.
2. Figure 19 shows the project in the CCSv6 *C/C++ Project* tab, which lists all the key files used in the project.

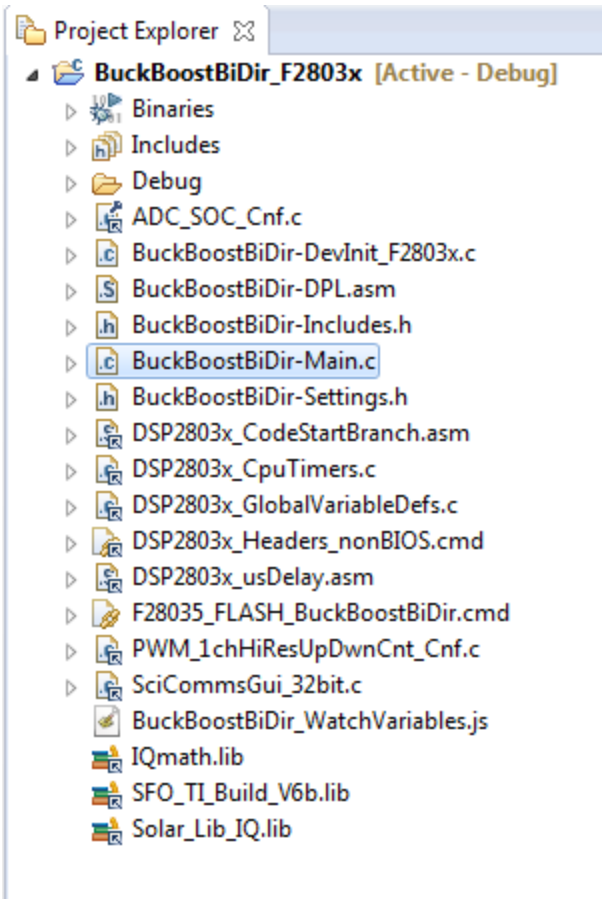


Figure 19. Project Explorer Tab

5.3 Procedure

1. Open and inspect BuckBoostBiDir-DevInit_F2803x.c by double clicking on the file name in the project window. Note that the system clock, peripheral clock prescale, and peripheral clock enables have been setup. Next, notice how the shared GPIO pins have been configured.
2. Open and inspect BuckBoostBiDir-Main.c. Notice the call made to DeviceInit() function and other variable initialization.
3. Locate and inspect the code in the main file under initialization code. Observe functions used for EPWM module initialization (PWM_1ch_UpDwnCnt_CNF, PWM_1ch_UpDwnCntCompl_CNF) and ADC module initialization (ADC_SOC_CNF) blocks. This is common for all incremental builds. Note the digital power library functions do not configure the deadband module, so this is done in the main file.
4. Also locate and inspect the following code in the main file under initialization code (see Figure 20). This location is where the ADC channels for different feedback signals are assigned and the start-of-conversion triggers are programmed.

```
#define Ipv_FB              AdcResult.ADCRESULT1
#define Vpv_FB              AdcResult.ADCRESULT2
#define Vout_FB             AdcResult.ADCRESULT4
#define Iout_FB             AdcResult.ADCRESULT5


//Map channel to ADC Pin
//The dummy reads are to account for first sample issue in Rev 0 silicon. Please refer to the Errata
and the datasheet. This would be fixed in later versions of the silicon
 ChSel[0] = 2;    // A2 - Ipv-FB Dummy,
 ChSel[1] = 2;    // A2 - Ipv-FB,
 ChSel[2] = 10;   // B2 - Vpv-FB,
 ChSel[3] = 12;   // B4 - Vout_FB,Dummy,
 ChSel[4] = 12;   // B4 - Vout_FB,
 ChSel[5] = 4;     // A4 - Iout_FB,

// Select Trigger Event
TrigSel[0] = ADCTRIG_EPWM1_SOCA;
TrigSel[1] = ADCTRIG_EPWM1_SOCA;
TrigSel[2] = ADCTRIG_EPWM1_SOCA;
TrigSel[3] = ADCTRIG_EPWM2_SOCA;
TrigSel[4] = ADCTRIG_EPWM2_SOCA;
TrigSel[5] = ADCTRIG_EPWM2_SOCA;

ADC_SOC_CNF(ChSel,TrigSel,ACQPS,-1,3);  //ADC auto clear mode, no interrupts
```

Figure 20. Code

5.3.1 Build and Load the Project

1. Hit the debug button to load the project into the controller. The perspective will now change to *Debug*. Once program load is complete, the debugger will halt the code at the beginning of the Main() routine.
2. Click *View* → *Scripting Console* to open the scripting console and open the "WatchVariables.js" script located inside the project folder. This script will populate the watch window with the appropriate variables needed to debug the system and the appropriate Q formats. Click the *Continuous Refresh* button on the watch window to enable continuous update of values from the controller.

5.3.2 Using Real-Time Emulation

Real-time emulation is a special emulation feature that allows the windows within Code Composer Studio to be updated at a rate up to 10 Hz while the MCU is running. This emulation not only 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. This emulation is very useful when tuning control law parameters on-the-fly, for example.

