

TI Designs

Two-Phase Embedded Metering Firmware Upgrade



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-2PHASE-SUBMTR-FW2](#)

Tool Folder Containing
Design Files

[MSP430i2041](#)

Product Folder



[ASK Our E2E Experts](#)
[WEBENCH® Calculator Tools](#)

Design Features

- Adds Functionality and Flexibility to Original Firmware on TIDM-2PHASE-SUBMTR
 - I²C Communication Capability in Parallel With Universal Asynchronous Receiver/Transmitter (UART) Communication
 - Fundamental Voltage, Current, and Power and Voltage; Current Total Harmonic Distortion (THD) Measurement of Voltage and Current
 - Flexible Assignment of Analog-to-Digital Converter (ADC) Channels in Application Level Without Touching the Metrology Library

Featured Applications

- Home Appliances
- Power Supplies
- Home Security and Automation



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

1 System Overview

1.1 Cautions and Warnings



CAUTION

Read the user guide before use.



WARNING

HIGH VOLTAGE: Electric shock is possible when connecting a board to a live wire. The board should be handled with care by a professional. For safety, TI recommends the use of isolated equipment with overvoltage and overcurrent protection.

This firmware upgrade is designed to run on the TIDM-2PHASESUBMTR hardware. Please read the TIDM-2PHASESUBMTR design guide before running the code on the corresponding hardware.

1.2 System Description

This document discuss the firmware upgrade to the original two-phase power measurement application (TIDM-2PHASE-SUBMTR) using a simple, low-cost MSP430i2041 microcontroller (MCU) from Texas Instruments. This design includes enhancements to the metrology library, which has measurements of fundamental voltage, fundamental current, and fundamental power as well as the total harmonic distortion for voltage and current. The design also modifies how the ADC is accessed so that the assignment of ADC channels for different voltage and current measurements can be done at the application level, instead of fixing these values at the library level. This design also modifies the communication architecture so that both Inter-Integrated Circuit (I²C) and UART are supported and can operate simultaneously. This design guide provides detailed descriptions of these features in later sections.

The TIDM-2PHASE-SUBMTR-FW2 original firmware includes features that were not considered during the design of the TIDM-2PHASE-SUBMTR [1] and corresponding firmware 1.0. The firmware 2.0 design implements several enhancements.

The purpose of the enhancement to the communication sub-system is to support the communication between connected units so that the measurement of more than two sets of voltage and current channels is possible when wiring several boards with the same hardware are connected. As an example, the I²C (slave mode) is chosen as the target of implementation. The same concept can be applied to easily add or move the support of other communication protocols, such as the serial peripheral interface (SPI).

Enhancements to the ADC driver sub-system allow an application-level change of the assignment of ADC channels that vary from the suggested hardware design without having to access and recompile the metrology library. This new feature may help simplify the PCB layout allowing better noise immunity and accuracy in a user application board.

The third major enhancement is the addition of measurement parameters, which some applications require.

2 Enhancement Details

2.1 Enhanced Communication System

Figure 1 and Figure 2 show the architectural change in the two versions of firmware of the communication sub-system. Notice that the individual communication ports HAL has been separated from the original communication layer to form the ports HAL and an intermediate layer. This change in architecture allows several communication ports to operate simultaneously on the same protocol or each on an individual protocol.

This change takes most parts of the original *emeter-communication.c* and modifies it to *emeter-uart.c*; keeping only the application program interfaces (API) *serial_config* and *serial_write*. These two APIs are then rewritten to perform the dispatch, port configuration, and port writing to the port specified in the API.

The change in *emeter-dlt645.c* is minimal and mainly to cope with the way the communication port is handled in the new architecture.

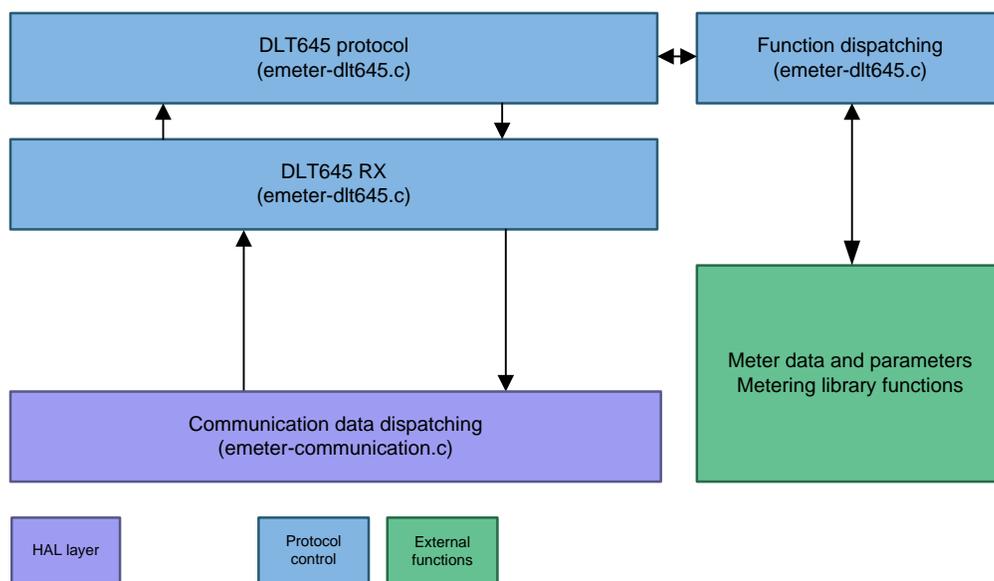


Figure 1. Communication Sub-System Architecture in Firmware 1.0

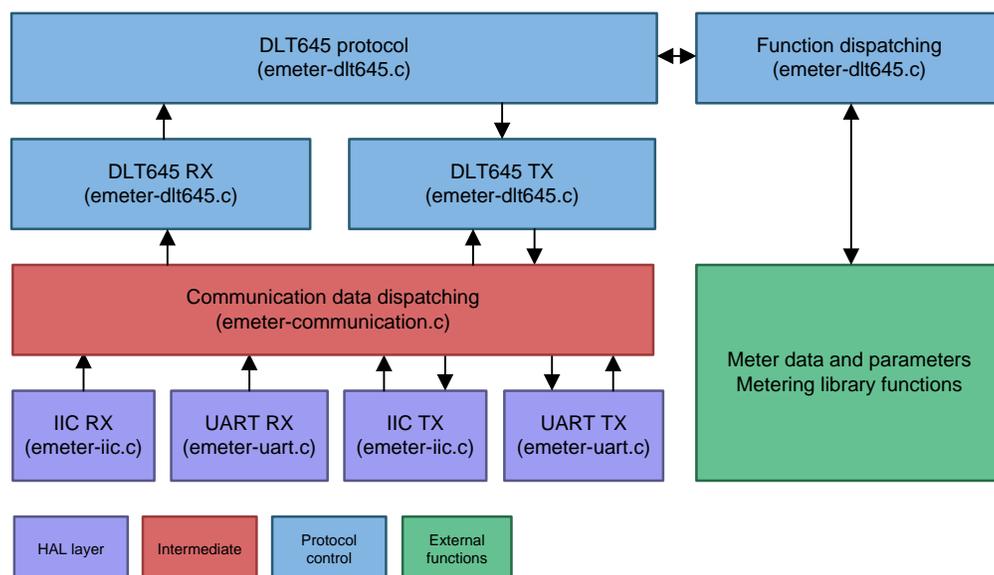


Figure 2. Communication Sub-System Architecture in Firmware 2.0

2.1.1 HAL Implementation

The HAL of the communication sub-system provides a unified view to the upper layer and removes the requirement for hardware specific code in the upper layer.

To keep the HAL for communication in this firmware as simple as possible, only the following APIs are required to provide for the HAL. The “xxx_” below indicates the communication port (for example, iic_).

- xxx_configure: Configure or initialize the communication port
- xxx_write: send data to the communication port
- serial_rx_callback: Callback function to the emeter-communication layer to handle the process when a byte is received
- serial_tx_callback: Callback function to the emeter-communication layer to handle the process when a byte has been transmitted
- serial_comm_abort: Callback function to the emeter-communication layer to abort a communication process in cases where an unrecoverable error has been encountered

In the implementation, the xxx_configure sets the port up with a proper pin configuration, register setting for the specified bit rate, and register setting for the interrupt, if necessary. The xxx_write performs the setup of necessary pointers and counters for data transmission and then enables the interrupt, if necessary. If a port interrupt is enabled, the HAL must also include the handling of interrupts generated and any error conditions encountered.

The *emeter-communication.c* is an intermediate layer to further abstract the HAL, creating a unified view of the upper layer. The *emeter-communication.c* is responsible for the following:

- Receives the call to configure/initialize all the communication ports involved and call to the HAL layer individually for performing the configuration/initialization.
- Receives the call to write data to a specific communication port.
- Transfer callback functions from HAL back to the upper layer for unified data byte transmit and receive handling.

When adding a HAL for another communication port, the user must also modify the *emeter-communication.c* and *emeter-communication.h* files to handle the additional port.

