Software User Guide: TIDM-SOLAR-DCDC

500W, High Voltage Maximum Power Point Tracking (MPPT) Solar DC/DC Reference Design

TEXAS INSTRUMENTS

Description

This reference design document presents the implementation details of a digitally controlled DC-DC converter that is used as a front-end converter for solar inverter (DC-AC) application. It implements an isolated DC-DC stage with maximum power point tracking (MPPT) algorithm in order to utilize the full capacity of a 500W solar panel. It maintains its input voltage at the reference set point generated by the MPPT algorithm and delivers power to a downstream DC-AC inverter when connected across its output. The DC-AC inverter transfers the power from the DC-DC stage to an emulated grid connected across its own output. A TMS320F280049 control card containing a TMS320F280049 micro-controller (MCU) and a 500W isolated DC-DC stage EVM are used to implement the complete DC-DC system.

Resources

TIDM-SOLAR-DCDC	Design Folder
TMS320F280049C	Product Folder
C2000WARE-DIGITALPOWER-SDK	Tool Folder
UCC27324	Product Folder
TMDSCNCD280049C	Tool Folder
ISO7240, ISO7242	Product Folder

Ask our TI E2E™ support experts

Features

- MPPT DC/DC converter with rated panel (string) voltage of 200V, max Power of 500W at 400V DC Output.
- MPPT control algorithm using TMS320F280049 MCU.
- 100kHz PWM frequency for the DC/DC boost stage. 50kHz sampling frequency for current and voltage loop control of the DC/DC boost stage. Isolated DC output using open loop LLC DC/DC converter running at 120kHz PWM and resonant frequency.
- Over-voltage protection for both input and output voltages.
- TMS320F280049 MCU implements full digital control with internal comparator sub-system (CMPSS) based fast over-current protection.

Applications

- String inverter
- Central inverter

DC Link

1 System Description

Photovoltaic (PV) systems based on solar energy offer an environmentally friendly source of electricity. A key feature of such PV system is the efficiency of conversion at which the power converter stage can extract the energy from the PV arrays and deliver to the load. The maximum power point tracking (MPPT) of the PV output for all sunshine conditions allows reduction of the cost of installation and maximizes the power output from the PV panel. Therefore, a DC-DC converter employing some MPPT algorithm is generally used as a front-end converter to efficiently extract the PV output power and convert the PV output voltage to a high voltage DC bus. The DC-DC converter, depending on the system requirement, can use either an isolated power stage or a non-isolated stage. The high voltage bus from the DC-DC converter is then fed to power the DC-AC inverter that eventually supplies the load and connects to the grid. This C2000 MCU based MPPT solar DC-DC reference design uses an isolated DC-DC stage as is shown in Fig 1. It consists of two DC-DC stages. These are, (1) a 2-ph interleaved boost converter and, (2) an isolated half bridge LLC resonant converter.

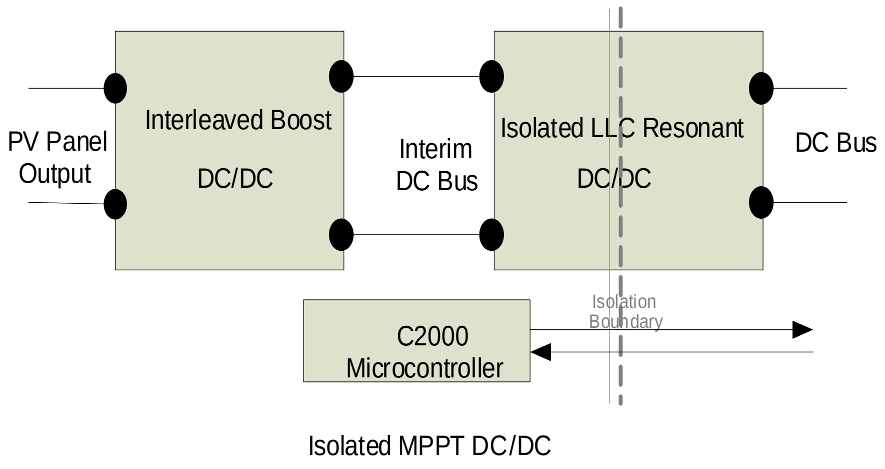


Figure 1. Isolated MPPT Solar DC-DC Converter Block Diagram

The DC-DC converter draws dc current from the PV panel such that the panel operates at its maximum power transfer point. This requires maintaining the panel output, i.e., the DC-DC converter input at a level determined by the MPPT algorithm. This is implemented in the 2-ph interleaved boost converter stage. The isolated LLC resonant converter simply provides high frequency isolation for the DC-DC stage. A C2000 microcontroller with its on-chip PWM, ADC and analog comparator modules is able to implement complete digital control of such MPPT DC-DC system.

1.1 Key System Specifications

Table 1-1. Key System Specifications

PARAMETER	SPECIFICATIONS
Output power	500W @400-VDC
PV Panel voltage	200-VDC
Output current	1.3A (max)
Nominal DC bus voltage	400-V DC
DC bus voltage range	350-V to 420-V DC
PWM switching frequency	100 kHz
Efficiency	96.5%

2 System Overview

2.1 Block Diagram

Fig 2.1 illustrates a C2000 based MPPT DC-DC converter control system. The PV panel output voltage, Vpv, is applied to the input of the 2-ph interleaved boost stage.

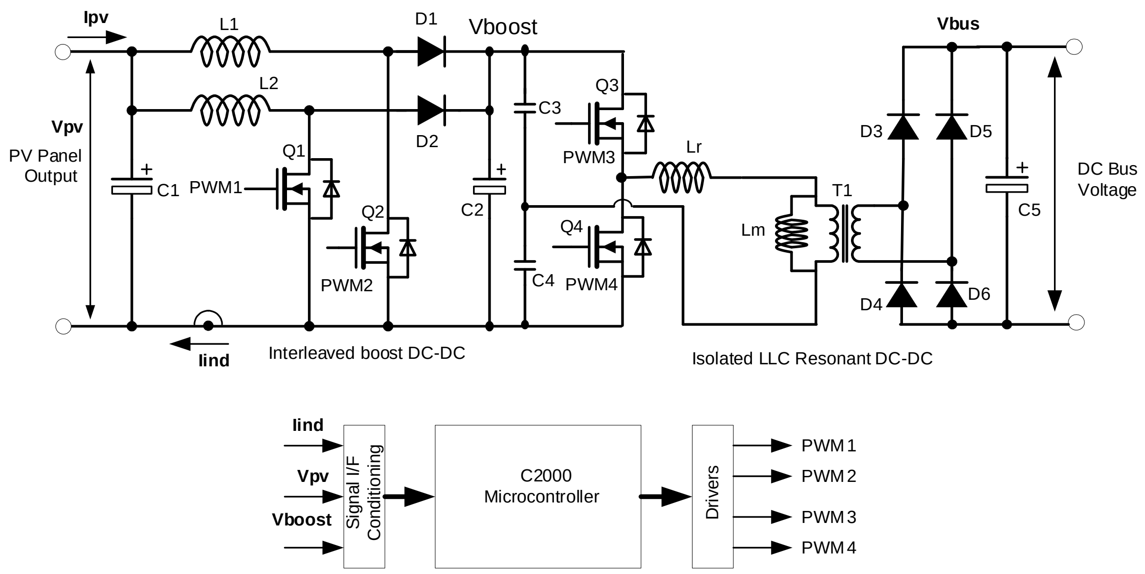


Figure 2.1 MPPT DC-DC Converter Control using C2000 Micro-controller

Inductor L1, MOSFET Q1, and diode D1 together form one of the boost stages while, L2, Q2, and D2 form the other. A capacitor C2 at the boost converter output acts as an energy reservoir and provides boost voltage to the resonant LLC stage. The H bridge LLC resonant stage consists of MOSFETs Q3~Q4, input capacitors C3~C4, resonant inductor Lr, resonant capacitor Cr, transformer T1, output rectifiers D3~D6 and output capacitor C5. This stage has a voltage ratio of 1 and provides the isolation between the primary and secondary side.
Figure 2.1 indicates all the interface signals needed for full control of this DC-DC converter using a C2000 micro-controller (MCU). The MCU controls the hardware using three feedback signals and four PWM outputs. The signals that are sensed and fed back to the MCU include the panel output voltage (Vpv) and the boost output voltage (Vboost) and the total boost inductor currents (I_{ind}). These sensed signals are used to implement the voltage and current control loops for the DC-DC boost stage. The interleaved boost DC/DC topology is chosen to boost the variable DC output to a fixed DC bus voltage. The main reason for using this topology is the wide input voltage variation. The PWM signals for the power switches Q1 and Q2 are phase-shifted by 180 degrees. This helps reduce the ripple in the PV panel current.
The LLC stage runs at open loop with its PWM frequency set to be the same as the resonant frequency. The C2000 MCU shares the common ground with the primary side of the LLC stage. There is an isolated feedback to the controller from the LLC secondary output terminals. But this signal is used only for over-voltage protection and not for closed loop control. Since the LLC is run under open loop it is necessary to maintain a voltage conversion factor of 1. This is achieved by making (1) the LLC PWM frequency the same as the resonant frequency and, (2) by maintaining a minimum load of about 10W across the LLC output.