1. Enable real-time mode by hovering your mouse on the buttons on the horizontal toolbar and clicking 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'. The DGBM is the debug enable mask bit. When the DGBM bit is set to '0', memory and register values can be passed to the host processor for updating the debugger windows.
3. Click the *Continuous Refresh* button for the watch view.
4. Run the project.

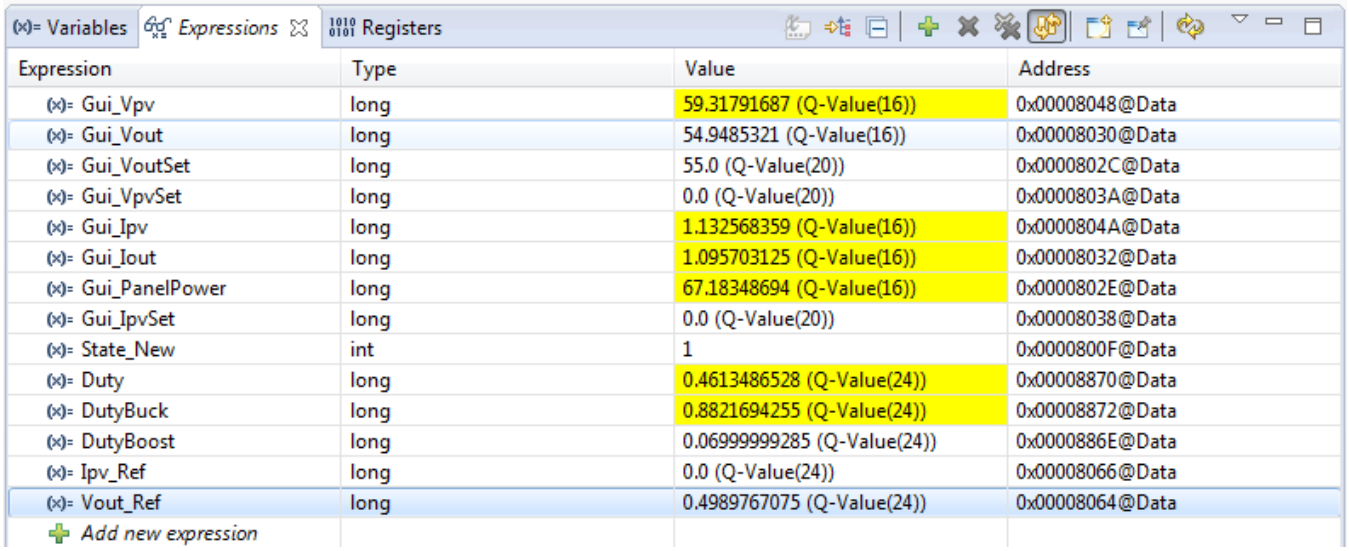
5.3.3 Check Output Voltage Control

1. In the watch view, check the value of Gui_Vpv, which will be the voltage at the input of the board. The board will show some power drawn because of the bias power supply and offsets on the ADC signals. The output voltage of the stage will be zero. By default, the mode of operation is in the output voltage control (State_New = 1).
2. Set a new voltage by entering a value in Gui_Vout (for example, 9 V). The voltage will be regulated at the set point voltage; because input voltage is at approximately 60 V, slowly increasing the set point changes the mode of operation of the converter from buck to br-A to br-B and boost. This change can be observed through the watch window.

(x)= Variables | Expressions | Registers

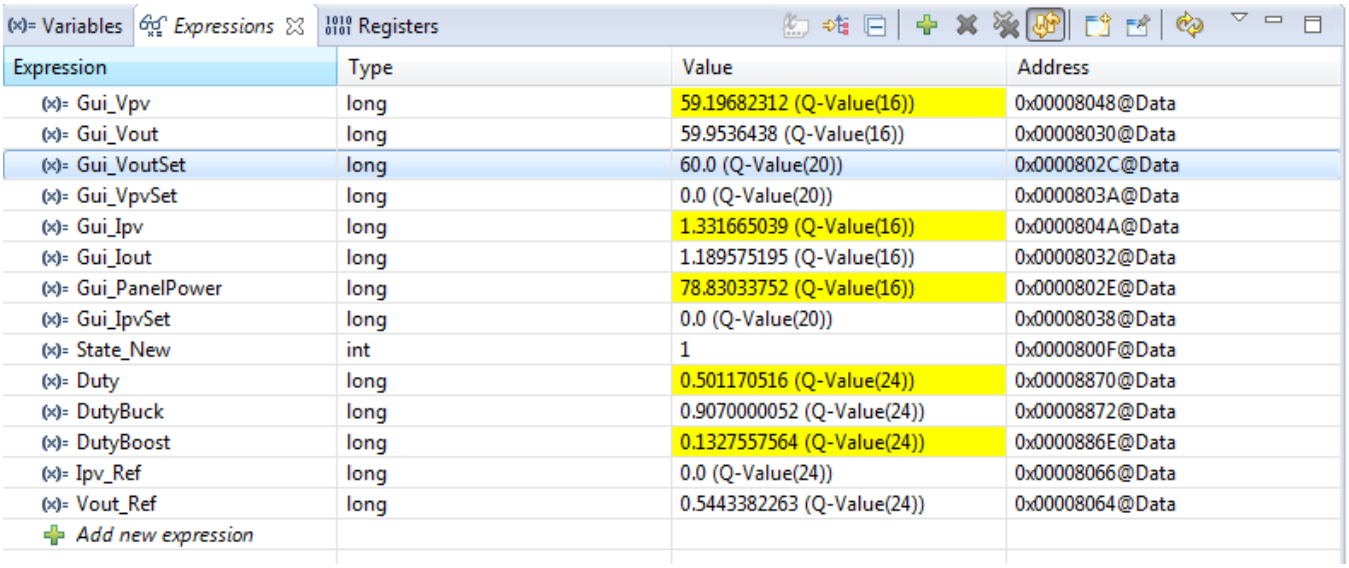
Expression	Type	Value	Address
(x)= Gui_Vpv	long	59.69631958 (Q-Value(16))	0x00008048@Data
(x)= Gui_Vout	long	39.948349 (Q-Value(16))	0x00008030@Data
(x)= Gui_VoutSet	long	40.0 (Q-Value(20))	0x0000802C@Data
(x)= Gui_VpvSet	long	0.0 (Q-Value(20))	0x0000803A@Data
(x)= Gui_Ipv	long	0.5944824219 (Q-Value(16))	0x0000804A@Data
(x)= Gui_Iout	long	0.7795410156 (Q-Value(16))	0x00008032@Data
(x)= Gui_PanelPower	long	35.49040222 (Q-Value(16))	0x0000802E@Data
(x)= Gui_IpvSet	long	0.0 (Q-Value(20))	0x00008038@Data
(x)= State_New	int	1	0x0000800F@Data
(x)= Duty	long	0.3377024531 (Q-Value(24))	0x00008870@Data
(x)= DutyBuck	long	0.674767375 (Q-Value(24))	0x00008872@Data
(x)= DutyBoost	long	0.0 (Q-Value(24))	0x0000886E@Data
(x)= Ipv_Ref	long	0.0 (Q-Value(24))	0x00008066@Data
(x)= Vout_Ref	long	0.3628921509 (Q-Value(24))	0x00008064@Data