2.2 Enhanced ADC Channel Assignment Scheme

Figure 3 and Figure 4 show the architectural change in the two versions of firmware in the ADC driver sub-system. In the firmware 1.0 the assignment of the ADC to different current and voltage channel sensors is fixed in the metrology library at compile time. The problem with this firmware version is that a different library version is required to assign the ADC differently. This firmware update adds a layer called ADC assignment matrix to improve this version and further isolate the metrology library from user applications.

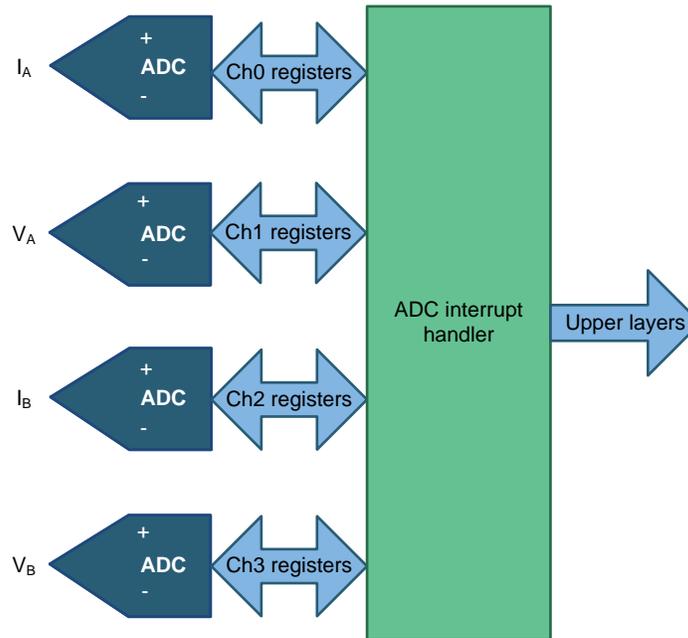


Figure 3. ADC Driver Sub-System Architecture in Firmware 1.0

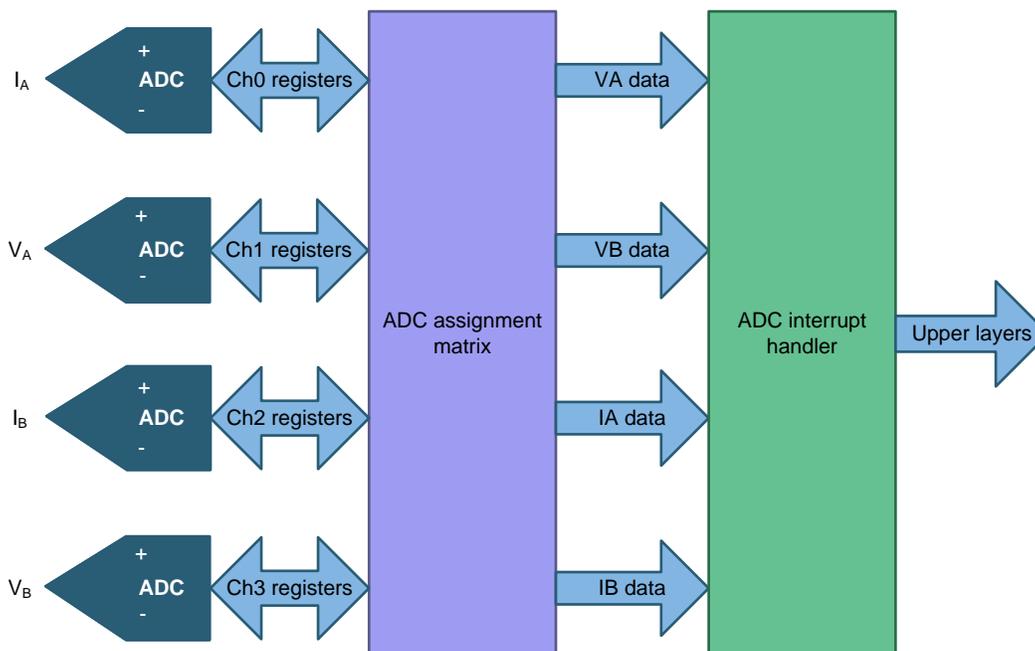


Figure 4. ADC Driver Sub-System Architecture in Firmware 2.0

With the addition of the ADC assignment matrix, the metrology computation no longer accesses the ADC through direct register access, but rather accesses the ADC abstracted as voltage channels and current channels. The assignment of an abstract channel to a physical channel is stored in an array, and the physical channel is accessed by using the abstract channel as an index into this array. When the physical channel is identified, the physical address for ADC access transmits from the matrix named SD24_Channel as defined in the *emeter-sd24.c* file (in the application layer).

The *emeter-sd24.c* file is the HAL for the SD24 ADC. The HAL includes the following APIs:

- SD24_Channel matrix: This matrix consists of the address of registers for accessing individual channels in SD24
- Function SD24_Init: This is function calls to initialize the SD24 to the proper clock and reference voltage
- Function ADC32_OSR256, ADC32_OSR128, and ADC32_OSR64: These are the functions called to get the 24-bit result for different OSR settings (the metrology library for the TIDM-2PHASE-SUBMTR device and this TI design fix the OSR to 256)

With the new architecture changing the channel assignment is simple. Channel assignment is defined in the *metrology-calibration-defaults.c* file and the assignment is defined in the array VoltageADCAssign and CurrentADCAssign. For example, in the following code the voltage channel 0 is using ADC channel 1, voltage channel 1 is using ADC channel 0, current channel 0 is using ADC channel 3, and current channel 1 is using ADC channel 2. The length of these two arrays is not important as long as it covers the required number of channels for voltage and current.

```
const uint16_t VoltageADCAssign[] = {1, 0, 0, 0, 0, 0, 0};
const uint16_t CurrentADCAssign[] = {3, 2, 0, 0, 0, 0, 0};
```

2.3 Enhanced Measurements

Firmware 1.0 for the TIDM-2PHASE-SUBMTR provides the most basic measurement required for nearly every application. The firmware 2.0 provides additional functionalities with the available CPU bandwidth. [Figure 5](#) shows the software data flow for the calculation of the measured parameters for a phase. For each phase each sample undergoes the process to yield the measurement parameters. The next subsection summarizes the measurements provided and equations used for computation in firmware 1.0, and the additional functionality provided in firmware 2.0.

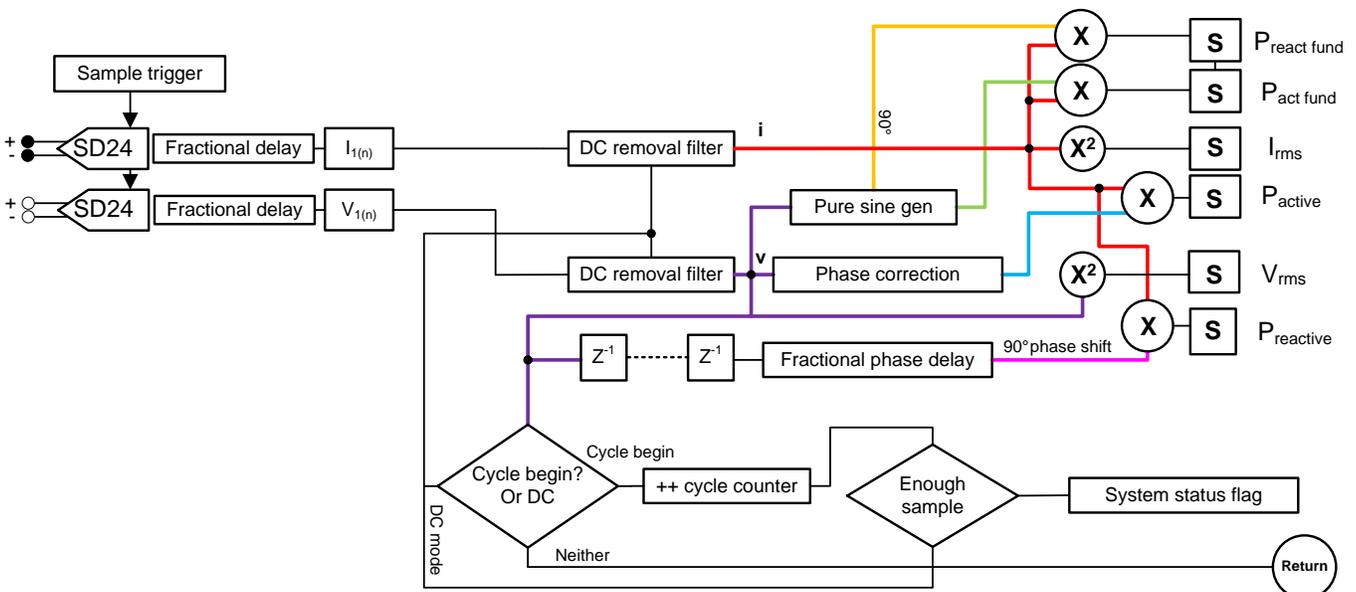


Figure 5. Software Data Flow of a Phase

2.3.1 Measurements Provided in Firmware 1.0

$$V_{\text{RMS}} = \text{VGAIN} \times \sqrt{\frac{1}{N} \sum_{i=1}^N V_{\text{samp}}(i) \times V_{\text{samp}}(i)}$$

