Fusion Digital Power Designer GUI for Isolated Power Applications

User Guide (for UCD3138, UCD3138A, UCD3138064, UCD3138A64, UCD3138128 applications)

1 Contents

1.1 Table of Contents

1.2 Figures

2 About This User's Guide

2.1 Introduction

The Fusion Digital Power Designer is a GUI tool that supports the evaluation and development of power supply solutions based on Texas Instruments' UCD3xxx and UCD9xxx family of digital power controllers.

This User Guide specifically addresses the following Texas Instrument controllers geared towards Isolated Power applications:

- UCD3138
- UCD3138064
- UCD3138A64

This tool is available for free download here: http://www.ti.com/fusion-gui.

There are a number of tools available upon installation. This user guide will focus on describing the functions of two important tools namely, the Fusion Digital Power Designer (Designer GUI) and the Device GUI. These two essential GUIs, in addition to providing key functionality, serve as a launch-pad to many of the other tools provided.

2.2 The Fusion Digital Power Designer (Designer GUI)

The Fusion Digital Power Designer or Designer GUI, essentially emulates a Host in a PMBUS-based power supply system (pmbus.org). If PMBUS commands are implemented in the firmware of the device under test, then the Designer GUI aids in establishing communication and delivering the supported PMBUS functions (such as telemetry etc). Additionally, when used in conjunction with TI-provided reference firmware, the Designer GUI provides certain additional capabilities related to optimizing the power supply such as adjusting loop compensation etc. Currently TI provides reference firmware for the 4 isolated power topologies listed in table below, which are supported by the Designer GUI and can be used in conjunction with associated EVMs available for purchase from www.ti.com:

Power Supply Topology	EVM Part #
Power Factor Correction	UCD3138PFCEVM-026
Phase Shifted Full Bridge	UCD3138PSFBEVM-027
Half-Bridge Resonant LLC	UCD3138LLCEVM-028
Hard Switching Full Bridge	UCD3138HSFBEVM-029

2.3 The Device GUI or Engineering GUI

The Device GUI or Engineering GUI, is a launch-pad for several invaluable device-related tools that are necessary for working with the UCD3138 (064, A64) devices and developing successful firmware. These tools allow the designer to execute critical tasks associated with the devices during the development phase such as switching between ROM mode and Program Flash mode, downloading firmware, debugging, investigating the contents of registers etc.

A PMBUS-based hardware interface, which allows communication between the GUI tool and the UCD3138 (064, A64) devices, is available from Texas Instruments (part #: USB-to-GPIO, http://www.ti.com/tool/usb-to-gpio). One unit of this interface adaptor is provided with the abovementioned EVMs, but the adaptor is available for stand-alone purchase, for use with other UCD3138 (064, A64) EVMs available from Texas Instruments that are not provided with one:

EVM Part #	Description
UCD3138CC64EVM-030	Control card featuring UCD3138RGC

UCD3138OL64EVM-031	Open Loop Evaluation board for UCD3138RGC
UCD3138OL40EVM-032	Open Loop Evaluation board for UCD3138RHA
UCD3138064EVM-166	Control card featuring UCD3138064RGC
UCD3138A64CEVM-660	Control card featuring UCD3138A64PFC

The reference firmware, EVMs, GUI and interface adaptor constitute a complete and powerful development system that is available for designers to successfully develop power supplies based on UCD3138 (064, A64).

2.4 Conventions

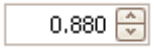
Any hexadecimal number will be prefixed by 0x. For example, 0xFF. Any other number should be assumed to be decimal.

2.5 User Interface Terminology & Tips

Checkbox
User can select any number of boxes.

Radio Button
User can only select one of the circles at a time. For example, clicking “High” will deselect “None.”

Spin Edit
Used for numeric entry. User can type in a number directly or click the up and down arrows to increment or decrement the number. The up/down usually changes the last decimal place (adding or subtracting 0.001 in this example).



Widget
A generic term used to describe a user interface component such as a button or checkbox.

Disabled (Grayed Out)
User cannot edit the widget. This is usually because the GUI has determined that a particular item is a “don’t care” or does not make sense given the setting of some other widget or PMBus command.

2.6 Terminology

‘Designer GUI’ or ‘GUI’ – refers to Fusion Digital Power Designer GUI (main tool), described above

‘Device GUI’ or ‘Engineering GUI’ – refers to UCD3xxx Device GUI that delivers device-related functions indispensable for development purposes

2.7 Additional Technical Support

For additional questions or clarifications please take advantage of TI’s E2E community:
http://e2e.ti.com/support/power_management/digital_power/default.aspx

Alternately, please contact your Texas Instruments local representative.

3 Getting Started

3.1 PC Requirements

The GUI requires the following:

- A PC running Windows XP/Windows 7
- Microsoft.NET Framework version 4.0

Microsoft.NET is the runtime application framework that the GUI uses. The GUI's installer will ensure version 4.0 of .NET is installed, and install if necessary.

3.2 USB Adapter

As mentioned earlier, the EVM is attached to the PC through a Texas Instruments serial bus adapter, part number USB-to-GPIO[i]. The user should have received this adapter with certain EVMs, but is also orderable stand-alone. The serial adapter must be running firmware v. 1.0.5 or higher. If the adapter's firmware does not meet this requirement, a warning message will appear when the GUI first starts. The GUI can be run in "Offline mode" without the serial bus adapter, which allows the user to edit an existing device configuration or experiment with a default "virtual device."

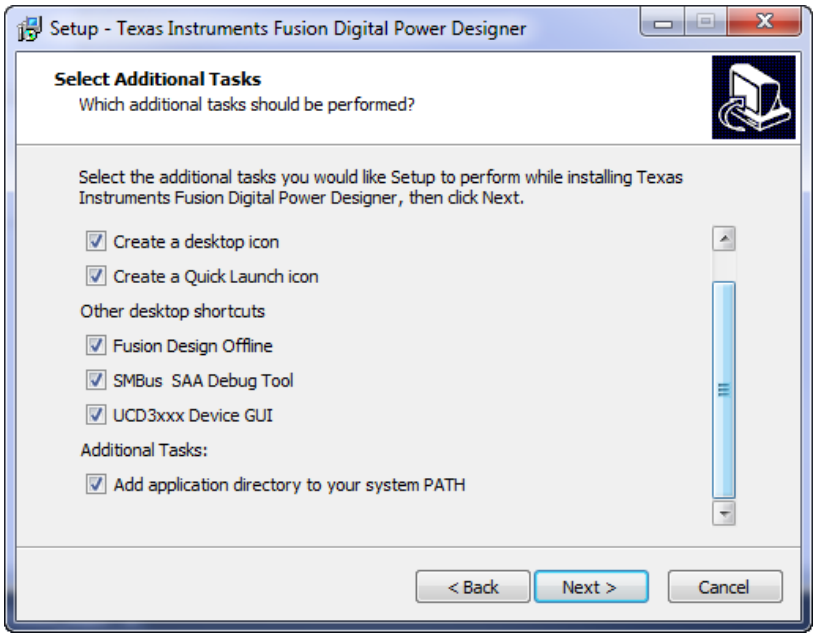
3.3 Download & Installation

The latest public production versions can be found at http://www.ti.com/tool/fusion_digital_power_designer. In addition to what is found at that address, your TI representative may provide you with more recent releases that are not available from the website mentioned.

If you would like to be added to our release mailing list for Isolated GUI builds send an email to: iso-fusion-gui-releases.owner@list.ti.com.

Download the ZIP file to your hard drive. You do not need to unzip the ZIP: you can launch the installer from within WinZip or similar ZIP utility.

The following figure displays some extra tools you can create shortcuts for in addition to the main Fusion Digital Power Designer GUI.



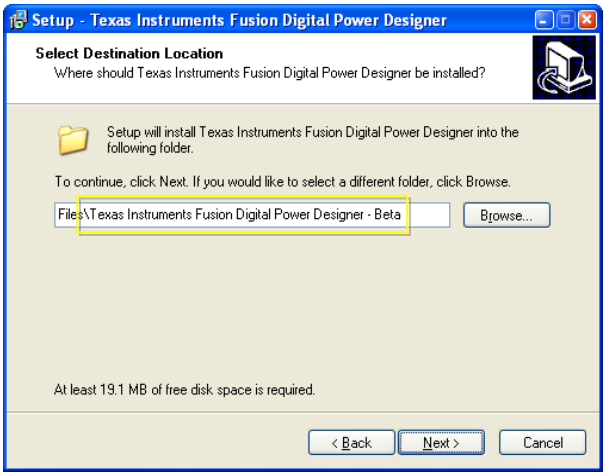
3.4 Upgrading the GUI

When upgrading to a new release of the GUI, there is no need to un-install the current installed version first. In fact, doing so will remove your program preferences, and is not recommended. The GUI installer will take care of updating all necessary files. The program preferences will not be modified by the installer.

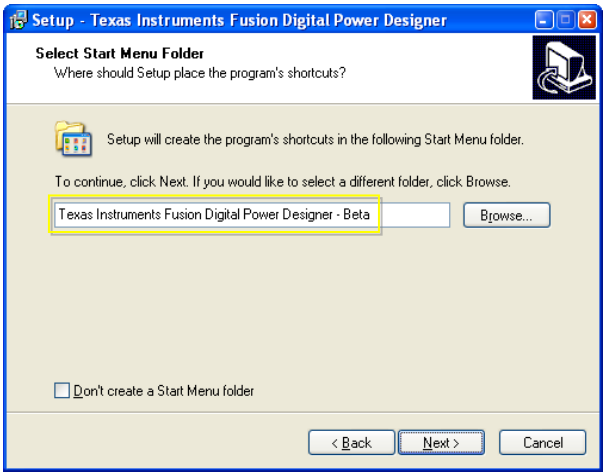
3.5 Multiple Installations of the GUI

You *can* install different versions of the GUI on same the PC. Because the preferences are stored within the program folder as described in Section 6.1.1.1, each version of the GUI installed on your PC will have its own set of preferences.

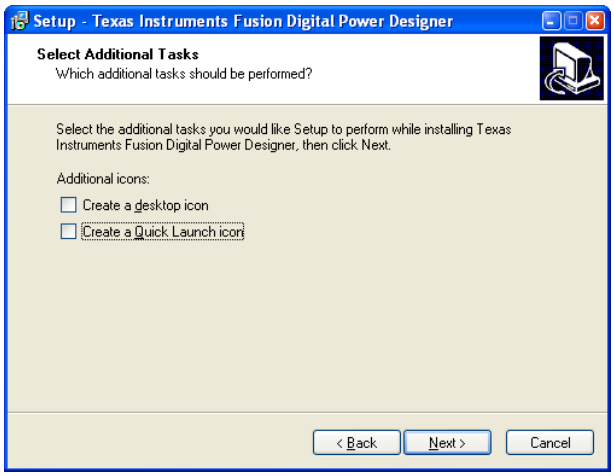
When you install a second copy of the GUI, you need to ensure the name of the folder for the additional copy is named differently from the default folder name, “Texas Instruments Fusion Digital Power Designer.” The easiest way to do this is to append something descriptive to the folder name. For example, in the following example “ – Beta” was appended to the installation folder pathname:



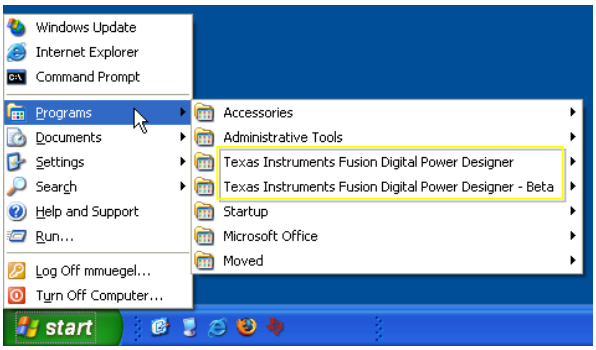
You will also need to rename the Start Menu folder that gets created. Again, “ – Beta” has been appended to the default:



Finally, you'll need to decide whether you want to install desktop or quick launch shortcuts for this version of the GUI. These shortcuts will overwrite any existing shortcuts. In the "beta" example used here, it is probably best to skip the creation of shortcuts:



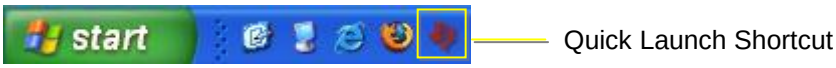
Using this technique, you'll be able to launch either version of the GUI from the Start Menu:



4 Fusion Digital Power Designer (Designer GUI)

4.1 Starting the GUI

The previous form in the installer controls whether GUI "shortcuts" are added to the desktop and quick launch area. The quick launch area is the area next to the Start menu which contains shortcuts to commonly used applications.



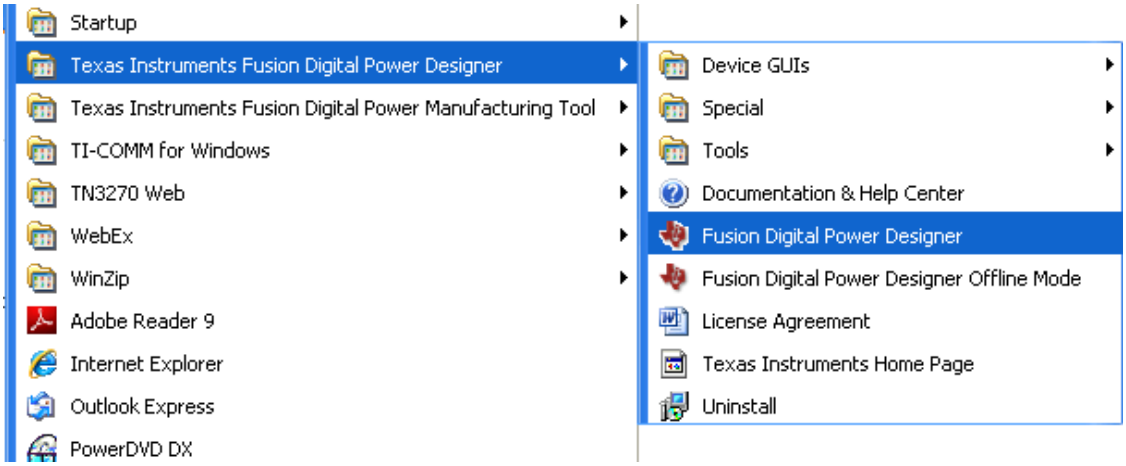


Figure 1 - Accessing Designer GUI from Start Menu

When you launch the GUI, it attempts to find a supported device attached to the PMBus. The following sequence is followed:

1. The GUI looks for an attached USB serial bus adapter. If it is not found you will see the following figure:

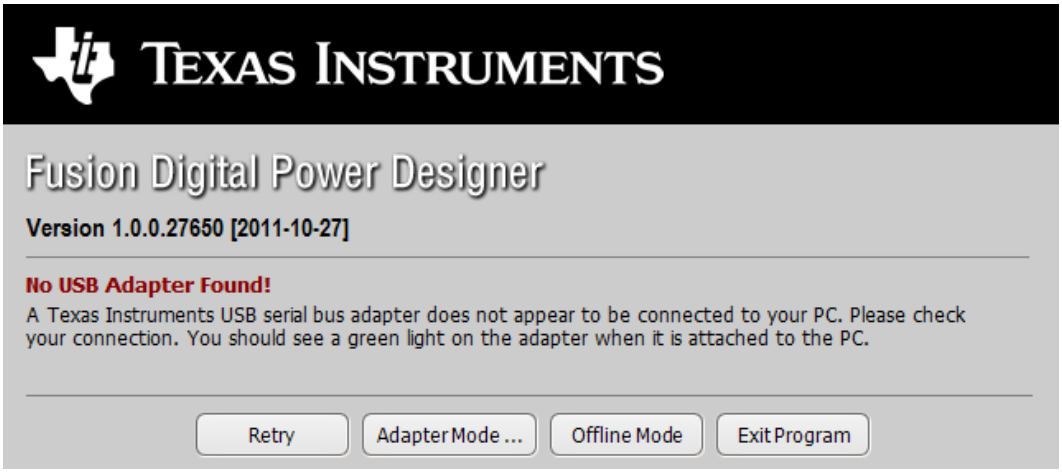


Figure 2 - No USB Adapter Found

2. The GUI sends SMBus commands to the “broadcast” address 11 telling any devices that are in ROM mode to execute their program (go to flash mode). While this is not necessary for production devices, it may be necessary for in-development products that are set to boot to ROM mode.

3. The GUI scans addresses 1 through 127 for an attached device. It does this by reading a special manufacturer command, DEVICE_ID, on each address. This parameter contains information about the device, including part number and firmware version. Address 12 is skipped because this is reserved for use in the SMBus Alert Response Protocol. After this command has been read then the SETUP_ID is analyzed. If the SETUP_ID is not recognized due to being part of new firmware, for example, then there are some steps that can be taken to still allow for communication with the GUI. See “Section 6.1.1.1 SETUP_ID in firmware is not recognized by the GUI”.
4. While the scanning process occurs, you will see a dialog box:

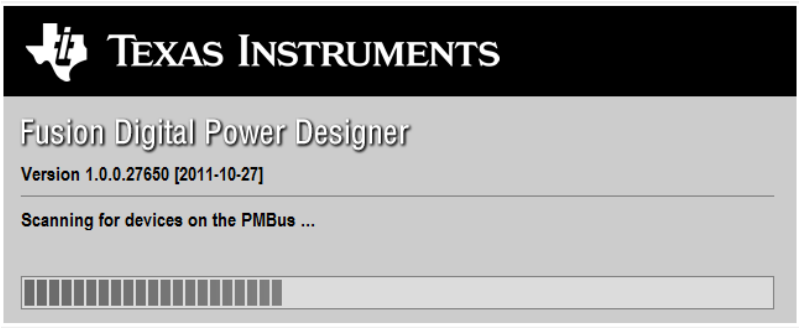


Figure 3 - Device scanning

5. If a supported device cannot be found, you will see this error message:

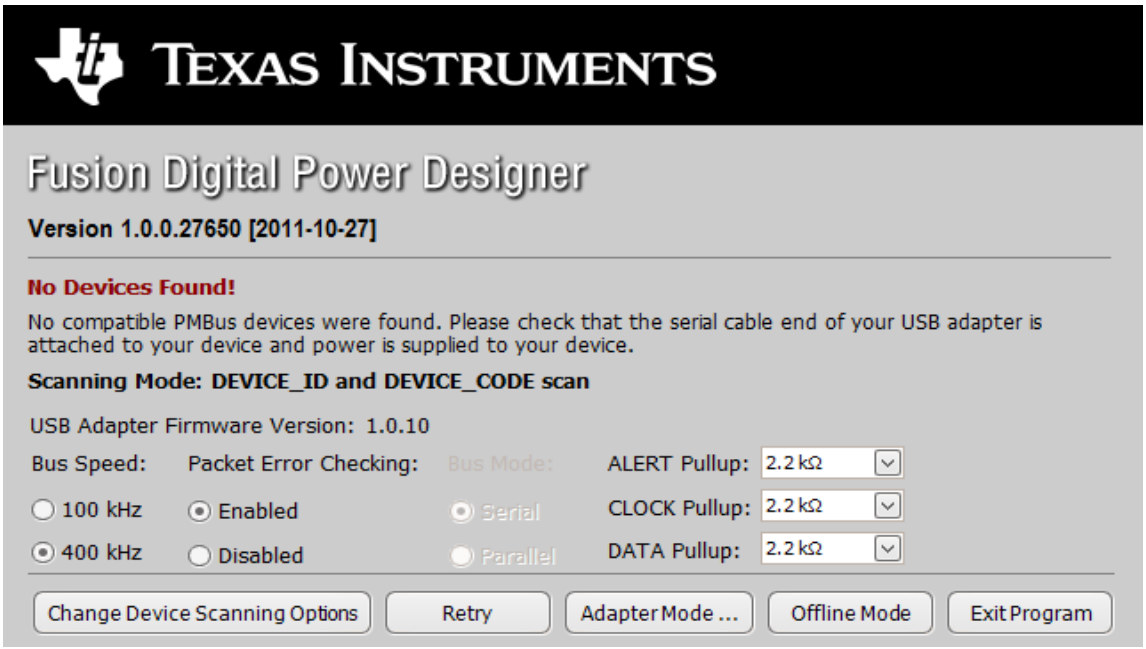


Figure 4 - No devices found

Double check your USB adapter connection and power to your device and click “Retry” to re-scan.

If the GUI is still unable to detect the device see the following troubleshooting tips in the Section “Connection Troubleshooting Tips 4.2.4”.

If you expect the device not to be detected and are interested in working with the offline features for your device, simply click “Offline Mode”. This allows you to use most of the GUI’s features while not electrically connected to a device. Offline Mode is described in more detail in Section 6.8.

4.2 Connecting to a Remote USB Adapter

The Fusion GUI supports connecting to a remote USB adapter on another PC running the Fusion GUI Adapter Server.

The client computer, i.e. the one without the USB Adapter desired, needs to configure the scan process with the IP, port, and password for the other PC. The computer with the desired USB Adapter will need to run a server that is part of the Fusion GUI. The client will be discussed first and then the server.

4.2.1 Client configuration

The following are two ways to get to the client configuration. The first way is from the "Start" menu:

Click "Start>Texas Instruments Fusion Digital Power>Tools>USB Adapter Mode Selector" as shown in Figure 5.

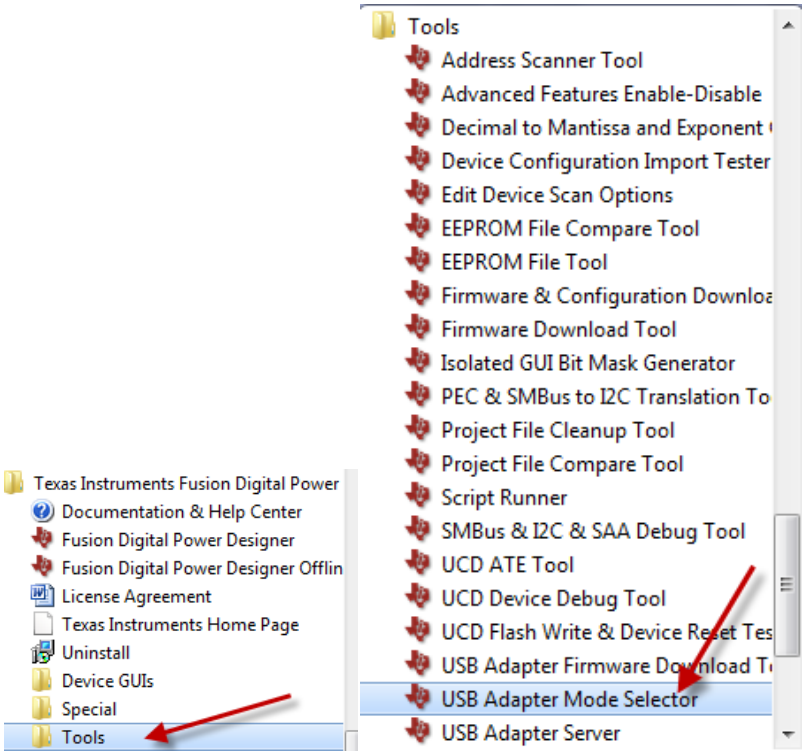


Figure 5 - USD Adapter Mode Selector from Start menu

The second way is after the initial startup scan fails, there will be an option to click "Adapter Mode …" as shown in Figure 6.

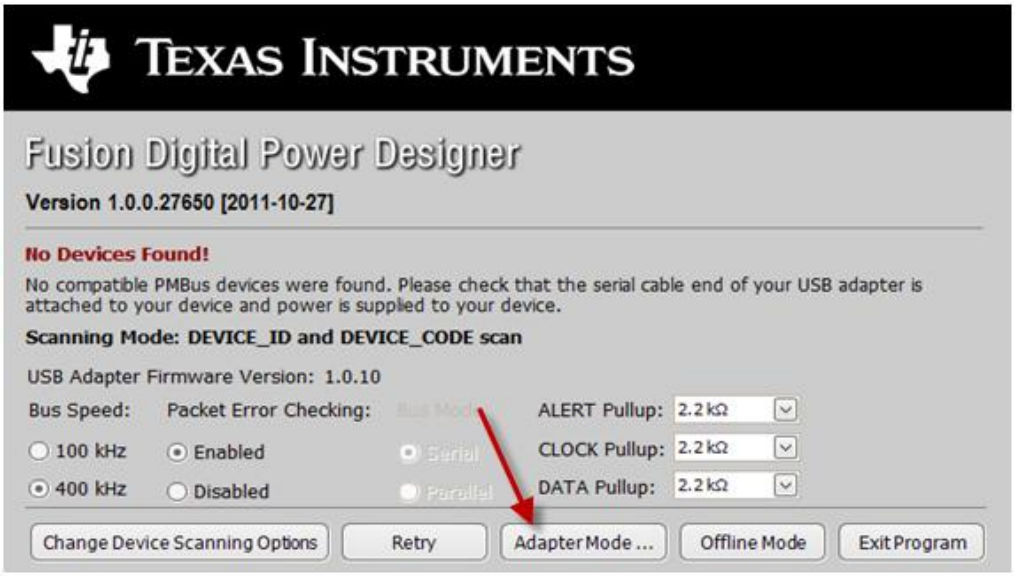


Figure 6 - Adapter Mode button from "No Devices Found!" screen

The default setting is to have the GUI use the local USB adapter. However to access the remote USB adapter the user will select "Remote" as shown in the figure below. The host's IP and Port will need to be specified (and password if one is set). This information is automatically available in the Fusion GUI Server running as shown in Figure 10 (a few figures below). After entering the server information click "Test Connection" as shown in Figure 7.

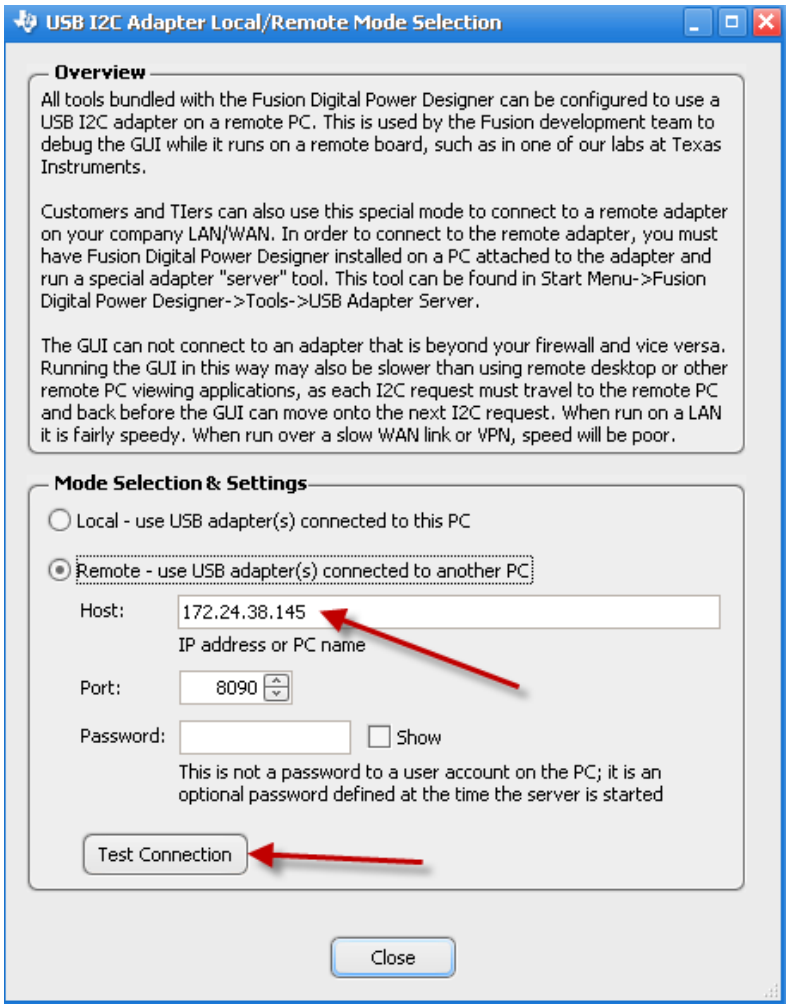


Figure 7 - Adapter Mode Selection

If you click “Test Connection” you will be able to test whether you are connected to the server and observe the following figure indicating success. If unsuccessful, ensure you are connected to the internet and that the information is entered correctly. Note: the GUI cannot connect to an adapter that is beyond your firewall and vice versa.

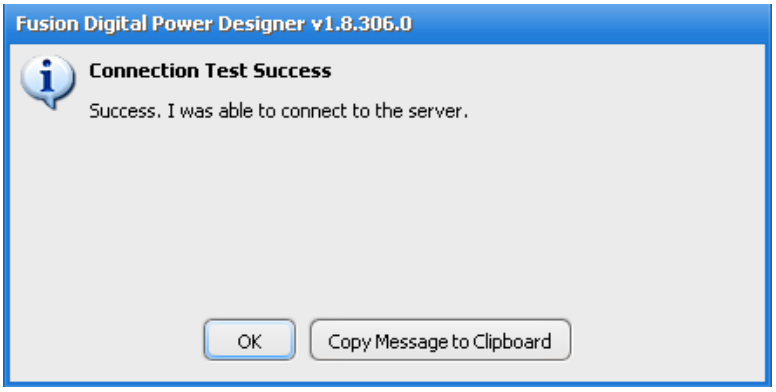


Figure 8 - Remote Adapter Connection Test Success

4.2.2 Remote Server Configuration

The server machine will need to run the “USB Adapter Server” to allow clients to connect. The USB Adapter Server can be found in the Start>Texas Instruments Fusion Digital Power>Tools menu as shown in Figure 9.

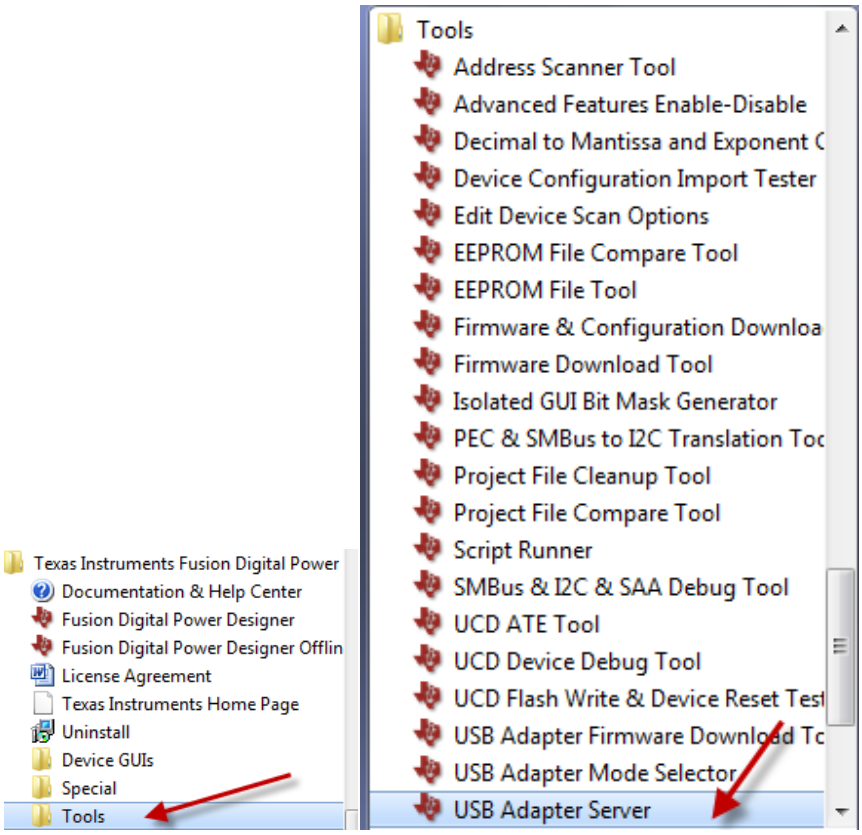


Figure 9 - USB Adapter Server from Start menu

The USB Adapter Server shows its IP Address that the client will need to use. Also, in order for clients to connect to the server the user must click the “Start Server” button.

