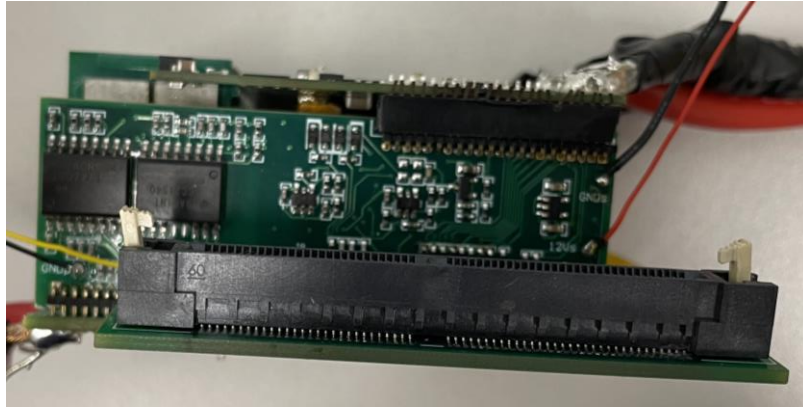


PMP41017 software User Guide

Procedures for Running the Incremental Builds

1. Hardware setup

- 1) Connect the interposer board to J8 on the control board (attention for the direction as below picture);



- 2) Insert the control card in the J6 slot on the interposer board;



- 3) Connect two isolated 12-V, 1-A DC power supply at 12Vs/PGNDs and 12Vp/GNDp.
- 4) To connect JTAG, use a USB cable from the control card and connect it into a host computer. Build and load the project.
- 5) A DC power supply can be connected to the HVBUS, PGND net. A constant current load of approximately 10 A can be connected to the output at the 12Vout, V0-out net.
- 6) Current and voltage probes can be connected to observe the resonant current, switching node voltage, SR drive signal and output voltages.
- 7) (***Noted***), the hardware board, without cooling or with weak air cooling, **CAN ONLY** be operated smaller than 80A load. Please **DO** make an air channel with strong cooling FAN, when operated equal or larger than 80A.

2. Lab 1: open loop check

The objective of this build is to:

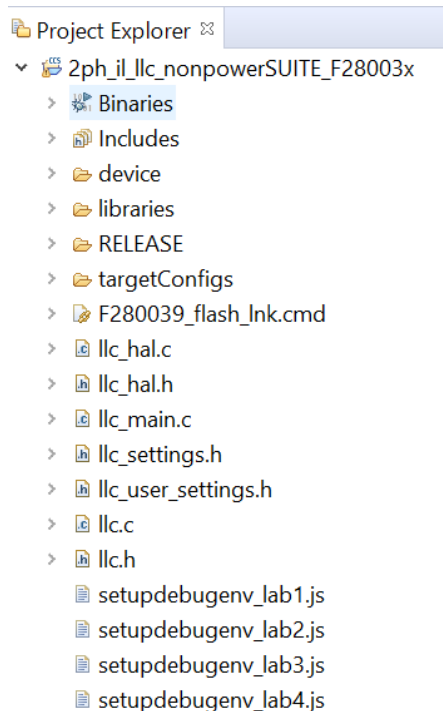
- Evaluate the open loop LLC operation, such as switching frequency, resonant frequency, deadtime and SR control.
- Verify the CMPSS, PWM and ADC driver modules
- Verify the FET driver circuit and sensing circuit on the board

Because this system is running open-loop, the ADC measured values are only used for instrumentation purposes in this build.

2.1. Start CCS and Open a Project

To quickly execute this build:

1. Connect an isolated DC power supply capable of providing at least 100 V at up to 1 A to the input terminals of the LLC stage. Do not turn the power supply on at this time.
2. Connect a 10-A constant current load (electric load) at the output.
3. Open CCSv11 (or newer). Maximize the program to fill the screen. Close the welcome screen if it opens up.
4. A project contains all the files and build options needed to develop an executable output file (.out), which can be run on the MCU hardware. On the menu bar click Project → Import Existing CCS/CCE Eclipse Project. Under Select root directory, navigate to the C2000Ware Digital Power SDK folder and select <install_location>\solutions\pmp_41017. Click Finish. This project invokes all the necessary tools (compiler, assembler, linker) to build the project.
5. In the project window on the left, click the plus sign (+) to the left of Project. Figure 15 shows an example project view.