$$I_{\text{RMS}} = \text{IGAIN} \times \sqrt{\frac{1}{N} \sum_{i=1}^N I_{\text{samp}}(i) \times I_{\text{samp}}(i)}$$

$$P_{\text{active}} = \text{PGAIN} \times \frac{1}{N} \sum_{i=1}^N V_{\text{samp}}(i) \times I_{\text{samp}}(i)$$

$$P_{\text{reactive}} = \text{PGAIN} \times \frac{1}{N} \sum_{i=1}^N V_{\text{samp},90}(i) \times I_{\text{samp}}(i)$$

$$P_{\text{apparent}} = \sqrt{P_{\text{active}}^2 + P_{\text{reactive}}^2}$$

$$\text{PF} = \cos\phi = \frac{P_{\text{active}}}{P_{\text{apparent}}}$$

2.3.2 Additional Measurements Provided in Firmware 2.0

$$V_{\text{RMS_fund}} = \text{VGAIN} \times \sqrt{\frac{1}{N} \sum_i V_{\text{samp}}(i) \times V_{\text{pure}}(i)}$$

$$P_{\text{active_fund}} = \text{PGAIN} \times \frac{1}{N} \sum_i I_{\text{samp}}(i) \times V_{\text{pure}}(i)$$

$$P_{\text{reactive_fund}} = \text{PGAIN} \times \frac{1}{N} \sum_i I_{\text{samp}}(i) \times V_{\text{pure}(\pi/2)}(i)$$

$$V_{\text{THD}} = \frac{\sqrt{V_{\text{RMS}}^2 - V_{\text{RMS_fund}}^2}}{V_{\text{RMS_fund}}}$$

$$I_{\text{RMS_fund}} = \frac{\sqrt{P_{\text{active_fund}}^2 + P_{\text{reactive_fund}}^2}}{V_{\text{RMS_fund}}}$$

$$I_{\text{THD}} = \frac{\sqrt{I_{\text{RMS}}^2 - I_{\text{RMS_fund}}^2}}{I_{\text{RMS_fund}}}$$

2.4 Other Changes

In addition to all of the major changes previously discussed, there were a few minor changes made to the firmware version 2.0.

The metrology version number is now retrieved by an API call to the metrology library rather than taking the constant defined by the application in *metrology-calibration-defaults.c* and *metrology-calibration-template.h*. This new way of retrieving the metrology version isolates the metrology library from the application and allows the reported version number to remain consistent.

The meter protocol version is now defined in “#define METER_PROTOCOL_VERSION” in the *dlt645-decs.h* file. This definition isolates the communication version from the user application and allows the reported version number to remain consistent.

The meter configuration is now retrieved by an API call to the metrology library rather than composing this configuration within the *emeter-dlt645.c* file. This new way of retrieving the meter configuration isolates the metrology library from the application and allows the reported meter configuration to report the functionality of the metrology library consistently.

The metrology library has a new name “emeter-metrology-i2041-2-phase.r43”, which has changed from “emeter-metrology-i2041.r43” in firmware 1.0.

Firmware 2.0 allows measurement of DC voltage and current up to 190-V DC or up to 380-V DC depending on the configuration set for a two phase measurement. Other operation ratings are the same as for the TIDM-2PHASE-SUBMTR device. Please refer to the design guide of TIDM-2PHASE-SUBMTR [\[1\]](#) for details about the two configurations for two-phase measurement and the operation ratings.

3 Tests

3.1 Accuracy Test

As the computation algorithm and the hardware is the same as in firmware 1.0, the accuracy test is merely testing whether the ADC assignment using the SD24_Channel matrix is correct or not.

3.1.1 Apparatus

TI recommends the following list of instruments to perform the calibration and test.

- AC meter test set (see <http://bit.ly/1CgeVH4>)
- A reference meter capable of giving AC parameter readings based on the voltage, current, and phase setting from the AC meter test set (see <http://bit.ly/1Fgh2LF>)

3.1.2 Setup

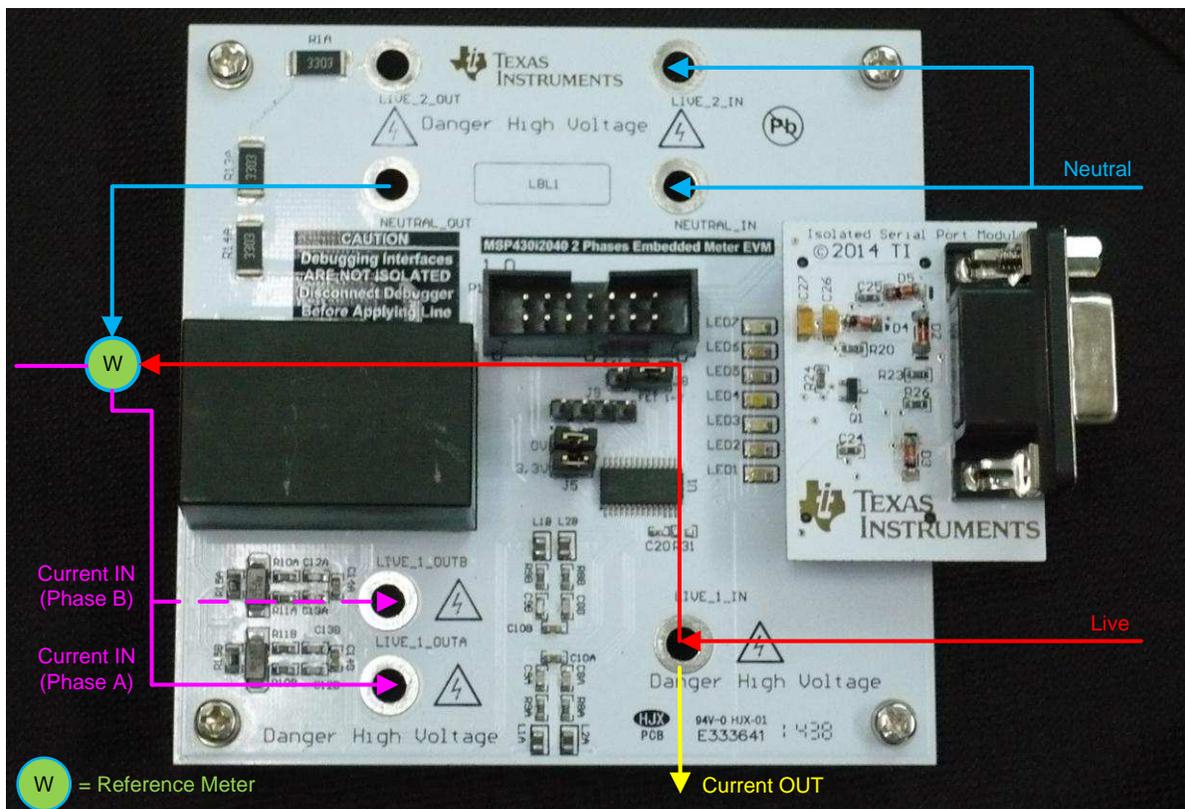


Figure 6. Connection to AC Meter Test Set

3.1.3 Test Results

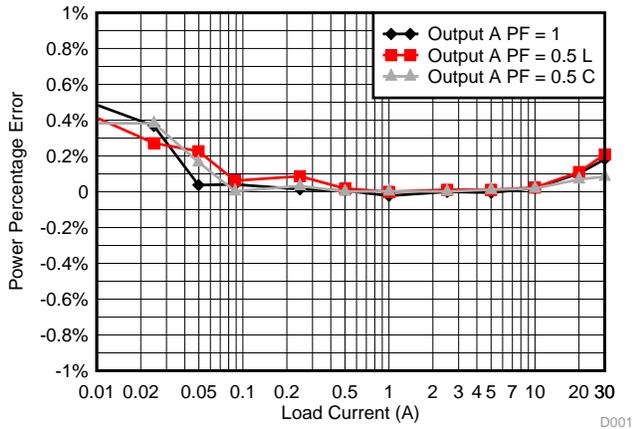


Figure 7. Phase A — Accuracy Test Result

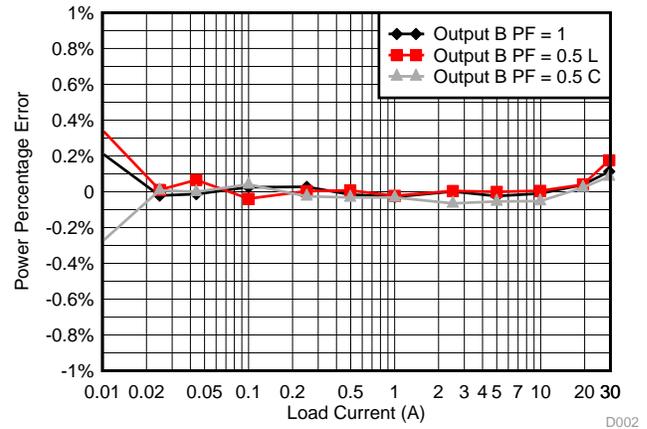


Figure 8. Phase B — Accuracy Test Result

3.1.4 Accuracy Test Summary

When comparing the accuracy of the 1.0 firmware to the 2.0 firmware there is no significant difference in the level of accuracy observed for these parameters.