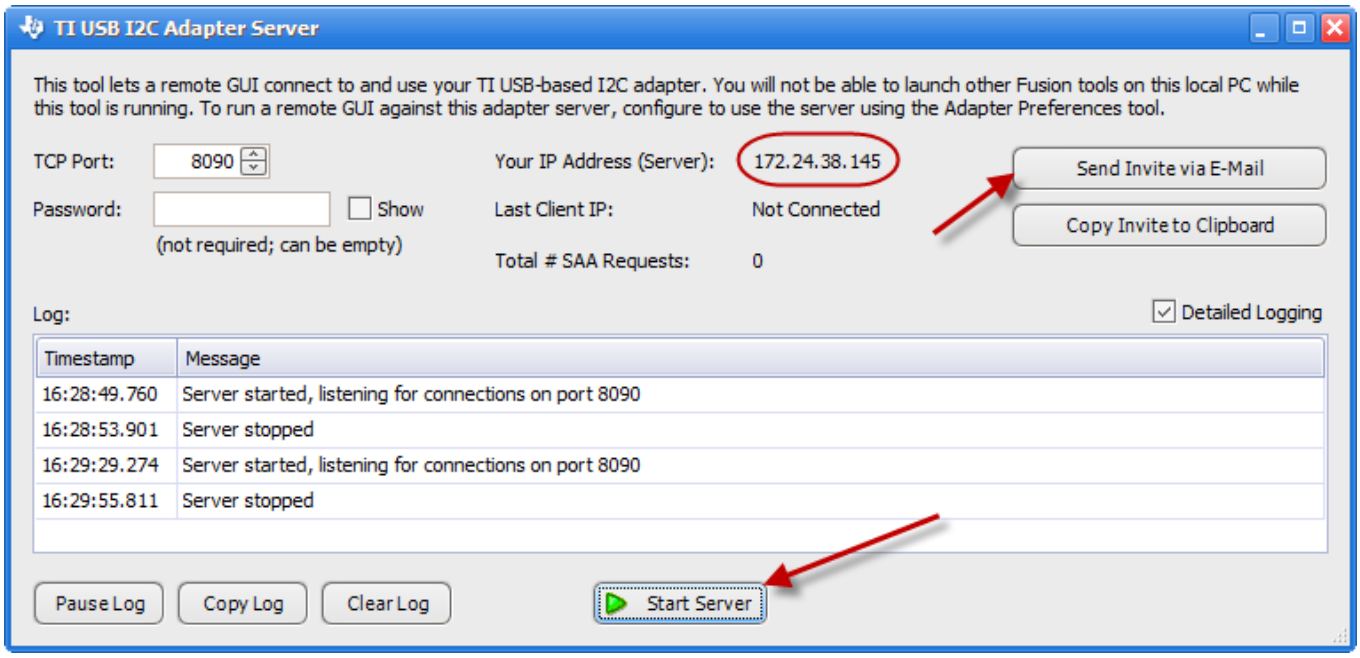


Figure 10 - Remote Adapter Server

If the user clicks “Send Invite via E-Mail”, a prefilled email will appear populated with the relevant information for the client to connect to the server. It includes the IP address, Port and Password if any. It also includes instructions on how to configure the client. The Figure 11 below shows the content of the prefilled email. The user will need to replace “my-client-friend@firend.com” in the “To:” with the appropriate email address of the client.

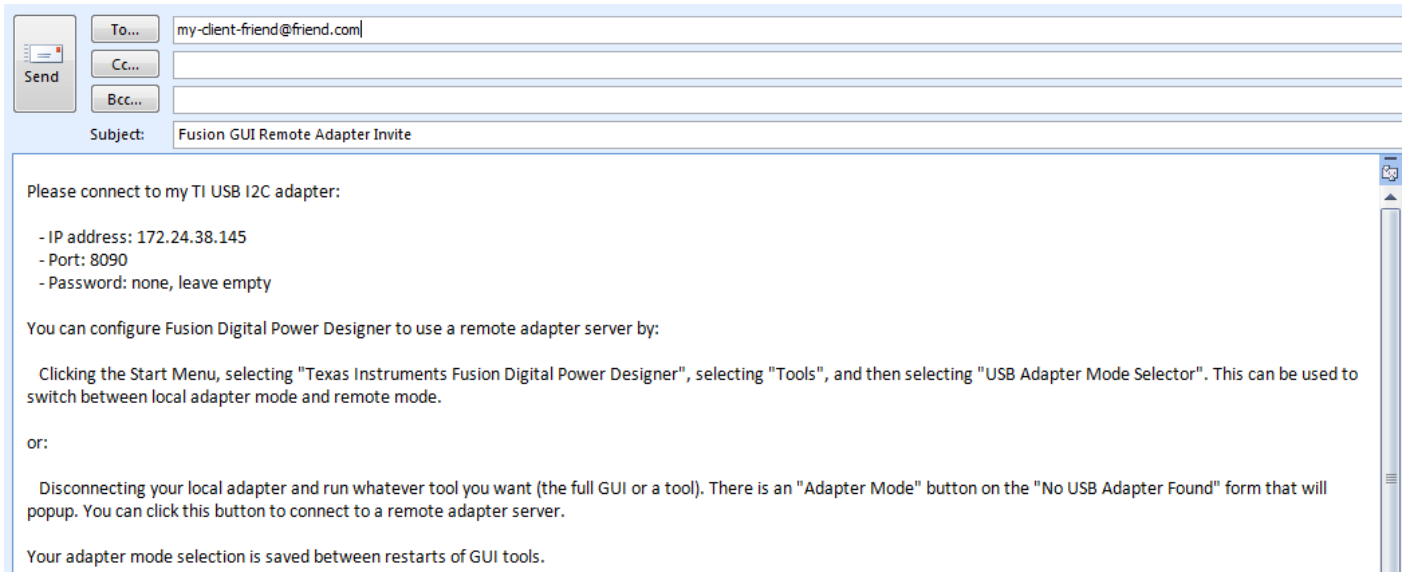


Figure 11 - E-mail invite with Remote Server settings

4.2.3 Client and Server running

The following are some figures of a live client and server interacting. For the client, the experience of running the GUI remotely would be the same as running it on a local USB adapter except for the speed being slower.

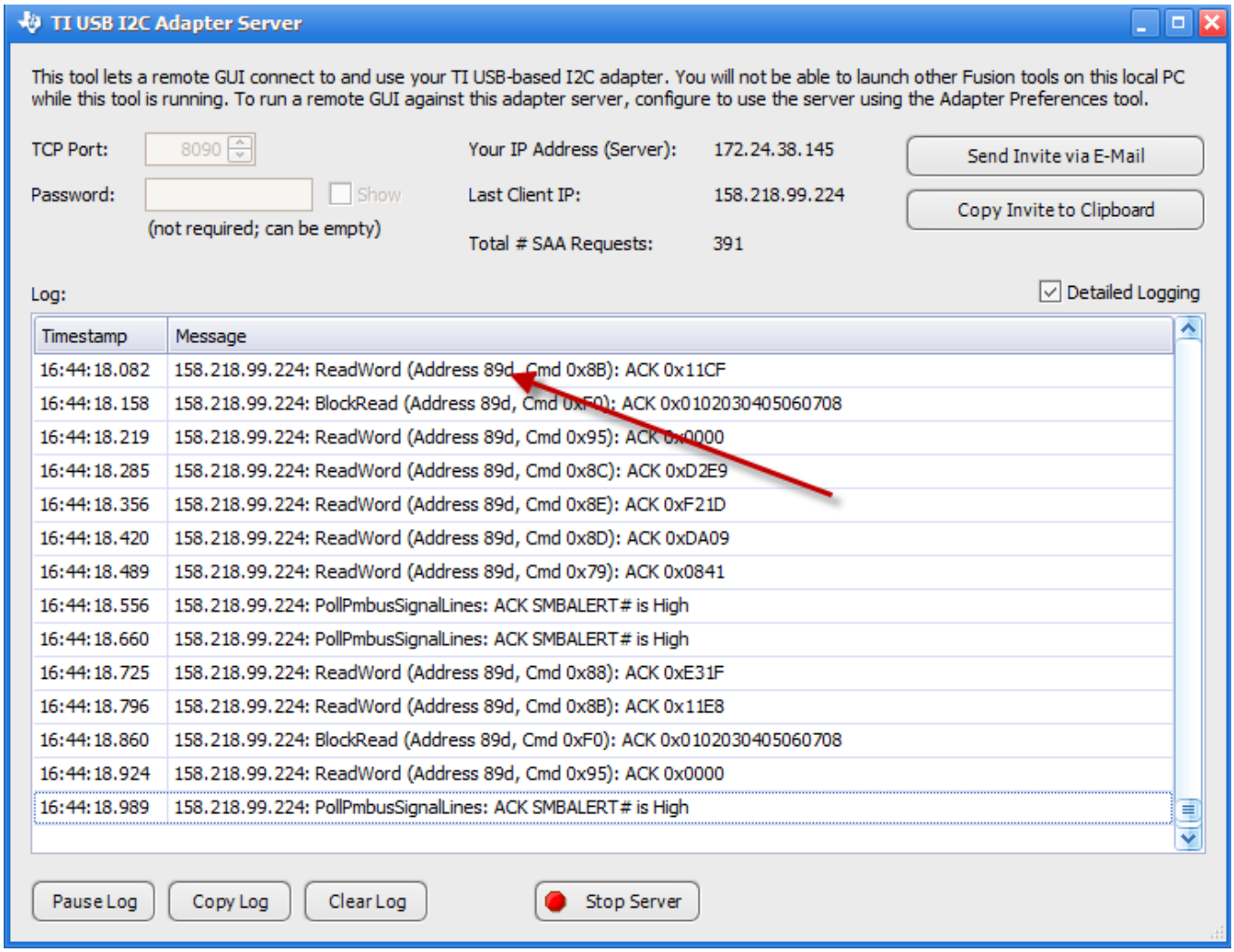


Figure 12 - Running a remote server connected to a client

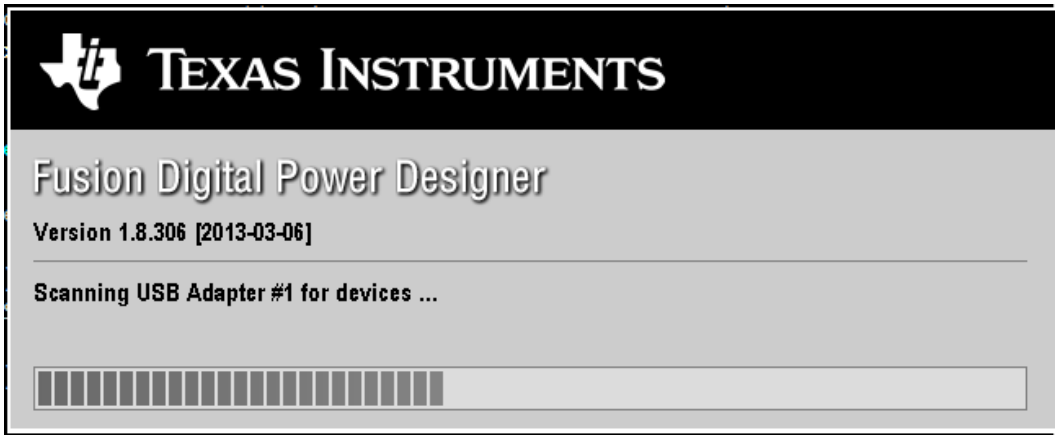


Figure 13 - Client's machine conducting a scan on the remote adapter

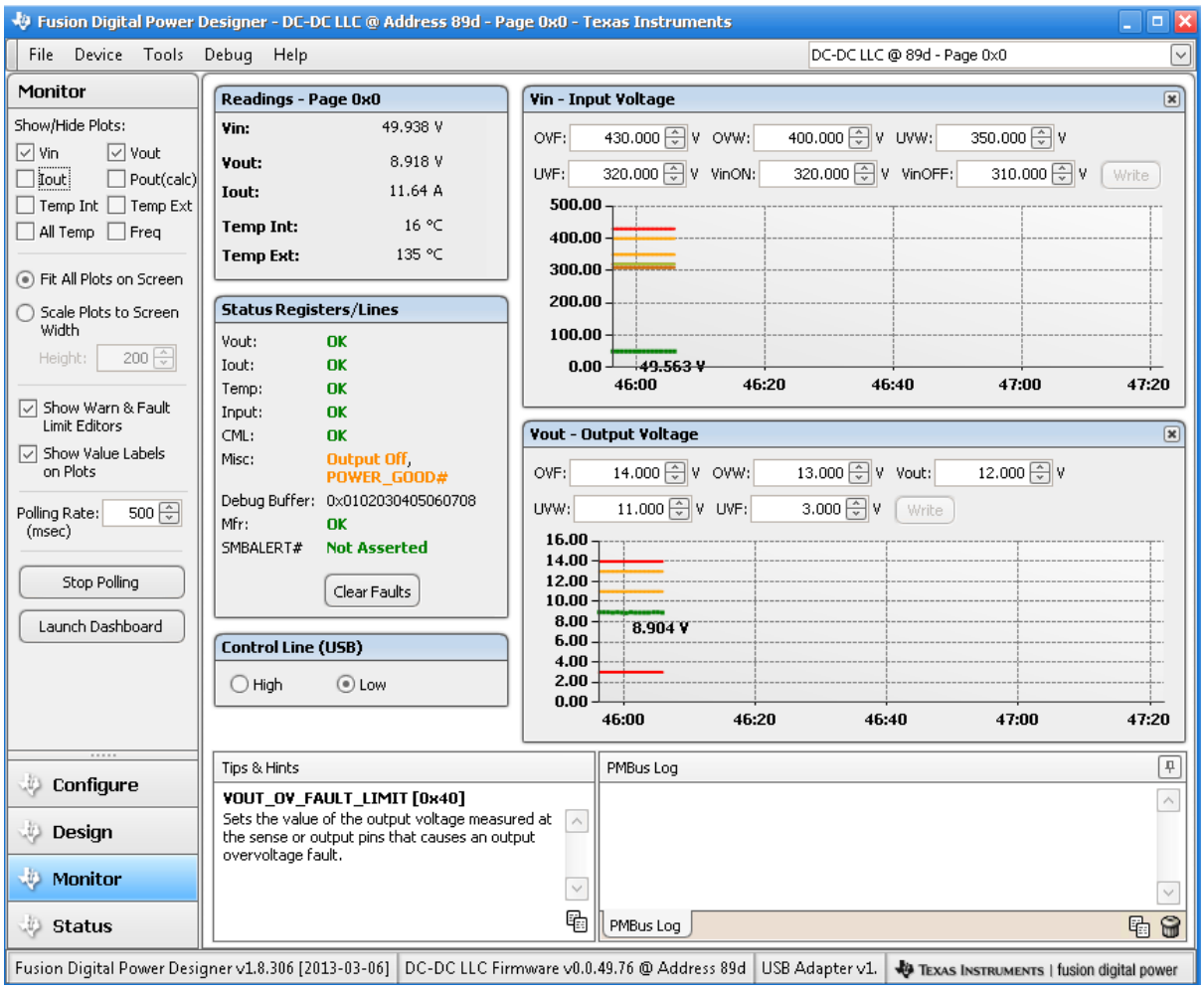


Figure 14 – Client's machine interacting with a remote USB Adapter connected to a device

4.2.4 Connection Troubleshooting Tips

5 Problem	6 Resolution
The scan never occurs. The GUI immediately comes up with the error form. When retry is clicked, the error form reappears immediately.	This usually indicates the USB serial adapter is not attached to the PC or is malfunctioning. Verify that the green LED on the serial adapter is ON. If it is not, unplug the adapter, power off your device, reconnect the adapter, and then power on your device.
The GUI scans each address, but cannot find the device	Verify that power is on to the device. Try re-applying power to the EVM. Also, try resetting the USB adapter as described above.

6.1.1.1 SETUP_ID in firmware is not recognized by the GUI

Generally in order for the GUI to recognize your firmware it needs to recognize the manufacturer commands Device_ID and SETUP_ID. However in the case where you are developing a new firmware and the SETUP_ID is not supported by the GUI you can change your scan preferences to ignore your SETUP _ID and continue to try to communicate with your device through the GUI. If communication can be established, then you will have the ability to interact with the PMBus commands that you have implemented in your firmware. You will not be able to access the Design features of model compensation and the stage of your topology since this requires knowledge of your SETUP_ID which indicates to the GUI the device's topology.

You can skip the SETUP_ID recognition scan by doing the following.

6.1.1.1.1 Change the Device Scanning Options

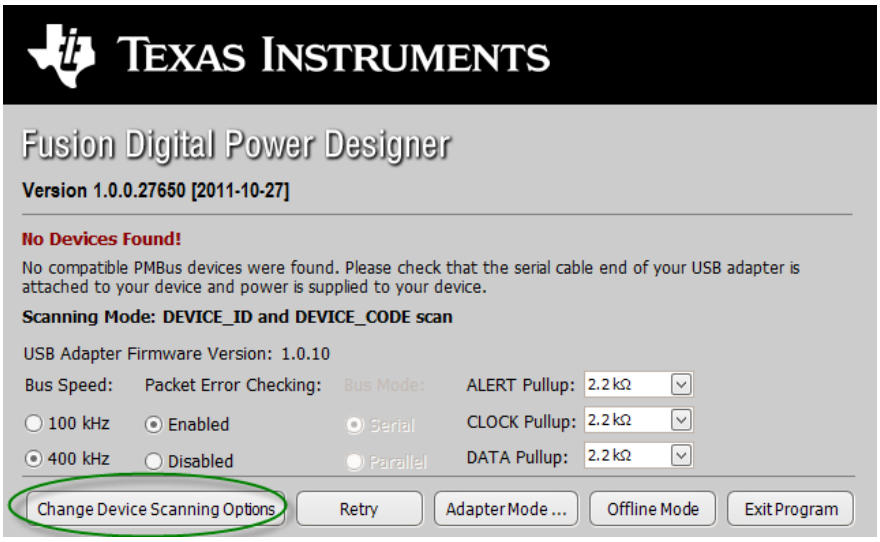


Figure 15 - Select Change Device Scanning Options

The following dialog allows you tell the scanner what type of device is to be expected at each address.

Click the button "UCD3XXX Isolated" at the top right. Click "OK" and then "Retry" the scan.

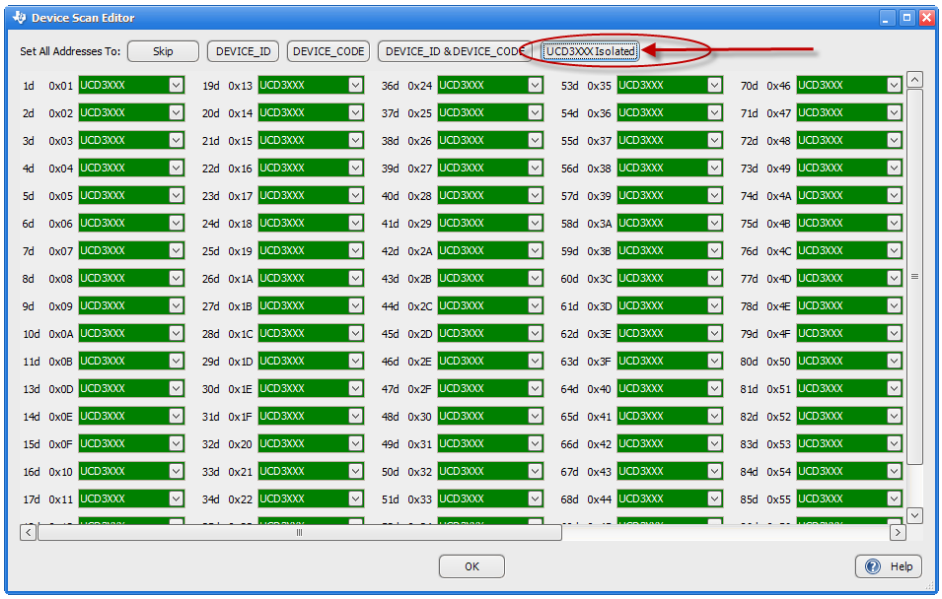


Figure 16 - Click UCD3XXX Isolated at top

6.1.1.1.2 Click Fallback Mode from Start Menu

An alternative way to change the scanning options is to select this scan mode from the Start Menu as shown below.

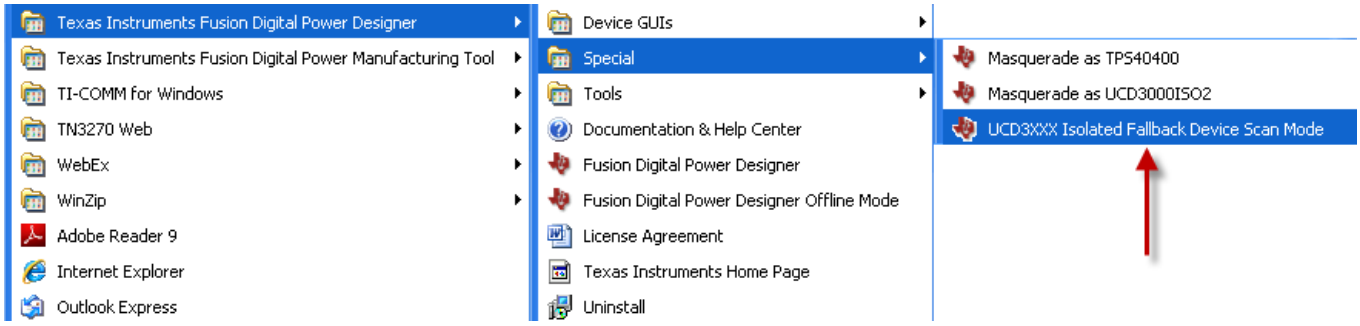


Figure 17 - Start>Texas Instruments...>Special>UCD3XXX Isolated Fallback Device Scan Mode

6.2 Enable GUI Protected Features

Figure 18 shows how to access the configuration screen to enable the GUI protected features. Figure 19 shows the screen. Make sure the selections are checked as shown and in the password box type the word "**forestIn**." Click OK and then many features will be available.

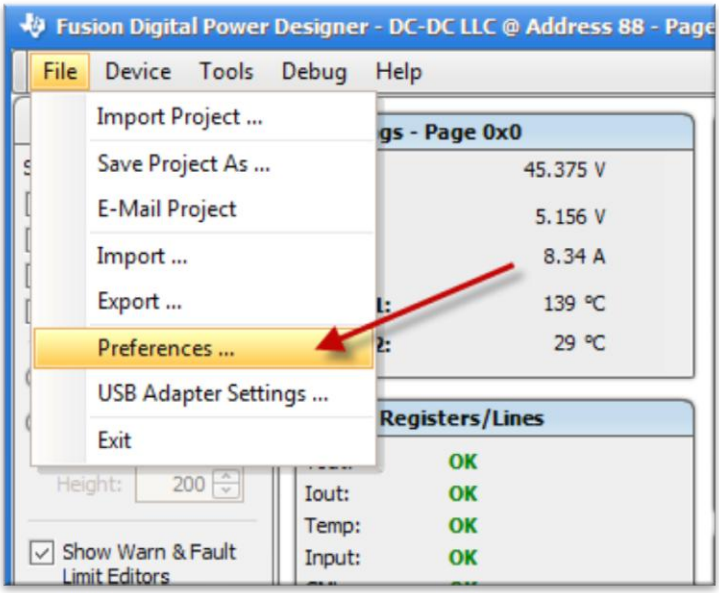


Figure 18: GUI Preferences

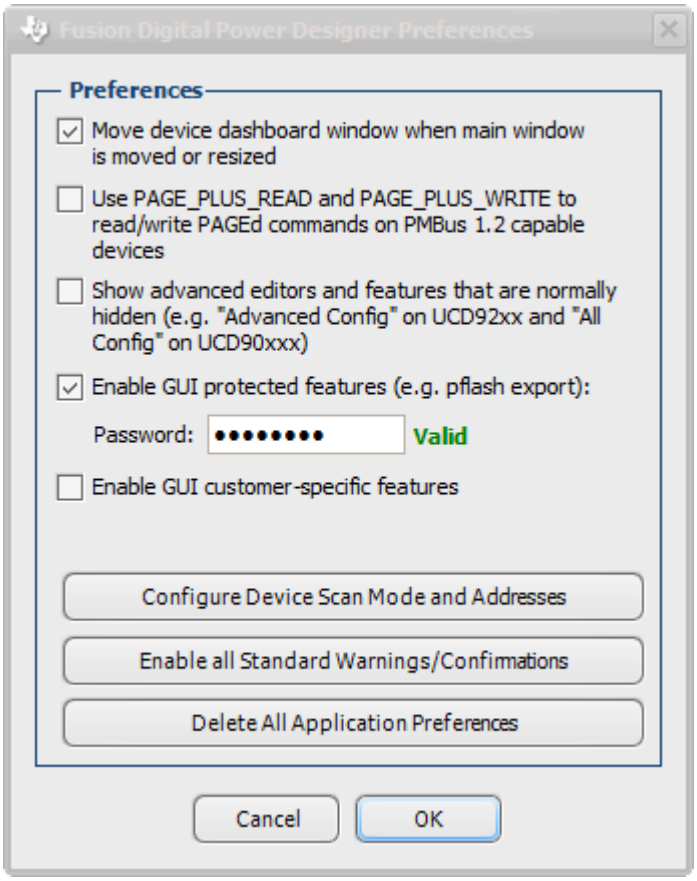


Figure 19: GUI Protected Features

6.3 Monitor

After a device is found, the first screen that appears is the Monitor Screen. Depending on which commands are implemented the corresponding monitor graphs will be available. In the figure below the commands for reading Vin, Vout, Iout, Pout, Temp 1, Temp 2, Frequency were all implemented so the graphs are available. The Monitor tab gives you a live view of the active power supply. In addition to plotting the values it also shows the latest values in the "Readings" group. It also shows a snapshot of the "Status Registers/Lines". The word "Fault" appears in red when a register is at fault, otherwise a green "OK" is visible. The polling of the parameters being read can also be halted by clicking "Stop Polling" on the left side.

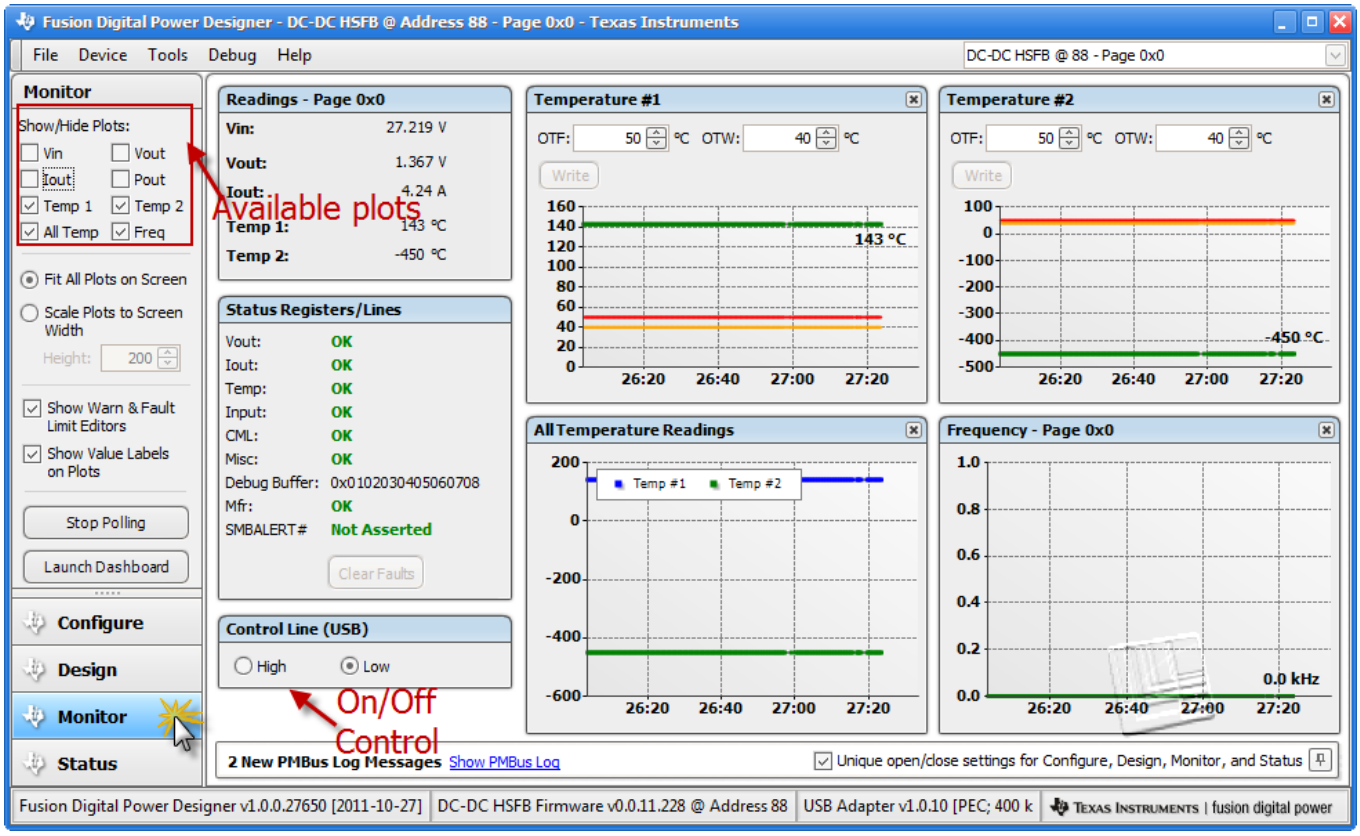


Figure 20 - Monitor mode displays some of the live parameters being read from the device.

6.4 Configure

As can be seen from the above figure there are a four clickable categories on the bottom left. To get to the Configure mode the user selects "Configure". The following figure displays some of the features of the Configure mode.