Figure 2.2 shows the DC-DC interleaved boost converter control loops. This uses current mode control. However, the goal is to control the PV panel output (Vpv) which is the input to the DC-DC stage. This allows the PV panel (array) operates at its maximum power point at all time. Input current is regulated by adjusting the duty cycles of the power switches Q1 and Q2. Input voltage is regulated by adjusting the input current. A Maximum Power Point Tracking algorithm described in the next section is responsible for determining the set point (Vpv_ref) for the PV panel voltage. Notice that the input voltage control loop works quite differently compared to conventional feedback used in output voltage control. Under this control scheme, when the PV panel voltage (Vpv) tends to go higher than the reference panel voltage (Vpv_ref) set by the MPPT algorithm, the control loop increases the panel current command (reference current for inner current loop Iind_ref) and thereby controls the panel voltage at its reference level (Vpv_ref). When the panel voltage tends to go lower than the reference, the control loop reduces the panel current command in order to reestablish the panel voltage to its reference level.

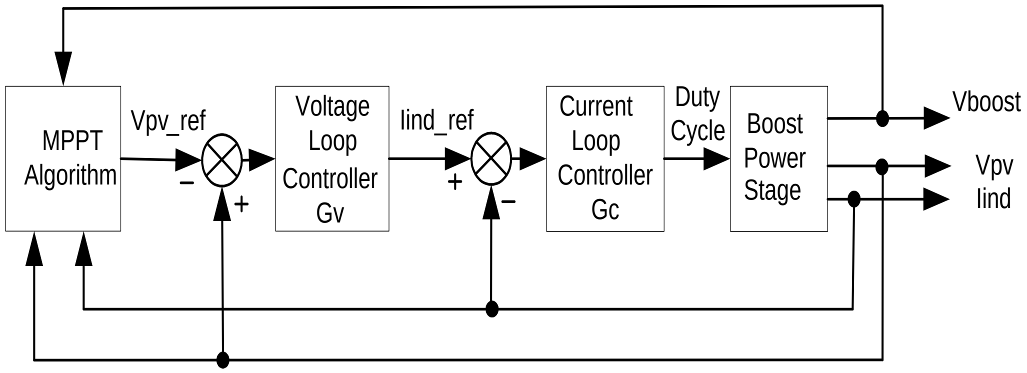


Figure 2.2 MPPT DC-DC Converter Control Loops

The panel voltage Vpv, sensed through one of the ADC channels, is compared against the reference voltage Vpv_ref set by the MPPT algorithm. The resulting error signal Ev is then input the voltage loop controller Gv which regulates the panel voltage at the reference level. The voltage controller Gv has the form of a PI compensator. The output of Gv is the reference current command for the inner inductor current loop. The average value of the inductor current is the panel current Ipv. Therefore, by controlling the average value of the inductor current the current controller Gc essentially controls the panel current. This reference current command Iind_ref for the current control loop is compared against the feedback inductor current Iind sensed through another ADC channel. The resulting current error signal is then input the current loop controller Gc which generates the boost converter PWM duty ratio command d for the boost switches Q1 & Q2.
In addition to implementing the voltage and current loop controllers, C2000 MCU also monitors the panel current for over-current protection (OCP). It also monitors the panel voltage and boost output voltage for over voltage protection (OVP). The ADC channel that monitors the panel current has an internal analog comparator with user programmable threshold. This threshold for the comparator is set by use of an internal 12-bit DAC. Whenever the current reaches the programmed upper limit corresponding to the user programmable comparator threshold, the comparator initiates a one shot turn OFF of all PWM signals. The over-voltage protection is implemented in code and it runs from B-task in the background loop at a user programmable frequency of 10Hz. Whenever the DC bus voltage or the panel voltage reaches the upper limit corresponding to the user programmable thresholds, the OVP code initiates a one-shot turn OFF of all PWM signals.
C2000 MCU also generates two PWM outputs to drive the isolated LLC stage. This stage is run in an open loop fashion with unity voltage conversion ratio (voltage gain). This means the boost voltage and the LLC output voltage is almost equal. However, this requires a small minimum load of about 10W across LLC stage output (16kohm at 400V). With no load connected across LLC output and the boost output voltage set to 400V, the LLC stage gain might be higher than 1, resulting in high voltage across LLC output. ***The user must prevent this condition by always maintaining a minimum load resistor of about 16kOhm across the LLC output.*** All the time critical functions related to the DC-DC control loops are implemented in a fast sampling loop enabled by the C2000 Micro-controller high speed CPU, interrupts, on chip 12-bit ADC module and high frequency PWM modules. A detailed description of the software algorithm is provided in the following chapters.

3 Hardware, Software, Testing Requirements, and Test Results

3.1 Required Hardware and Software

The hardware design for the TIDM-SOLAR-DCDC has been detailed in the associated hardware design files. Please refer to the h/w design files for details. This section details the hardware setup required for the experiments and testing of different software labs, Lab 1 through Lab 3 as outlined in this user guide.

3.1.1 Hardware

The hardware in this design is set up and operated in several pieces:

- One TIDM-SOLAR-DCDC mother board
- TMDSCNCD280049C Control Card:
  - S2 Switch : Change from the DOWN to the UP position
  - S3 Switch : Changed from the UP to the DOWN position
- Mini USB cable
- Laptop or Desktop PC with CCS installed

The test equipment required to power and evaluate the design is as follows:

- >12-V, 1-A bench style supply for low voltage board power
- >1kW, 0-400V DC voltage source
- >1kW, 400-V resistive load
- Solar Panel Emulator similar to Agilent E4360A with two E4362A DC modules (0-130V, 600W) connected in series to provide 200V input.

3.1.2 Software

This section details the CCS software project for this TIDM-SOLAR-DCDC and testing of different software labs, Lab 1 through Lab 3:

3.1.2.1 Getting Started with Firmware

The software of this design is available inside C2000Ware_DigitalPower_SDK. To open the project:

1. Install CCS (version 12.1 or above)
2. Install C2000Ware DigitalPower SDK from the tool page
3. Open CCS, and create a new workspace
4. Inside CCS, go to View -> Resource Explorer. Under Resource Explorer, go to Software -> C2000Ware DigitalPower SDK - <version> -> development kits and select this solution; that is, TIDM-SOLAR-DCDC, and click import project by browsing to Solutions→ TIDM-SOLAR-DCDC →f28004x→ccs→ mpptdcdc_f28004x.projectspec

3.1.2.2 Digital Power SDK Software Architecture

Once the project is imported, the Project Explorer will appear inside CCS.

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 " **mppt_dcdc**" which is also referred to as the module name.

The module name for all the variables and defines used in the solution are also defined similarly.

Therefore, all variables and function calls are prepended by the **mppt_dcdc** name (for example, **MPPTDCDC**_vBus_sensed_pu). This naming convention lets the user combine different solutions while avoiding naming conflicts.

3.1.2.3 Interrupts and Lab Structure

The software flow diagram for this project is shown in Figure 3.1. The project consists of two ISRs (ISR1 and ISR2) with ISR1 being the fastest and non-nestable ISR. ISR1 is reserved for the control loop and the PWM update. ISR1 is triggered by the EPWM1_BASE → EPWM_INT_TBCTR_D_CMPC event. ISR1 runs at 50kHz rate which half the PWM switching frequency.

ISR2 runs at 10kHz and is triggered by CPU Timer2 INT which is initiated by an overflow on CPU timer. It is used to run housekeeping functions such as, power board fault condition status check and activation of protection function (bus voltage OVP, panel volt OVP), software frequency response analyzer (SFRA) background tasks and toggling heartbeat LED etc.