With the proper test settings and calibration, this design achieves an accuracy of < 0.2% error over a 30-mA to 30-A range, and a < 0.5% error over a 10-mA to 30-A range. The higher percentage error in the low current range is due to noise present on the shunt resistor. The effect of the noise diminishes as the test current increases. The error starts to increase at 20 A due to the heat generated on the shunt, which causes thermal drift to the shunt resistance.

3.2 THD Test

The THD test is conducted using the same generator as the accuracy test with a reference meter that measures THD in current. See the Yokogawa website for the meter used in this test: <http://bit.ly/1aCONil>.

In the test, a 5-A sinusoidal current is applied with an additional odd harmonic selected at approximately 25% in magnitude. The reading of the THD from the reference meter and the unit under the test is recorded. The percentage deviation from the reading of the reference meter is calculated.

Table 1. THD Accuracy Test Result

| HARMONIC | REF METER READING (%) | UUT READING (%) | DIFFERENCE |
|----------|-----------------------|-----------------|------------|
| 3rd | 22.63 | 22.55 | -0.08 |
| 5th | 22.38 | 22.05 | -0.33 |
| 7th | 22.96 | 22.15 | -0.81 |

3.3 Communication Test

The I²C slave communication capability is tested with three units connected together by an I²C host to an RS232 converter. As Figure 9 shows, the test unit has a three-phase voltage applied (120° between phases) and each of the test boards applies two of the three phases. Note that there are more than one power supply units in all of the boards together. As Figure 10 shows, all but one power supply is removed. The VDD and VSS of the boards is connected with the remaining power supply so that the system runs on a single rail.

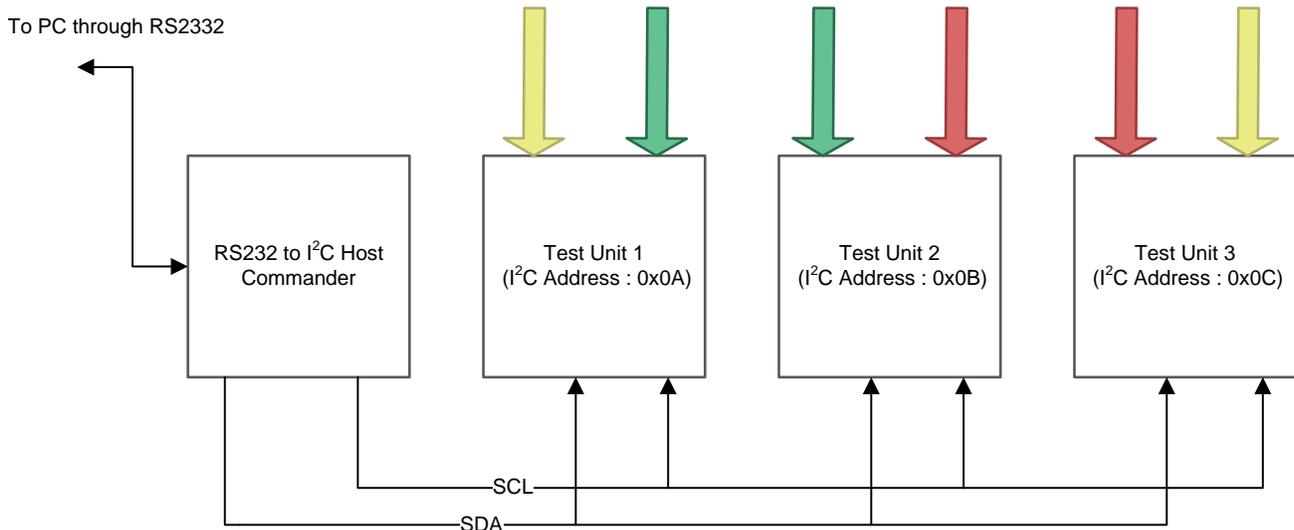


Figure 9. I²C Functional Test Setup

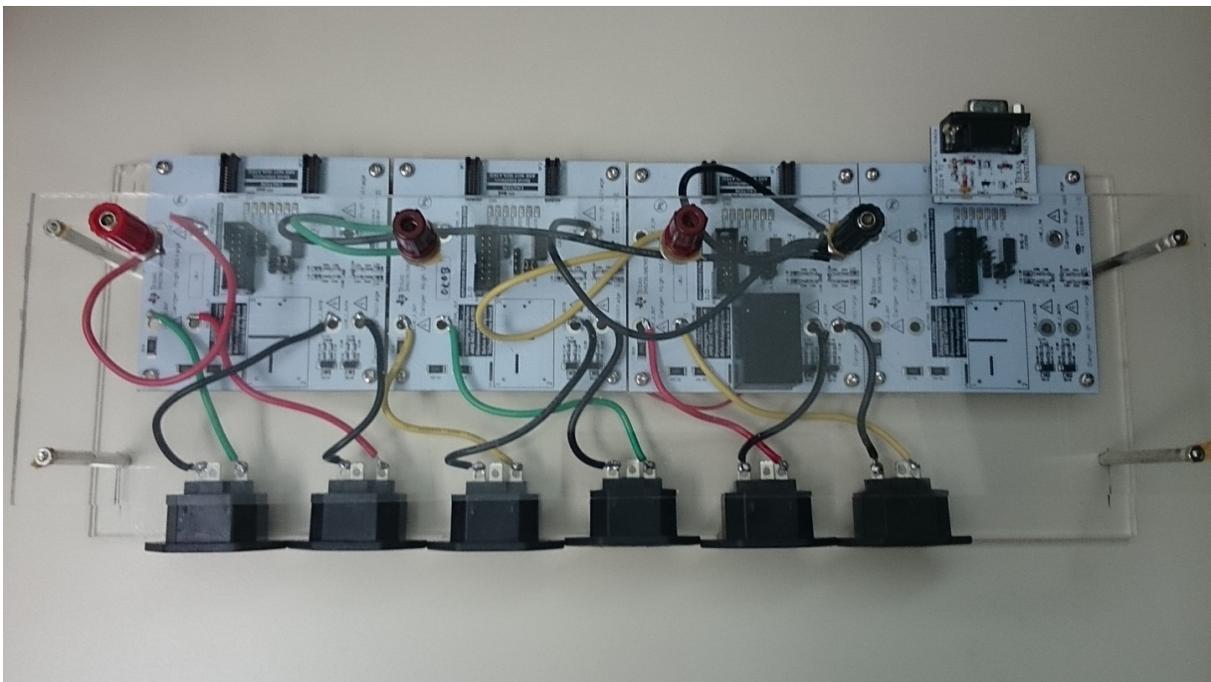


Figure 10. I²C Test Unit

3.4 I²C Functionality Test Result

The test setup is set to run for four hours of continuous reading from each board at a reading interval of 160 ms with no read error reported.

4 Design Files

4.1 Software Files

To download the software files, source code, and the metrology library, see the software files at: [TIDM-2PHASE-SUBMTR-FW2](#).

5 References

1. Texas Instruments, *Two-Phase Embedded Metering*, User's Guide, ([TIDU641](#))
2. Texas Instruments, *MPS430i2xx Family*, User's Guide, ([SLAU335](#))

6 About the Author

MARS LEUNG received his Bachelor of Engineering at Hong Kong Polytechnic University and Master of Science at Chinese University of Hong Kong. He is an experienced field application engineer specialized in MCU application support and development; senior smartcard application engineer specialized in smart card payment system definition and implementation; staff engineer specialized in MCU and new module definition; staff engineer in analog system applications specializing in digital system and video processing of dynamic LED backlight control. He is now a staff engineer in Texas Instruments in the Smartgrid Application Team, which specializes in embedded electricity metering applications.

Appendix A Example Application Code

A.1 Introduction

Project structure:

- *emeter-communication.c* – source code for intermediate level communication interface
- *emeter-dlt645.c* – source code for the polling mode protocol implementation
- *emeter-iic.c* – source code for I²C driver
- *emeter-iic.h* – header code for I²C driver
- *emeter-main.c* – source code for system initialization, main loop, callback functions implementation, and interrupt vector placement
- *emeter-metrology-i2041-2-phase.r43* – embedded metering library object code
- *emeter-setup.c* – source code for low-level system initialization
- *emeter-template.h* – source code for configuration
- *emeter-uart.c* – source code for UART driver
- *emeter-uart.h* – header code for UART driver
- *metrology-calibration-default.c* – source code to put the user defined default calibration parameter into a proper data structure (should only modify the const VoltageADCAssign and CurrentADCAssign for assignment of ADC and must not modify the other part of this file)
- *metrology-calibration-template.h* – source code of user defined default calibration parameter

A.2 RS232 to I²C Master Application

The *emeter-iic-master-i2040* is a standalone application project written to test the I²C functionality of firmware 2.0. The application acts as a bridge to convert the RS232 command and the I²C command to and from three units of the TIDM-2PHASE-SUBMTR device. The result makes the whole connected test setup function as a six-channel metering unit. The application is also designed to run on the same hardware as TIDM-2PHASE-SUBMTR.

A.3 Preparing the Application Code to Run

1. After launching the IAR Embedded Workbench® IDE Version 5.5, click on *File* → *Open* → *Workspace*
2. Select *emeters.eww* when prompted to open *Workspace*.