6.4.1 PMBus commands, Edits, and Writing to Hardware

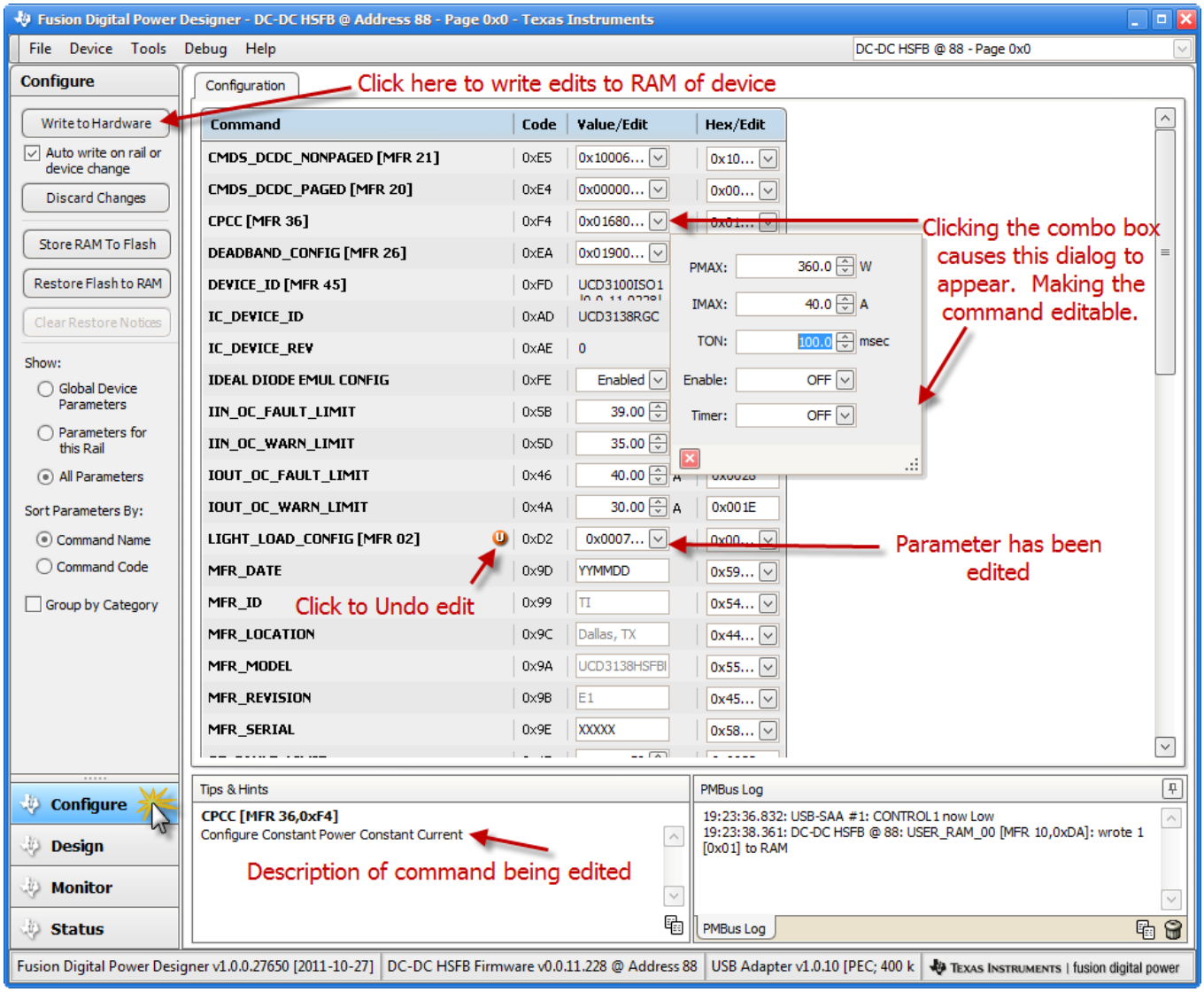


Figure 21 - Configure mode

When the Configure mode appears all of the implemented PMBus Commands are visible. A discussion of the relationship between what is visible and what is implemented in the firmware will be discussed in "Section6.4.2 How does Implemented Commands on the Firmware Appear in the GUI?." A read was done on all the PMBus Commands and their values are immediately visible.

On the left there are some controls to decide how they can be ordered to help view them. They may be listed by category, or sorted by name, or by hex code.

Some values are read-only (uneditable) and some are writable. In the above figure the parameter

LIGHT_LOAD_CONFIG was edited by changing the value. When a command is edited a appears beside it. This indicates that the value can be undone, or reverted back to the device value stored in RAM. As a command is edited the value is not automatically written to the device. To write all edits to the device the user needs to click "Write to Hardware." Then if the user would like to store those to flash the button "Store RAM To Flash" would need to be clicked. Section "6.7 Capturing the State of the Device - Saving a Project File" discusses storing the current state of all commands to a local file that can be used to write to another device.

Another feature that is highlighted in this figure is the dialog box that appears to edit the Constant Power Constant Current “CPCC” command. Not all commands are direct value edits like “IIN_OC_WARN_LIMIT” that is set for “35 A” rather some of them are more complex and require unique dialogs to edit them. CCPC is just one example from many.

6.4.2 How does Implemented Commands on the Firmware Appear in the GUI?

The Designer GUI is dynamic. It automatically lays out the PMBus commands that are implemented in the firmware. The firmware developer can make a change and then restart the GUI noticing the change immediately without a new Designer GUI installation. How does the GUI know which commands are implemented? The answer is there are certain Manufacturer commands that indicate which commands are implemented. The command “CMDS_DCDC_NONPAGED [MFR 21] 0xE5” is one such important command that helps the GUI to configure itself. It contains a bitmask. That bitmask is determined in firmware. Each bit in the bitmask indicates whether a command is implemented or not. Each bit refers to a specific command according to the PMBus 1.2 spec. When the GUI reads this bitmask it looks for all the “1”s and then displays those commands in the GUI.

The Isolated Bitmask Tool, discussed in section 7.6 of this document, is a valuable tool to help firmware developers set this important bitmask. The figure below displays the read-only command “CMDS_DCDC_NONPAGED [MFR 21] 0xE5”.

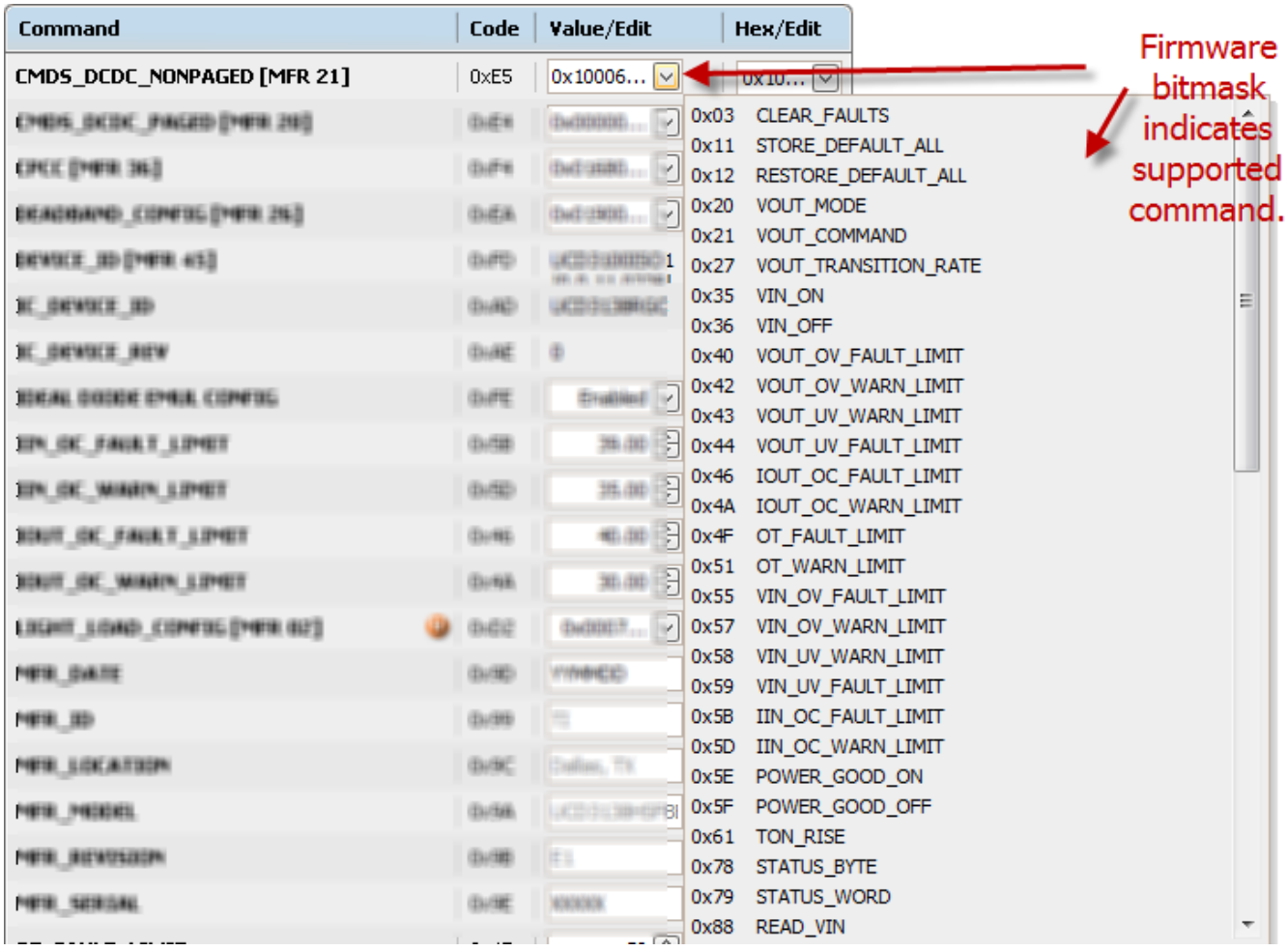


Figure 22 - Displays the list of commands the firmware supports

6.5 Design – Model Stage and Compensator

To get to the Design mode click the “Design” button on the bottom left. The following figure should appear. The number of loops to configure and parameters in the power stage may differ depending on which of the 4 power supply topologies is represented by the firmware in the device. The example below illustrates what is available with the “Hard Switching Full Bridge (HSFB)” firmware (featured in UCD3138HSFBEVM-029).

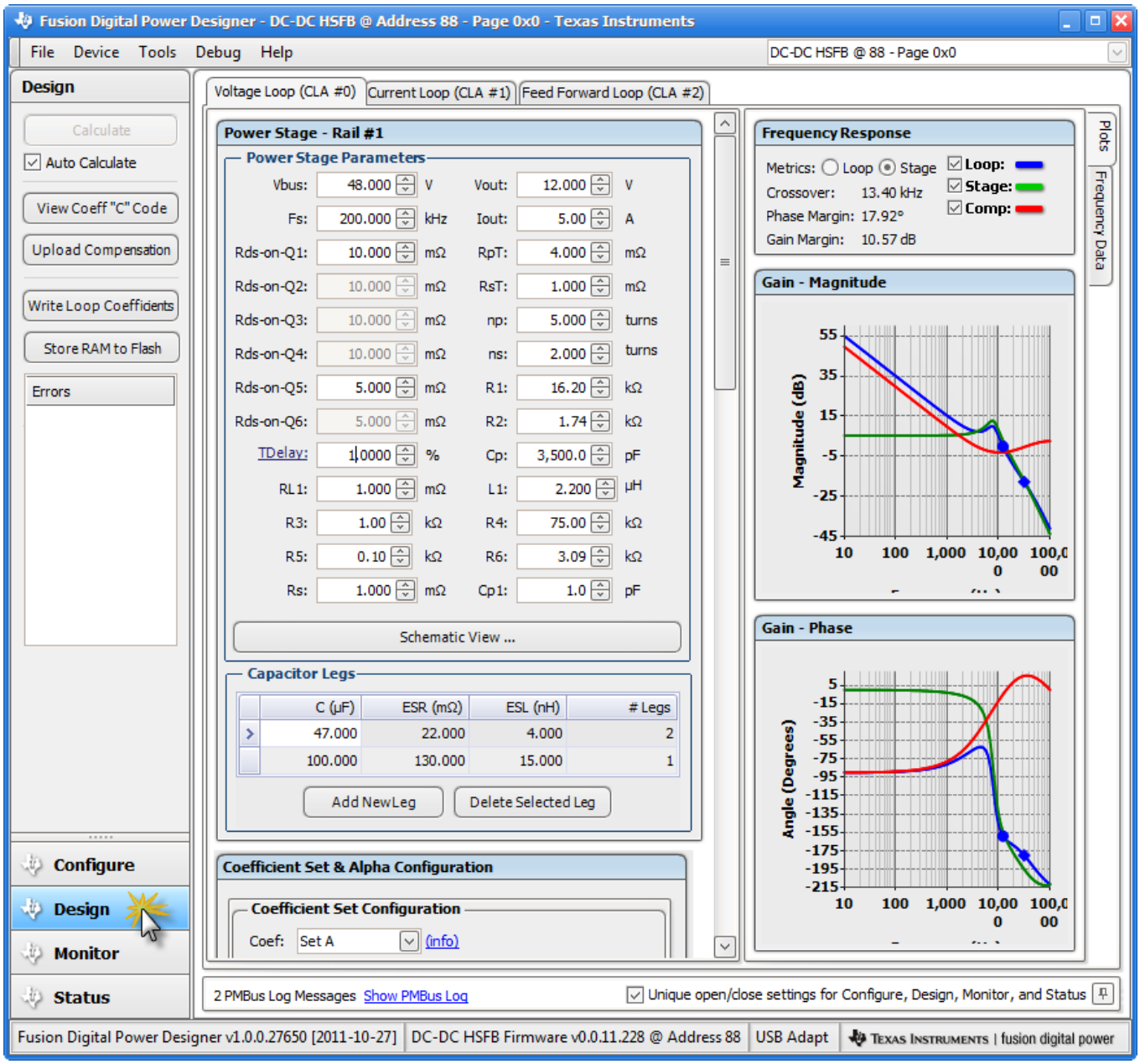


Figure 23 - Design mode selected

6.5.1 Power Stage

Depending on which topology is being modeled, the relevant parameters for the stage will be displayed. In the example shown above for HSFB the following parameters for the stage were shown:

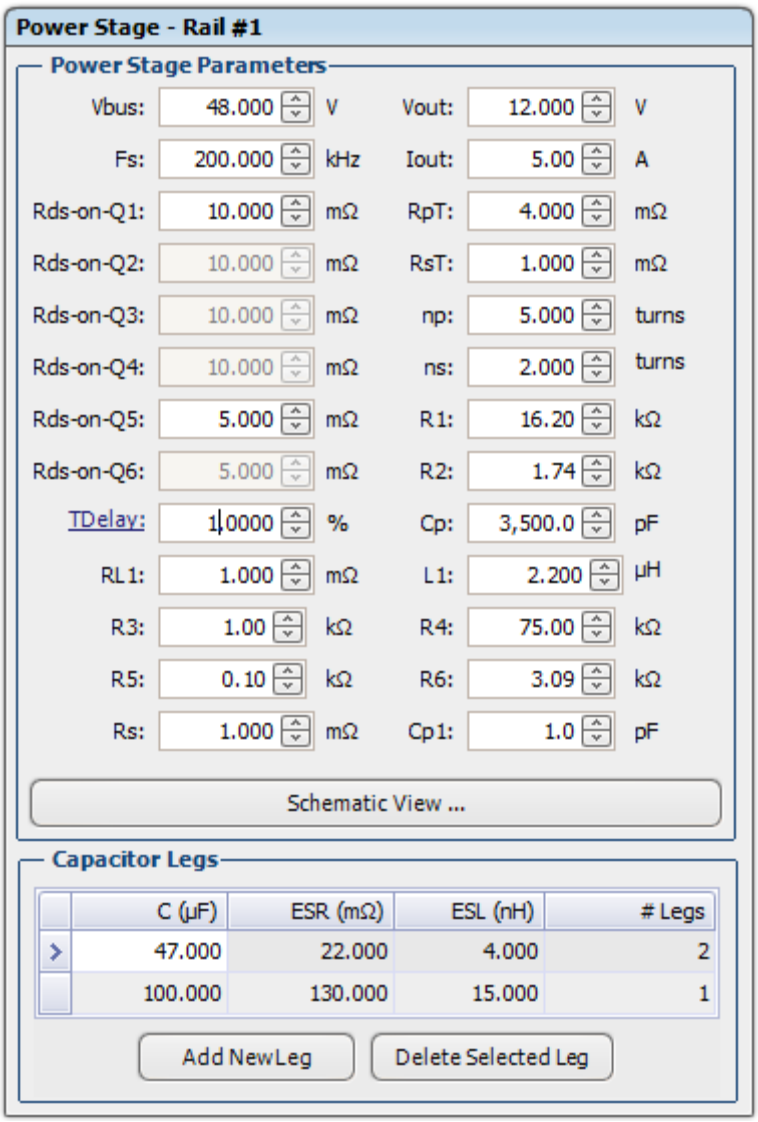


Figure 24 - Stage parameters for HSFB for Voltage Loop (CLA #0)

To model the power stage for the topology, certain parameters need to be specified. Based on the values set, the Bode plot for the power stage is calculated and displayed on the right. This powerful feature is provided to aid designers with fast loop compensation based on analytical models built inside the GUI, representing the power stage equations appropriate for the topology. The power stage equation differs from loop to loop. The figure above is part of the voltage loop as shown in Figure 23.

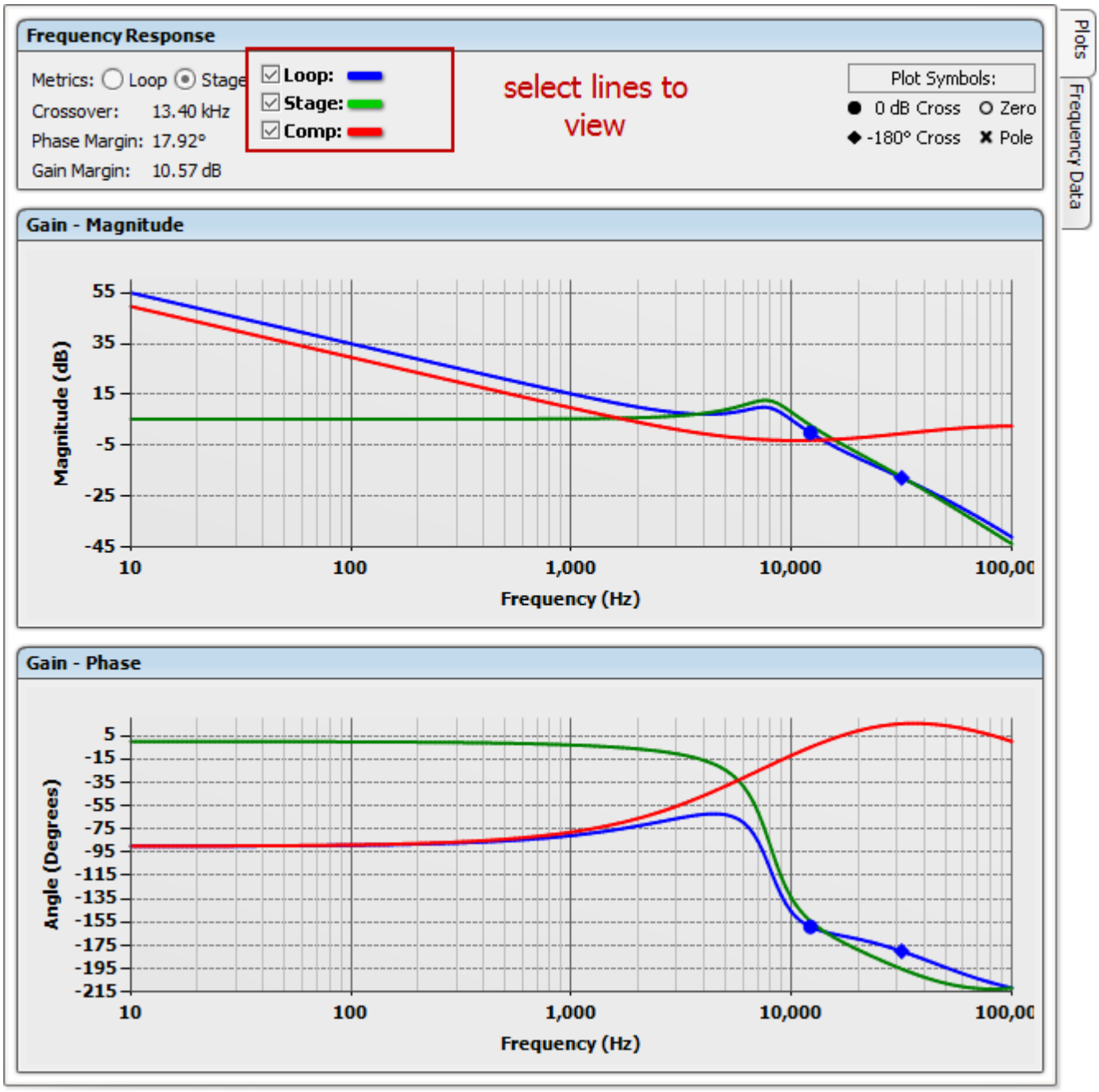


Figure 25 - Bode plots

There are three lines. The green line indicates the power stage. The other two lines are the Compensator and the Loop. Lines can be deselected as shown in the figure above. The Compensator will be discussed in “Section 6.5.2 Compensator.”

Clicking “Schematic View” in Figure 24 will open a dialog with a picture of the schematic. See below.

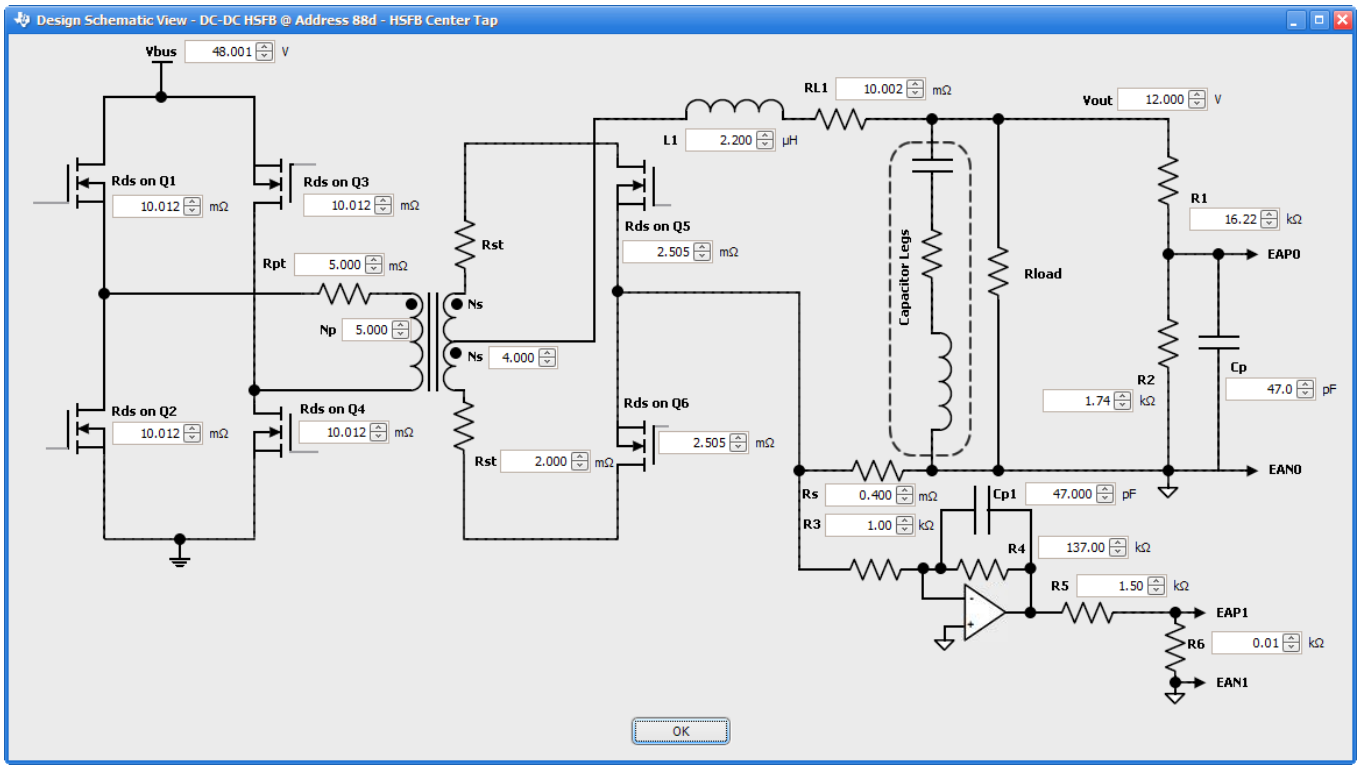


Figure 26 - HSFB schematic being modelled

The bode plots are updated automatically as the values are set.

6.5.2 Compensator

To model the compensator there are a number of values to configure. The values to configure for the compensator are the Coefficient Sets (A to G), Alphas (0 and 1), Bins (0 to 6) and Threshold Limits (0 to 5). This needs to be done for each loop. The compensator area is just below the Power Stage Parameters. Simply scroll down to bring the controls into view.

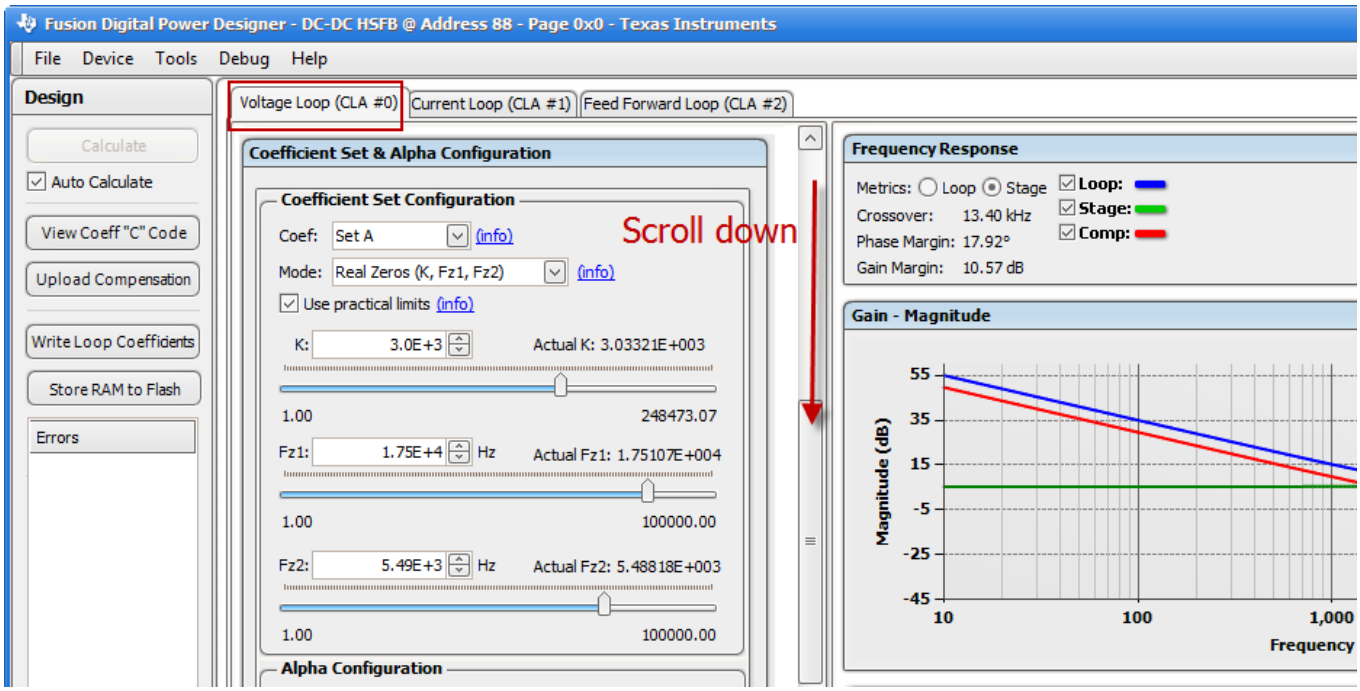


Figure 27 - Scroll down the stage parameters to see the compensator

The GUI comes equipped with 3 different ways to program the UCD3138 digital compensator. The figure below lists these options. The compensator hardware is described by the forth equation (Device PID). In this context; Kp, Ki, Kd and α are the raw register values used to configure the positions of the poles and zeros of the compensator. SC is a gain scaling term. Although it is normally set to zero, it provides additional gain for situations where the power stage gain may be low. PRD is used to configure the minimum operating period and KCOMP is used to configure the maximum operating period. In the context of the compensator they are simply gain terms that modify the overall transfer function by a fixed value. It is important to be aware that the proper way to configure PRD and KCOMP varies based on the control topology implemented. Please consult the relevant EVM user guide and training materials for details.

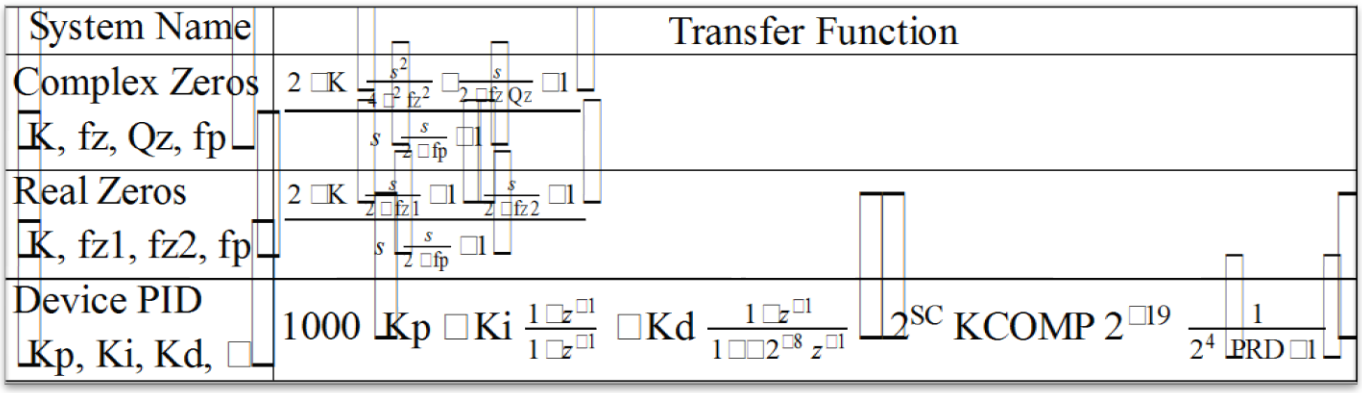


Figure 28 - Three ways to program the compensator

6.5.2.1 Coefficient sets and Alpha

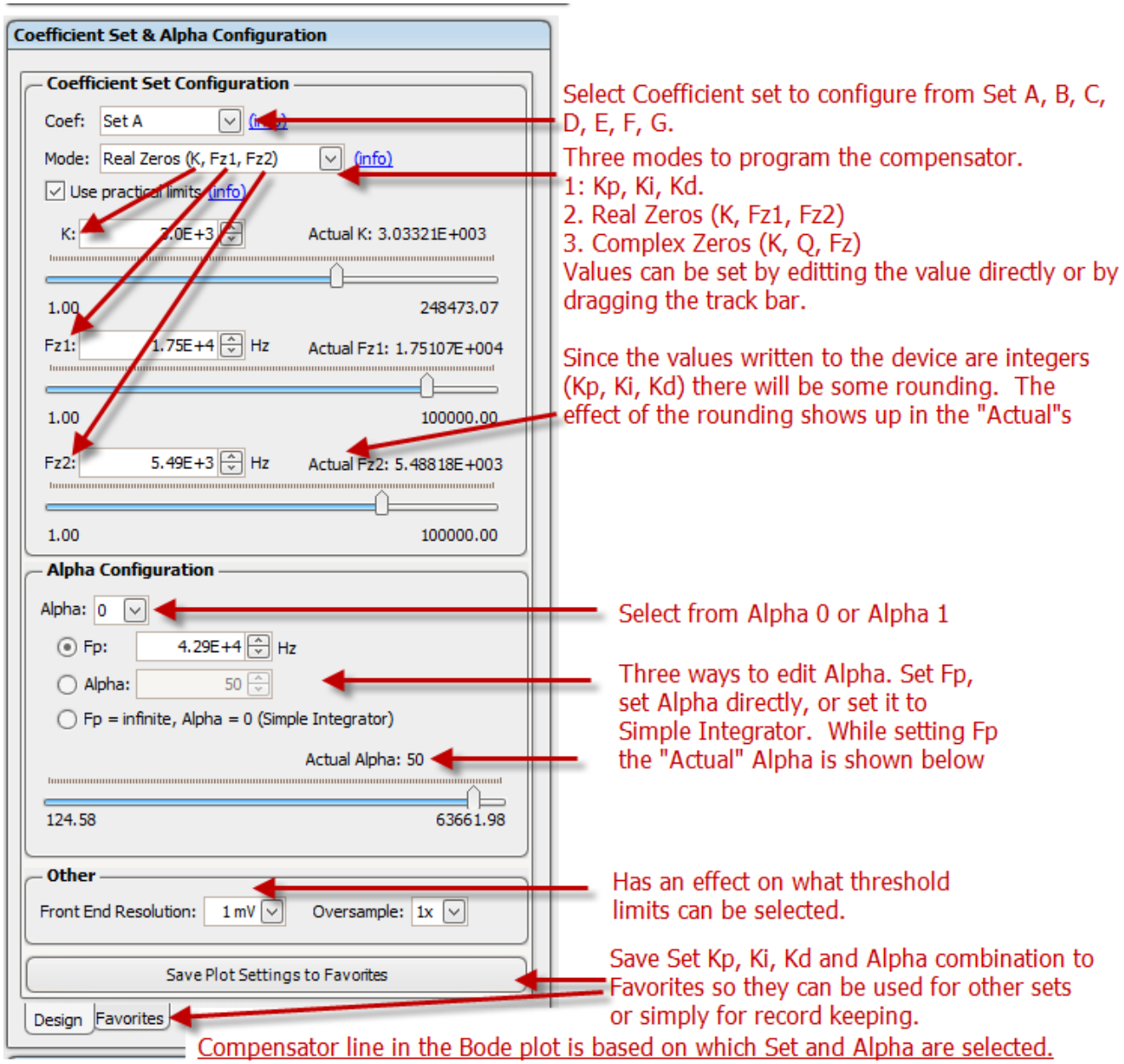


Figure 29 - Coefficient Set and Alpha Configuration

6.5.2.2 Bode Plot

The Bode plot located on the right of Figure 27 is based on the selected Set and Alpha.