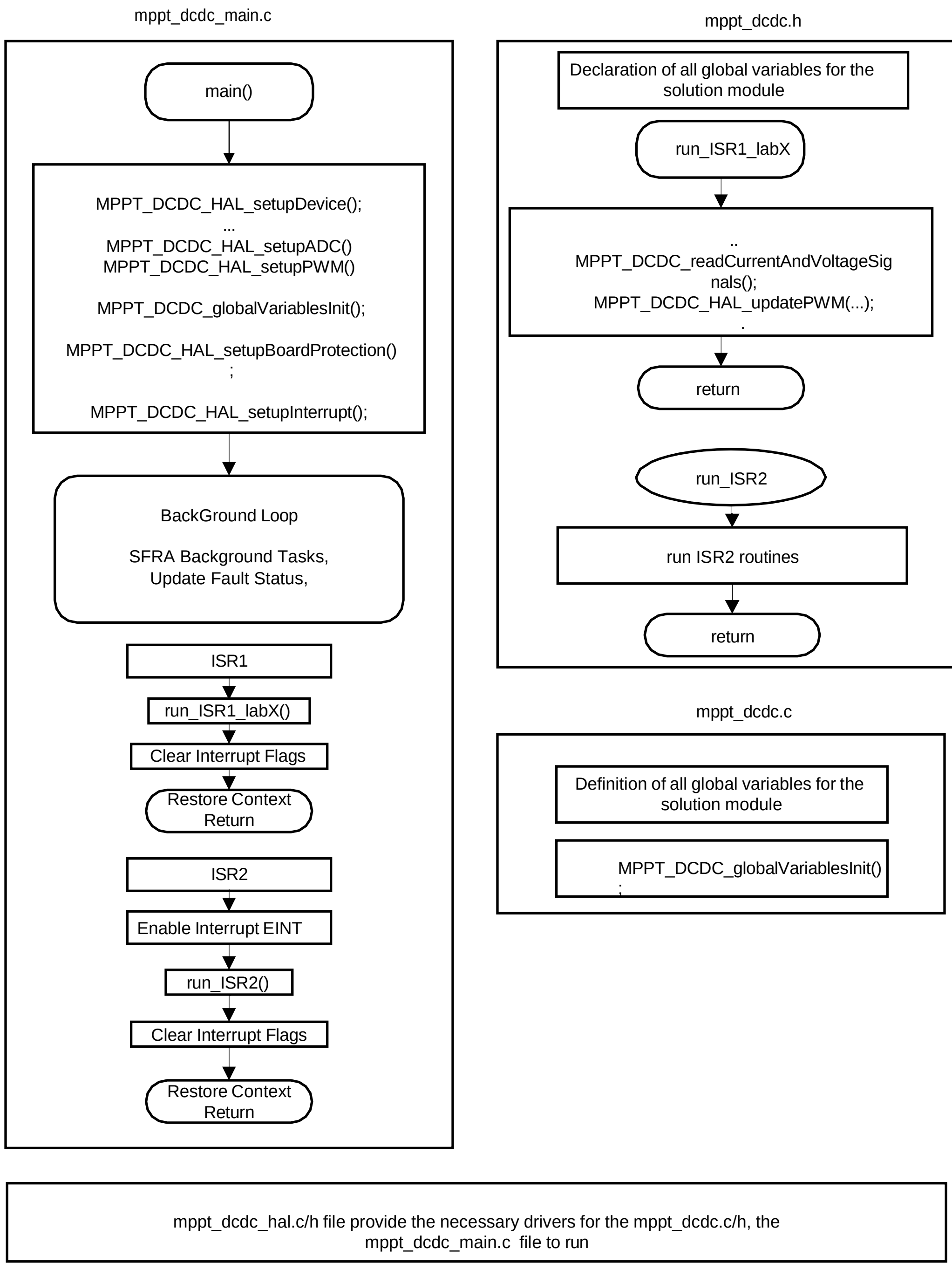

mppt_dcdc_main.c
main()
MPPT_DCDC_HAL_setupDevice();
...
MPPT_DCDC_HAL_setupADC()
MPPT_DCDC_HAL_setupPWM()
MPPT_DCDC_globalVariablesInit();
MPPT_DCDC_HAL_setupBoardProtection()
;
MPPT_DCDC_HAL_setupInterrupt();
BackGround Loop
SFRA Background Tasks,
Update Fault Status,
ISR1
run_ISR1_labX()
Clear Interrupt Flags
Restore Context
Return
ISR2
Enable Interrupt EINT
run_ISR2()
Clear Interrupt Flags
Restore Context
Return
mppt_dcdc.h
Declaration of all global variables for the solution module
run_ISR1_labX
..
MPPT_DCDC_readCurrentAndVoltageSignals();
MPPT_DCDC_HAL_updatePWM(...);
.
return
run_ISR2
run ISR2 routines
return
mppt_dcdc.c
Definition of all global variables for the solution module
MPPT_DCDC_globalVariablesInit()
;
mppt_dcdc_hal.c/h file provide the necessary drivers for the mppt_dcdc.c/h, the mppt_dcdc_main.c file to run

Figure 3-1. Software Flow Diagram

The software for this reference design is organized in 3 labs, Table 3-1 lists the labs and how they have been tested. All the labs are run using the C28x Main CPU.

Table 3-1. Overview of Labs to Test Reference Design

LAB NUMBER	DESCRIPTION	COMMENTS	TEST ENVIRONMENT
1	PWM and ADC check. Open loop check	PWM Check, ADC check, Protection Check, open loop mode, DC source connected and resistive star network as load	Control Card + Power Stage Hardware + HV DC source
2	Closed Current Loop for DC/DC boost, Resistive or electronic load connected at DC output, DC source connected.		Control Card + Power Stage Hardware + HV DC source+Load
3	Closed Voltage loop + Current Loop, Resistive or electronic load connected at DC output, PV panel emulator connected as input power source.		Control Card + Power Stage Hardware + PV panel emulator+Load

To build the project, right-click on the project name and click *Rebuild Project*. The project builds successfully.

To load the project, first make sure in the Project Explorer the correct target configuration file is set as Active under targetConfigs (**F28004x-ControlCard**.ccxml file). Then, click *Run* → *Debug* to launch a debugging session. 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.

To debug the system, one would monitor the variables in the watch/expressions window. To populate this window with the correct variables, click View → Expressions. Then right click on the Expressions tab on CCS and select Remove All. This will clear the Expressions window. Now right click again on the cleared Expressions window and select Import. Then browse to the project folder (solutions --> tidm_solar_dcdc --> source --> mppt_dcdc --> debug) and select lab1.txt. This will populate the watch window with the appropriate variables needed to debug the system for Lab 1. Enable Continuous Refresh button on the watch window to enable continuous update of values from the controller.

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 be updated. It 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. To enable real-time mode, click on this button on the top bar of CCS . 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.

In some of the labs for this project, the measured currents and voltages or the control variables need to be verified by viewing the data in a graph window. To do this, CCS Graph window tool can be used. This Graph window, in conjunction with a piece of code that runs on the controller, can show a snapshot of how the values are being sensed by the controller. The values are logged by the datalogger typically in the slower ISRs. In this case the datalogger is run in Lab2 from the fast ISR1. To import the graph into the CCS view, select Tools => Graph => DualTime, and click Import and point to the graph1.GraphProp file inside the project folder (solutions --> tidm_solar_dcdc --> source --> mppt_dcdc --> debug). Two graphs will appear in CCS. Click Continuous Refresh on these graphs. A second set of graphs can also be added by importing the graph2.GraphProp file.

CPU Loading

The main control ISR (ISR1) in Lab 3 takes approximately 12 MIPS at 50-kHz ISR frequency, that is approximately 12% of the CPU when running from 100-MHz F28004x processor. This includes the ADC drivers, MPPT calculation, PWM generation, current control loop, voltage control loop, and the SFRA call.

3.1.2.4 Protection Scheme

Figure 3-2 explains the software functions used to setup the trip behavior on the design.

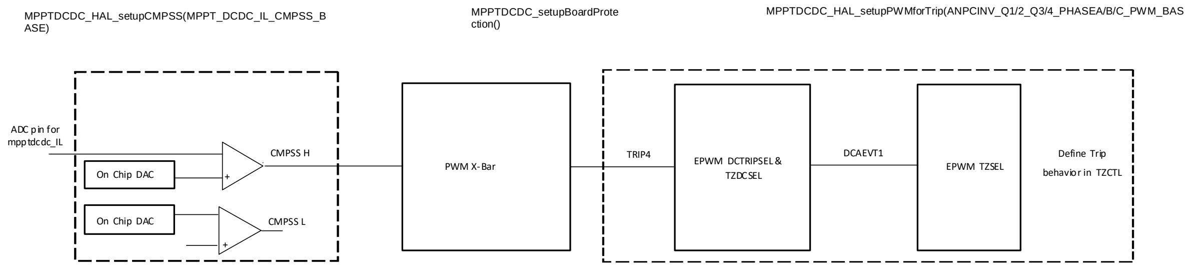

MPPTDCDC_HAL_setupCMPSS(MPPT_DCDC_IL_CMPSS_BASE)
MPPTDCDC_setupBoardProtection()
MPPTDCDC_HAL_setupPWMforTrip(ANPCINV_Q1/2_Q3/4_PHASEA/B/C_PWM_BAS
ADC pin for mpptdcdc_IL
On Chip DAC
CMPSS H
On Chip DAC
CMPSS L
PWM X-Bar
TRIP4
EPWM DCTRIPSEL & TZDCSEL
DCAEVT1
EPWM TZSEL
Define Trip behavior in TZCTL

Figure 3-2. Software Diagram for Trip Setup

The figure shows that the overcurrent protection (OCP) has been implemented using the comparator sub-system (CMPSS) internal to the MCU. Over-voltage protection (OVP) has also been implemented. But this is implemented in software. Under the fault condition (OCP or OVP) the function MPPTDCDC_updateFaultStaus() reports the fault status. This function is called periodically in a slow background task for updating Trip Flags and also resetting the latch if needed.