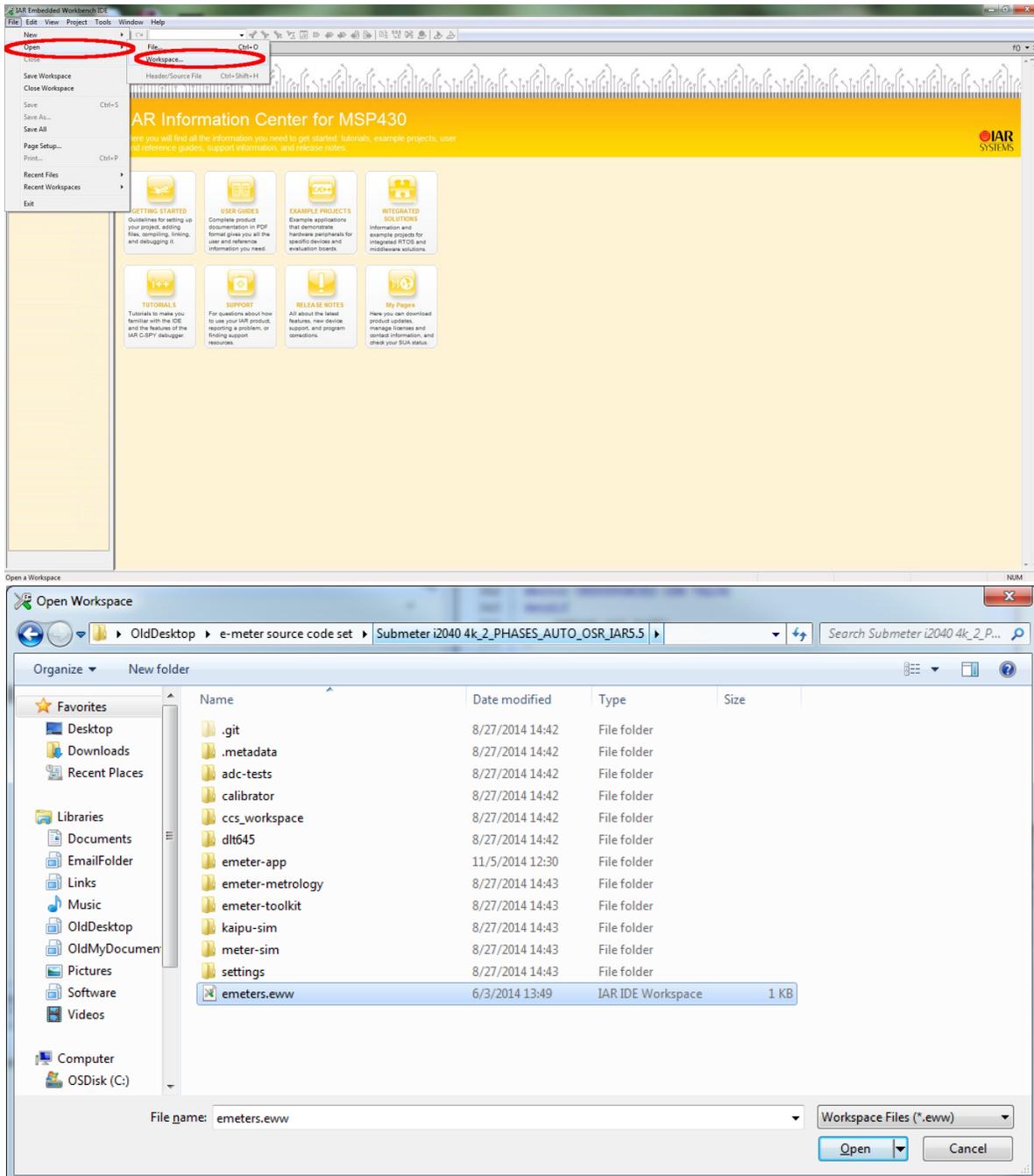


Figure 11. Opening Workspace

3. Select the *emeter-app-i2041* project tab at the bottom of the *Workspace* window.

4. Check project options by right clicking the project name and select *Options...* from the pop-up menu (see [Figure 12](#)).

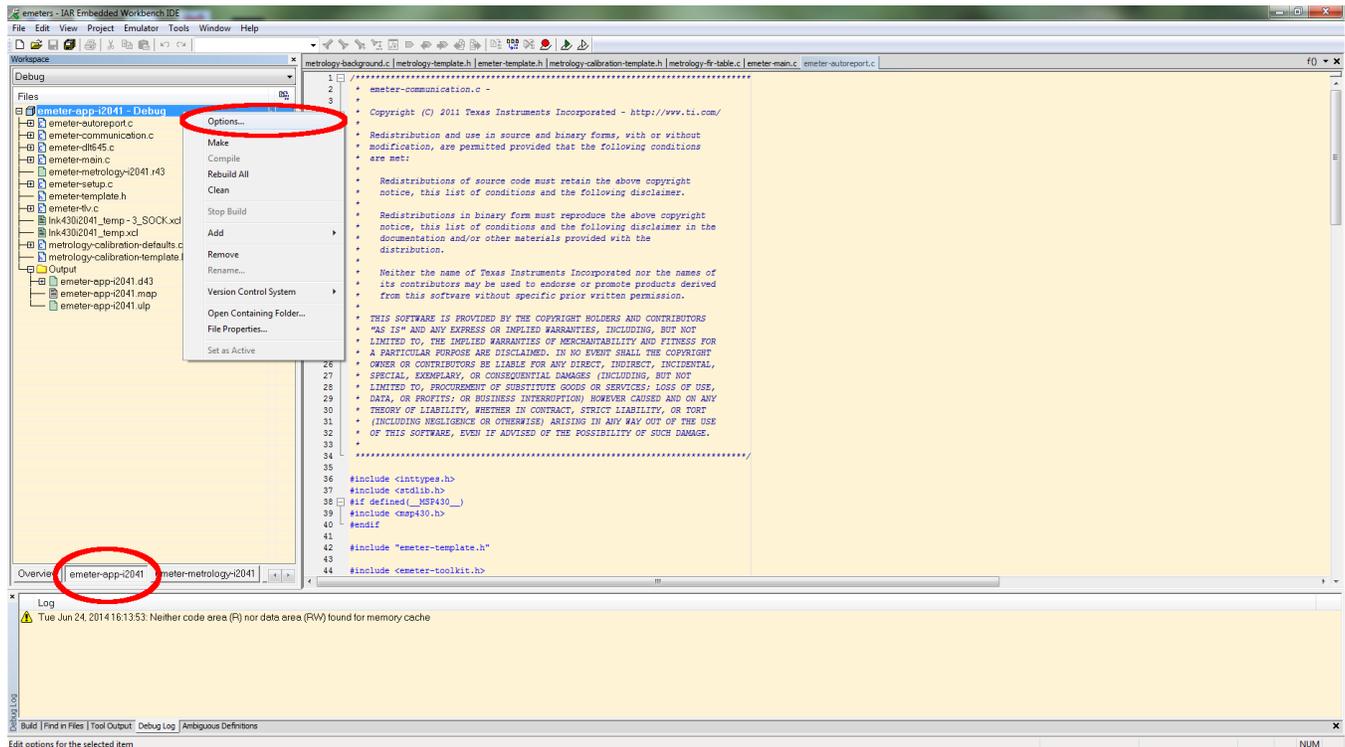


Figure 12. Project Tab

5. When the options appear, select *C/C++ Compiler* on the left-hand column. Then select the *Optimizations* tab on the right-hand side and check the optimization settings as shown in [Figure 13](#).