6.5.2.3 Saving Favorites

Sometimes the user would like to keep copies of their Sets and Alphas so they may use them later or apply them to another Set and Alpha. This is possible by clicking the “Save Plot Settings to Favorites” button in Figure 29.

Users can also access the “Favorites” tab directly to view all their Alpha-Set combinations. They can also copy favorites and add descriptions. See Figure 30.

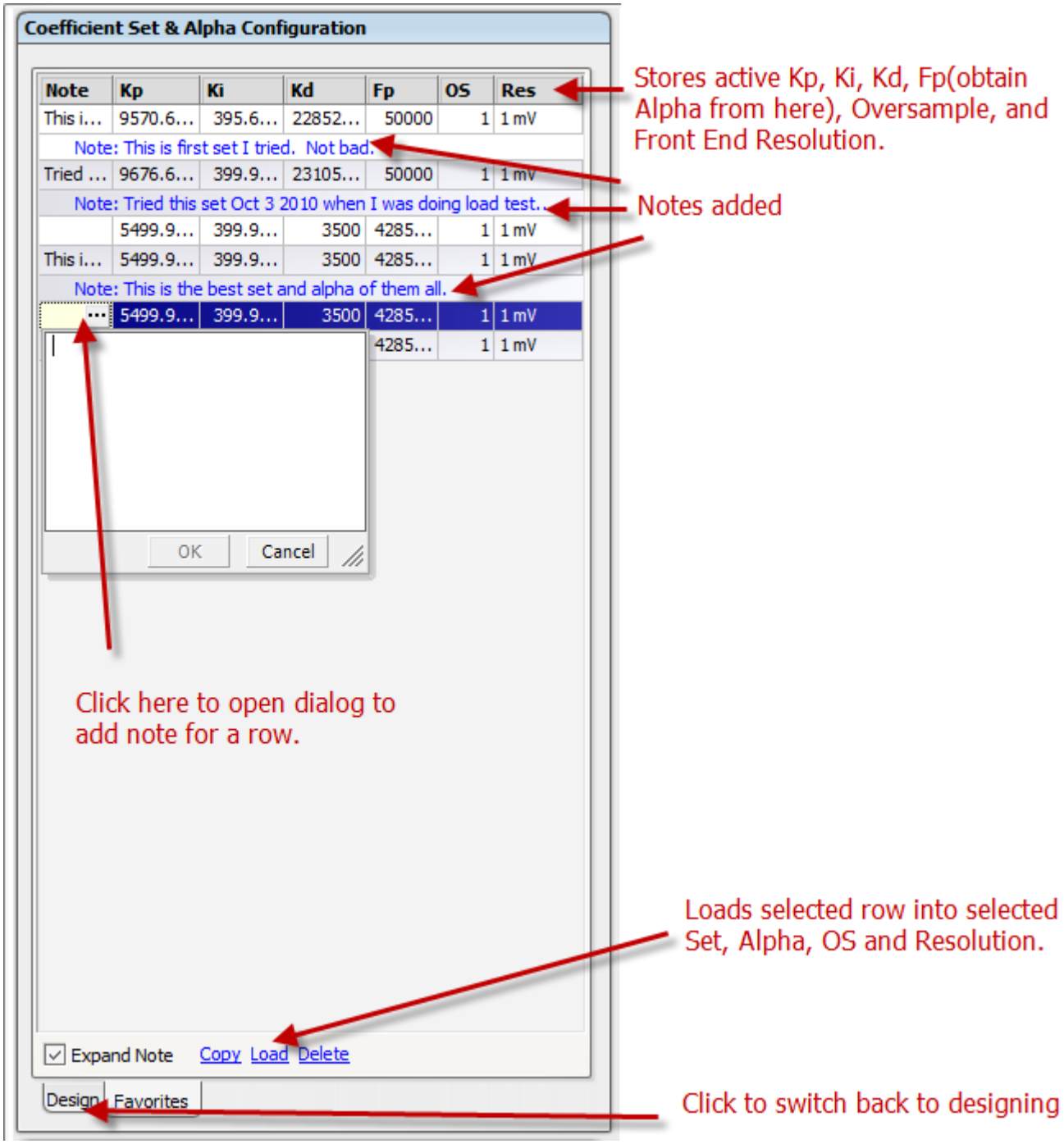


Figure 30 - Favorites

6.5.2.4 Coefficient Set and Alpha Summary

Immediately below the Set configuration is the “Coefficient Set and Alpha Summary.” This section displays all the alphas and coefficient sets.

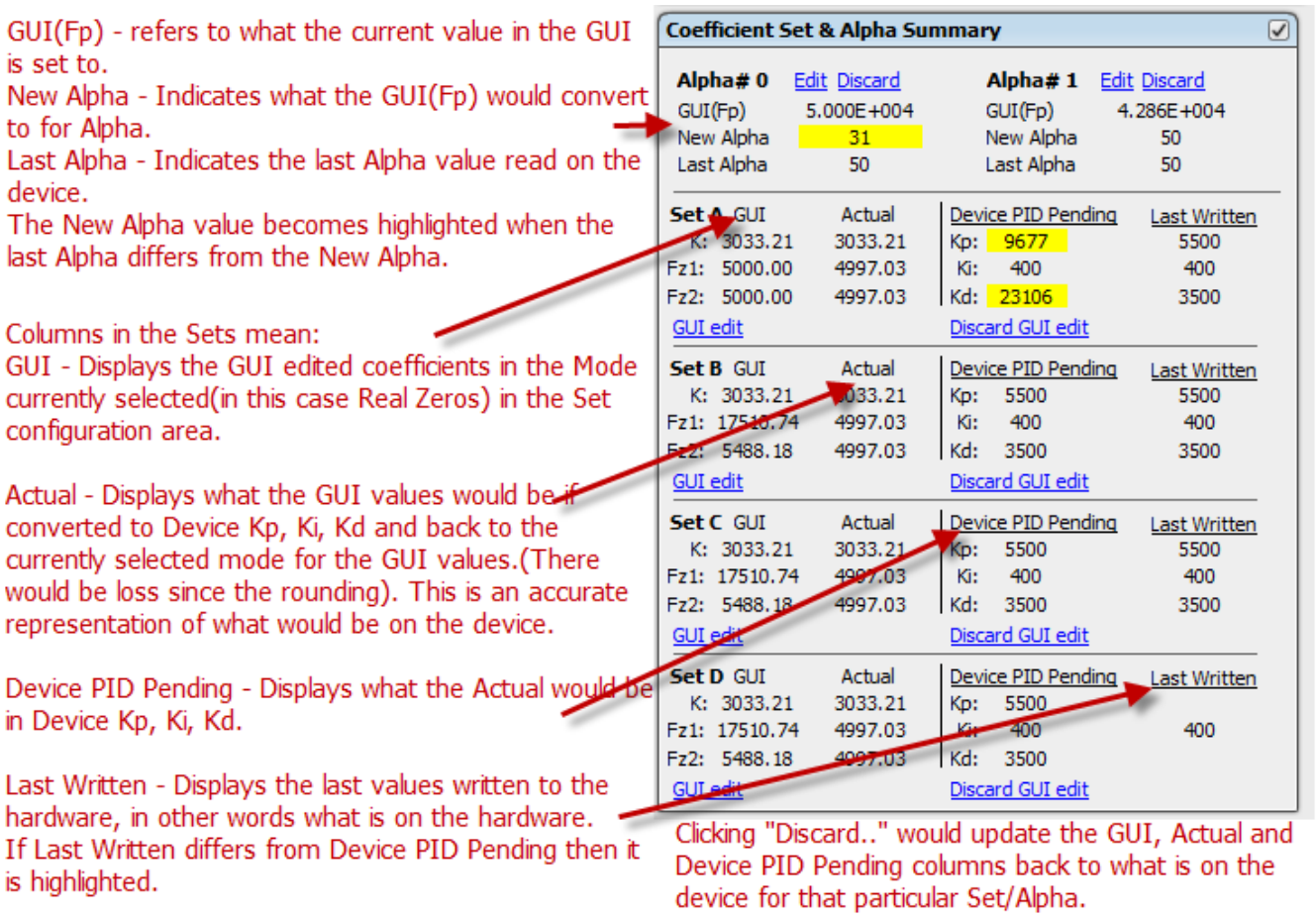


Figure 31 - Coefficient Set & Alpha Summary

Another way to discard all GUI edits globally is to click “Upload Compensation” as described in Section 6.5.2.5

6.5.2.5 Bin Assignment & Non-Linear Table Configuration

To configure the non-linear table the user specifies which sets and alphas are to be used within the configurable limits. One of the rules of the limits is that Lim 0 should be less than Lim 1 and Lim 1 should be less than Lim 2 etc... Lim (n) < Lim (n+1). If the limits are not configured validly then the “Write Loop Coefficients” button will be disabled.

6.5.2.5.1 Make Non-Linear table Linear – Apply Bin 0 to all.

If the user wishes to simply use the same Set and Alpha for all the limits, making it essentially Linear, then the user can select the convenience option “Apply Bin 0 configuration to all bins”. All the errors will be removed in this case even though all the Limits are the same. See figure below where all the bins are configured for Set C and Alpha 1.

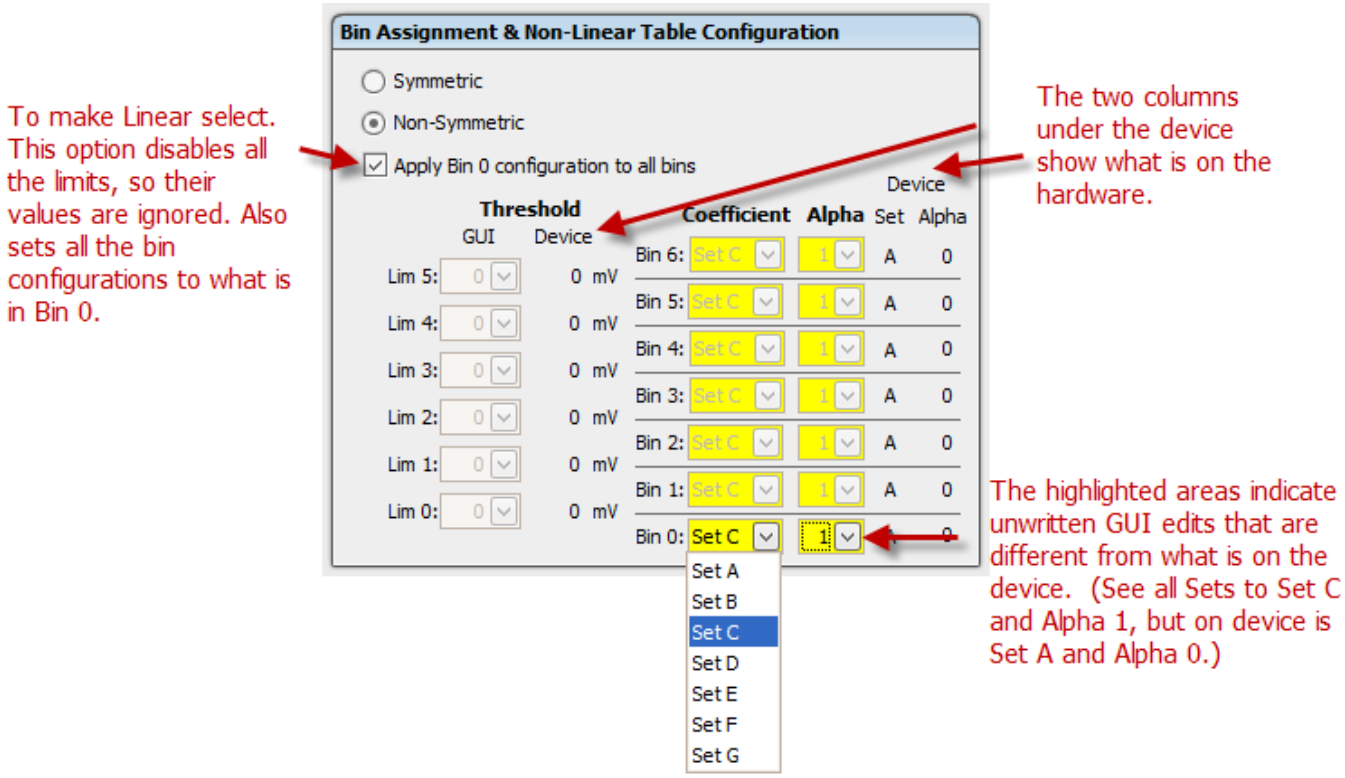


Figure 32 - Apply Bin 0 to all bins (Linear)

6.5.2.5.2 Non-Symmetric and Symmetric

There is an option to make the Limits Symmetric or Non-Symmetric. For Non-Symmetric the limits can be positive or negative. For Symmetric the limits specified must be positive since the symmetric part is automatic and negates all the positive limits. See figure below.

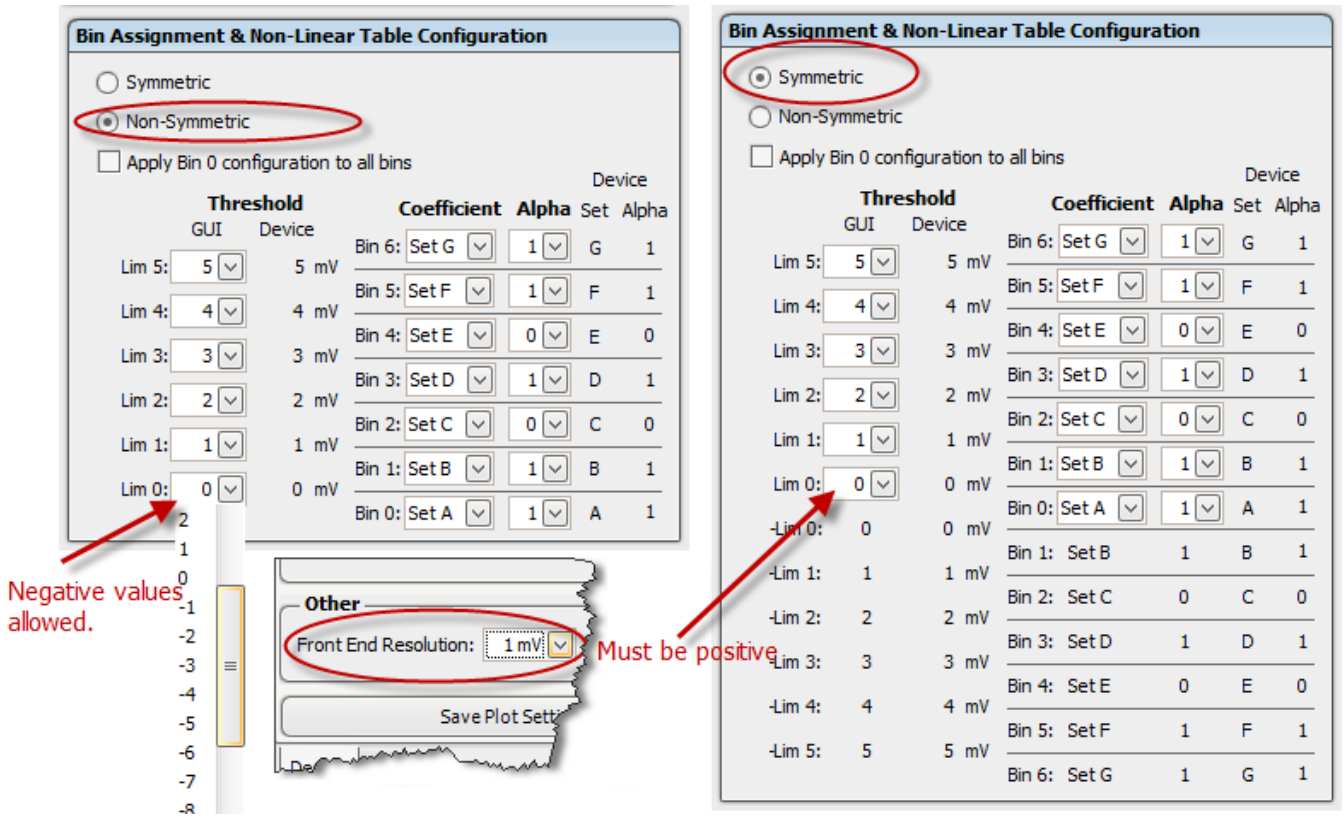


Figure 33 - Symmetric and Non-Symmetric

6.5.2.6 Writing Loop Coefficients, C code, Upload Compensation

After the user is satisfied with their configuration they can then proceed to writing it to the hardware. This does not happen automatically but requires the user to "Write Loop Coefficients." If there are errors they need to be corrected before the writing can proceed. What will be written? All the highlighted values are an indication of what is different from what is on the device so those values will be written. If the user wishes to discard all their GUI edits, or the highlighted values they can do a global discard by simply clicking "Upload Compensation." These buttons mentioned are located on the left side. The user can also view the C code that represents the coefficients in firmware by clicking "View Coeff 'C' Code". See figure below.

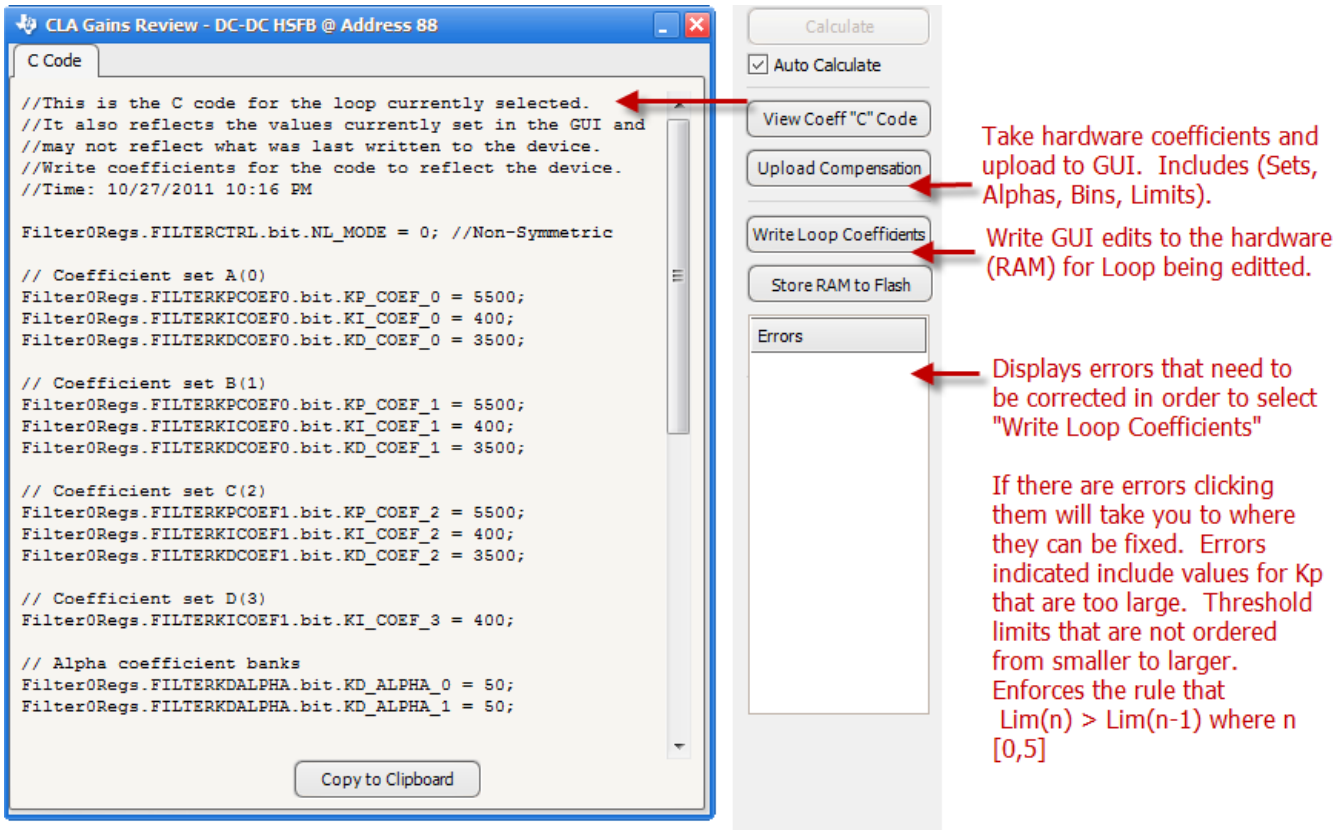


Figure 34 - Writing Loop Coefficients, and global reset of GUI edits to hardware coefficients

6.6 Status

The final mode is the status tab. It provides additional details on the type of fault or warning. Figure 35 - Status Mode shows a screen shot of this tab.

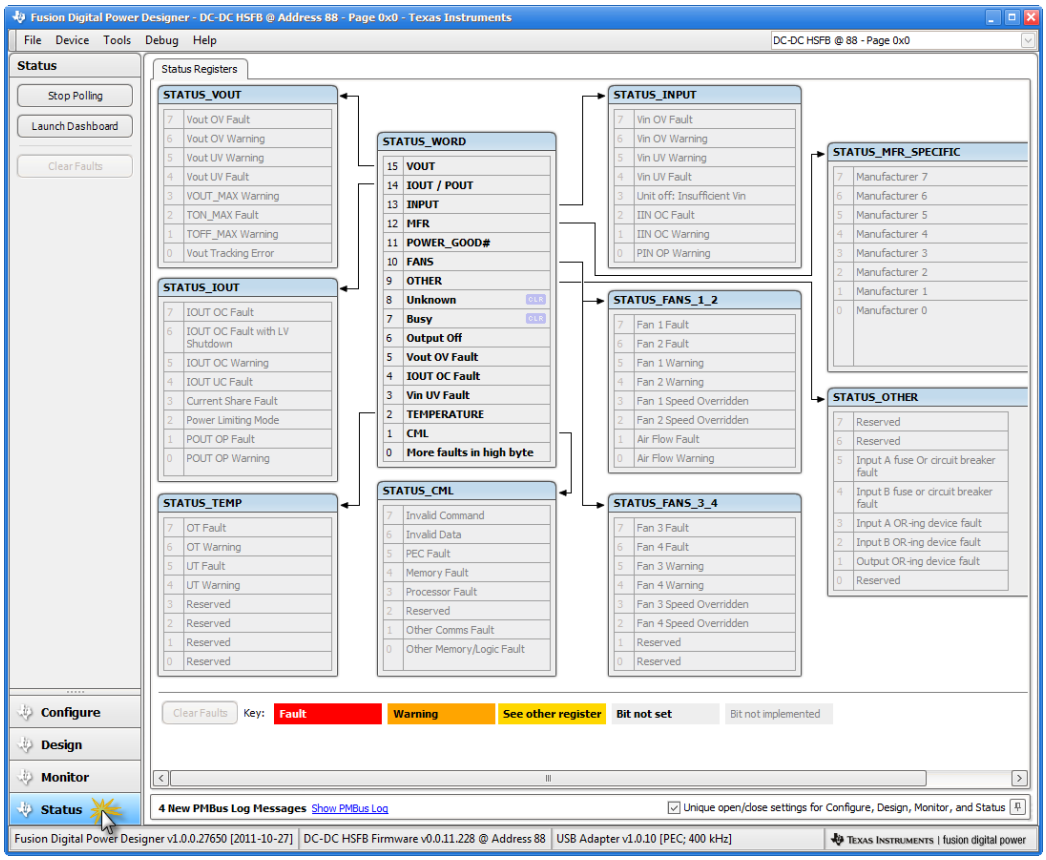


Figure 35 - Status Mode

6.7 Capturing the State of the Device - Saving a Project File

After editing PMBus commands in Configuration Mode or editing the Compensation, users can simply click the "Write …" button on the left to commit those changes to the hardware's RAM. They can then follow that with a "Store RAM to Flash" to save the hardware changes to Flash so that they would remain after the device undergoes a reset. If the changes on the hardware are not flashed then a reset would simply restore what is in flash and overwrite what was previously written to RAM.

However, the above only covers writing device-related parameters. What about the parameters set in the Power Stage in Design mode? These are not stored on the device. The only way these can be stored is by saving a "Project File". The Project File is an .XML file stored on the PC. Not only does it contain design parameters, but it also stores the current state of all PMBus commands. So it is a snapshot of the device and more.

To save a "Project File" simply click File> Save Project As …

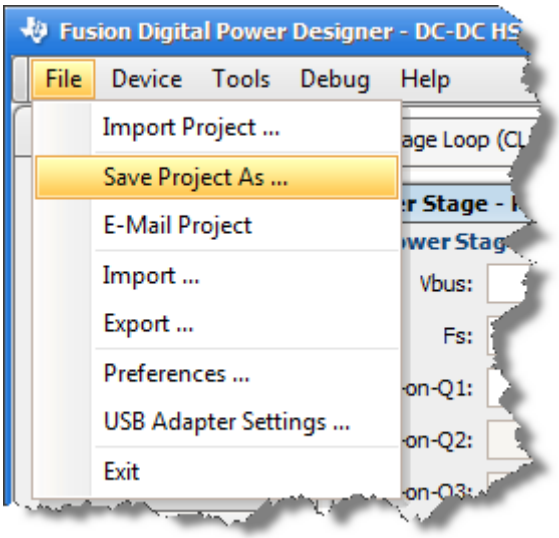


Figure 36 - Save Project File

What can be done with a project file? If a new device was hooked up to the PC the user can simply import the project file and write that to the device. The project file can also be used in Offline mode and act as a virtual device.

6.8 Miscellaneous Tools

6.8.1 Multi image

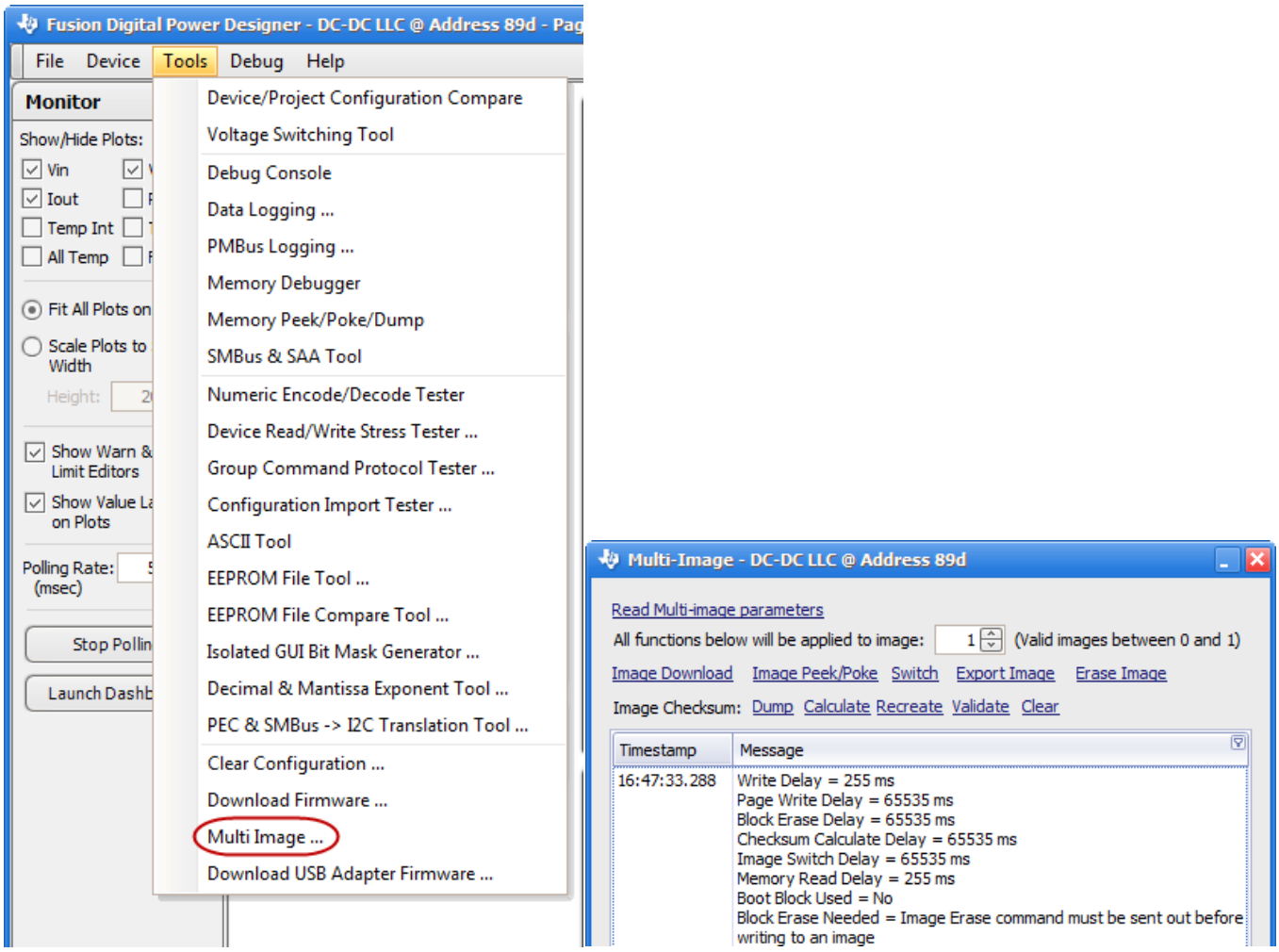


Figure 37 - Multi image

There are a number of other functions that can be performed from the “Tools” menu. Clicking “Multi Image …” will show a dialog with a number of multi image functions as shown in Figure 37. These functions are also available from the Device GUI and are covered in detail in “Section 7.5 Multi Image Functions”. One feature that can be observed in the Fusion Designer that isn’t seen in the Device GUI is the ability to download to a non-executing image and still observe the device monitoring various parameters. This can be seen in the background of Figure 38.