If a trip event has occurred, the trip flags need to be cleared separately in order to enable the PWM. This part is typically handled in the ISR by calling MPPTDCDC_clearPWMTrips().

3.1.2.5 PWM Switching Scheme

Figure 3-3A shows the EPWM and ADC sampling configuration for the MPPT boost stage. The boost PWM signals are generated at a frequency of 100 kHz i.e. a period of 10 us. With the controller operating at 100MHz, one count of the time base counter of ePWM1 corresponds to 10ns. This implies a PWM period of 10us is equivalent to 1000 counts of the time base counter (TBCNT1, TBCNT2). The ePWM1 and ePWM2 modules are configured to operate in up-down count mode as shown in Fig 3-3A. This means a time base period value of 500 (period register value) will give a total PWM period value of 1000 counts (i.e. 10 us). Figure 3-3A also shows that fastest ISR, ISR1, runs at every other PWM cycle resulting in a fastest sampling frequency of 50kHz.

The LLC PWM signals are generated at a frequency of 120 kHz i.e. a period of 8.33 us. With the controller operating at 100MHz, one count of the time base counter of ePWM3 corresponds to 10ns. This implies a PWM period of 8.33us is equivalent to 833 counts of the time base counter (TBCNT3). The ePWM3 module is configured to operate in up-down count mode. This means a time base period value of 416 (period register value) will give a total PWM period value of 832 counts (i.e. 8.32 us).

To maintain synchronous operation all conversions are triggered as follows: MPPT_DCDC_HIGH_FREQ_PWM_A_BASE; that is, EPWM1 TBCTR_U_CMPB → EPWM1_SOCA, triggered every 2nd cycle (ADC sampling and PWM duty calculation done at half the PWM frequency).

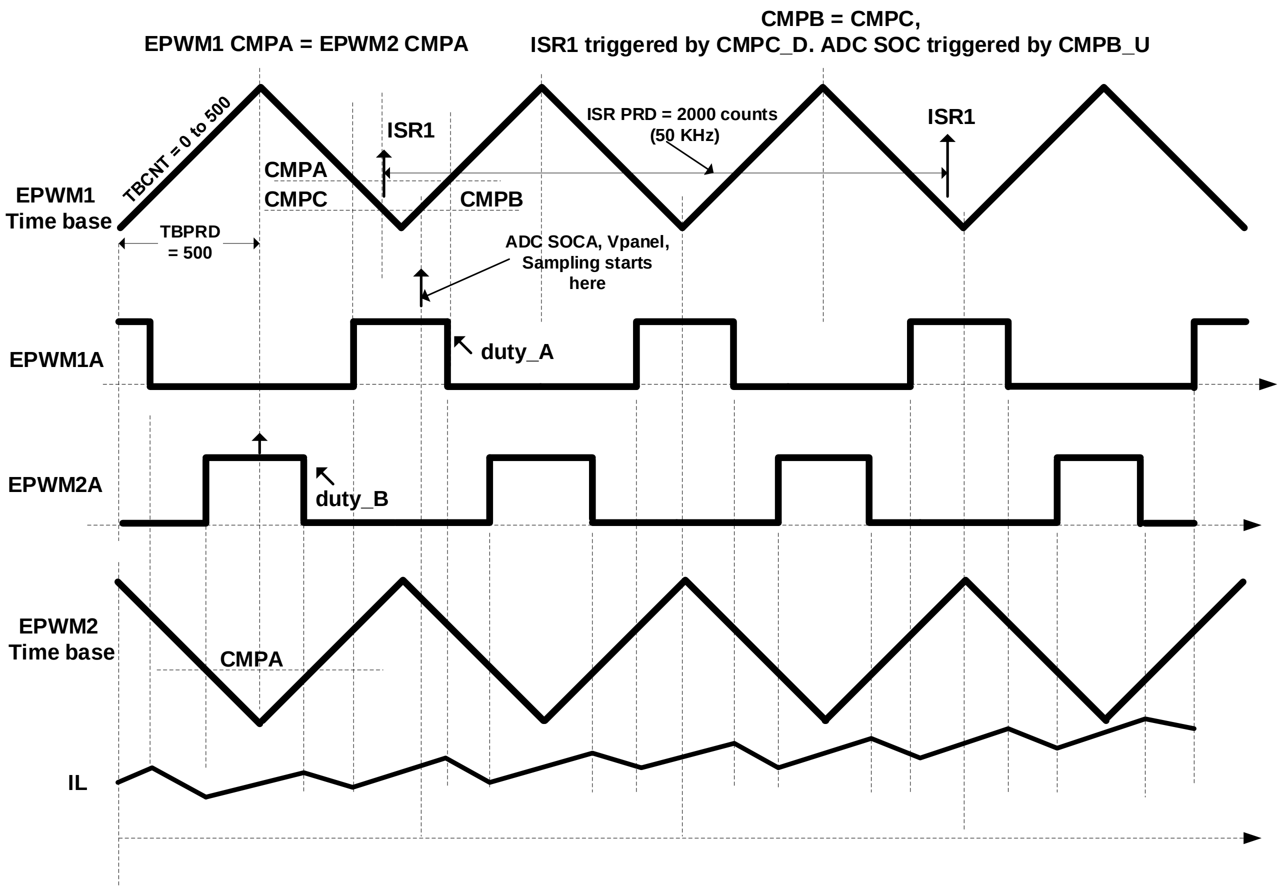


Figure 3-3A. PWM Generation and ADC sampling

3.1.2.6 ADC Loading

Table 3-4 shows the mapping with F280049C on the TIDA-010210 hardware.

Table 3-4. ADC Loading Architecture

	ADC-A	ADC-B	ADC-C
SOC0	Vpanel → ADC-A0,		Not used
SOC1	Vbus → ADC-A1,		
SOC2	IL → ADC-A2, CMPSS1		
SOC3		VOUT → ADC-B0	

3.2 Testing and Results

3.2.1 Lab 1

This lab is to check the open loop PWM and ADC signals. It also allows checking the protection function.

Set the project to Lab 1 by changing the Lab Number in the <mppt_dcdc_settings.h> file.

```
#define MPPT_DCDC_LAB 1U
```

Right-click on the project name and click *Rebuild Project*. The project builds successfully.

To load the project, first make sure in the Project Explorer the correct target configuration file is set as Active under targetConfigs (*.ccxml file). Then, click *Run* → *Debug* to launch a debugging session. 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. Click Run → Resume to run the program.

To populate this window with the correct variables, click View → Expressions. Then right click on the Expressions tab on CCS and select Remove All. This will clear the Expressions window. Now right click again on the cleared Expressions window and select Import. Then browse to the project folder (solutions --> tidm-solar-dcdc --> source --> mpptdcdc --> debug) and select lab1.txt. This will populate the watch window with the appropriate variables needed to debug the system. Enable Continuous Refresh button on the watch window to enable continuous update of values from the controller.

To check the boost PWM waveforms first enable the PWMs by entering a ‘1’ for the ‘Value’ field corresponding to the expression “MPPTDCDC_clearPWMTrip”. Now enter any duty ratio, say 0.25, in the ‘Value’ field corresponding to the expressions “MPPTDCDC_duty_A” and “MPPTDCDC_duty_B”. Finally enter a ‘1’ in the ‘Value’ field corresponding to the expression “MPPTDCDC_updateDuty”. Now use a scope probe to check the boost PWM outputs, PWM-1 and PWM-2 on the board. These probe points are located inside M1 block on the main power board. These should be 100kHz interleaved PWM with 0.25 duty ratio. The interleaving should show a phase shift of 180 deg between the two PWM outputs. Check the two PWM outputs related to the LLC stage. These probe points are located inside M5 block on the main power board and are also labelled as PWM-1 and PWM-1. These signals should be 120kHz PWM, with 50% duty and phase shifted by 180 deg.

CCS Expressions window also shows some other variables related to the DC bus voltage (MPPTDCDC_guiVbus), the panel voltage (MPPTDCDC_guiVpanel) and the per unit panel current (MPPTDCDC_iL_sensed_pu). Since no high voltage is applied to the board at this time, the values corresponding to these expressions will be almost 0.

The MPPT function is disabled in Lab 1 and Lab 2. Therefore, the expressions related to MPPT, grouped under the single expression “MPPTDCDC_incc1”, will all be 0 at this time.

Figure 3-4 shows the Lab 1 SW diagram when the code is running.