Figure 21. Buck Mode



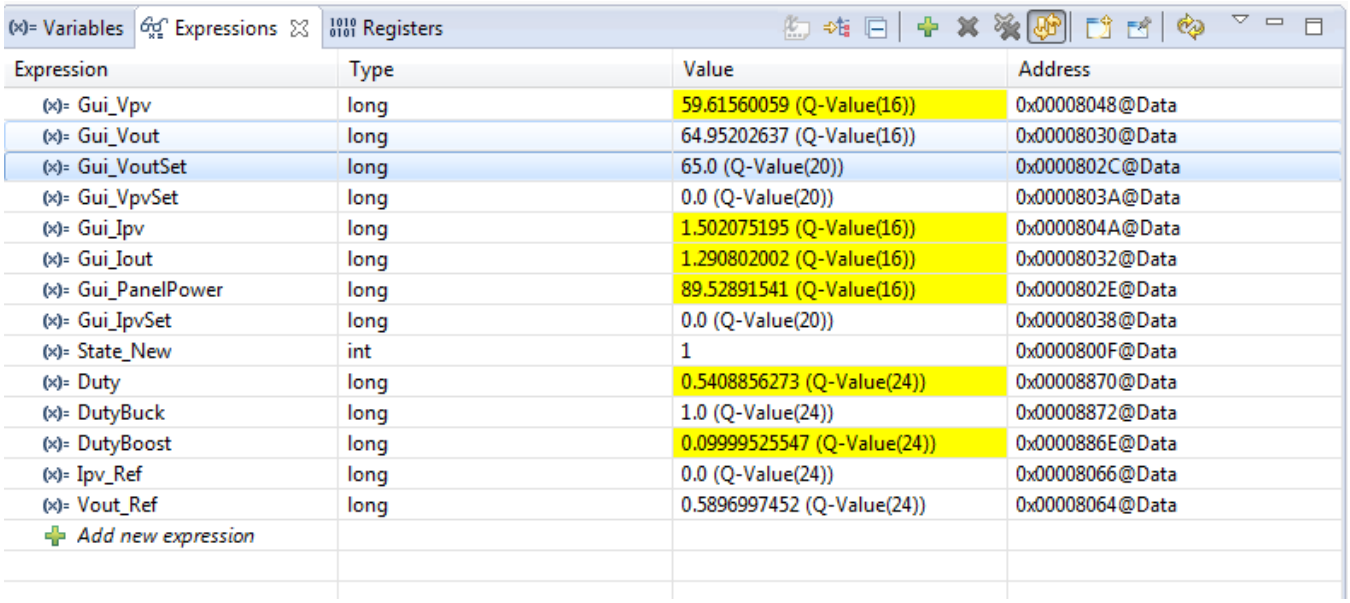
Expression	Type	Value	Address
Gui_Vpv	long	59.31791687 (Q-Value(16))	0x00008048@Data
Gui_Vout	long	54.9485321 (Q-Value(16))	0x00008030@Data
Gui_VoutSet	long	55.0 (Q-Value(20))	0x0000802C@Data
Gui_VpvSet	long	0.0 (Q-Value(20))	0x0000803A@Data
Gui_Ipv	long	1.132568359 (Q-Value(16))	0x0000804A@Data
Gui_Iout	long	1.095703125 (Q-Value(16))	0x00008032@Data
Gui_PanelPower	long	67.18348694 (Q-Value(16))	0x0000802E@Data
Gui_IpvSet	long	0.0 (Q-Value(20))	0x00008038@Data
State_New	int	1	0x0000800F@Data
Duty	long	0.4613486528 (Q-Value(24))	0x00008870@Data
DutyBuck	long	0.8821694255 (Q-Value(24))	0x00008872@Data
DutyBoost	long	0.06999999285 (Q-Value(24))	0x0000886E@Data
Ipv_Ref	long	0.0 (Q-Value(24))	0x00008066@Data
Vout_Ref	long	0.4989767075 (Q-Value(24))	0x00008064@Data
Add new expression			