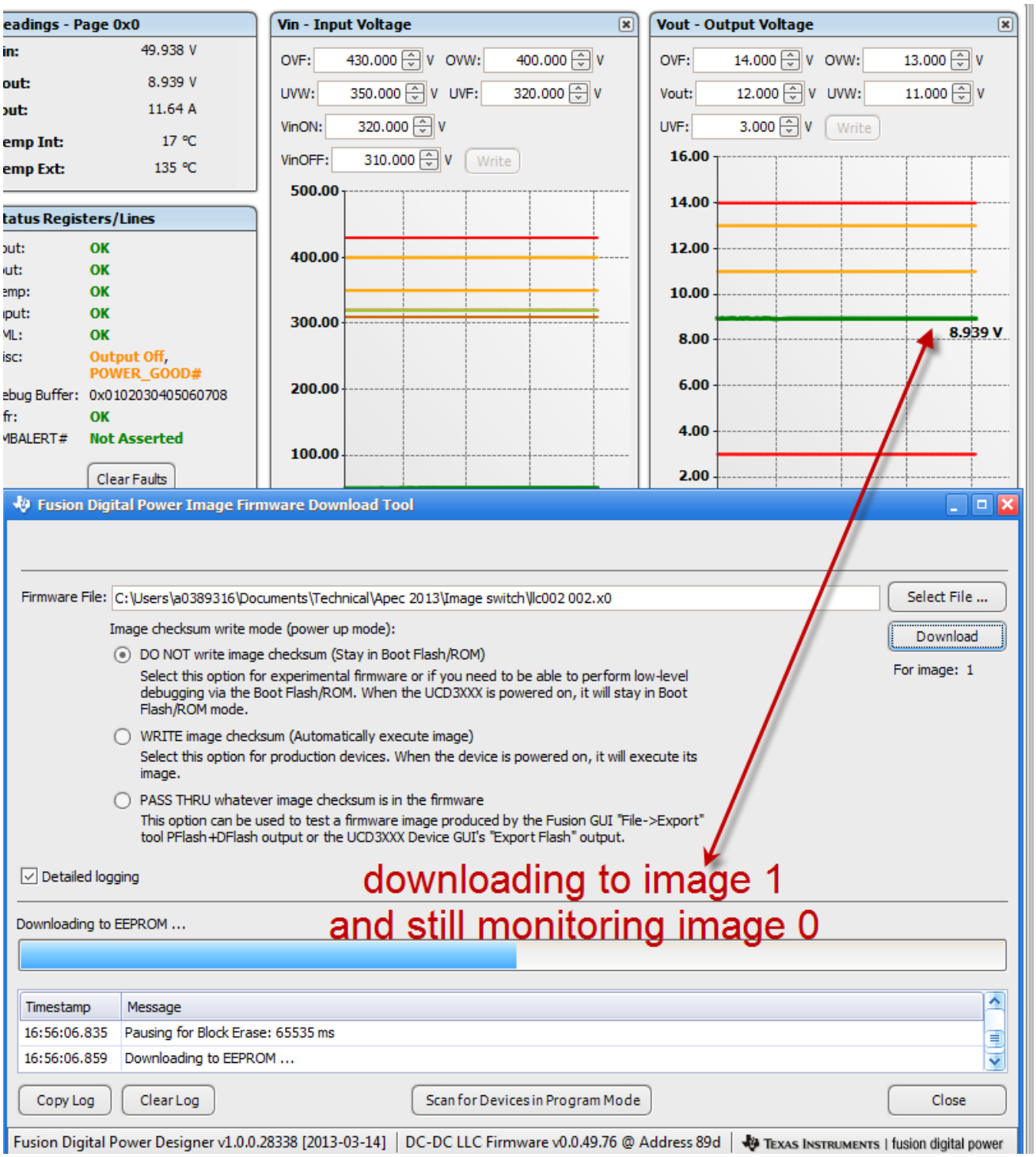


Figure 38 - Downloading to an image while monitoring at the same time

Clicking “Switch” in the Multi Image window will activate the new image. The GUI will need to restart to load the new image. NOTE: The power supply is not reset.

6.8.2 Isolated Bitmask Tool

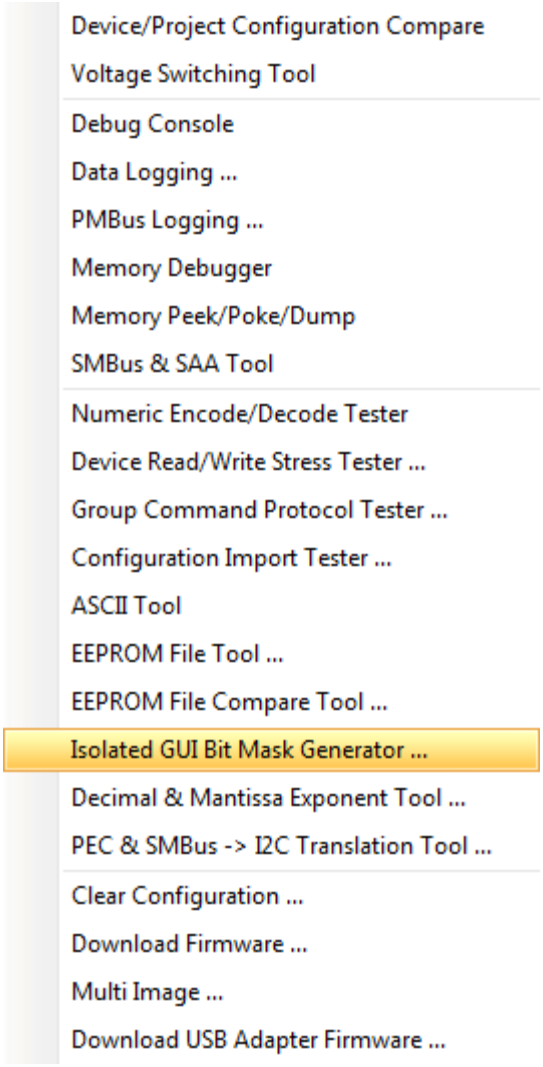


Figure 39 - Tools> Isolated GUI Bit Mask Generator …

The “Isolated GUI Bit Mask Generator” is also detailed in the part of this document describing the functions of the Device GUI in “Section 7.6 Isolated Bitmask Tool.” One feature that is available in the Online Fusion Designer that is not in the Device GUI is the ability to view the PMBus command bitmasks set in the firmware. The user simply clicks “Upload bitmask from device” as shown in Figure 40. This is a quick way to debug why a command may not be visible in the configuration tab if the reason is the command’s bit was not set in the bitmask.

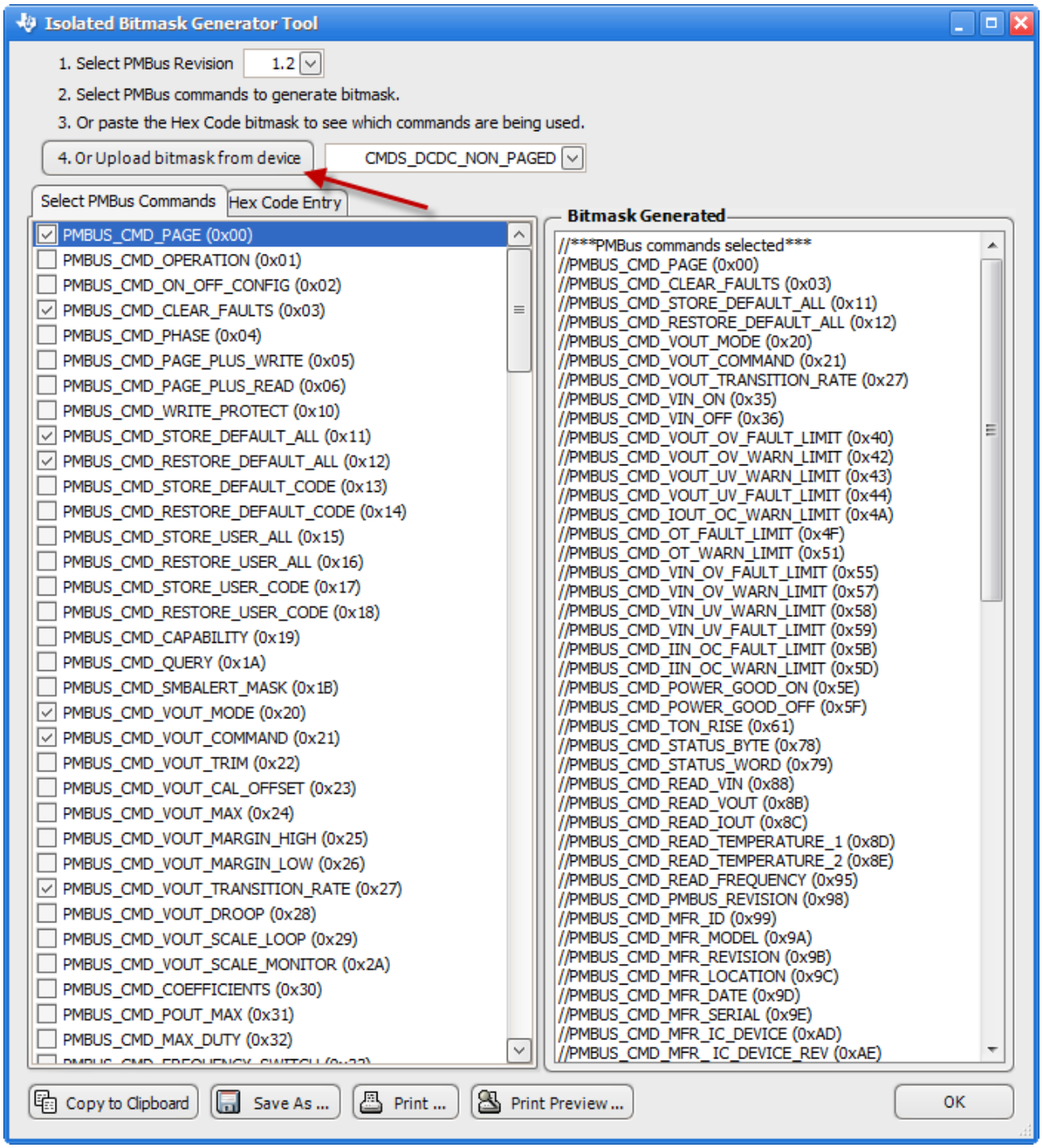


Figure 40 - Isolated Bitmask Tool in Fusion Designer(Online)

6.9 Offline Mode

So far all the discussion has been related to communicating with a device that is connected and online. There is also a concept of working with the device in offline mode. This is done by working with a previously saved Project File as discussed in the last section or by working with Sample Project Files that are already embedded in the GUI. In offline mode the user can write PMBUS commands to a “virtual device” and they can also do modeling in Design mode. When the user gets a device they can simply import this project file that they’ve worked offline with and sync the device to that.

6.9.1 Starting in Offline mode

To start offline you can click the other shortcut that came when the GUI was installed. See following figure.

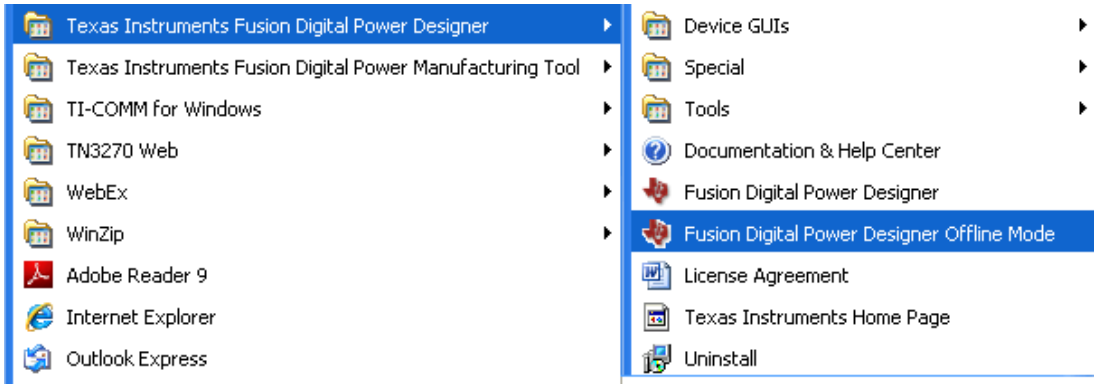


Figure 41 - Starting in offline mode

Another way to start in offline mode is to unplug any connected devices and start the GUI normally with the other shortcut. This will cause the GUI to scan for devices and then upon the fail will prompt the user to Retry, or work in offline mode.

6.9.2 Open Existing Project File

In offline mode the user selects from three options. The first option is to open an existing project file that has been previously saved.

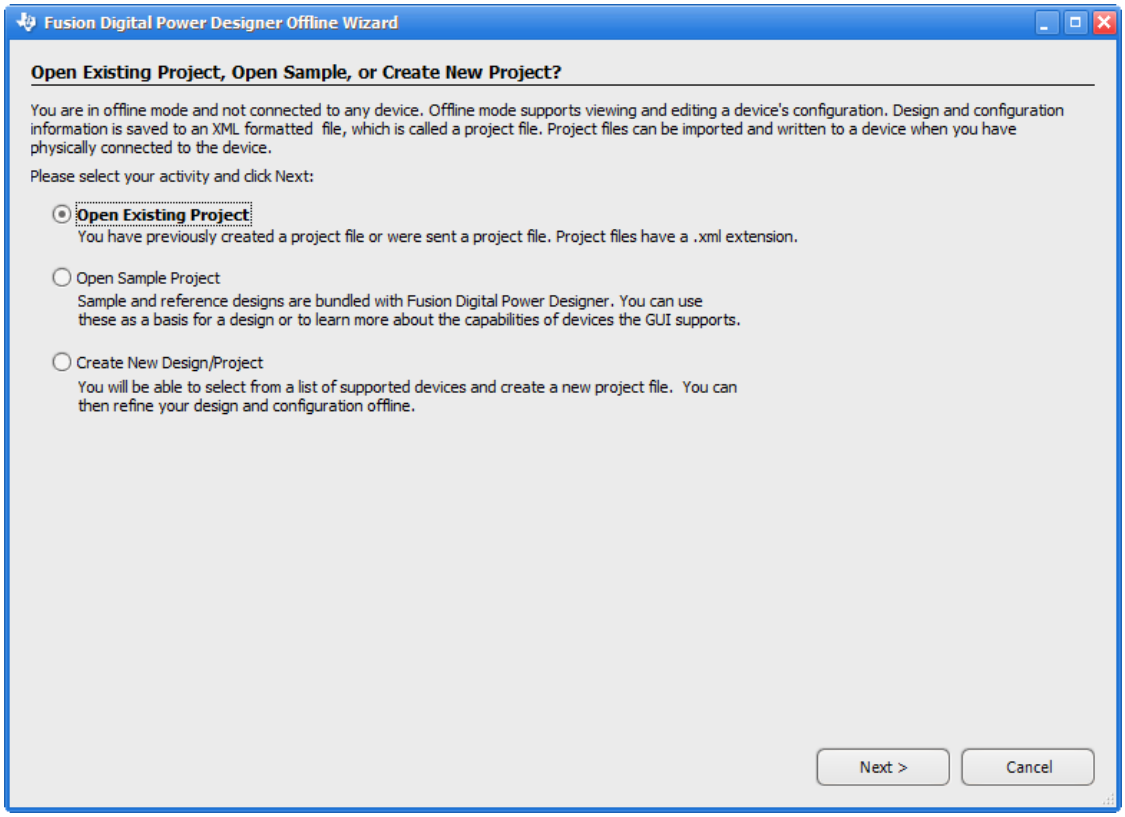


Figure 42 - Offline options

6.9.3 Open Sample Project

The user can also open a sample project file and work with that. They can then save that afterwards as a project file to their PC and use it later to import to a device. The following Sample projects are available at this time. Isolated UCD3138 (064) users should click “UCD31xx Isolated Digital Power Controllers” See figure below.

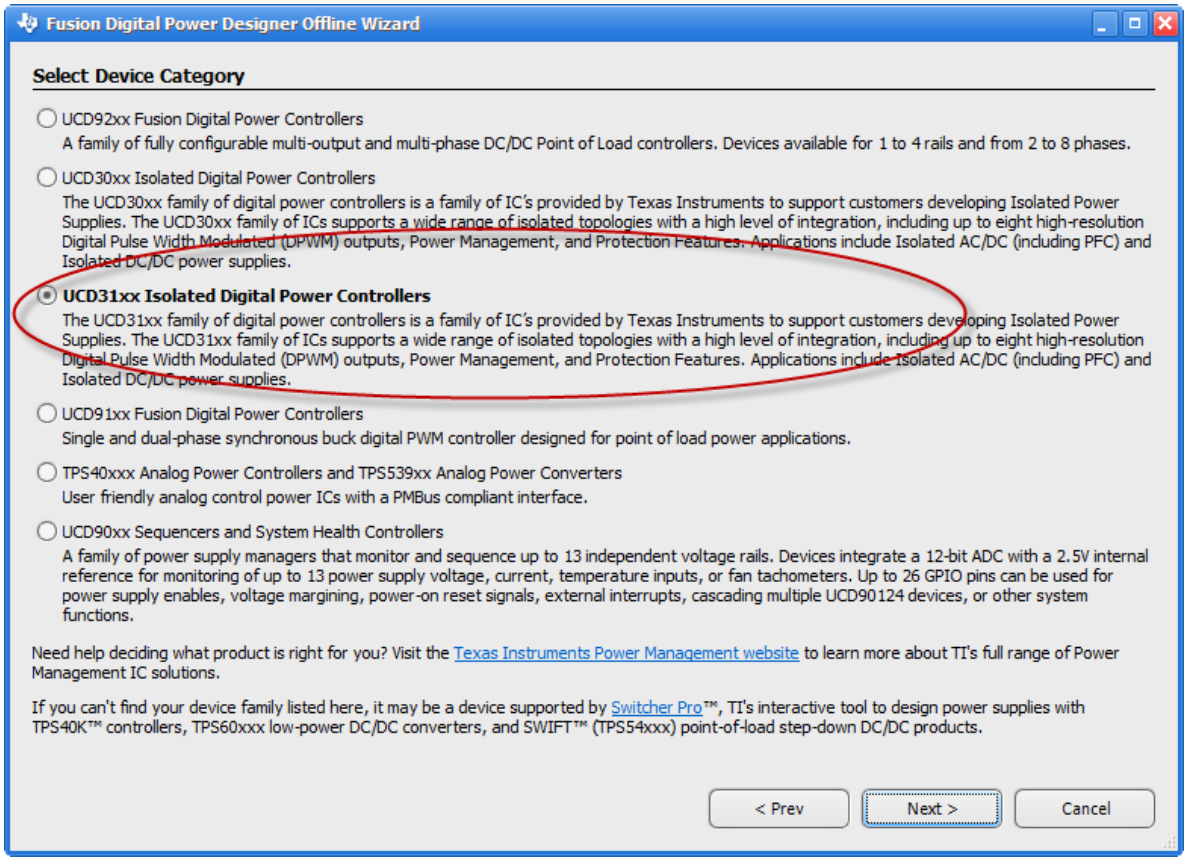


Figure 43 - UCD3138(064) sample projects

After clicking "Next" the sample projects will appear. This list will increase as new topologies are supported.

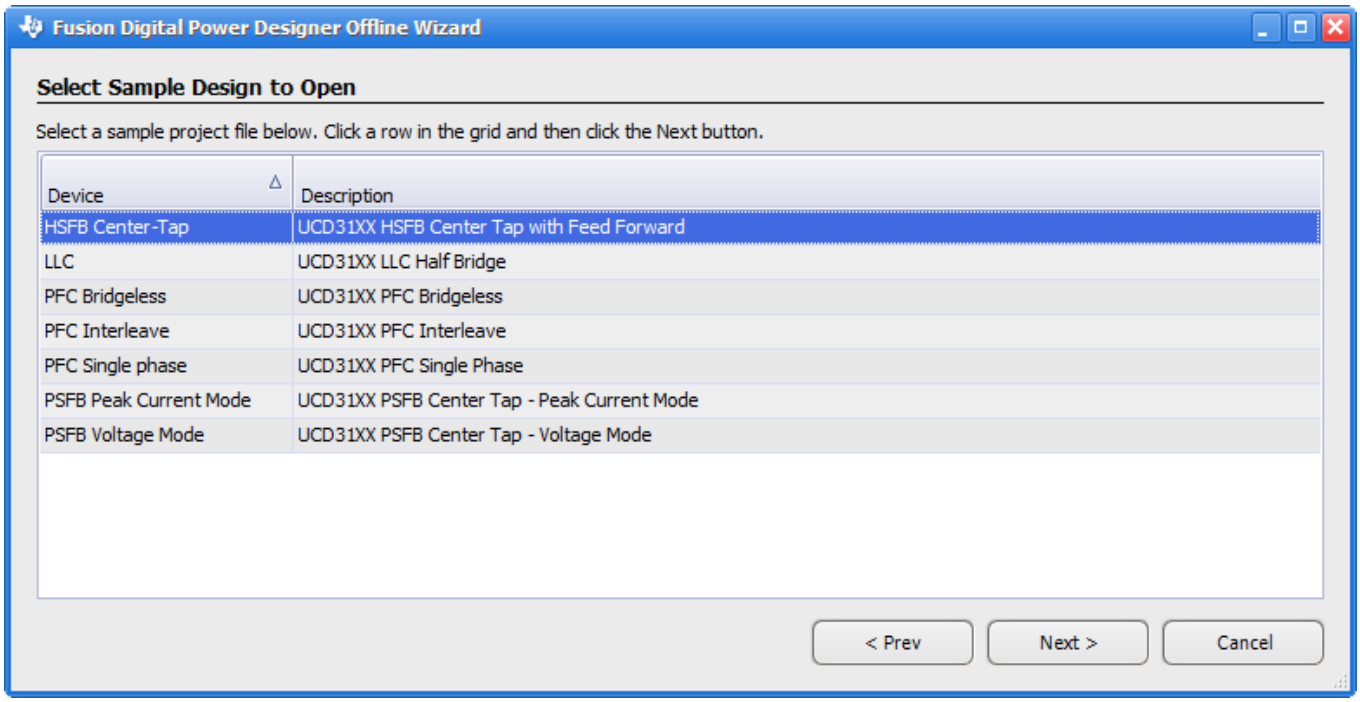


Figure 44 - Offline sample topologies

7 Device GUI (Engineering GUI)

In the previous section the Fusion Designer GUI was described. In this section the Device GUI will be described. The device GUI provides an entry point to a number of important development tools indispensable for working with the UCD3138(064, A64) devices. Users will also find out that a number of these tools are also available in the Designer GUI under the Tools menu. Users may use whichever entry point they wish to launch these tools. The following figure shows the entry point to some of the tools that will be described now from the Designer GUI previously discussed. Note you will need to enable the "Protected Features" with the password “forestln” in the Designer GUI to see this. See Section 6.2 Enable GUI Protected Features. This password should also be used for the Device GUI if prompted for a password.

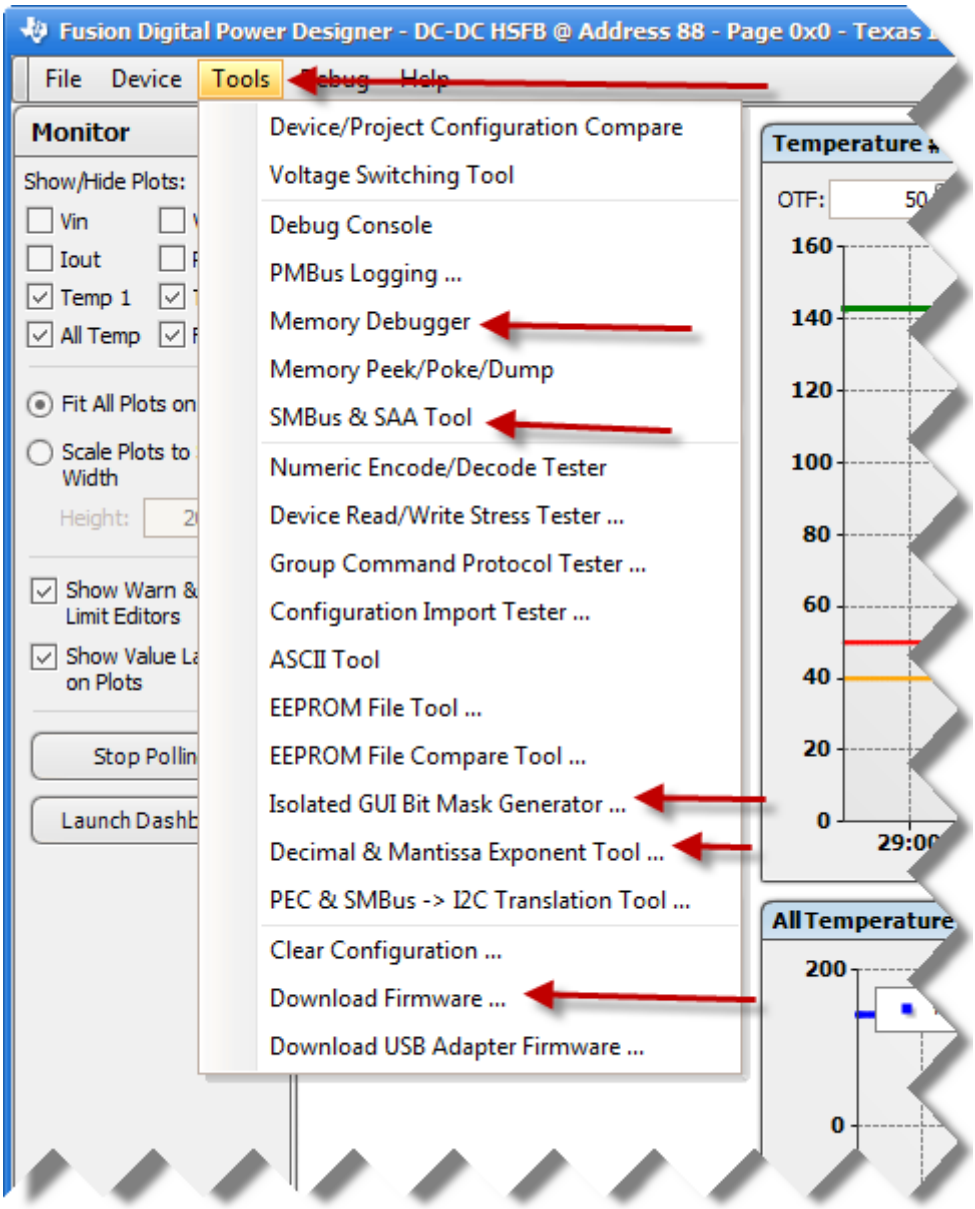


Figure 45 - Designer GUI Tools menu

7.1 Launching Device GUI

During the installation users had the option to create a shortcut for the UCD3xxx Device GUI. If that option was not selected the UCD3xxx Device GUI can be accessed from the Start Menu.

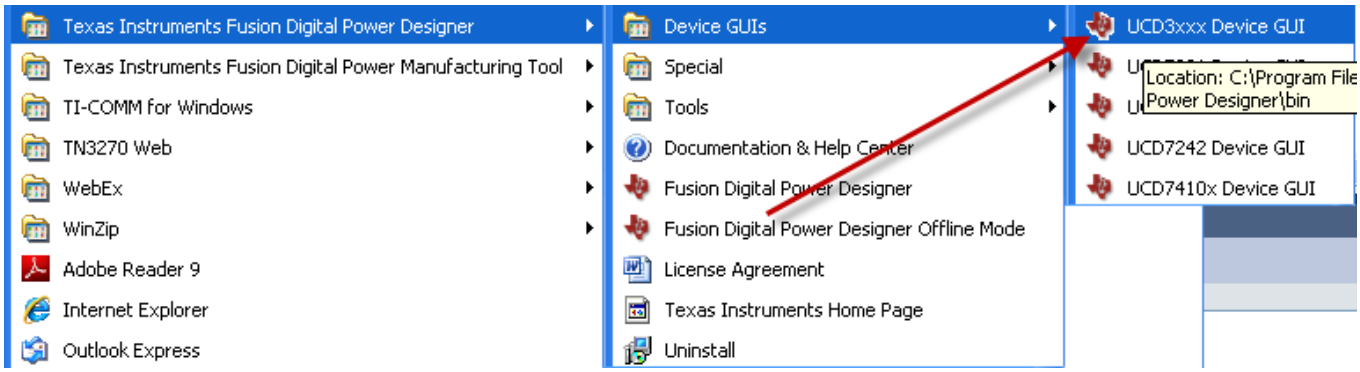


Figure 46 - Opening UCD3xxx & UCD9xxx Device GUI

The Device GUI looks as follows,

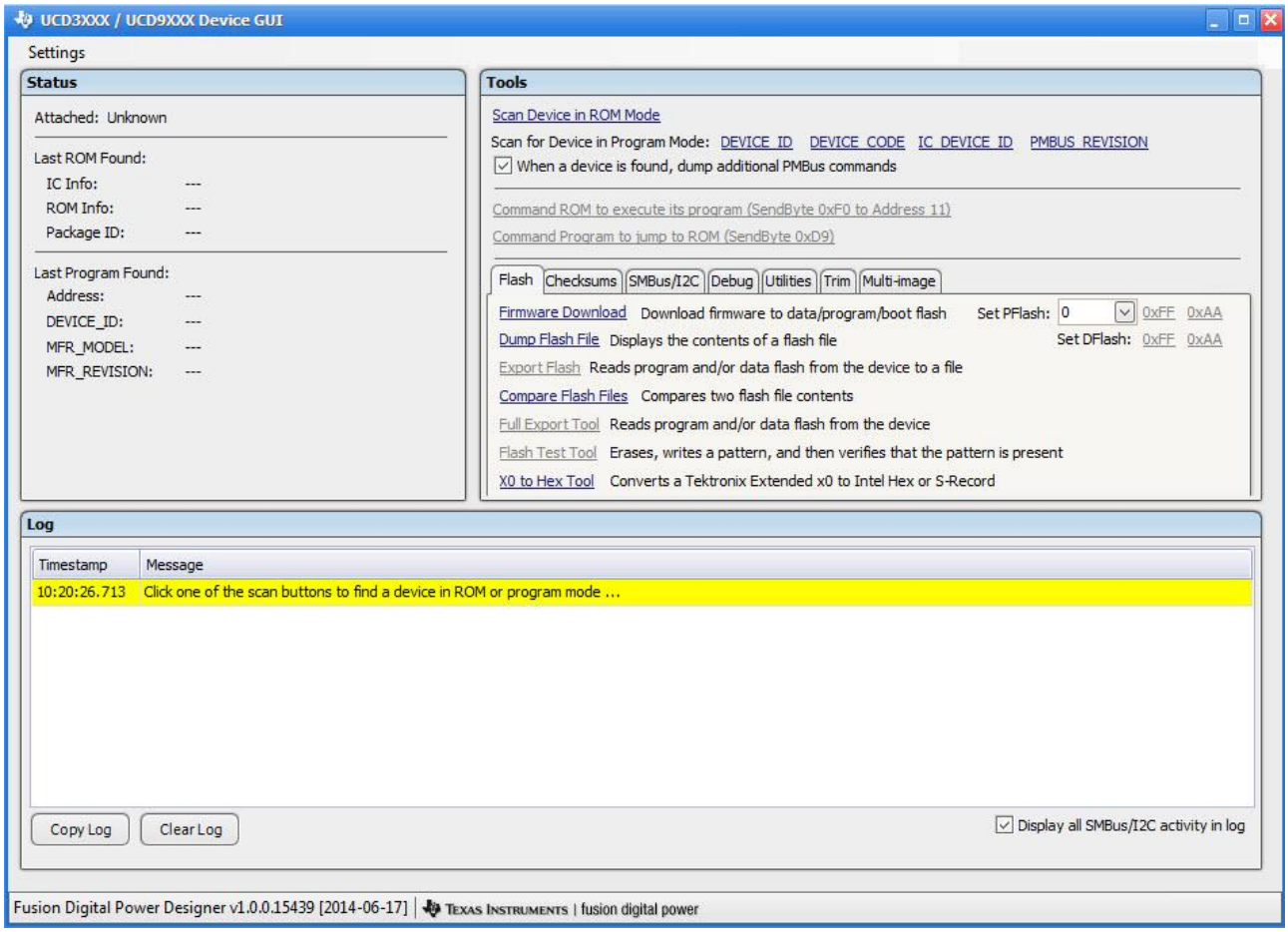


Figure 47 - UCD3XXX Device GUI

After the Device GUI starts up there are a number of links that are enabled and some disabled. Which links are clickable depends on whether the GUI is in ROM mode or Program mode. To start off the user should click “Scan Device in Rom Mode” if the device is in ROM mode. If the user clicks this and the device isn’t in ROM mode a message will be logged that there is No ROM detected. If the device is in Program mode then the user should select “Device ID” or “PMBus REVISION”.