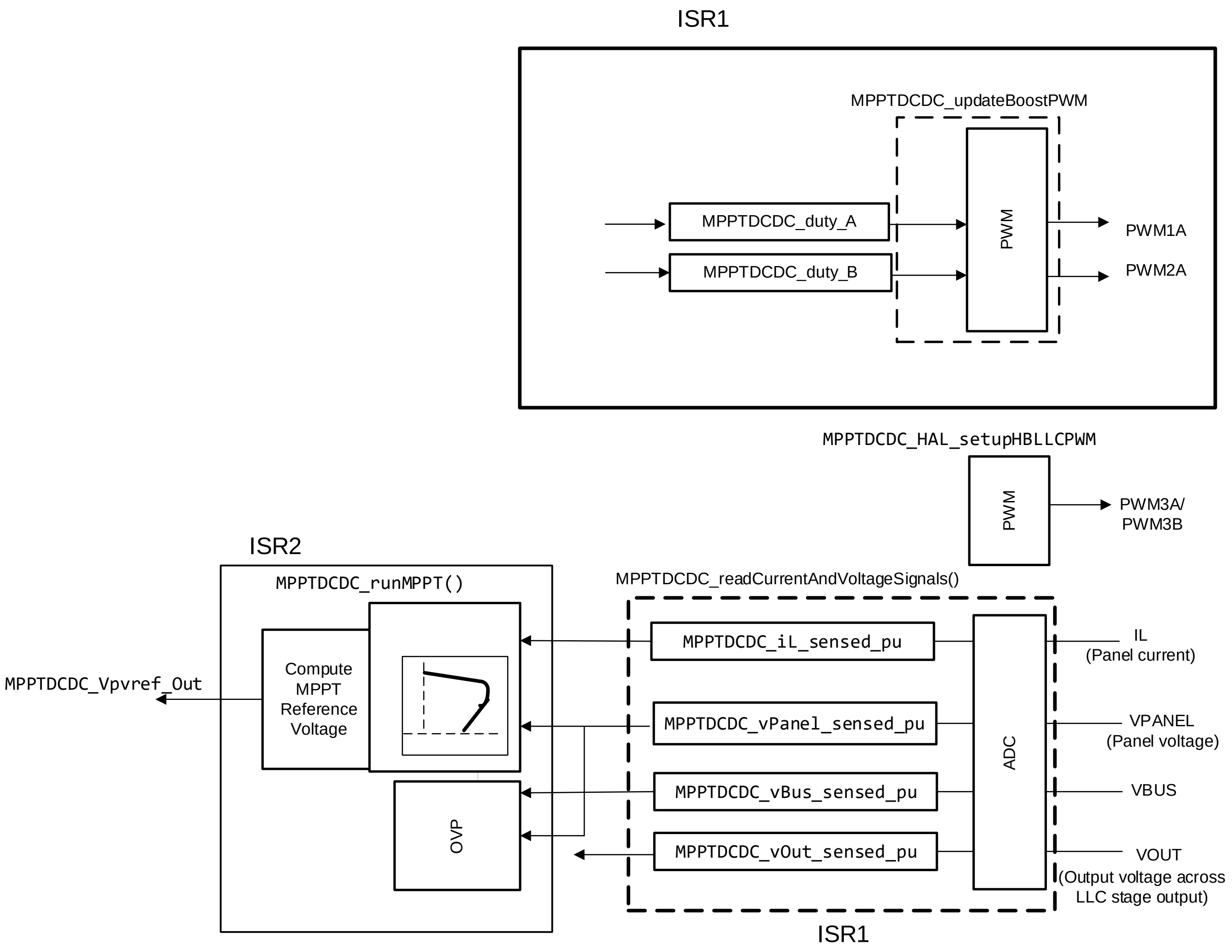


Figure 3-4. Lab1 Software Diagram

3.2.2 Testing Labs related to MPPT Operation

Labs 1, 2, and 3 elaborate the steps for running the MPPT DCDC power stage. Lab 1 is the converter operation in open loop, Lab 2 is the converter operation with closed current loop. Lab 3 is the converter operation under closed voltage and current loops where the reference voltage for the voltage loop is set by use of MPPT control. This Lab 3 is also checked only with a solar panel emulator connected across the input. The input voltage (< 200 VDC) is applied across terminals BS1 (Vin+) and BS2 (Vin-Ret). 12-V auxiliary power supply is connected to JP1 located inside M3 block on the power board. The resistive load is connected across LLC stage output terminals BS6(Vo-R) and BS7(GND-S). Verify that the DC/DC boost stage output terminal BS4(400V) is connected to LLC input terminal BS5(Vin-R). This way DC/DC boost stage output is applied to the isolated LLC stage input.

3.2.2.1 Lab 2

In this lab the power stage is run under open voltage loop and closed current loop where the reference current is set by the user from the CCS watch window. Since the voltage loop is open, MPPT is disabled and there is no MPPT operation under this Lab 2. For this lab setup, connect the DC voltage source (specified in section 3.1.1) across the DC/DC boost input terminals BS1(Vin+) and BS2(Vin-Ret). Now connect the resistive load across the LLC output terminals BS6(Vo-R) and BS7(GND-S).

Figure 3-5 shows the details of some of the SW functions implemented for Lab 2 operation. The incremental functions added to this Lab 2 compared to Lab 1 are the current loop controller (MPPTDCDC_GI_RUN), the data logger (DATALOG) and the averaging function (EMAVG_MACRO). Overvoltage protection for both the boost input (panel output volt) and DC bus voltage output (DC/DC boost output) are also tested to this lab.

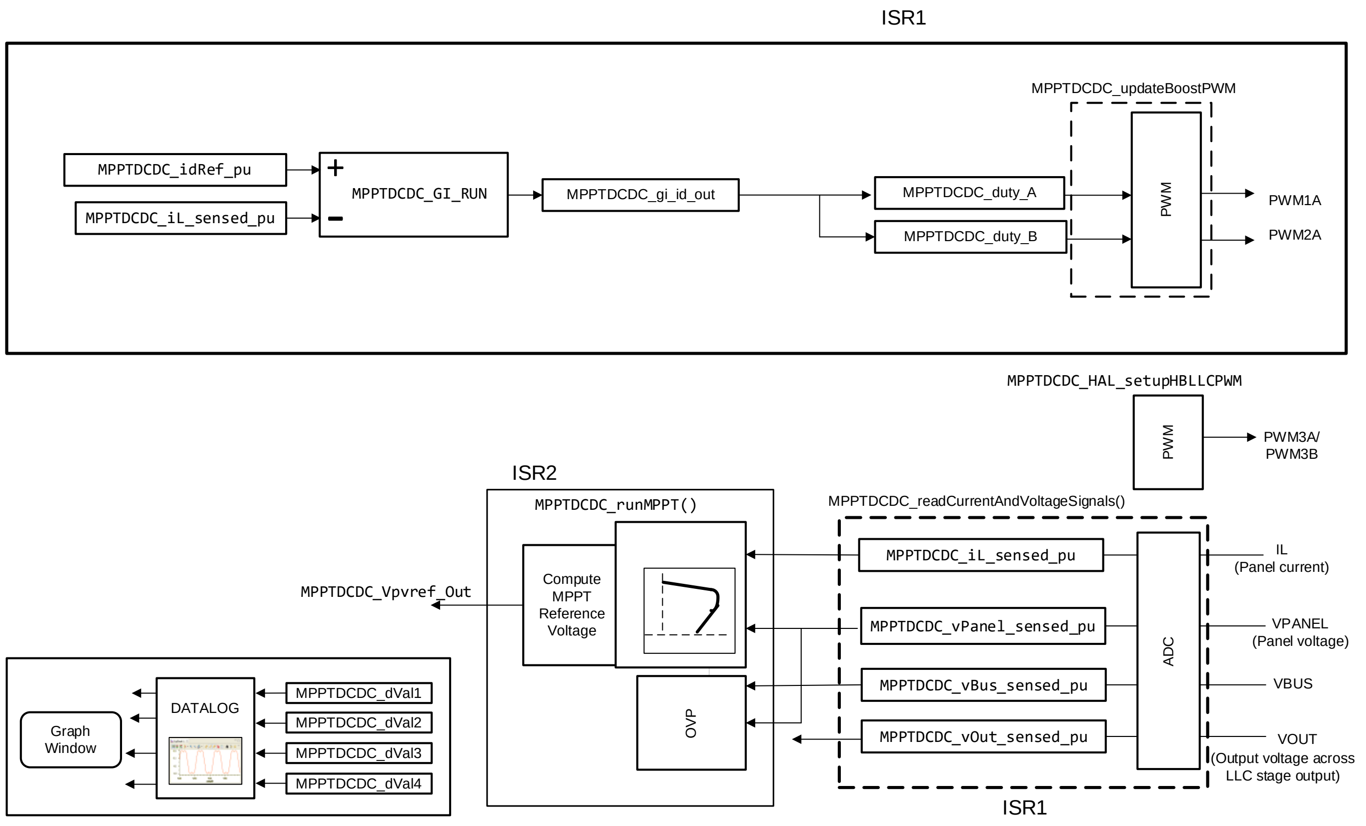
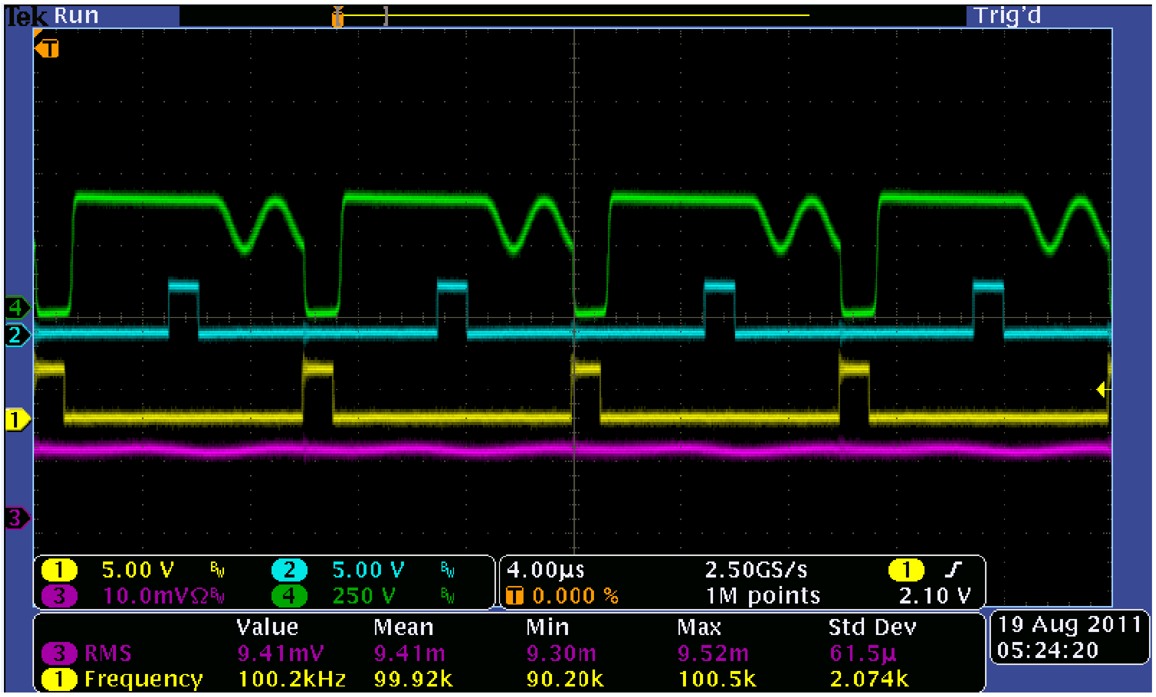