Figure 22. Constant Boost + Buck Mode



Expression	Type	Value	Address
Gui_Vpv	long	59.19682312 (Q-Value(16))	0x00008048@Data
Gui_Vout	long	59.9536438 (Q-Value(16))	0x00008030@Data
Gui_VoutSet	long	60.0 (Q-Value(20))	0x0000802C@Data
Gui_VpvSet	long	0.0 (Q-Value(20))	0x0000803A@Data
Gui_Ipv	long	1.331665039 (Q-Value(16))	0x0000804A@Data
Gui_Iout	long	1.189575195 (Q-Value(16))	0x00008032@Data
Gui_PanelPower	long	78.83033752 (Q-Value(16))	0x0000802E@Data
Gui_IpvSet	long	0.0 (Q-Value(20))	0x00008038@Data
State_New	int	1	0x0000800F@Data
Duty	long	0.501170516 (Q-Value(24))	0x00008870@Data
DutyBuck	long	0.9070000052 (Q-Value(24))	0x00008872@Data
DutyBoost	long	0.1327557564 (Q-Value(24))	0x0000886E@Data
Ipv_Ref	long	0.0 (Q-Value(24))	0x00008066@Data
Vout_Ref	long	0.5443382263 (Q-Value(24))	0x00008064@Data
Add new expression			

Figure 23. Constant Buck + Boost Mode



Expression	Type	Value	Address
Gui_Vpv	long	59.61560059 (Q-Value(16))	0x00008048@Data
Gui_Vout	long	64.95202637 (Q-Value(16))	0x00008030@Data
Gui_VoutSet	long	65.0 (Q-Value(20))	0x0000802C@Data
Gui_VpvSet	long	0.0 (Q-Value(20))	0x0000803A@Data
Gui_Ipv	long	1.502075195 (Q-Value(16))	0x0000804A@Data
Gui_Iout	long	1.290802002 (Q-Value(16))	0x00008032@Data
Gui_PanelPower	long	89.52891541 (Q-Value(16))	0x0000802E@Data
Gui_IpvSet	long	0.0 (Q-Value(20))	0x00008038@Data
State_New	int	1	0x0000800F@Data
Duty	long	0.5408856273 (Q-Value(24))	0x00008870@Data
DutyBuck	long	1.0 (Q-Value(24))	0x00008872@Data
DutyBoost	long	0.09999525547 (Q-Value(24))	0x0000886E@Data
Ipv_Ref	long	0.0 (Q-Value(24))	0x00008066@Data
Vout_Ref	long	0.5896997452 (Q-Value(24))	0x00008064@Data
Add new expression			

Figure 24. Boost Mode

3. As a new voltage is set, you will see the Gui_Vout rise to the set point. You can vary the output voltage set point and see the buck, boost, or bridge mode operation. Now set Gui_Vout equal to zero.
4. The software is configured to connect to the SFRA GUI. Open the SFRA gui.exe, located at controlSUITE/libs/app_libs/SFRA/version/GUI.
5. Select fixed-point math.
6. Click *Setup Connection* and set the baud rate to be 57600 on the pop-up window.
7. Uncheck *boot on connect* and select the appropriate COM port. If this is the first time using the control card, this is the higher number of the COM port displayed in this list. To ensure everything is correct, turn the power off on the board, physically disconnect the USB cable, and connect back and observe which COM port appears on the list.
8. Click *OK* to close the pop-up window and return to the main screen.
9. On the main window, click *Connect*. Once connected, the GUI will parse the current settings for the FRA sweep from the controller. These settings include the start frequency of the sweep, the length of the frequency sweep array (this is fixed in the code and therefore cannot be changed through the GUI), injection amplitude, and steps per decade. Leave these settings as default for now.
10. Click the *Start Sweep* button.
11. Wait for the status bar in the GUI to change to *Sweep Complete*.
12. The results of the SFRA sweep are now displayed on the window. You can check the SFRA results at different operating points of the board.

NOTE: The code uses a fixed set of coefficients. A bank of coefficients can maintain better margin across the load and operating range.