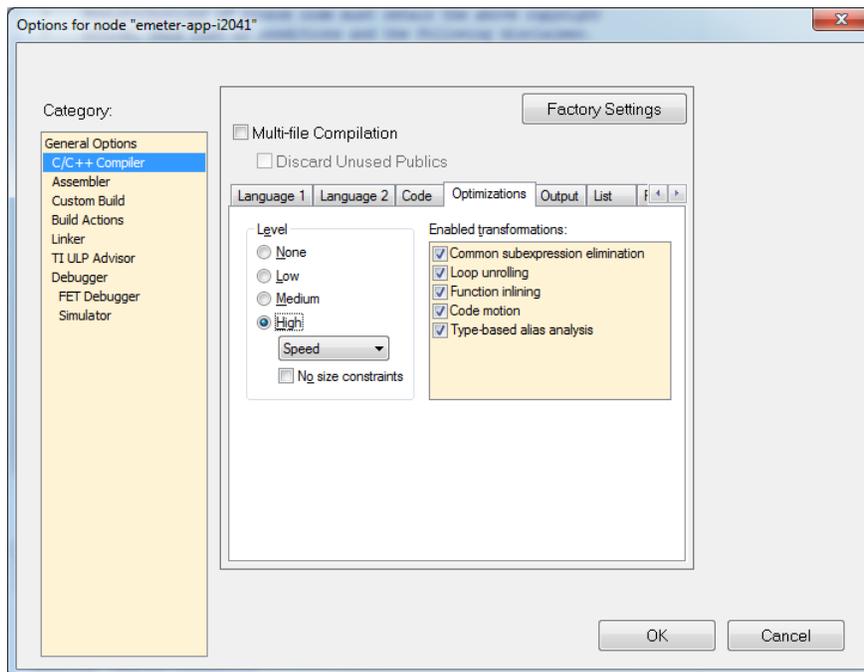
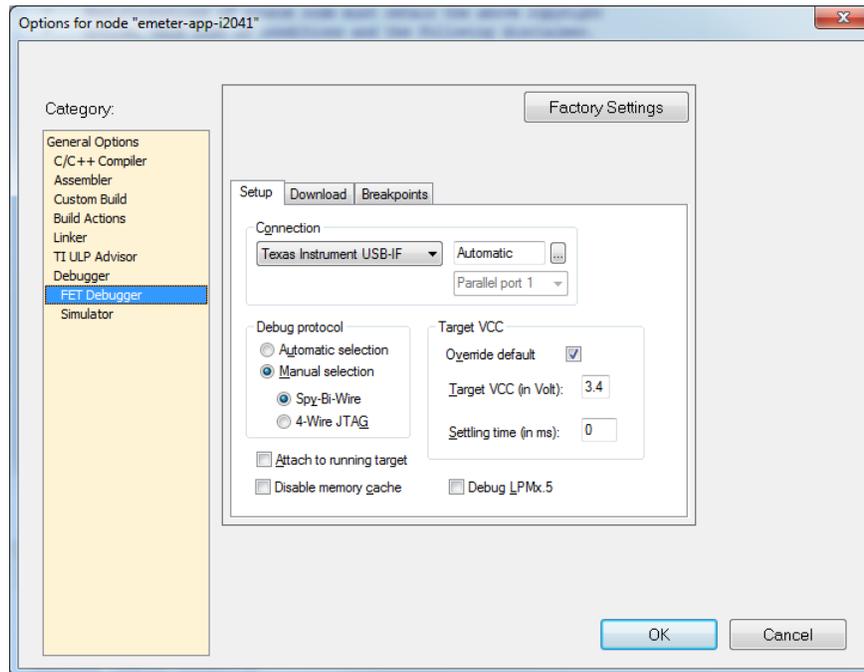
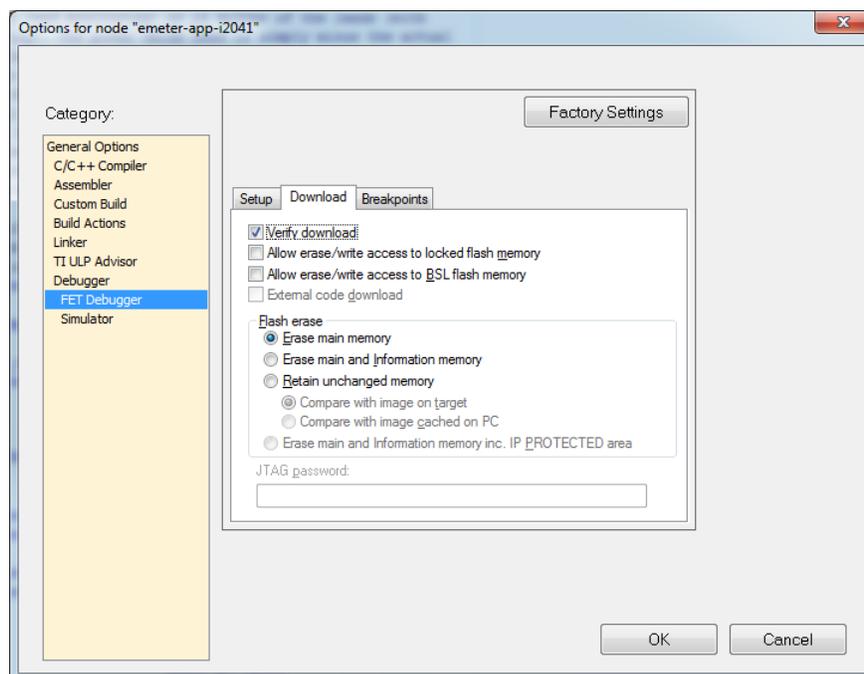


Figure 13. Optimization Options

6. Select *FET Debugger* on the left-hand column, then select the *Setup* tab. The EVM uses *Spy-Bi-Wire* for its code downloading and debugging. Check to make sure the options are as shown in [Figure 14](#).


Figure 14. Debugger Options

7. Select the *Download* tab. Under *Flash erase*, do not choose *Erase main memory and Information memory*, this option erases both sets of data and cannot be recovered. Instead, choose *Erase main memory* as the download option to preserve these factory parameters: system clock calibration, ADC calibration, and internal reference calibration (see [Figure 15](#)). However, metrology calibration stored in the main memory, such as VGAIN, IGAIN, PGAIN, and so on, are always erased after downloading.


Figure 15. Download Options

8. Click *OK* after completing all of the changes.

- Rebuild the project by right-clicking on the project and select *Rebuild All* from the pop-up menu (see [Figure 16](#)). Three warnings will be reported during rebuilding (see [Figure 17](#)), which are safe to ignore. To open the project workspace and modify, compile, and download the code, the user must have IAR Embedded Workbench 5.5 installed with a valid license. If a valid license is not available, the user can still download the object code. See [Section A.4](#) for downloading procedures.

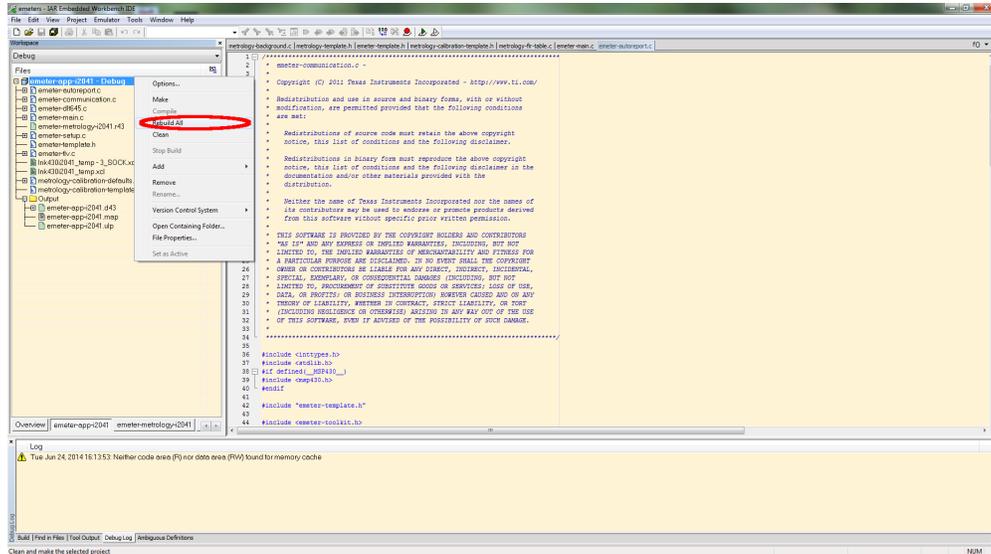


Figure 16. Compiling the Application

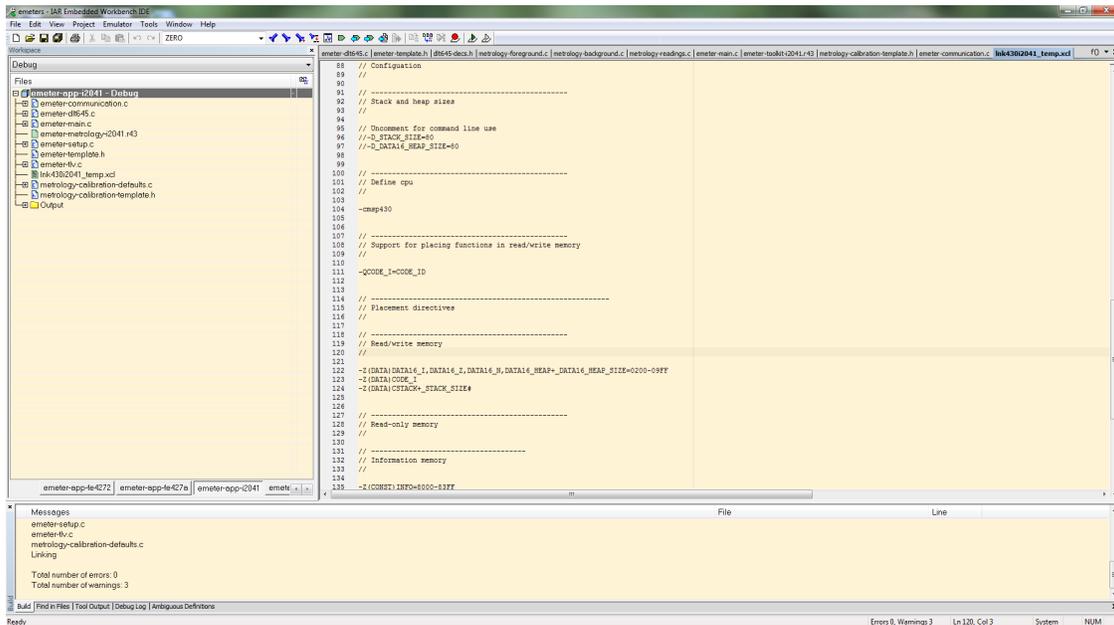


Figure 17. Warnings

10. Make sure the jumper on J8 is short properly and the the jumpers on J5 are open. Connect the 14-pin connector P1 to MSP-FET430UIF by a flat cable as [Figure 18](#) shows.