Figure 3-5. Lab2 Software Diagram

Set the project to Lab 2 by changing the Lab Number in the < mppt_dcdc_settings.h > file.

Build and load the code, use the lab2.txt file to populate the watch variables in the CCS window.

- Use an appropriate resistive load (specified section 3.1.1) with minimum rating of 400V,1kW. Now set the resistor value to about 1K ohm. This will allow about 160W load when the output is 400V.
- In the CCS expression window set MPPTDCDC_idRef_pu to 0.05. This variable sets the magnitude of the reference current command for the current control loop.
- Turn on the isolated DC source (specified section 3.1.1) with a dc voltage of about 20V. Slowly increase the DC input voltage to the board from the DC source. Monitor the LLC stage output voltage as the input voltage is raised slowly to 300V. Slowly adjust (in steps of 0.01) the value for MPPTDCDC_idRef_pu to set the output voltage to about 385V. Use an oscilloscope with voltage and current probes to observe the input voltage, input current, boost MOSFET voltage, LLC primary current and the PWM outputs. With a 300V boost input and 1.0kohm resistive load when the boost output voltage is set to 373V you should see the LLC output voltage (board output) of 385V. The following scope plot is captured under this condition. Here Ch1 and Ch2 show the boost PWM outputs. Ch3 is the boost input current and Ch4 is the voltage across the boost MOSFET.
-

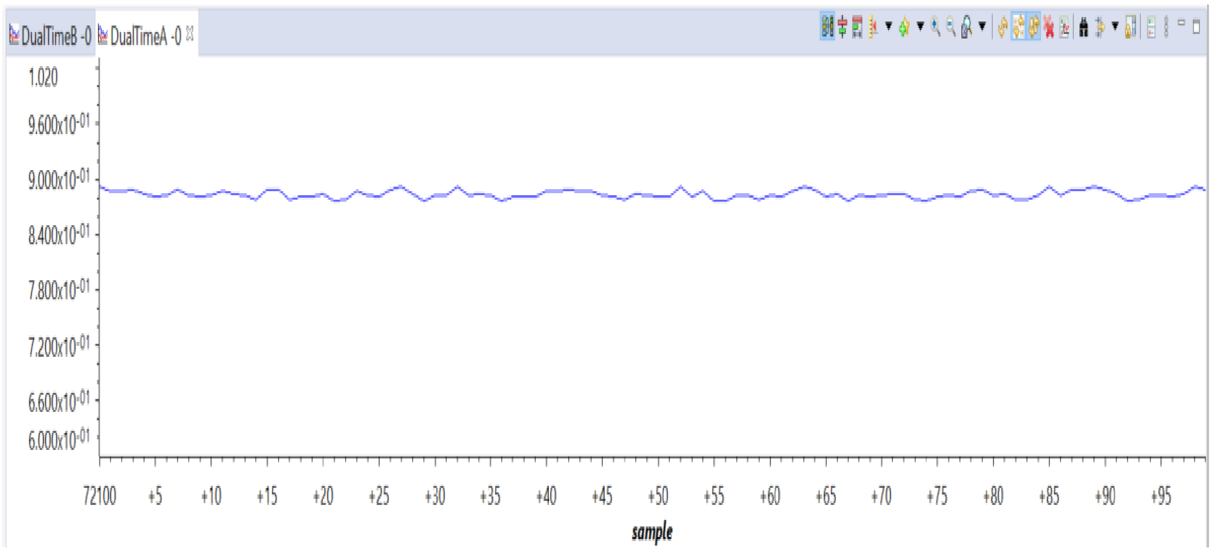


1 5.00 V 2 5.00 V
3 10.0mVΩ 4 250 V
4.00µs 2.50GS/s 1 2.10 V
T 0.000 % 1M points

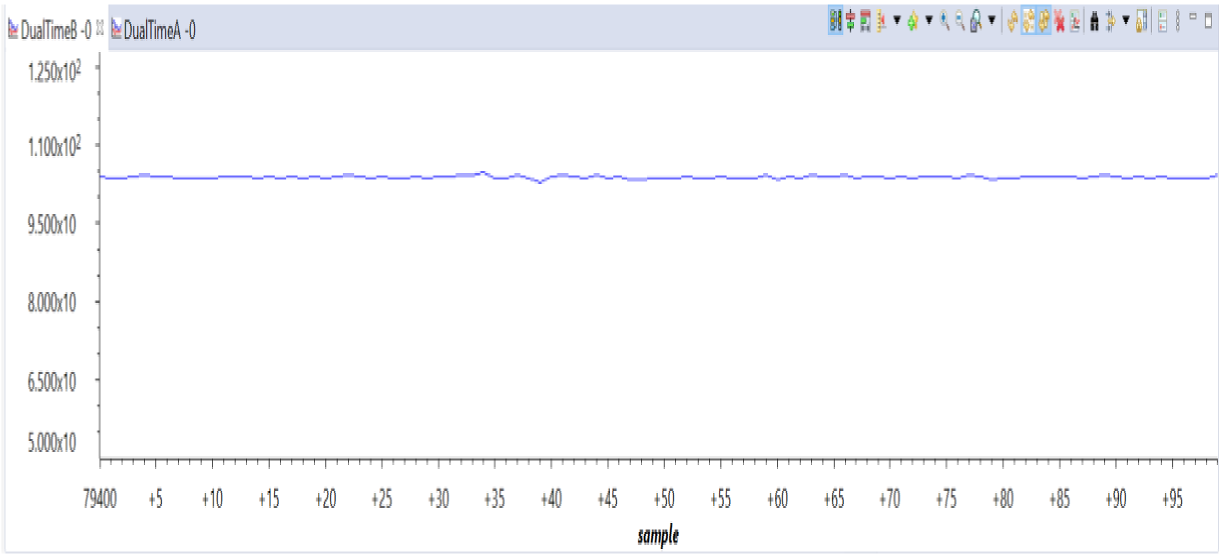
	Value	Mean	Min	Max	Std Dev
3 RMS	9.41mV	9.41m	9.30m	9.52m	61.5µ
1 Frequency	100.2kHz	99.92k	90.20k	100.5k	2.074k

19 Aug 2011
05:24:20

- .
- Increase MPPTDCDC_idRef_pu slightly (in steps of 0.01) and observe the bus voltage settles to a higher value. Increasing *MPPTDCDC_idRef_pu* increases the magnitude of the current reference signal and the bus voltage will rise. Therefore, apply caution and set the overvoltage protection threshold to a value less than 400V. Verify that the current loop controller is working properly by checking the value of sense inductor current MPPTDCDC_iL_sensed_pu. Under all conditions, this inductor current value should match the reference current command value represented by MPPTDCDC_idRef_pu.
- Verify the sensed voltage and current measurement data in the graph window before proceeding to Lab 3. Figure 3-6 shows the graph window for sensed MPPT stage input current by using ADC module of C2000 together with the input voltage signal. These graph window plots are obtained with boost input voltage of 100V, current reference command of 0.1 pu corresponding to absolute current of 0.88A and an output load of about 90W.