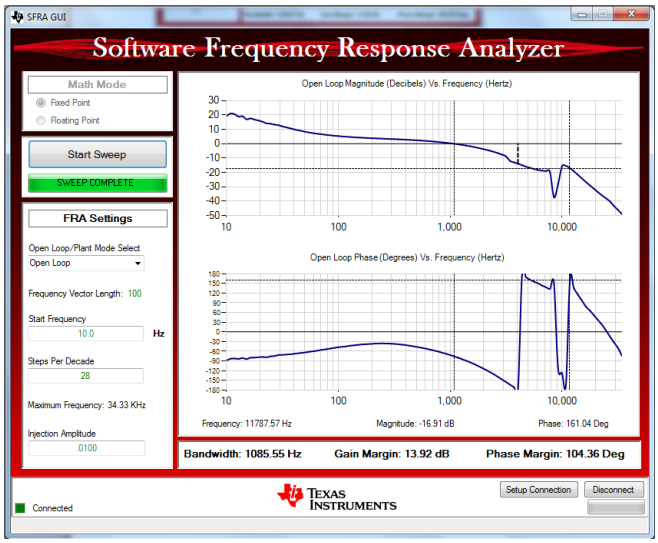


Figure 25. Vin = 60 V, Output Load = 50 Ω, Vout = 40 V

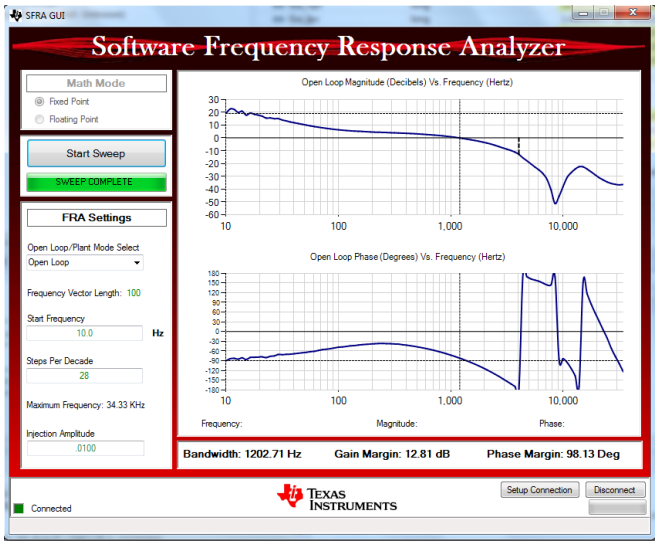


Figure 26. Vin = 60 V, Output Load = 50 Ω, Vout = 60 V

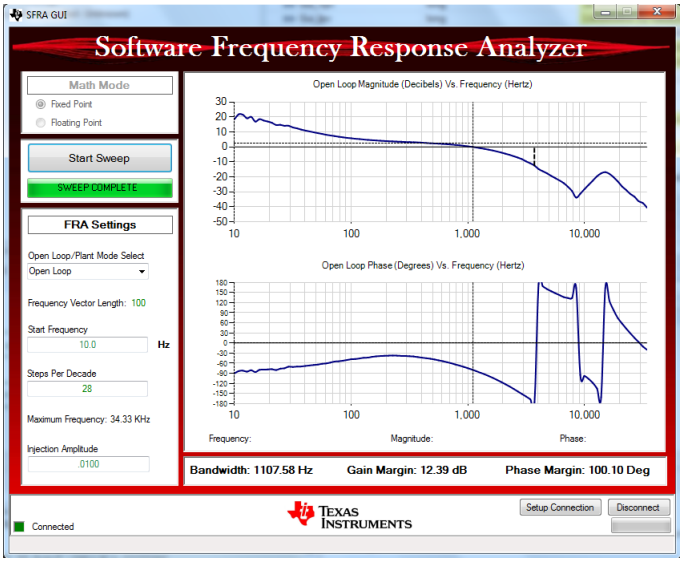


Figure 27. Vin = 60 V, Output Load = 50 Ω, Vout = 65 V

5.3.4 Check MPPT Mode

1. To check MPPT mode, make sure the input is connected to a panel, a panel emulator, or to a power supply with a series connected resistor to emulate the PV characteristics. To enable MPPT mode, just set State_new = 2, and make sure that if the power supply is used for the input stage, then it is limited current. The supply will reach this max current limit as soon as MPPT state is enabled. In case of PV the panel the MPP will be reached.
2. To end the experiment, write a 0 for Gui_Vout, now set State_new = 1. Wait until the Gui_Vout = 0, then 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 *[image]*. Finally, reset the MCU.