CAUTION

The debugging interface is NOT ISOLATED. Make sure to properly isolate between the EVM and the PC used for debugging with the AC or DC high voltage connected.

NOTE: Connection to debugging interface is optional for the operation of the EVM. The EVM can operate without a debugger connected.

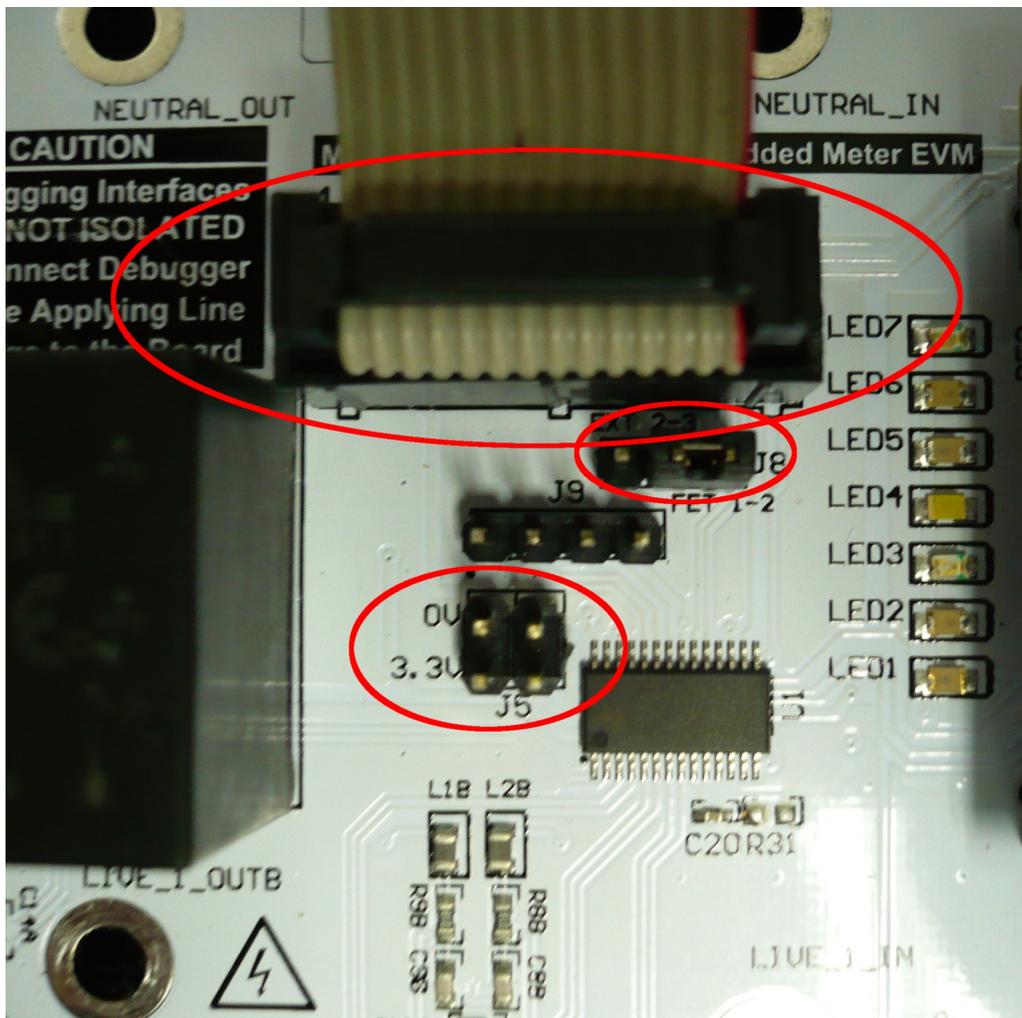


Figure 18. Connecting EVM and FET

11. Click the *Download and Debug* button to download and debug (see [Figure 19](#)).

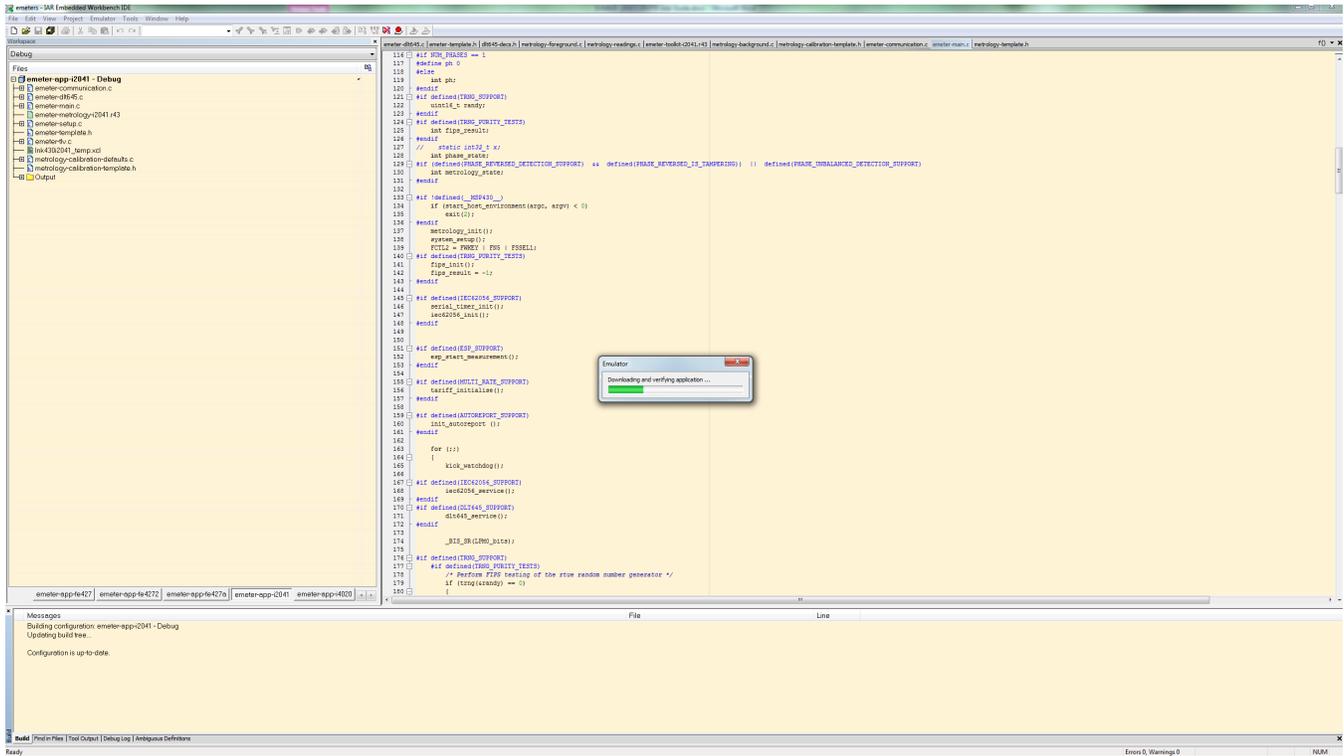


Figure 19. Code Downloading

12. After successfully completing the download, [Figure 20](#) appears. Click the *Go* button to run the application.

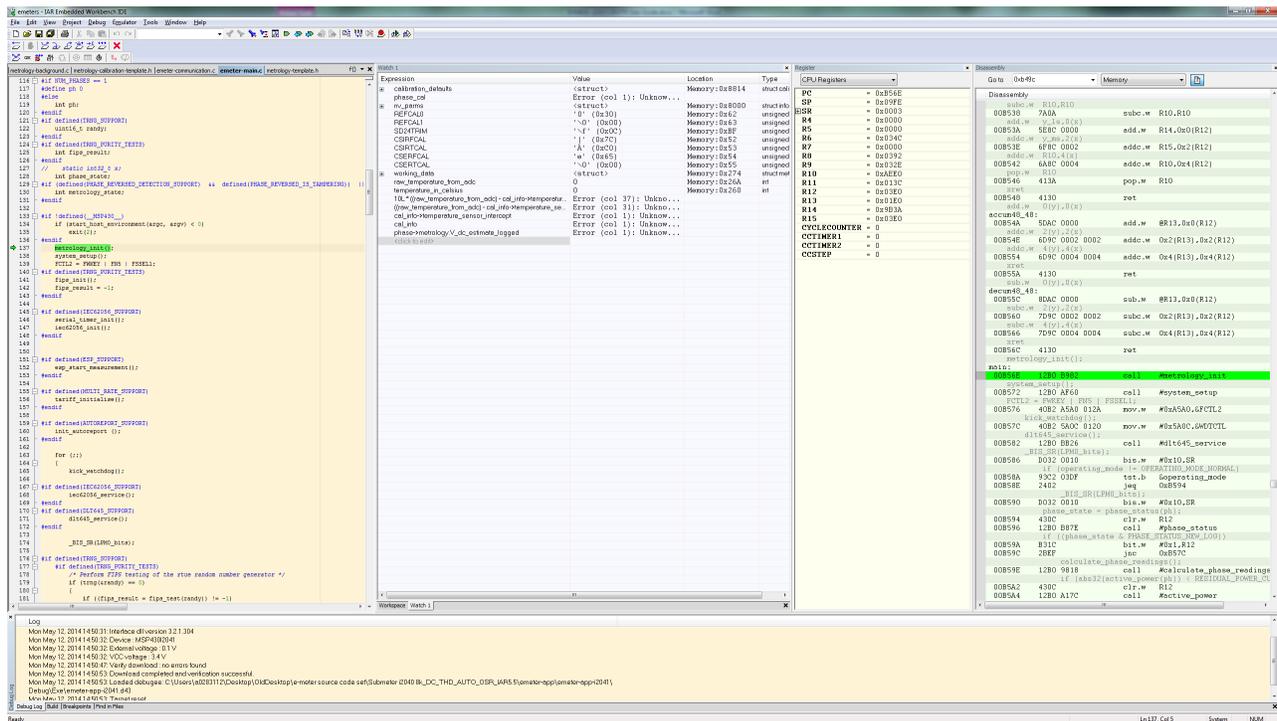


Figure 20. Debugger Screen

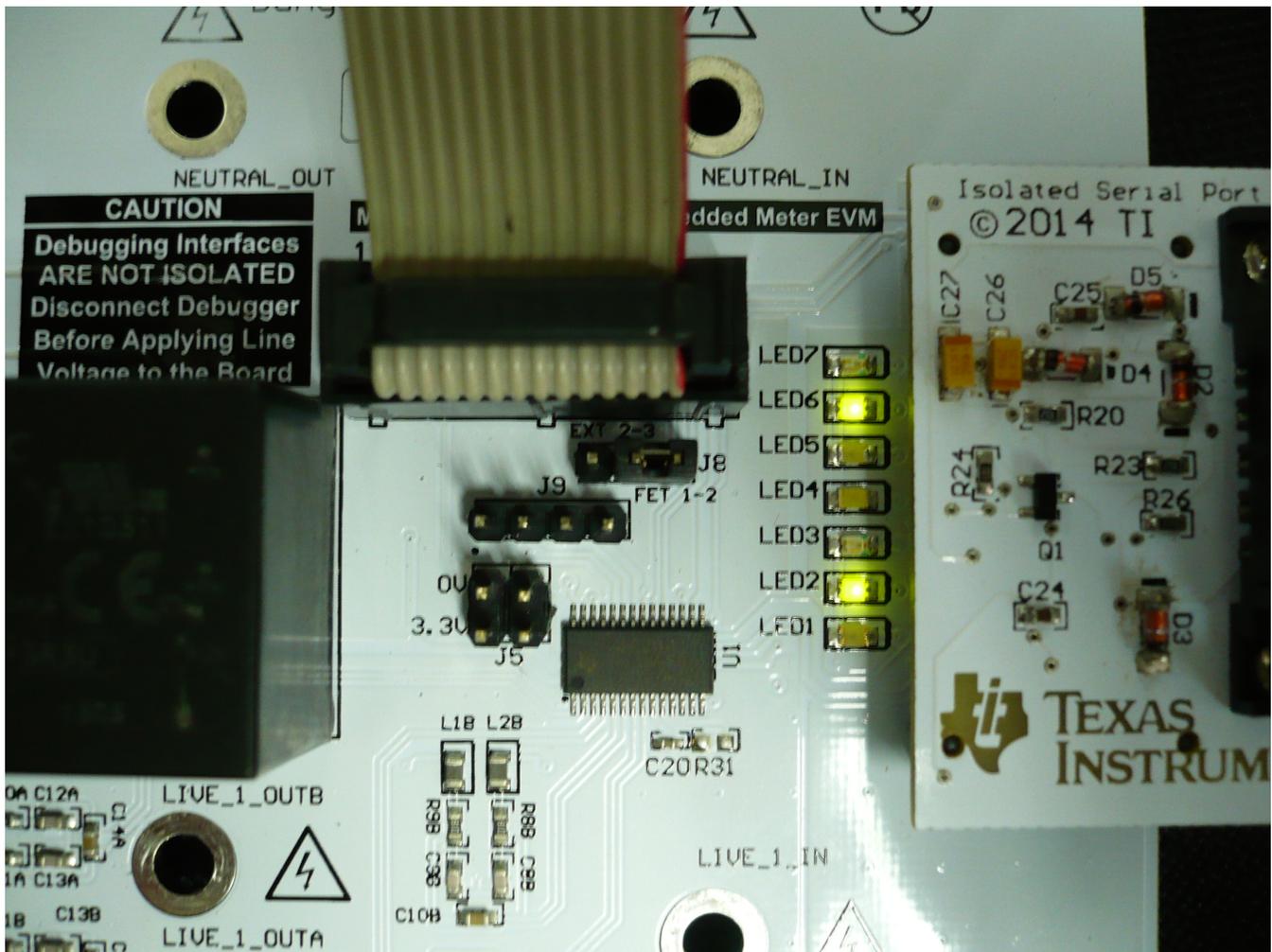


Figure 21. EVM Running

A.4 Downloading Without an IAR License