DC/DC Boost inductor current (A)



DC/DC Boost input voltage (V)

Figure 3-6. Datalogger waveforms for MPPT-Solar-DCDC Voltage and Current Signal

3.2.2.2 Lab 3

In this lab the power stage is run under closed voltage loop and closed current loop control with MPPT algorithm generating the reference voltage for the voltage loop. This also requires a solar panel emulator (specified in section 3.1.1) be connected across the input terminals BS1 and BS2. Figure 3-7 shows the SW diagram for Lab 3 with the new functions added in this lab. This incremental function compared to Lab 2 is the voltage loop controller (MPPTDCDC_GV_RUN). Also, in this lab the MPPT function is enabled and its output is applied as the reference input to the voltage loop. The default code setting uses incremental conductance algorithm for the MPPT. As indicated in the diagram the MPPT block runs from a 10kHz ISR2.

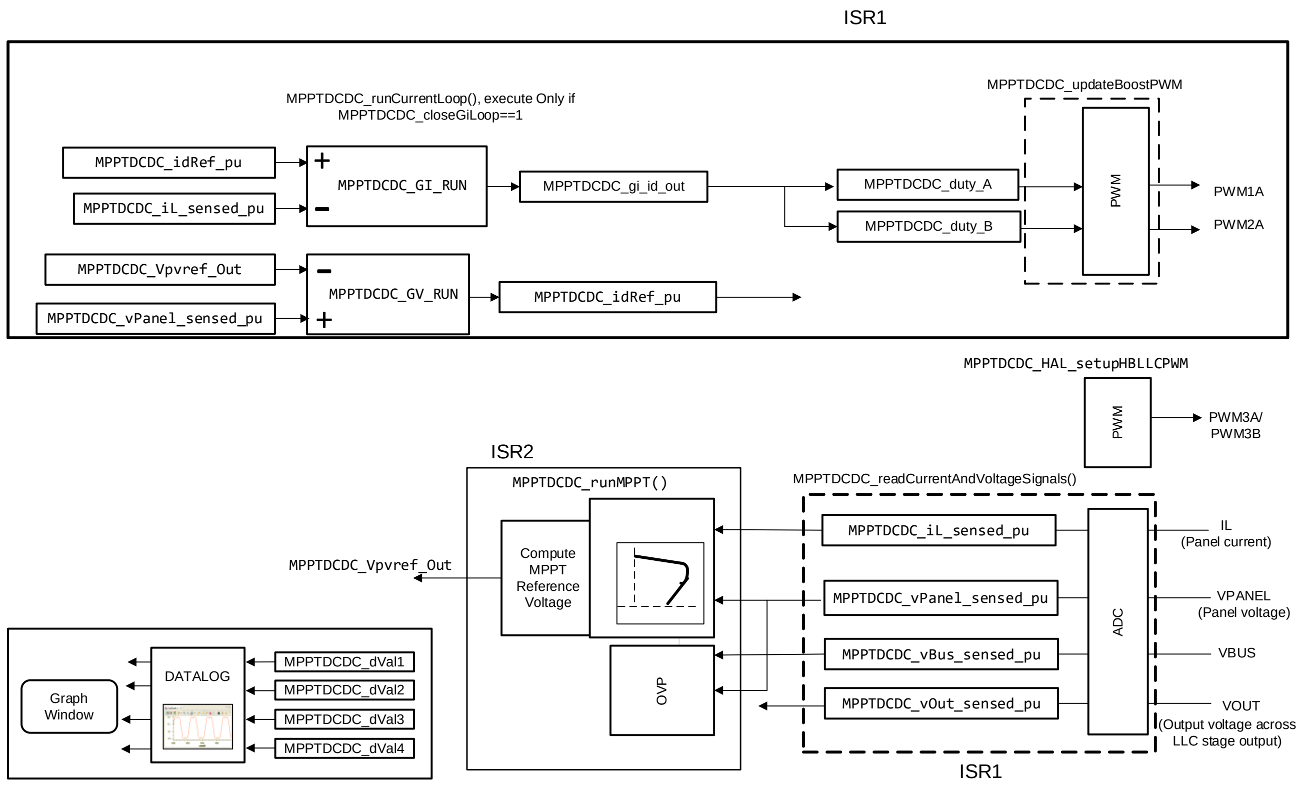


Figure 3-7 Lab3 Software Diagram

Set the project to Lab 3 by changing the Lab Number in the < mppt_dcdc_settings.h > file.

At this time, do not supply any HV power to the board.

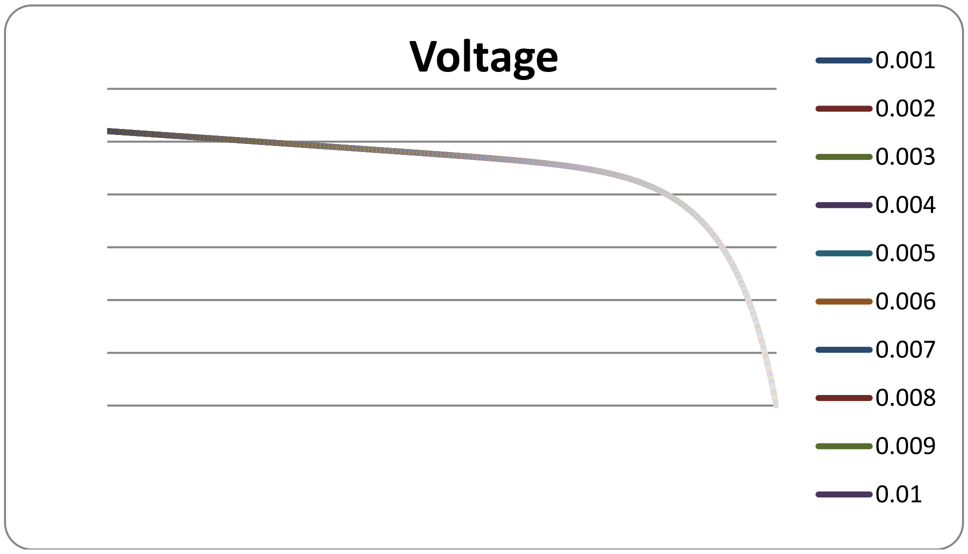
- The current and voltage compensator coefficients used for running the control loop are shown in the following code. The user can modify these coefficients to meet the necessary loop bandwidth and phase margin.

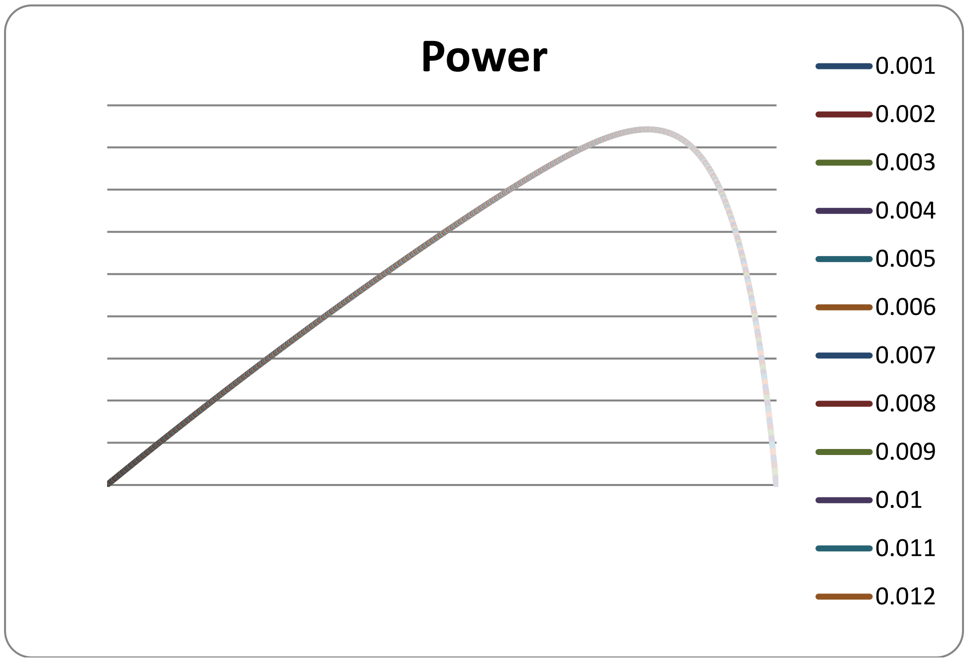
```
#define MPPT_DCDC_GI_PI_KP ((float32_t)0.1)
#define MPPT_DCDC_GI_PI_KI ((float32_t)0.005)

#define MPPT_DCDC_GV_PI_KP ((float32_t)0.5)
#define MPPT_DCDC_GV_PI_KI ((float32_t)0.0005)
```