5.3.5 Voltage Control in the Reverse Direction

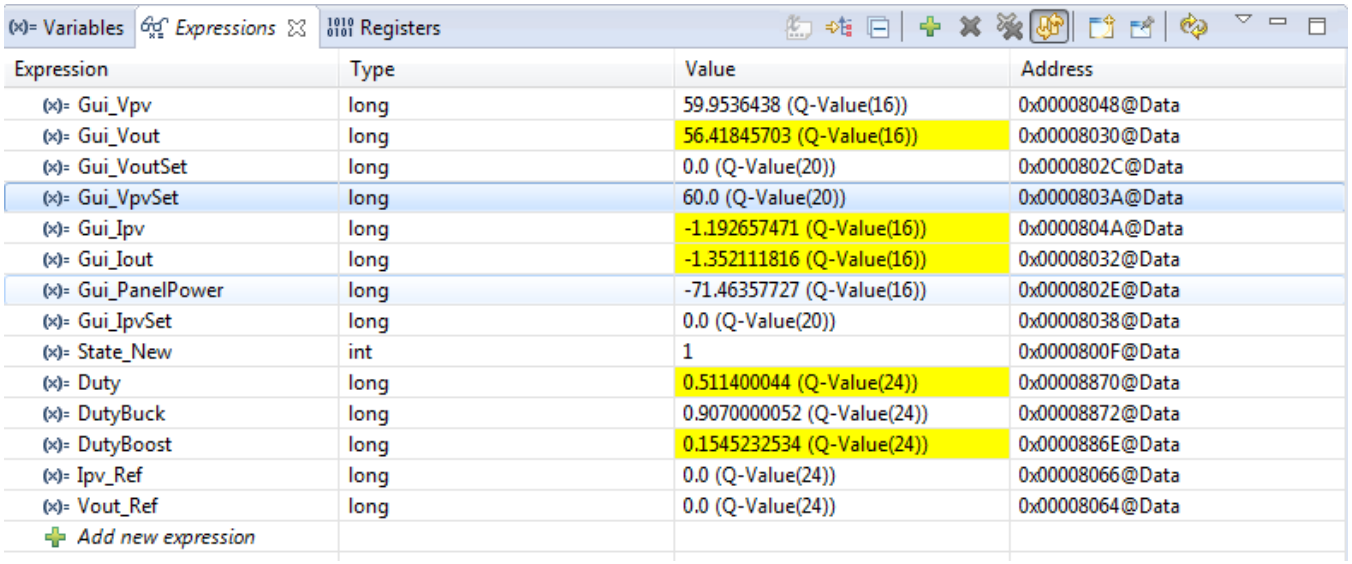
1. Note the bias supply is only from one direction (the PV side); therefore, to see the reverse direction of power flow, disconnect the bias from the PV side by depopulating the jumper LK201. Connect a controller power source of 9-V DC between the PWR pin and the bias ground to externally supply the controller power.

   Also the software uses V_{pv} as the term for the input terminal in the forward direction; however, in reality the V_{pv} should never sink in current. This process is just used for software compatibility of the board designed for solar application.
2. To begin a reverse power test, disconnect everything from the board and make sure there is no energized source.
3. Remove the LK201 jumper and connect a power source of 10-V DC between PWM pin of this jumper and the control ground test point. This source provides the controller power, signified by the light from the controller power LED.
4. Connect a resistive load of 30 Ω on the panel input side, and connect a controller DC source on the output side.
5. Go to the (ProjectName}-settings.h file and make the changes for DIRECTION==2. Also, make sure the OPEN_LOOP==0.
6. Rebuild the code and load the code.
7. Once the code is loaded, enable real-time mode and run the code.
8. Enter Gui_VpvSet in _IQ20 format in the watch window, which should be 0.0 initially.
9. Slowly raise the input voltage (connected at the output terminal), and see if the Gui_Vout matches this value.
10. Set the Gui_VpvSet to _IQ20(10.0), and see the Gui_Vpv value regulate at that voltage. As a 60-V supply is connected to the terminal, varying the Gui_VpvSet point will take the power supply across Buck, Br-A, Br-B, and Boost modes. This process can be seen through the watch window. Also note the negative value of current and power illustrating reverse power flow.

(x)= Variables | Expressions | Registers

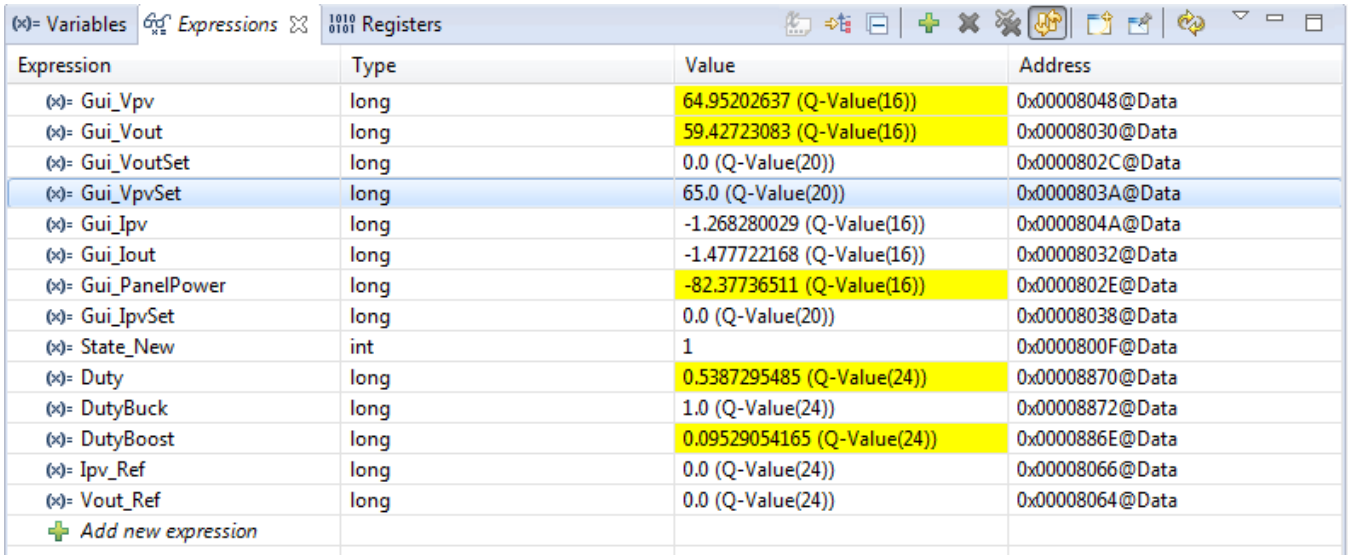
Expression	Type	Value	Address
(x)= Gui_Vpv	long	49.95352173 (Q-Value(16))	0x00008048@Data
(x)= Gui_Vout	long	57.08277893 (Q-Value(16))	0x00008030@Data
(x)= Gui_VoutSet	long	0.0 (Q-Value(20))	0x0000802C@Data
(x)= Gui_VpvSet	long	50.0 (Q-Value(20))	0x0000803A@Data
(x)= Gui_Ipv	long	-1.000305176 (Q-Value(16))	0x0000804A@Data
(x)= Gui_Iout	long	-0.9158325195 (Q-Value(16))	0x00008032@Data
(x)= Gui_PanelPower	long	-49.96542358 (Q-Value(16))	0x0000802E@Data
(x)= Gui_IpvSet	long	0.0 (Q-Value(20))	0x00008038@Data
(x)= State_New	int	1	0x0000800F@Data
(x)= Duty	long	0.4377933741 (Q-Value(24))	0x00008870@Data
(x)= DutyBuck	long	0.8742867708 (Q-Value(24))	0x00008872@Data
(x)= DutyBoost	long	0.0 (Q-Value(24))	0x0000886E@Data
(x)= Ipv_Ref	long	0.0 (Q-Value(24))	0x00008066@Data
(x)= Vout_Ref	long	0.0 (Q-Value(24))	0x00008064@Data
Add new expression			