2.2. running the project

2.2.1. Build and Load the Project

1. Open llc_settings.h, and change settings to **#define LLC_LAB 1**
2. Turn the 12-V power supply ON. The LED on the control card lights up and indicates the device is powered. Click on the *Debug* button or click *Run* → *Debug*. The build 1 code should compile and load.
3. Notice the CCS Debug icon in the upper right-hand corner, indicating that the user is now in the Debug Perspective view. The program should be stopped at the start of *main()*.

2.2.2. Debug Environment Windows

It is standard debug practice to watch local and global variables while debugging code. There are various methods for doing this in CCS, such as memory views and watch views.

1. Populate the expressions window entries by clicking on *View* → *Scripting* console on the menu bar and then opening the *setupdebugenv_lab1.js* file from the project directory `solutions\pmp_41017\source\hhcllc\debug` using the scripting console *Open File* command.
2. Right click the filename of 'setupdebugenv_lab1.js' in the window of Project Explore, then click 'Run Script'.

2.2.3. Using Real-Time Emulation

Real-time emulation is a special emulation feature that allows the windows within CCS to be updated at up to a 10-Hz rate *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 have those changes affect the MCU behavior.

This is very useful when tuning control law parameters on the fly, for example. Enable real-time mode by hovering the mouse on the buttons on the horizontal toolbar and clicking



Enable Silicon Real-time Mode (service critical interrupts when halted, allow debugger accesses while running)

1. 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 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.
2. When a large number of windows are open, as bandwidth over the emulation link is limited, updating too many windows and variables in continuous refresh can cause the refresh frequency to bog down. Right click on the button in the Expressions window and select *Continuous Refresh Interval*. One can slow down the refresh rate for the expressions window variables by changing the continuous refresh interval (milliseconds) value. A rate of 1000 ms is usually enough for these exercises.
3. Click on the *Continuous Refresh* button () for the expressions view.

2.2.4. Run the Code

1. Run the code by using the <F8> key, or by using the Run button on the toolbar.
2. In the expression window, input '1' to the 'LLC_startFlag', and then PWM will generated. (Note: the value of 'LLC_startFlag' will be reset to '0' after started)
3. With a 10A constant current load, gradually increase power supply input to 100-V. The output should go up to about 3V. The expression window should be close to Figure ???. `LLC_iSec_PH1_Amps`, `LLC_iSec_PH2_Amps`, `LLC_iSec_Amps`, `LLC_vPri_Volts`, and `LLC_vSec_Volts` should correctly reflect the real parameters. If this is the case, proceed to the next step. If this is not the case, an option is to debug the sensing circuit for the parameters that do not update correctly.
4. Gradually increase *input voltage* to 380V. The output voltage should increase to about 11.5 V. It is now safe to experiment with different `LLC_periodSetSlewed_pu`, `LLC_dutySet_PH1_pu` / `LLC_dutySet_PH2_pu`, `LLC_PWMdbRedPri`, `LLC_PWMdbFedPri`, and `LLC_phaseshift_pu`
5. You may also experiment with different load values and different input voltage values. Please make sure that the board is never operated out of specifications.
6. Shut down the input power supply to end build 1 test.

3. Lab 2: closed loop check

The objective of this build is to:

- Evaluate the closed loop LLC operation

3.1. Build and Load the Project

1. Open `llc_settings.h`, and change settings to `#define LLC_LAB 2`
2. Change `LLC_VSEC_REF_VOLTS` in `settings.h` to 12 (referring to 12V), if it is not the value.
3. Rebuild the project. Click on the Debug button or click Run → Debug. The build 2 code should load.

3.2. Run the Code

4. Run the code by using the <F8> key, or by using the Run button on the toolbar.
5. Turn on DC power supply with 400V, and add a 10A constant current load.
6. In the expression window, input '1' to the 'LLC_startFlag', and then the power stage will softstart to 12V.
7. It is now safe to experiment with different load values and different input voltage values. Also note that the input voltage should be within 350V to 420V.

8. *Note*: Reduce the load **to less than 10A** before shutdown the input DC source, to avoid inrush current during shutdown, which is harm to the board.