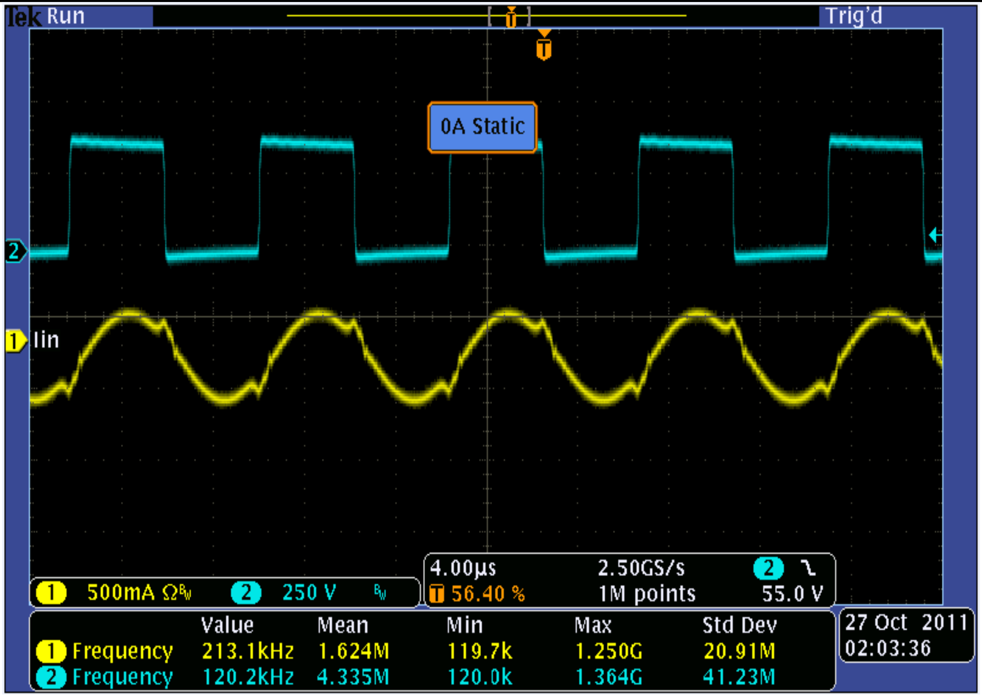
- Build and load the code, use the lab3.txt file to populate the watch variables in the CCS window.
- In the CCS expression window view, locate the two variables MPPTDCDC_guiVpanel and MPPTDCDC_guiVbus. These two variables represent the boost input and output voltages in volts respectively. These will slowly increase as the DC-DC starts up when PV panel emulator power is applied and the MPPT is turned on. Verify these values with those obtained from a voltmeter reading across the input terminals (BS1 & BS2) and the DC bus terminals (BS4-BS3) in the power board. Configure the solar panel emulator to provide input power to the board. Configure the panel emulator to emulate the following solar panel characteristics. Connect it to the board input but **do not turn on panel power at this time.**

Example Panel Emulator Parameters		
Voc	Open circuit panel voltage	**260V**
Vmpp	Panel voltage for max power point tracking (MPPT)	**220V**
Impp	Panel current for max power point tracking (MPPT)	**0.75A**
Isc	Short circuit panel current	**1A**

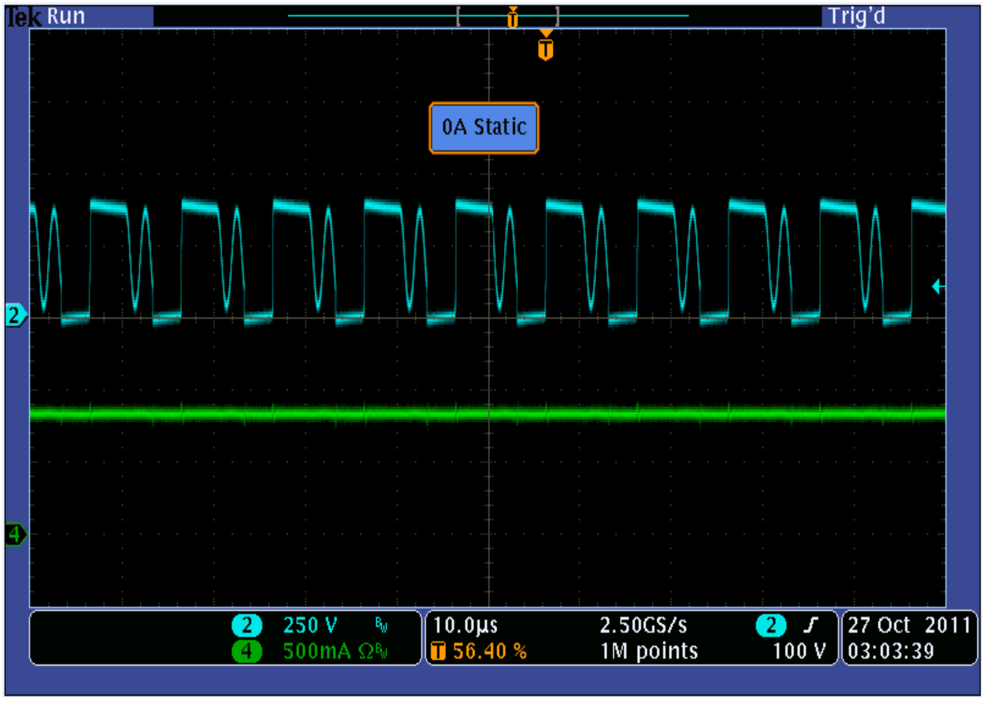




- Connect an appropriate resistive load to the output terminals BS6-BS7 (Vo-R & GND-S terminals). As in the example above, if the panel emulator is configured to supply 165W of power at MPPT point, then select a load resistor value of 970 ohm so that the EVM output voltage is limited to about 400V (P = 400*400/970 ≈ 165W). A smaller resistor will also work as long as the output voltage does not fall below 350V. This means that the smallest resistor that can be chosen for this load set up (165W) is about 742 ohm (R = 350*350/165 = 742 ohm). A resistor value larger than 970 ohm will cause output voltage higher than 400V for this load set up. **This output overvoltage condition must be prevented by choosing the maximum resistor value of 970 ohm for this load set up of 165W.** It is recommended that the resistor with a power rating > 200W is used for this load setting.
- Starting the MPPT algorithm (in Lab 3), with the PV panel emulator power applied to the board input, will require a minimum boost DC bus voltage. This minimum DC bus voltage threshold is set in the code as 0.3 per unit which translate to about 153V (511V*0.3) because of the max DC bus sense voltage of 511V.
- Use a voltmeter to monitor the DC bus voltage across the boost stage output. Now turn on the PV panel emulator with the setup described above. At this point the MPPT will still remain off and so the input voltage should be around 260V and the output voltage, with a 1kohm load resistor, will also be at the same level. From the CCS watch window now set the variables MPPTDCDC_clearPWMTrip to 1. This will start the MPPT algorithm and the output will rise to around 400V. With the MPPT turned on and the input voltage loop controller connected, the panel output voltage (i.e., the boost input voltage) will be around 220V. The board will deliver 165W of panel power at the MPPT reference voltage of 220V.
- Use an oscilloscope to capture the switch node voltage and transformer primary current from the LLC stage under this operating condition. This is shown in figure below where Ch2 represents the LLC primary switch node voltage and Ch1 represents the LLC primary current.

- 

	Value	Mean	Min	Max	Std Dev
1 Frequency	213.1kHz	1.624M	119.7k	1.250G	20.91M
2 Frequency	120.2kHz	4.335M	120.0k	1.364G	41.23M

- Now use the scope probes to capture the boost stage MOSFET drain to source voltage and the PV emulator current under this operating condition. This is shown in figure below where Ch2 represents the boost MOSFET drain to source voltage and Ch4 represents the panel current.
- 

- Observe the variables on the watch window. The variable MPPTDCDC_guiVpanel should show a value of about 220V. The variable MPPTDCDC_guiVbus should show a value of about 400V.
- Check the register MPPTDCDC_boardFaultFlags. It should indicate no trip condition.
- To bring the system to a safe stop, turn off the output of the solar panel emulator.

4 Trademarks

TMS320C2000™, C2000™ are trademarks of Texas Instruments.
All trademarks are the property of their respective owners.

5 About the Author

Shamim Choudhury has been with Texas Instruments Inc. since December 1997 where he is currently serves as a Senior Member of the Tech Staff in the C2000 Systems team in Sugarland, Texas. As a member of C2000 Systems team for many years his areas of interest have been on digital control of power converters, switching power supplies, Power Factor Correction, DC/DC converters, UPS and Solar Inverters.
He received his M.S. degree in Electrical Engineering from Texas A&M University in College Station, TX, in December 1991. He is a Senior Member of the IEEE Power Electronics Society.

6 Revision History

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

Revision A. December 2022

IMPORTANT NOTICE AND DISCLAIMER