If a valid IAR Embedded Workbench 5.5 license is not available, download the executable code to the board with the following steps using the installed IAR Embedded Workbench 5.5.

1. Open the project workspace as described in [Section A.3, Steps 1 through 7](#). Then connect the board to the MSP-FET430UIF as described in [Step 10](#) in [Section A.3](#).
2. Select *Project*→*Download*→*Download File...* from the menu (see [Figure 22](#)).
3. When prompted to select a file, go to the folder `[Submeter i2040 4k_2_PHASES_AUTO_OSR_IAR5.5]\emeter-app\emeter-app-i2041\Debug\Exe` and select the file named “emeter-app-i2041.d43” ([Figure 23](#)).

The executable code then downloads to the board.

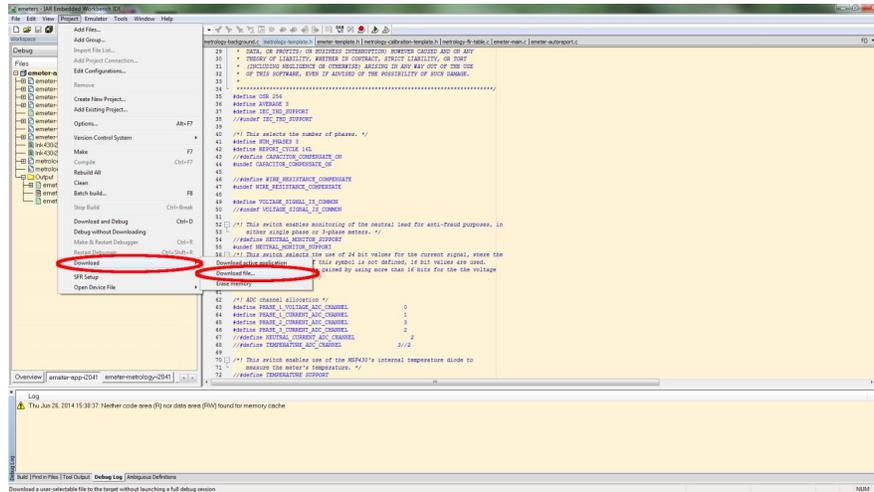


Figure 22. Download Executable File

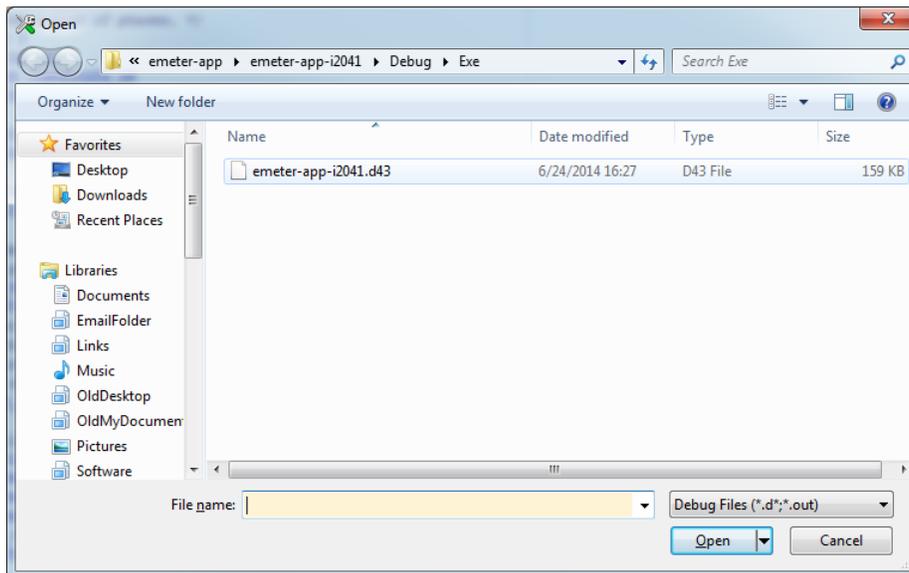


Figure 23. Select File to Download

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.