Figure 28. Buck



Expression	Type	Value	Address
Gui_Vpv	long	59.9536438 (Q-Value(16))	0x00008048@Data
Gui_Vout	long	56.41845703 (Q-Value(16))	0x00008030@Data
Gui_VoutSet	long	0.0 (Q-Value(20))	0x0000802C@Data
Gui_VpvSet	long	60.0 (Q-Value(20))	0x0000803A@Data
Gui_Ipv	long	-1.192657471 (Q-Value(16))	0x0000804A@Data
Gui_Iout	long	-1.352111816 (Q-Value(16))	0x00008032@Data
Gui_PanelPower	long	-71.46357727 (Q-Value(16))	0x0000802E@Data
Gui_IpvSet	long	0.0 (Q-Value(20))	0x00008038@Data
State_New	int	1	0x0000800F@Data
Duty	long	0.511400044 (Q-Value(24))	0x00008870@Data
DutyBuck	long	0.9070000052 (Q-Value(24))	0x00008872@Data
DutyBoost	long	0.1545232534 (Q-Value(24))	0x0000886E@Data
Ipv_Ref	long	0.0 (Q-Value(24))	0x00008066@Data
Vout_Ref	long	0.0 (Q-Value(24))	0x00008064@Data
Add new expression			

Figure 29. Constant Buck



Expression	Type	Value	Address
Gui_Vpv	long	64.95202637 (Q-Value(16))	0x00008048@Data
Gui_Vout	long	59.42723083 (Q-Value(16))	0x00008030@Data
Gui_VoutSet	long	0.0 (Q-Value(20))	0x0000802C@Data
Gui_VpvSet	long	65.0 (Q-Value(20))	0x0000803A@Data
Gui_Ipv	long	-1.268280029 (Q-Value(16))	0x0000804A@Data
Gui_Iout	long	-1.477722168 (Q-Value(16))	0x00008032@Data
Gui_PanelPower	long	-82.37736511 (Q-Value(16))	0x0000802E@Data
Gui_IpvSet	long	0.0 (Q-Value(20))	0x00008038@Data
State_New	int	1	0x0000800F@Data
Duty	long	0.5387295485 (Q-Value(24))	0x00008870@Data
DutyBuck	long	1.0 (Q-Value(24))	0x00008872@Data
DutyBoost	long	0.09529054165 (Q-Value(24))	0x0000886E@Data
Ipv_Ref	long	0.0 (Q-Value(24))	0x00008066@Data
Vout_Ref	long	0.0 (Q-Value(24))	0x00008064@Data
Add new expression			

Figure 30. Boost

This completes the reverse direction demonstration.

11. To end the experiment, write a '0' to Gui_VpvSet. Wait until the Gui_VpvSet equals zero, and make the input voltage zero. Now, halt the processor by using the *Halt* button on the toolbar or by going to *Target → Halt*. Then take the MCU out of real-time mode by clicking *[image]*. Finally, reset the MCU.
12. Finally, remove the controller power as well if all experiments are finished.

6 Design Files

6.1 Schematics

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

6.2 Bill of Materials

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

6.3 Layer Plots

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

6.4 Gerber Files

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

6.5 Software Files

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

7 References

1. *High-Efficiency, Wide-Load Range Buck/Boost Photovoltaic Microconverter*, Dave Freeman, Dick Hester, APEC 2010

8 About the Author

MANISH BHARDWAJ is a Systems Application Engineer at Texas Instruments, responsible for developing system solutions for C2000™ microcontrollers. Before joining TI in 2009, Manish received his Masters of Science in Electrical and Computer Engineering from Georgia Institute of Technology in Atlanta, and Bachelor of Engineering from Netaji Subhash Institute of Technology University of Delhi, India. He can be reached at mbhardwaj@ti.com.