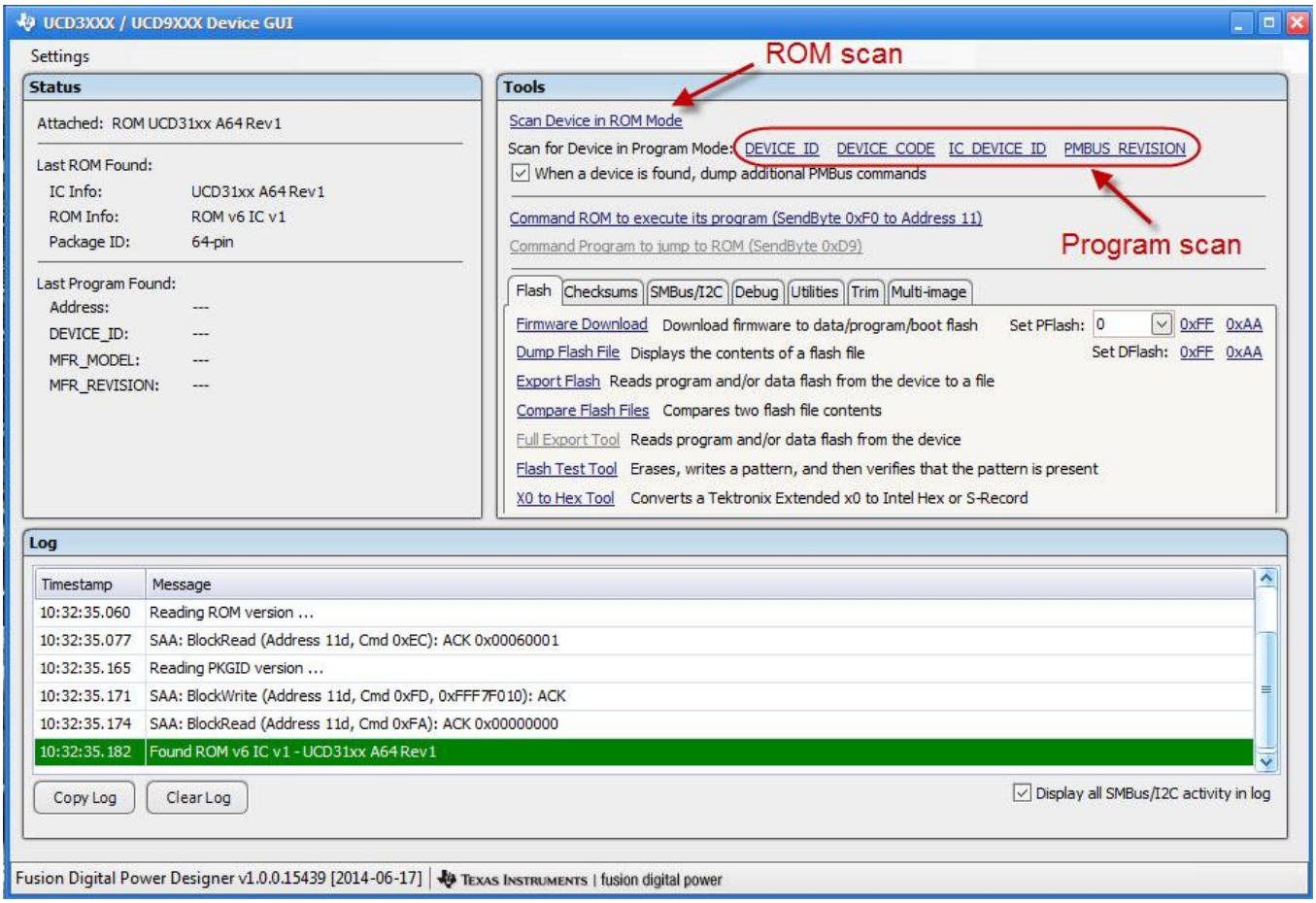


Figure 48 - Program scan and Rom scan

7.2 Moving between ROM and Program mode

To move between ROM mode and Program mode the user can select the following links respectively:

- Command ROM to execute its program (SendByte 0xF0 to Address 11)
- Command Program to jump to ROM (SendByte 0xD9 to Address xx)

Figure 49 displays these links in the Device GUI.

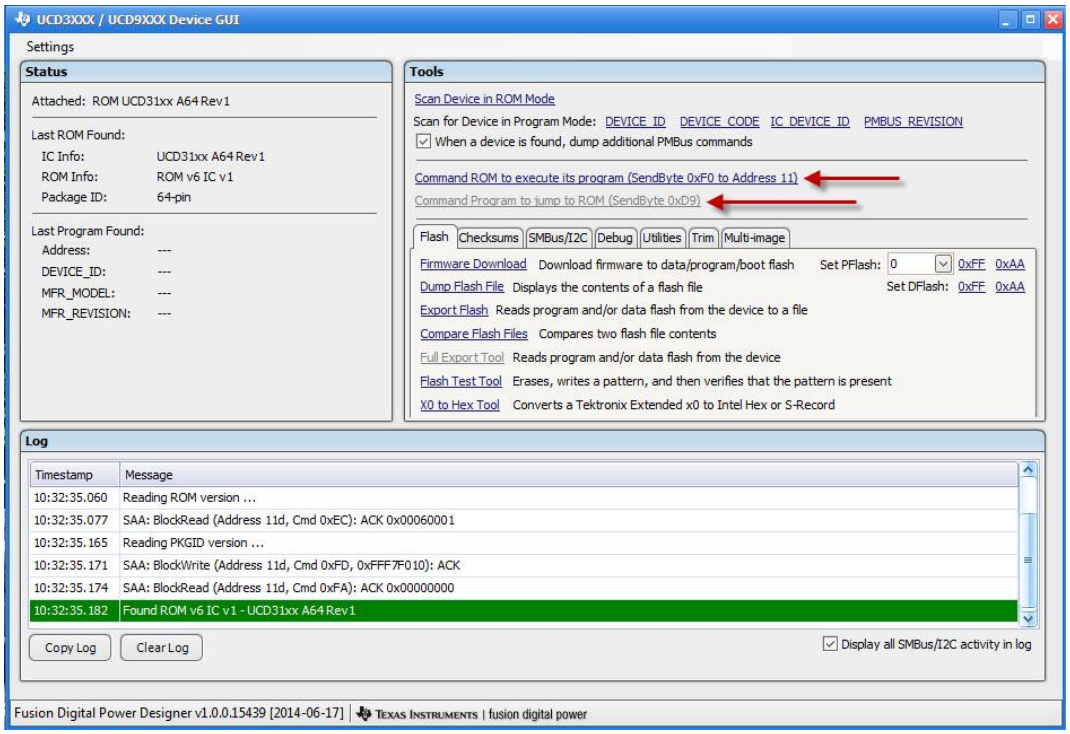


Figure 49 - Moving between ROM mode to Program mode

7.2.1 ROM mode to Program mode for multiple flashes

In devices that have multiple flash blocks, the user has more than one option when commanding ROM to execute its program. This applies to devices that allow execution from more than one block. For example in the UCD3138064, a device with two flash blocks, the user would send a different byte depending on which block they wanted to execute. They would send byte 0xF0 to execute Block 0. This would be the same byte to send if the firmware they wanted to run was the size of both blocks. This is due to the address beginning at the same place as Block 0. To execute Block 1 the user would send 0xF7. See Figure 50 and Figure 51 showing what to click to send the device from ROM to Program mode. The options for the two blocks appear after scanning for the device in ROM mode.

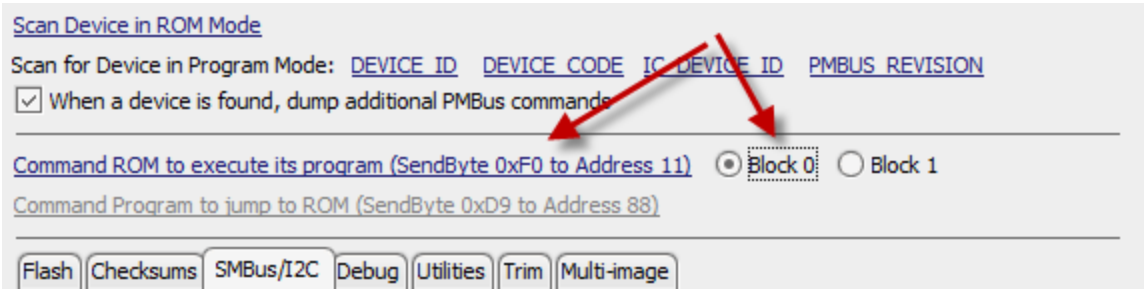


Figure 50 - Executing Program for Block 0 (0xF0)

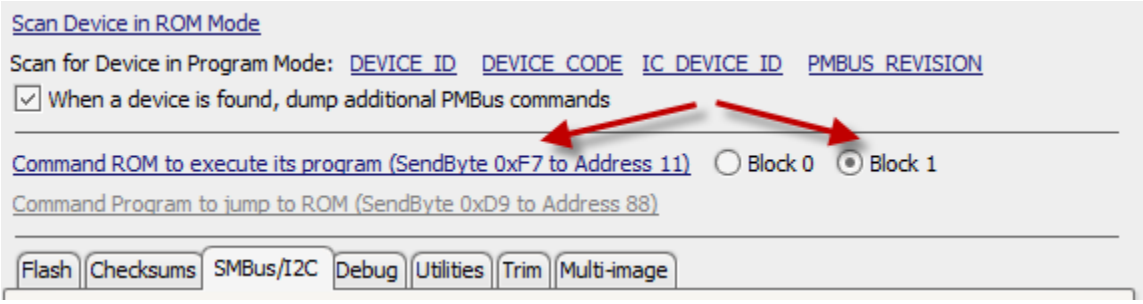


Figure 51 - Executing Program for Block 1 (0xF7)

7.3 Firmware Download Tool

To open the Firmware Download tool click “Firmware Download” as shown in Figure 52.

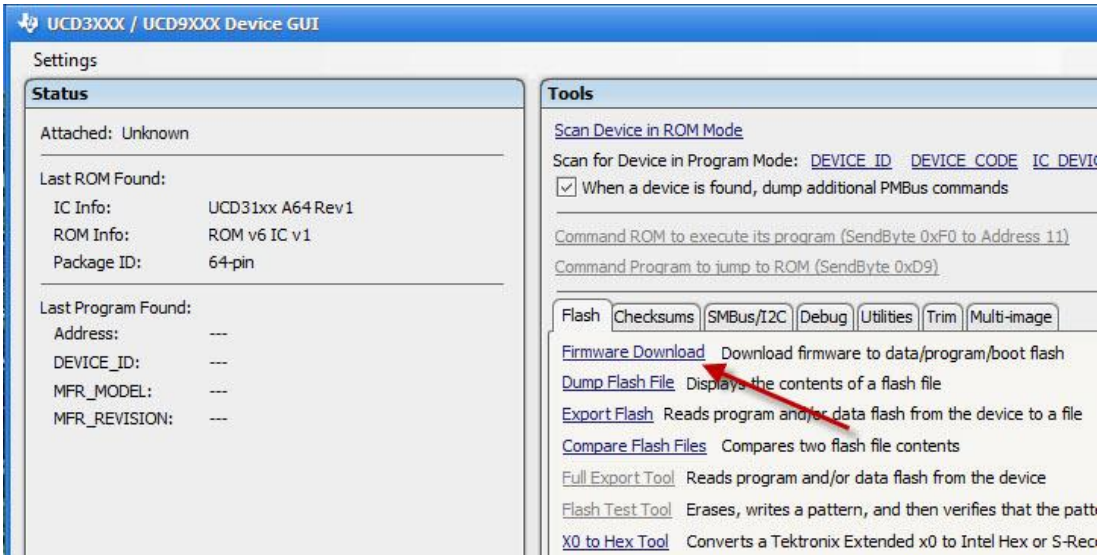


Figure 52 - Firmware Download

The firmware download screen launched will differ due to the available block configurations specific to each IC.

For UCD3138 the screen will look as follows:

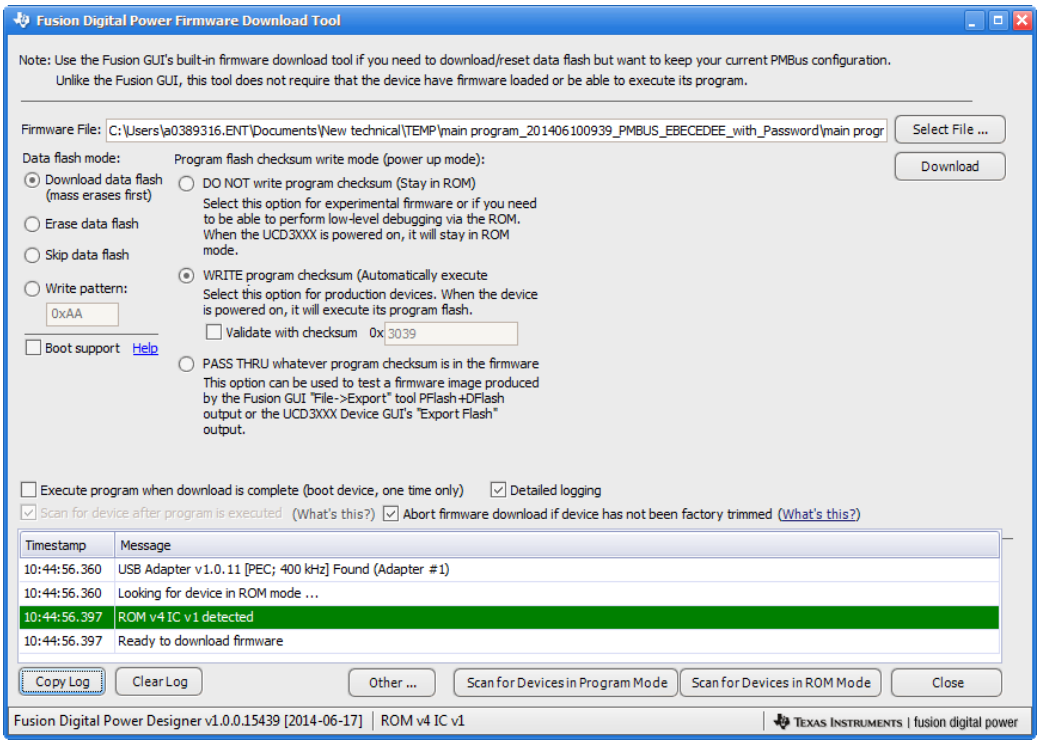


Figure 53 - Firmware download screen for UCD3138

For UCD3138064, notice the flash block selection available:

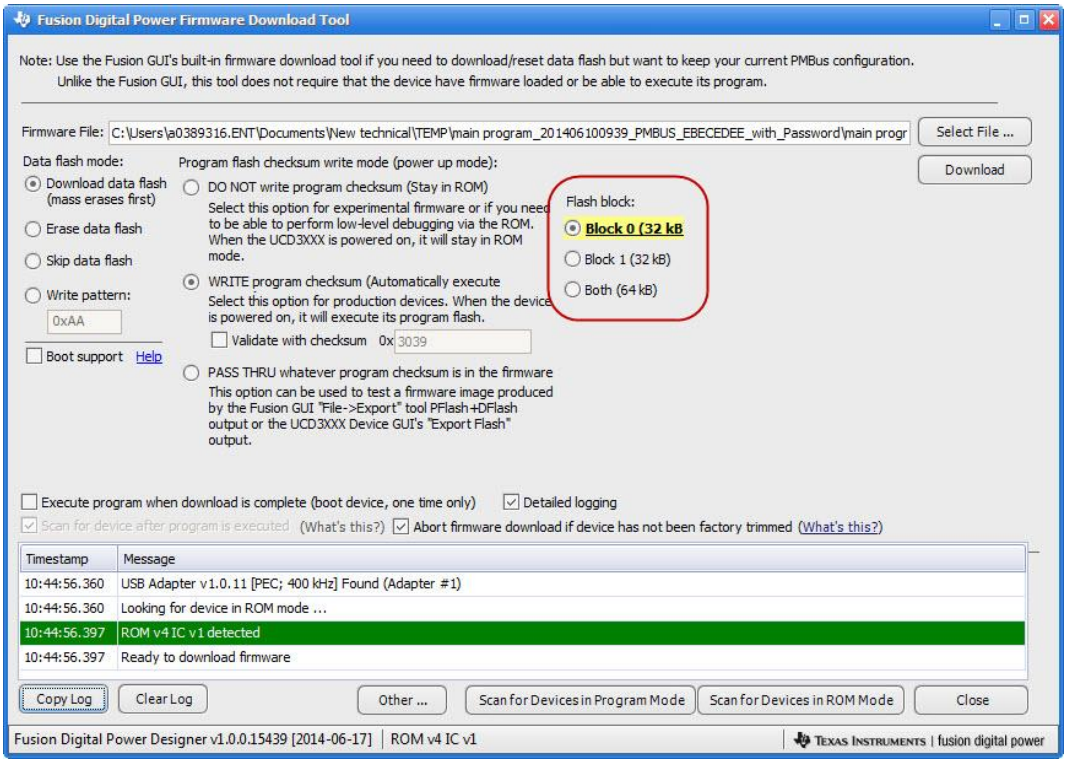


Figure 54 - Firmware download for UCD3138064

For UCD3138A64:

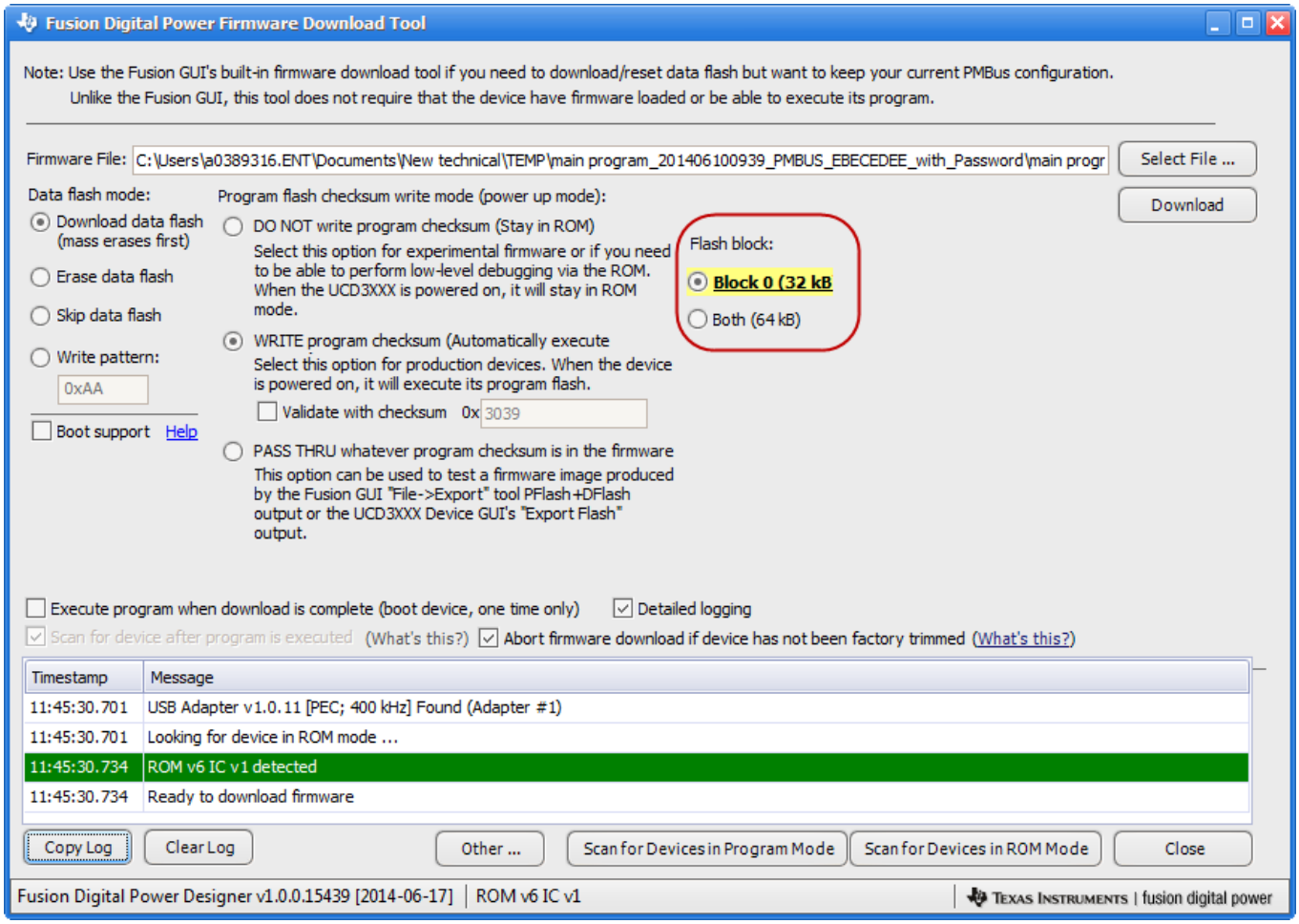


Figure 55 - Firmware download UCD3138A64

The user can choose what they would like to download with regards to the Program Flash, and Data Flash.

WARNING: It is important to note that if the program checksum is written, the device will boot up in program mode upon a reset. This may be a source for a device lockup if the firmware has not implemented the commands to jump back to ROM. Hence, it is advised not to write the program checksum for firmware in initial stages or implement the commands to jump back to ROM first.

For devices that have multiple flashes an extra set of radio buttons will appear for the user to decide which block to download to as shown in the previous figures.

The user picks the firmware file and clicks download.

NOTE: Sometimes this tool may be launched when the device is running in program mode. In that case they can use the button “Other …” at the bottom to put the device in ROM mode so that they can proceed with the download.

7.3.1 Boot support

To write firmware to the boot flash click “Boot support” as shown in Figure 56.

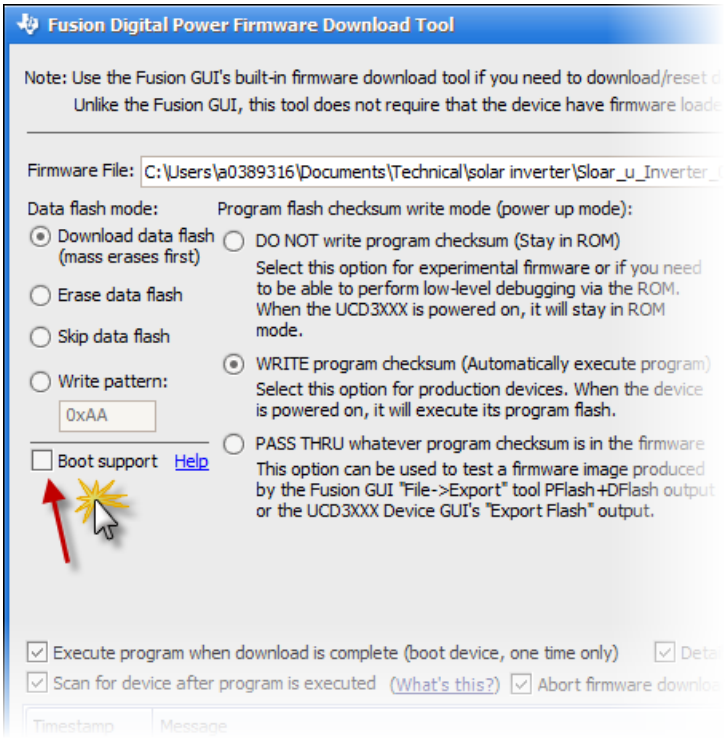


Figure 56 - Boot support

The following screen shows the new options circled below related to boot flash.

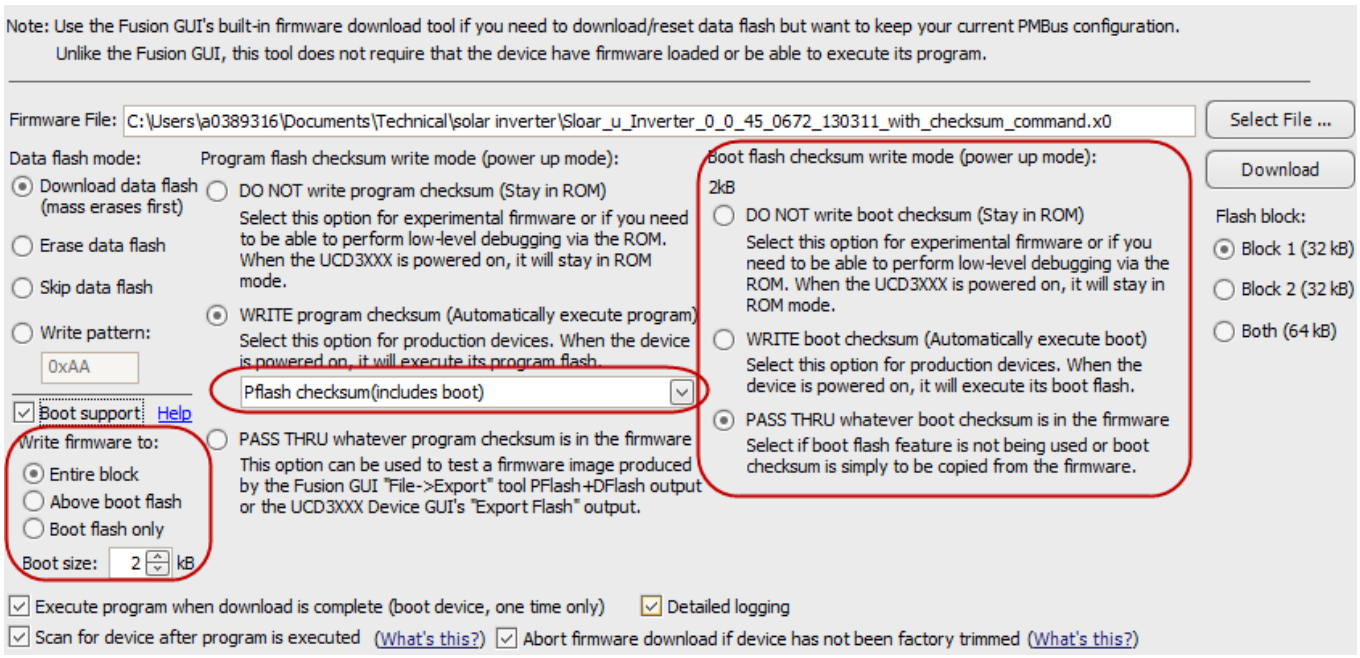


Figure 57 - Bootflash options

Each of the options will be described. Figure 58 shows the “Help” screen describing the various options that the firmware can be written to and the checksums related to it.

The first option to configure for Boot Support is “Write firmware to:” as circled in Figure 57.

- Write firmware to “Entire block”: The program and the boot will be taken from the firmware file.
- Write firmware to “Above boot flash”: Only the program will be taken from the firmware file.
- Write firmware to “Boot flash only”: Only the boot will be taken from the firmware file.
- “Boot size”: Can range from 2 kB to 31 kB. For a boot size of 2 kB then there is only one option for the boot flash checksum as shown in Figure 57. If the boot size is greater than 2 kB there is another option to set a checksum for the remainder of the boot flash as shown in Figure 59.

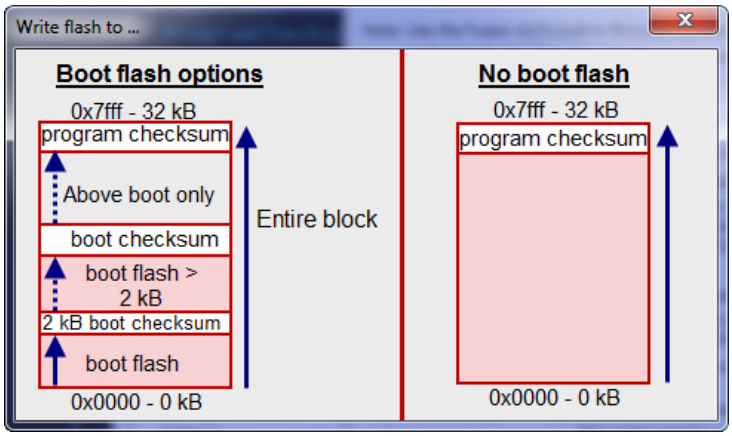


Figure 58 - Firmware writing options

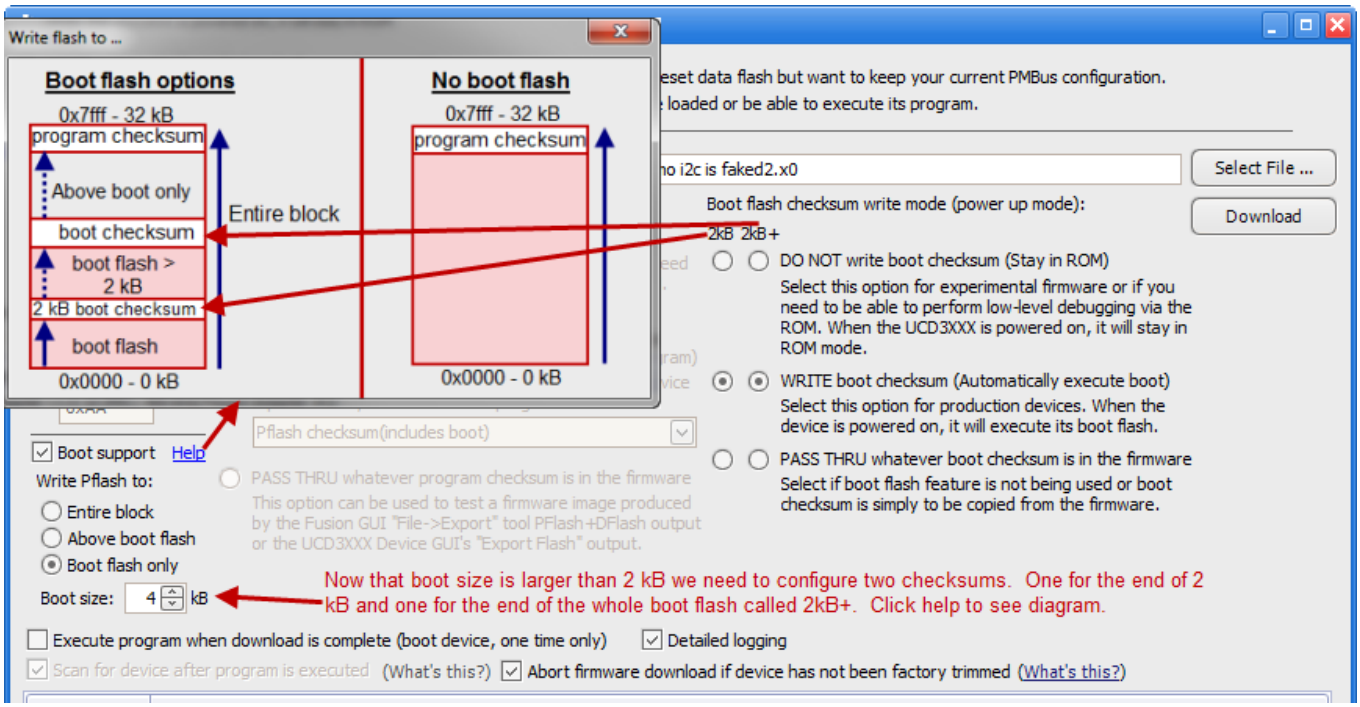


Figure 59 - Two checksums for boot flash greater than 2kB

There are two options if the user is writing the program checksum after downloading program flash. The user needs to specify if the checksum calculated should include in addition to the program, the boot or not. See Figure 60.

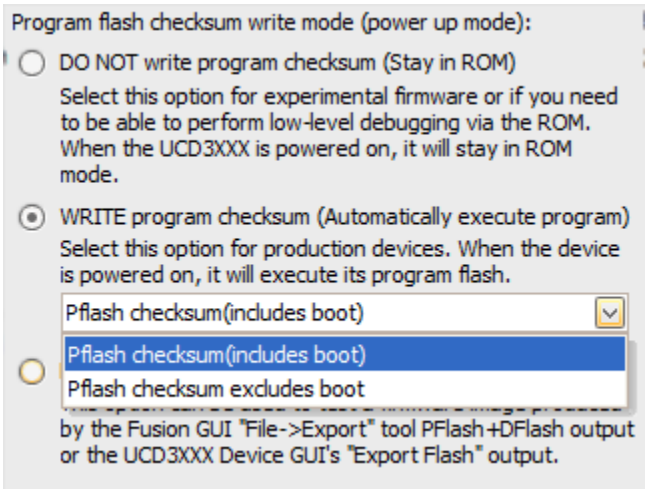


Figure 60 - Writing pflash checksum options

7.3.2 Data flash download

7.3.2.1 Data flash download options

There are three options regarding downloading of data flash.

7.3.2.1.1 Download

The option “Download data flash” writes the data flash portion defined in the .x0 file to the data flash location on the device. Before the writing of data flash, a mass erase is issued where all the pages are cleared simultaneously.

7.3.2.1.2 Erase

The option “Erase data flash” simply issues the mass erase without downloading the .x0 file.

7.3.2.1.3 Partial download

The second option is “Download partial.” For this case the user must specify an initial start page index and a final page index of the pages defined in their .x0 they wish to download. The data flash pages outside the range of these indices on the device will not be edited.

7.3.2.2 Download partial flash clarification

7.3.2.2.1 Erase time

Before the continuous set of pages (defined by the start and final page indices) are written, the page erase command is issued sequentially beginning with the “Start page.” This erase is done sequentially, one page at a time, including the appropriate wait time after a page erase has been issued. Therefore, if there are 10 pages and “y” is the wait time per page erase, then the total wait time needed would be 10y. For the first option above, the wait time is only “y”, as the mass erase applies a simultaneous erase to all the pages as opposed to the sequential erase in this option.

7.3.2.2.2 Identifying the pages

Once the data flash beginning address, and the address of the data variables with their respective data lengths are known then finding the start page index and final page index for a partial download can be found as follows:

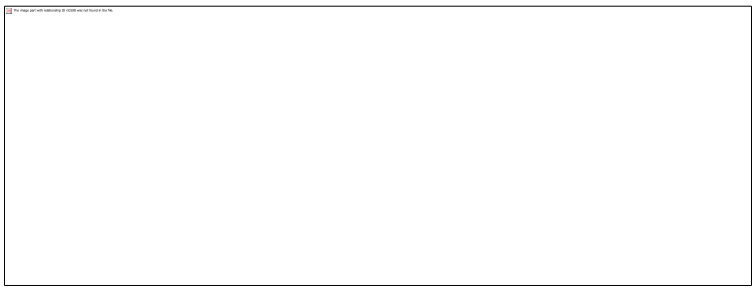
Start_page_index = (data_variables_begin_address – data_flash_begin_address)/0x20

Final_page_index = Start_page_index + (sum_of_data_lengths/0x20) -1

Note: usually the data that is being partially downloaded to the device is defined in the firmware along page boundaries.

7.3.2.2.3 Helpful tools

The “Memory Peek/Poke” tool is helpful for observing the flash.



After the user specifies the begin and end address they can view the flash contents in the “Memory Dump” tab.

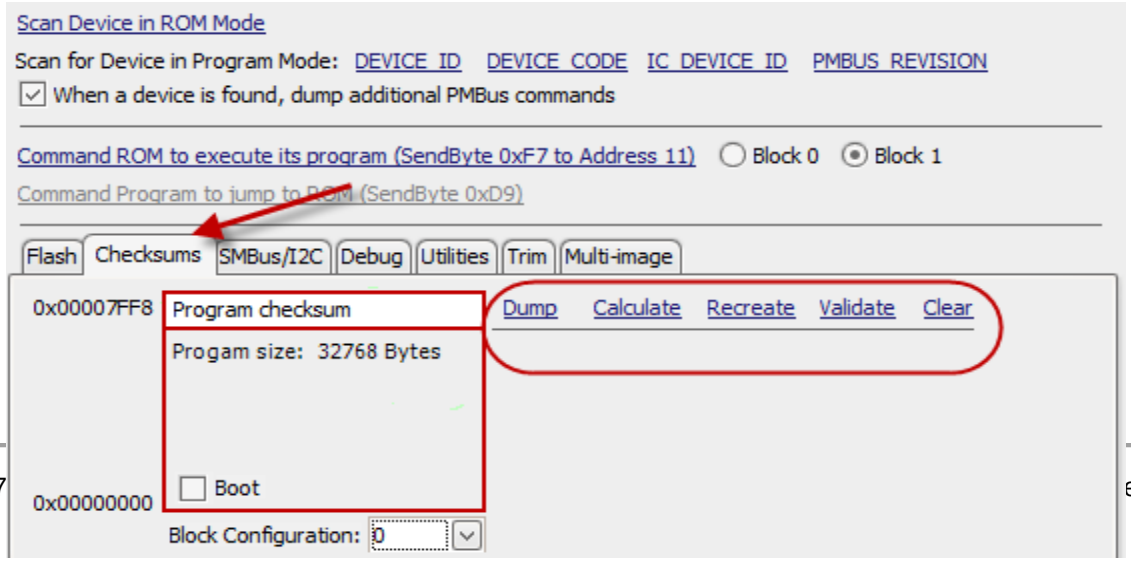
In the image above the data flash(0x18800-0x18FFF for UCD3138) was set initially to all 0xAA. Then a data flash partial download was done where the start and final page indices were defined to be 3 and 7 respectively.

Note: To set the data flash to 0xAA click the 0xAA link found in the “Flash” tab as shown below:

7.4 Checksum functions

In the Checksums tab there are a number of functions available to view, calculate, create, validate and clear checksums on the device as shown in

Figure 61. The tab visually displays the checksums to more easily apply the appropriate function. Depending on the boot flash size or whether boot flash is even needed the visualization of the checksums will update as shown in the following figures.



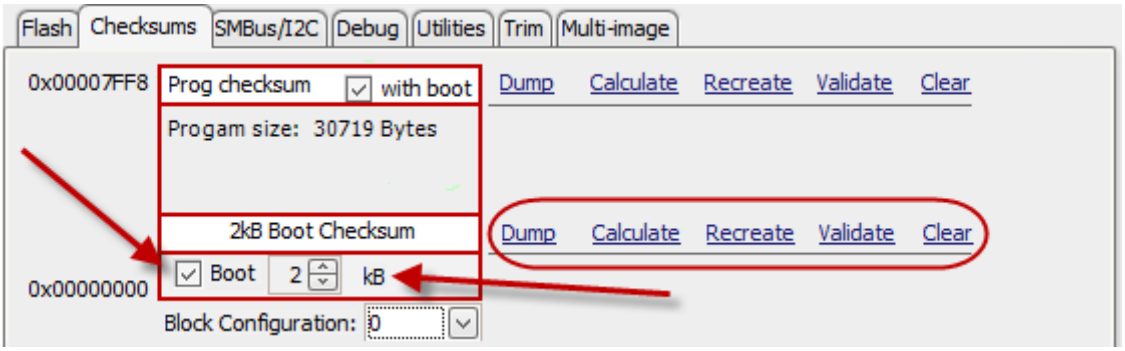


Figure 61 – Checksum functions

Figure 62 - 2 kB boot checksum functions

7.5 Multi Image Functions

Click the Multi-image tab to use the functions for firmware that implement multiple images. See Figure 64 .

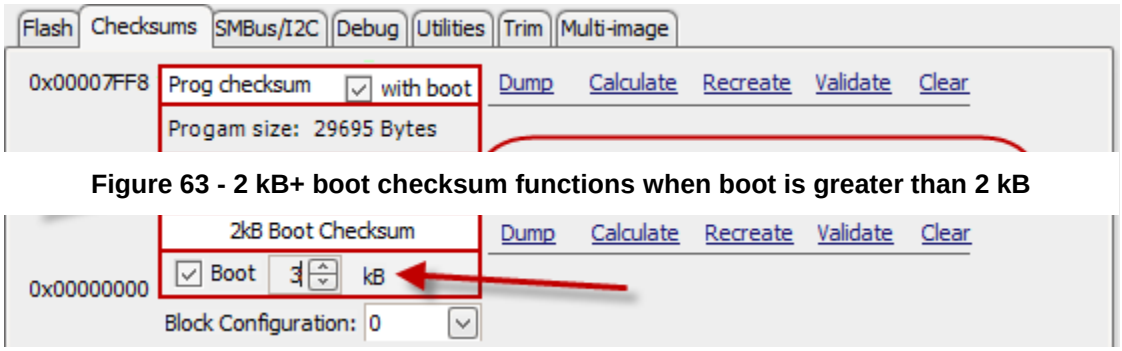


Figure 63 - 2 kB+ boot checksum functions when boot is greater than 2 kB

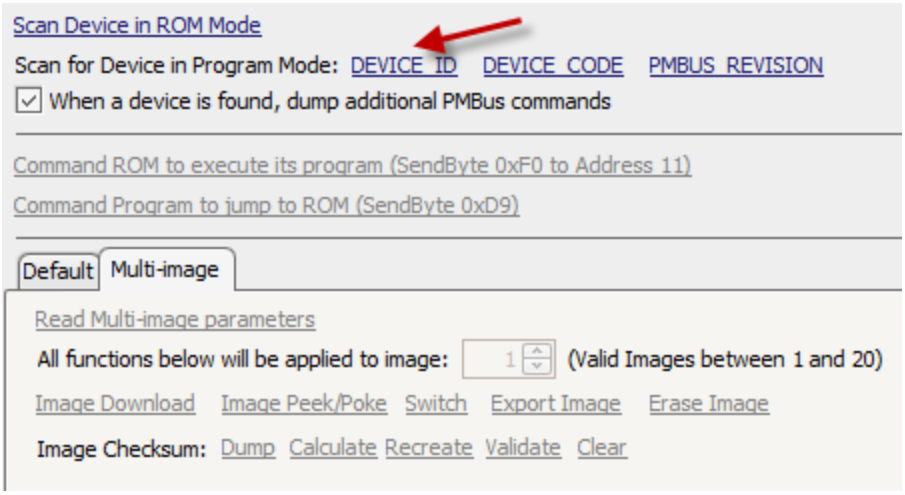


Figure 64 - Scan for Device ID to activate Multi-image

The Multi-image tab provides functions for working with other images while an image is executing. After scanning for "Device ID" as shown in the figure above, the user will see the link "Read Multi-image parameters" become enabled. The user then clicks this to read important parameters that describe the images and how the GUI will interact with them as shown in Figure 65 and Figure 66.

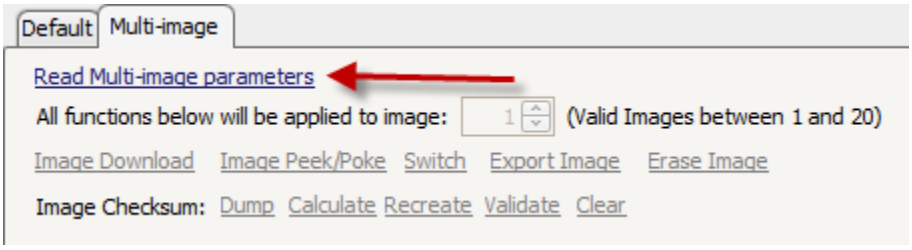


Figure 65 - Click Read Multi-image parameters to activate functions

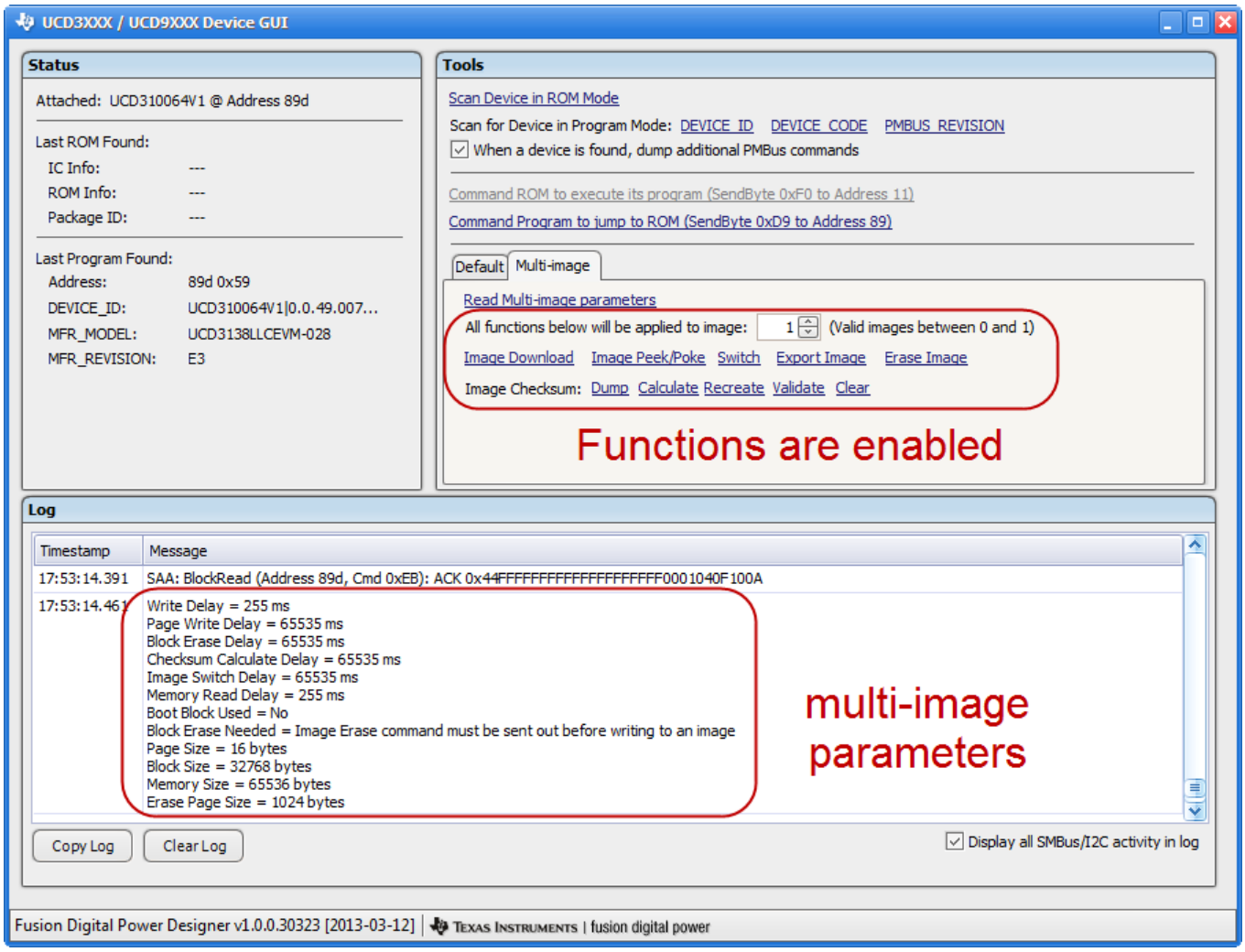


Figure 66 - Functions enabled after reading multi-image parameters

The following sections are descriptions of the functions for multi images.

7.5.1 Setting Image Index

Before using any of the functions shown in Figure 66 above the user must set which image index they will be working with.

7.5.2 Multi-image Download

After setting the appropriate image index and clicking "Image Download", the following image will be displayed.

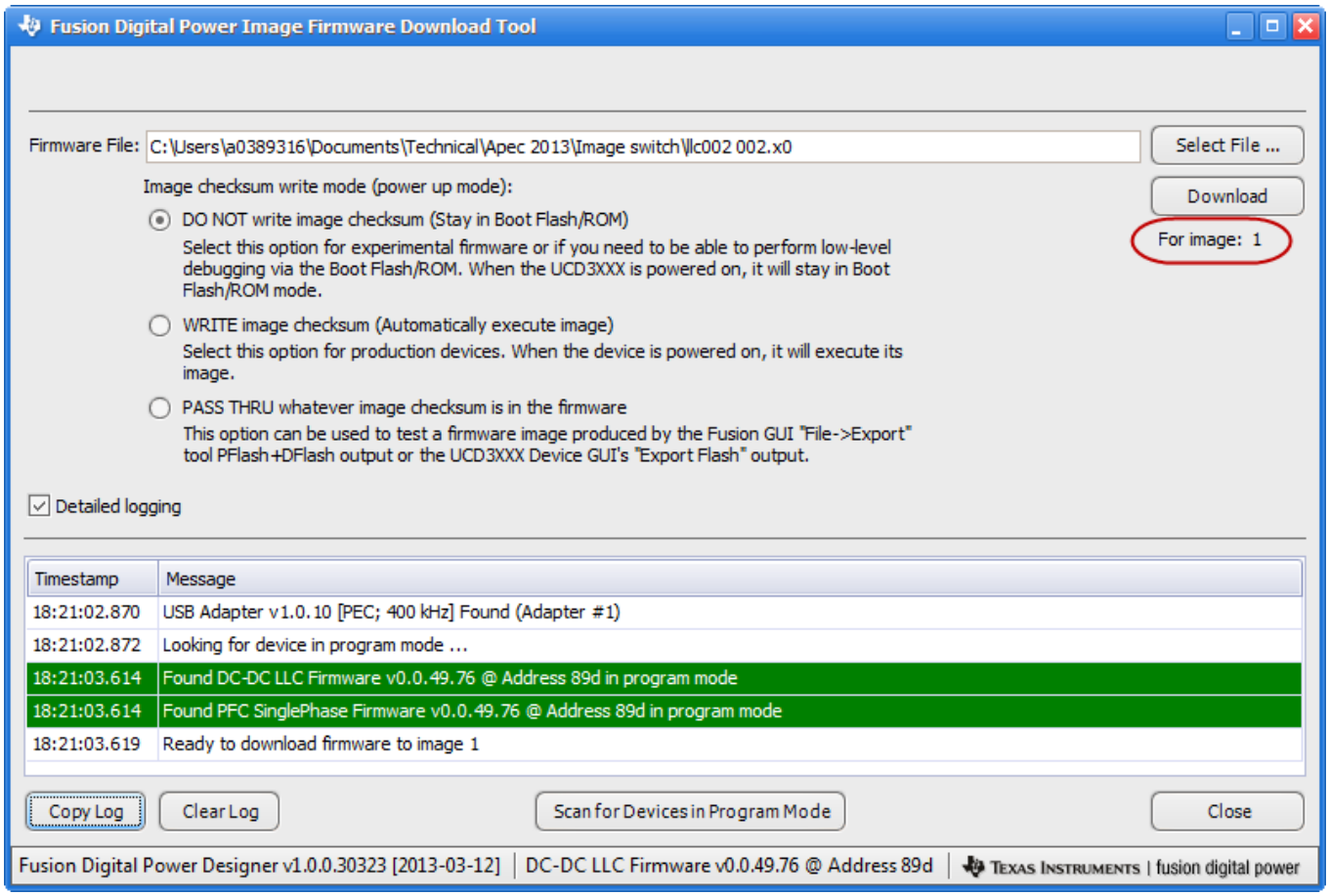


Figure 67 - Image download

7.5.3 Switch

In order to activate the image downloaded the user will need to click “Switch”. See Figure 68.

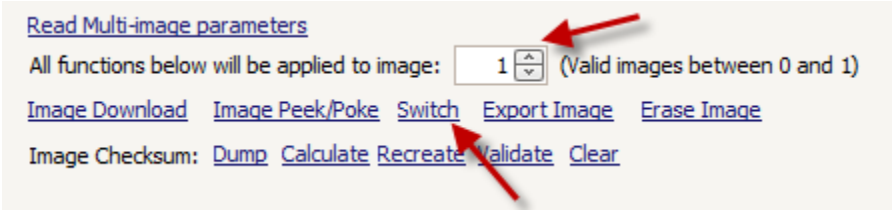


Figure 68 - Image to switch to

7.5.4 Image Peek/Poke/Dump

The user can specify which address to read/write to as shown in Figure 69 and Figure 70.

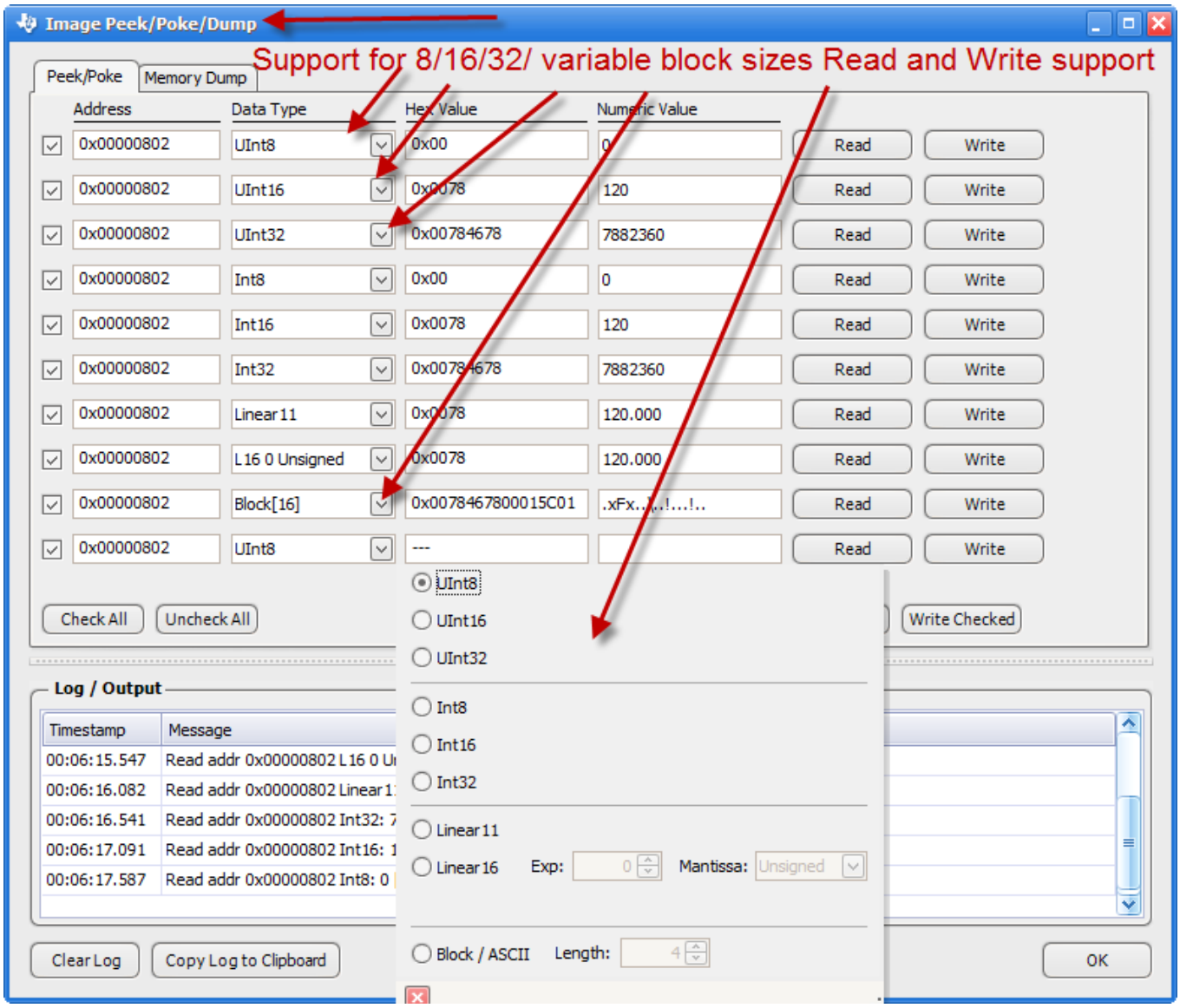


Figure 69 - Image Peek/Poke

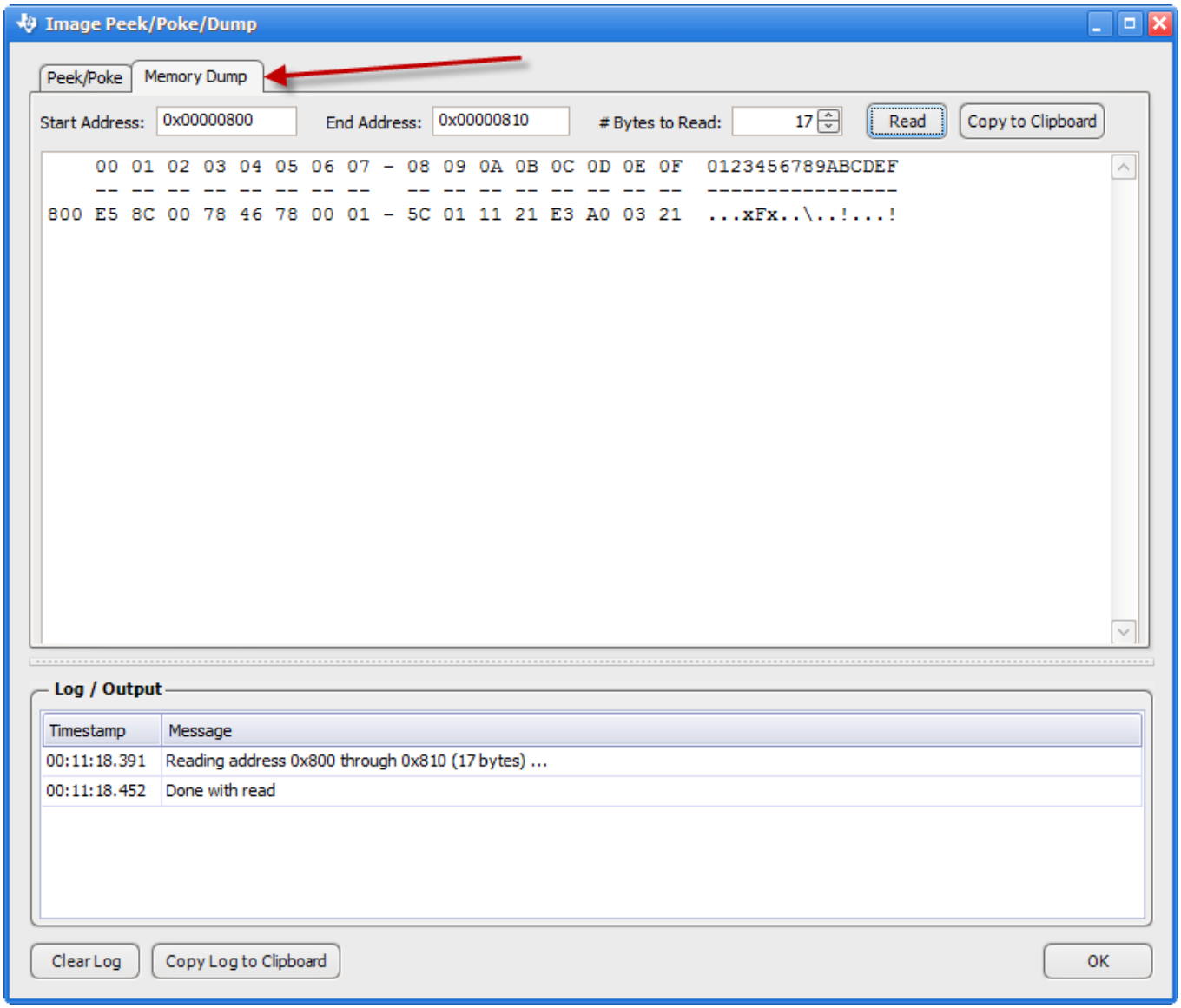


Figure 70 - Image Dump

7.5.5 Erase Image

Click Erase image to send the firmware command to erase the image selected as shown in Figure 71.