IMPORTANT NOTICE FOR TI REFERENCE DESIGNS

Texas Instruments Incorporated ("TI") reference designs are solely intended to assist designers (“Buyers”) who are developing systems that incorporate TI semiconductor products (also referred to herein as “components”). Buyer understands and agrees that Buyer remains responsible for using its independent analysis, evaluation and judgment in designing Buyer’s systems and products.

TI reference designs have been created using standard laboratory conditions and engineering practices. **TI has not conducted any testing other than that specifically described in the published documentation for a particular reference design.** TI may make corrections, enhancements, improvements and other changes to its reference designs.

Buyers are authorized to use TI reference designs with the TI component(s) identified in each particular reference design and to modify the reference design in the development of their end products. HOWEVER, NO OTHER LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE TO ANY OTHER TI INTELLECTUAL PROPERTY RIGHT, AND NO LICENSE TO ANY THIRD PARTY TECHNOLOGY OR INTELLECTUAL PROPERTY RIGHT, IS GRANTED HEREIN, including but not limited to any patent right, copyright, mask work right, or other intellectual property right relating to any combination, machine, or process in which TI components or services are used. Information published by TI regarding third-party products or services does not constitute a license to use such products or services, or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

TI REFERENCE DESIGNS ARE PROVIDED "AS IS". TI MAKES NO WARRANTIES OR REPRESENTATIONS WITH REGARD TO THE REFERENCE DESIGNS OR USE OF THE REFERENCE DESIGNS, EXPRESS, IMPLIED OR STATUTORY, INCLUDING ACCURACY OR COMPLETENESS. TI DISCLAIMS ANY WARRANTY OF TITLE AND ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, QUIET ENJOYMENT, QUIET POSSESSION, AND NON-INFRINGEMENT OF ANY THIRD PARTY INTELLECTUAL PROPERTY RIGHTS WITH REGARD TO TI REFERENCE DESIGNS OR USE THEREOF. TI SHALL NOT BE LIABLE FOR AND SHALL NOT DEFEND OR INDEMNIFY BUYERS AGAINST ANY THIRD PARTY INFRINGEMENT CLAIM THAT RELATES TO OR IS BASED ON A COMBINATION OF COMPONENTS PROVIDED IN A TI REFERENCE DESIGN. IN NO EVENT SHALL TI BE LIABLE FOR ANY ACTUAL, SPECIAL, INCIDENTAL, CONSEQUENTIAL OR INDIRECT DAMAGES, HOWEVER CAUSED, ON ANY THEORY OF LIABILITY AND WHETHER OR NOT TI HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES, ARISING IN ANY WAY OUT OF TI REFERENCE DESIGNS OR BUYER’S USE OF TI REFERENCE DESIGNS.

TI reserves the right to make corrections, enhancements, improvements and other changes to its semiconductor products and services per JESD46, latest issue, and to discontinue any product or service per JESD48, latest issue. Buyers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. All semiconductor products are sold subject to TI’s terms and conditions of sale supplied at the time of order acknowledgment.

TI warrants performance of its components to the specifications applicable at the time of sale, in accordance with the warranty in TI’s terms and conditions of sale of semiconductor products. Testing and other quality control techniques for TI components are used to the extent TI deems necessary to support this warranty. Except where mandated by applicable law, testing of all parameters of each component is not necessarily performed.

TI assumes no liability for applications assistance or the design of Buyers’ products. Buyers are responsible for their products and applications using TI components. To minimize the risks associated with Buyers’ products and applications, Buyers should provide adequate design and operating safeguards.

Reproduction of significant portions of TI information in TI data books, data sheets or reference designs is permissible only if reproduction is without alteration and is accompanied by all associated warranties, conditions, limitations, and notices. TI is not responsible or liable for such altered documentation. Information of third parties may be subject to additional restrictions.

Buyer acknowledges and agrees that it is solely responsible for compliance with all legal, regulatory and safety-related requirements concerning its products, and any use of TI components in its applications, notwithstanding any applications-related information or support that may be provided by TI. Buyer represents and agrees that it has all the necessary expertise to create and implement safeguards that anticipate dangerous failures, monitor failures and their consequences, lessen the likelihood of dangerous failures and take appropriate remedial actions. Buyer will fully indemnify TI and its representatives against any damages arising out of the use of any TI components in Buyer’s safety-critical applications.

In some cases, TI components may be promoted specifically to facilitate safety-related applications. With such components, TI’s goal is to help enable customers to design and create their own end-product solutions that meet applicable functional safety standards and requirements. Nonetheless, such components are subject to these terms.

No TI components are authorized for use in FDA Class III (or similar life-critical medical equipment) unless authorized officers of the parties have executed an agreement specifically governing such use.

Only those TI components that TI has specifically designated as military grade or “enhanced plastic” are designed and intended for use in military/aerospace applications or environments. Buyer acknowledges and agrees that any military or aerospace use of TI components that have ***not*** been so designated is solely at Buyer's risk, and Buyer is solely responsible for compliance with all legal and regulatory requirements in connection with such use.

TI has specifically designated certain components as meeting ISO/TS16949 requirements, mainly for automotive use. In any case of use of non-designated products, TI will not be responsible for any failure to meet ISO/TS16949.

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