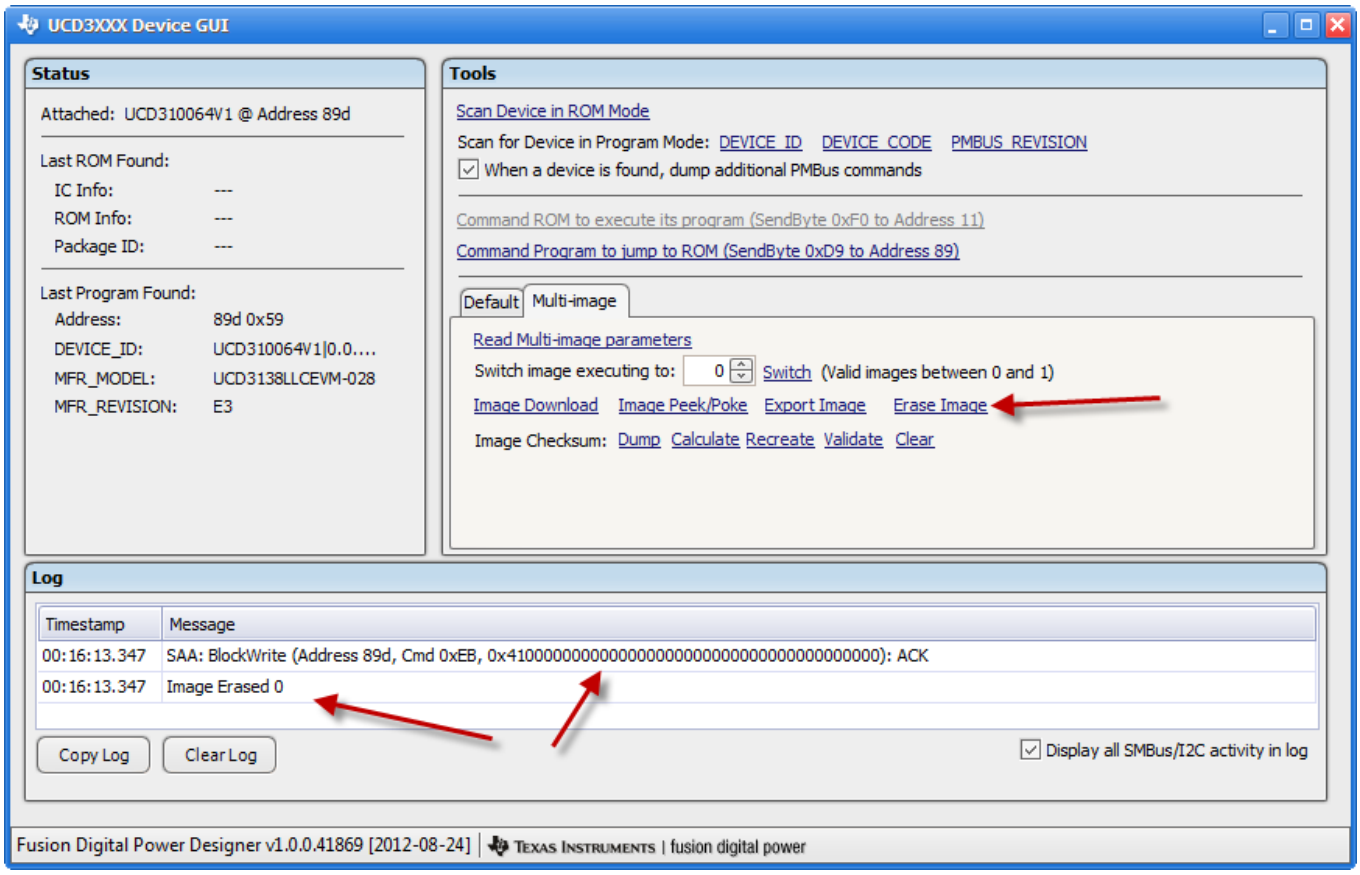


Figure 71 - Erase Image

7.5.6 Export Image

The user can also export the image currently on the device.

7.5.7 Image Checksums

The following pictures illustrate the image checksum commands.

7.5.7.1 Calculate Image Checksum

To calculate a checksum based on the image selected click “Calculate.”

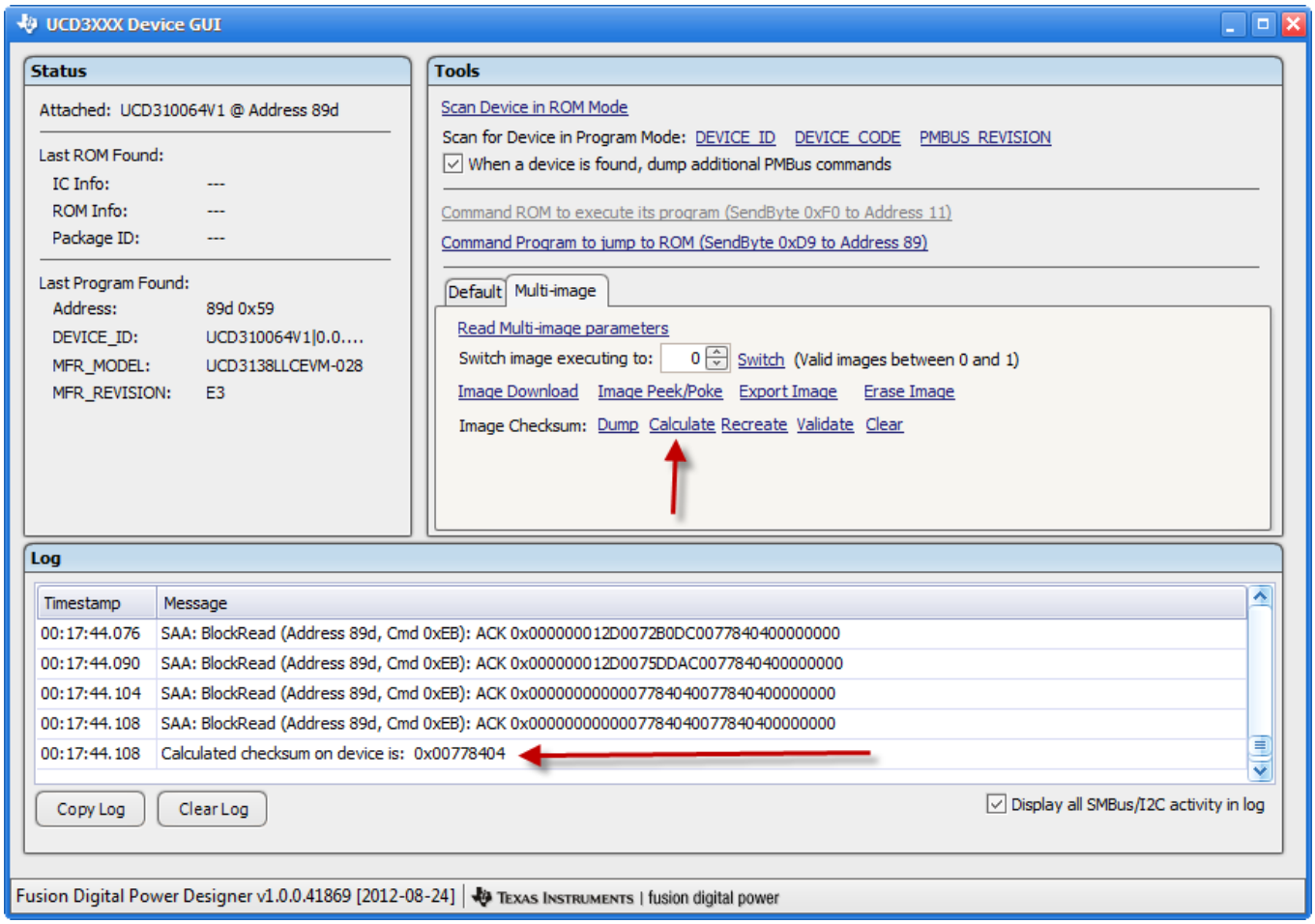


Figure 72 - Calculate image checkum

7.5.7.2 Dump Image Checksum

To display the last written checksum or bytes currently in the location of where the image checksum would be click “Dump.”

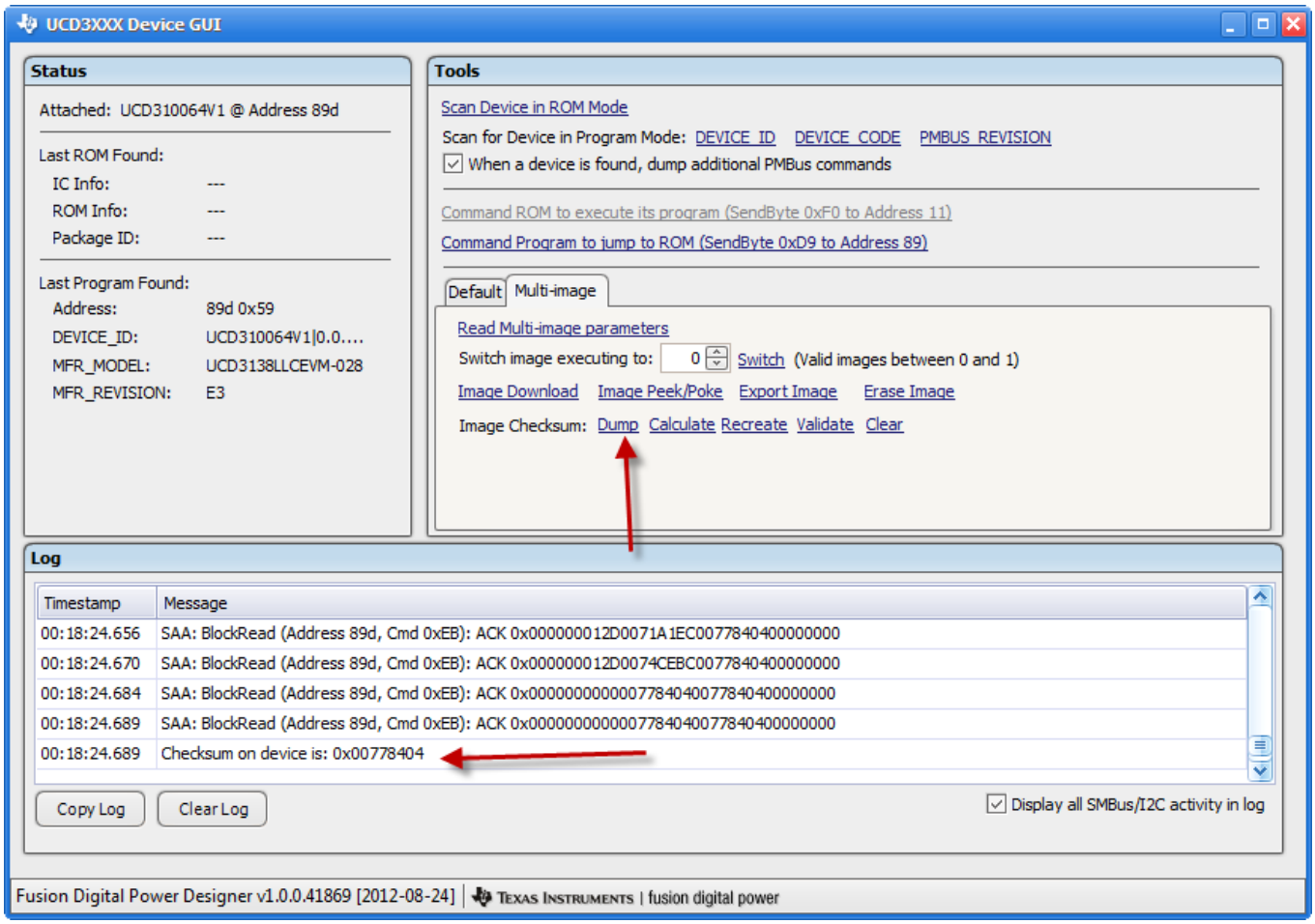


Figure 73 - Dump image checksum

7.5.7.3 Create Image Checksum

To create a checksum in the checksum location for the image selected click “Recreate.”

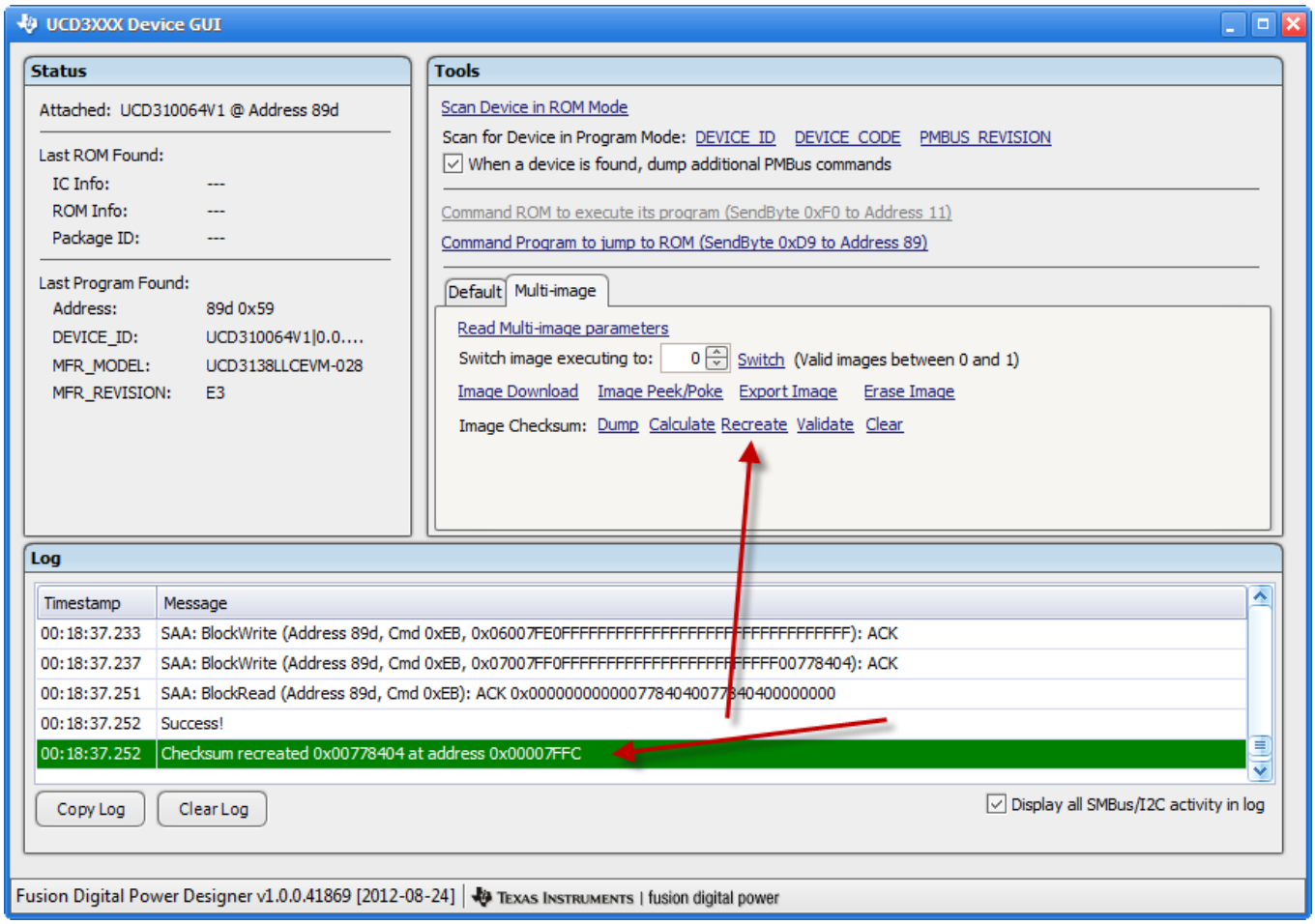


Figure 74 - Recreate image checksum

7.5.7.4 Validate Image Checksum

To validate that the calculated checksum equals the dump checksum click “Validate.”

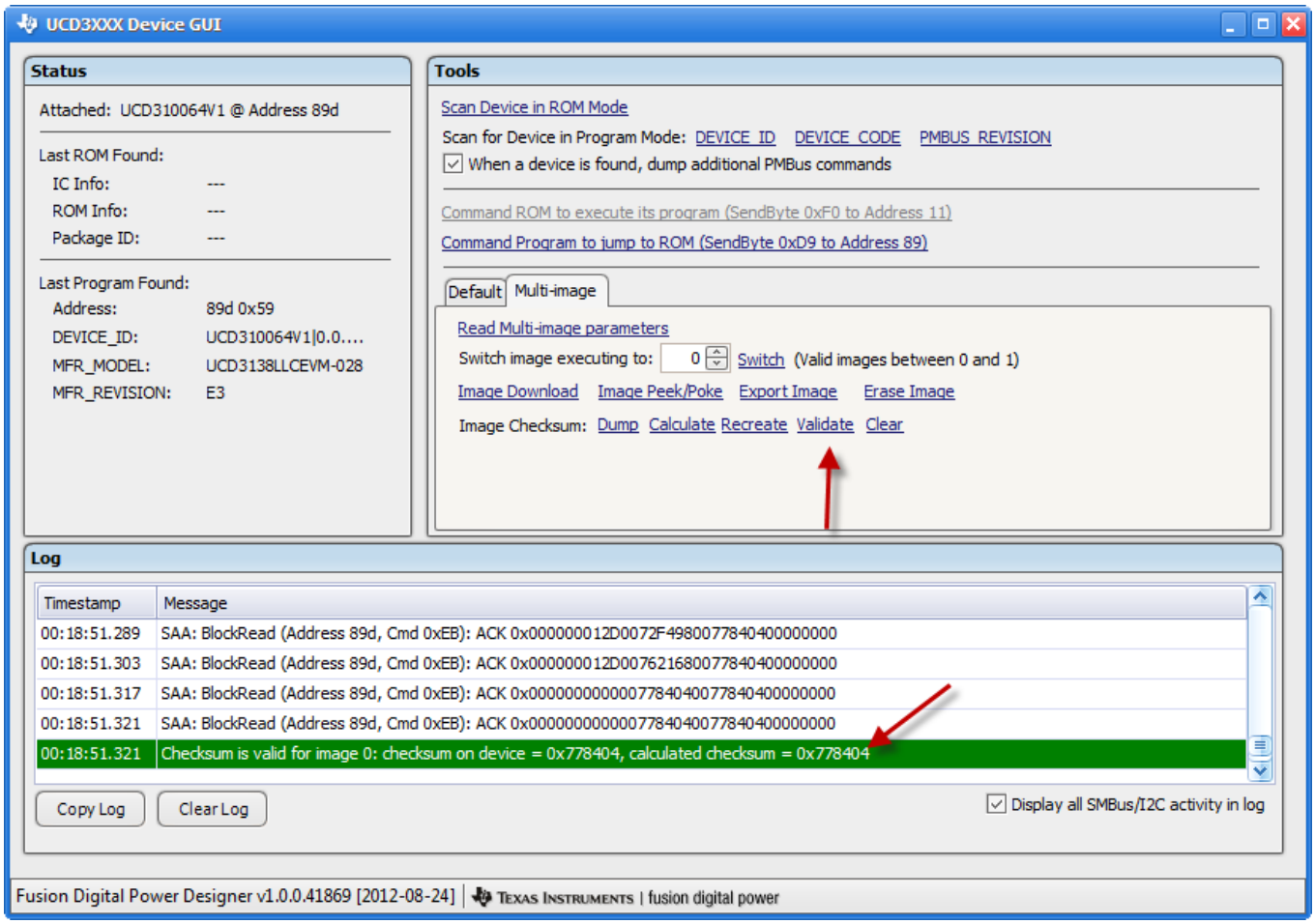


Figure 75 - Validating image checksum

7.5.7.5

To clear the checksum for the image selected click “Clear” and 0xFFFFFFFF will be written to that location.

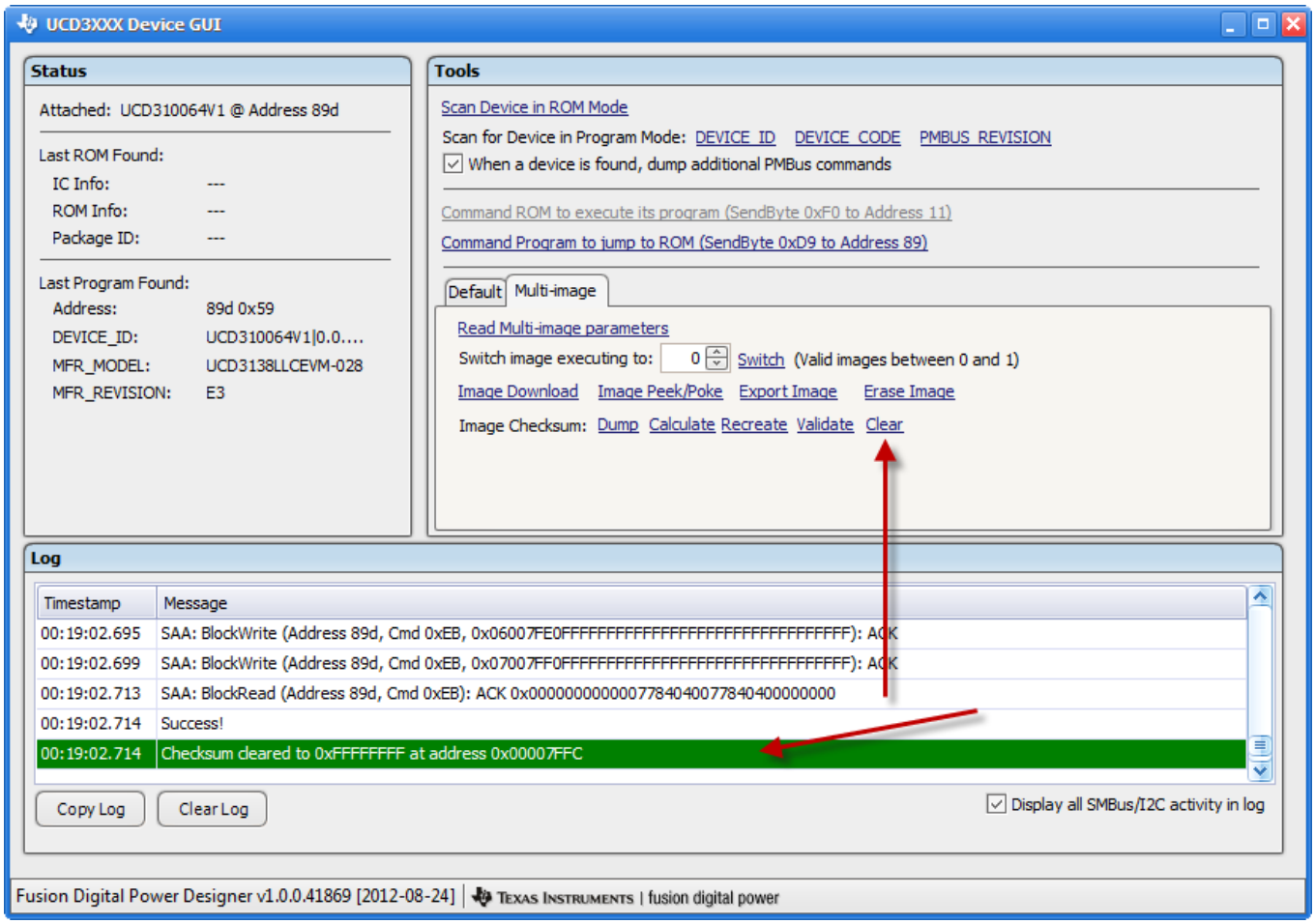


Figure 76 - Clearing image checksum

7.6 Isolated Bitmask Tool

The Isolated Bitmask Tool provides firmware developers with a tool to help them set the bitmask for the commands that inform the GUI of what PMBus commands are supported. See “Section 6.4.2 How does Implemented Commands on the Firmware Appear in the GUI?”

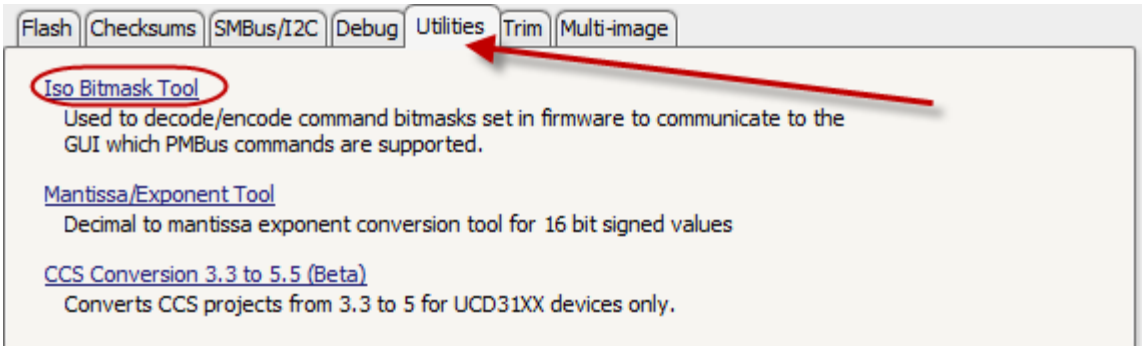


Figure 77 - Click Iso Bitmask Tool

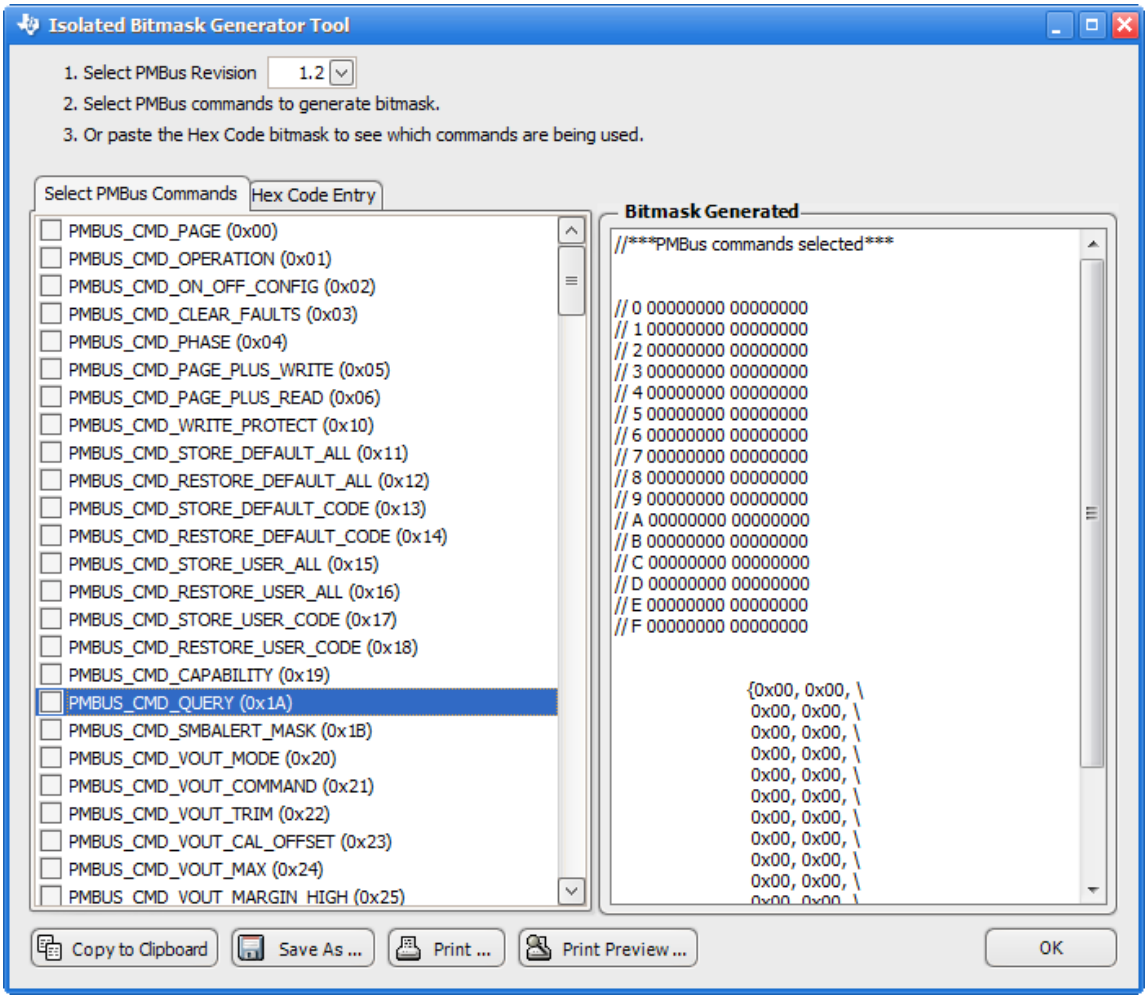
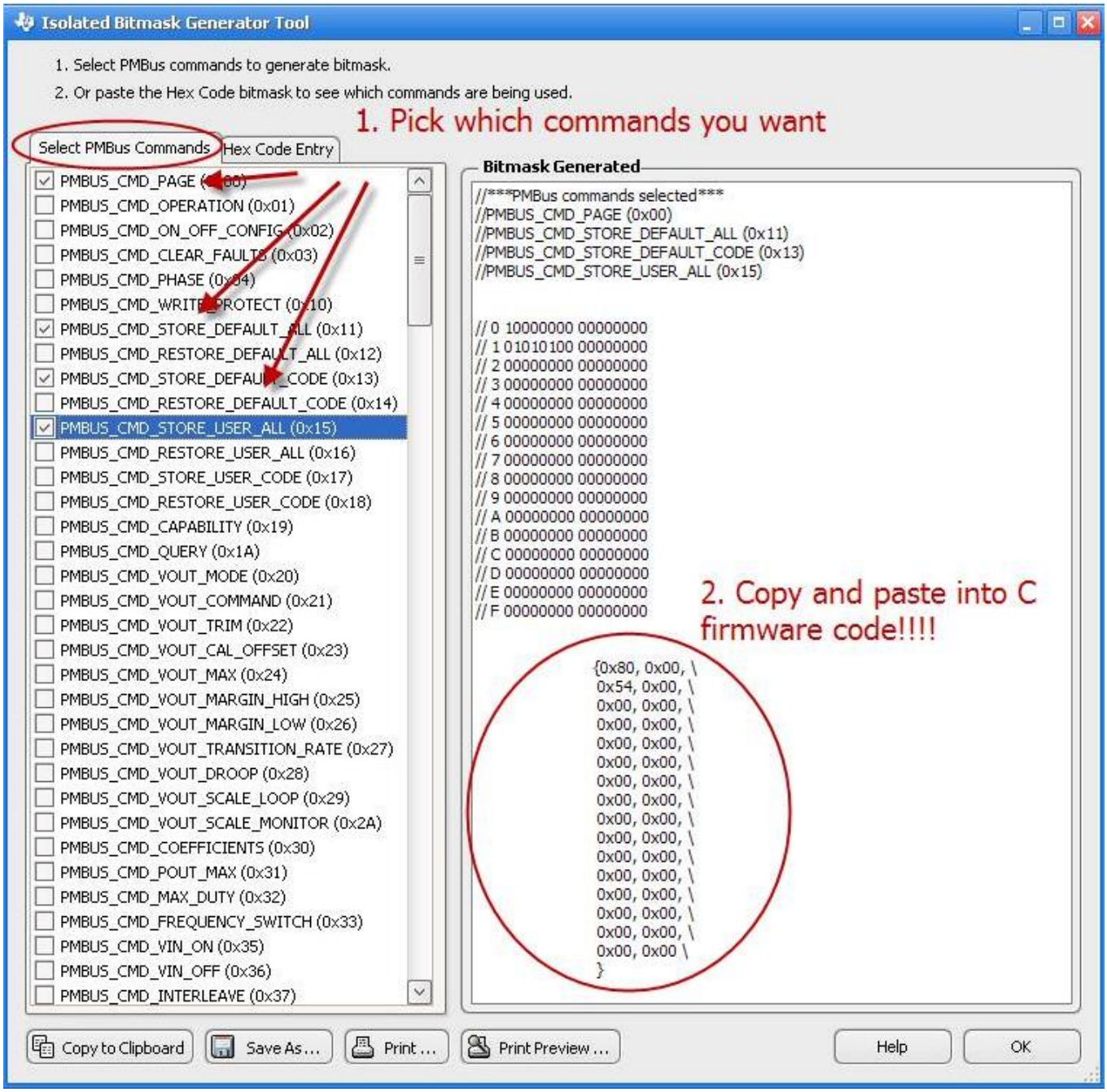
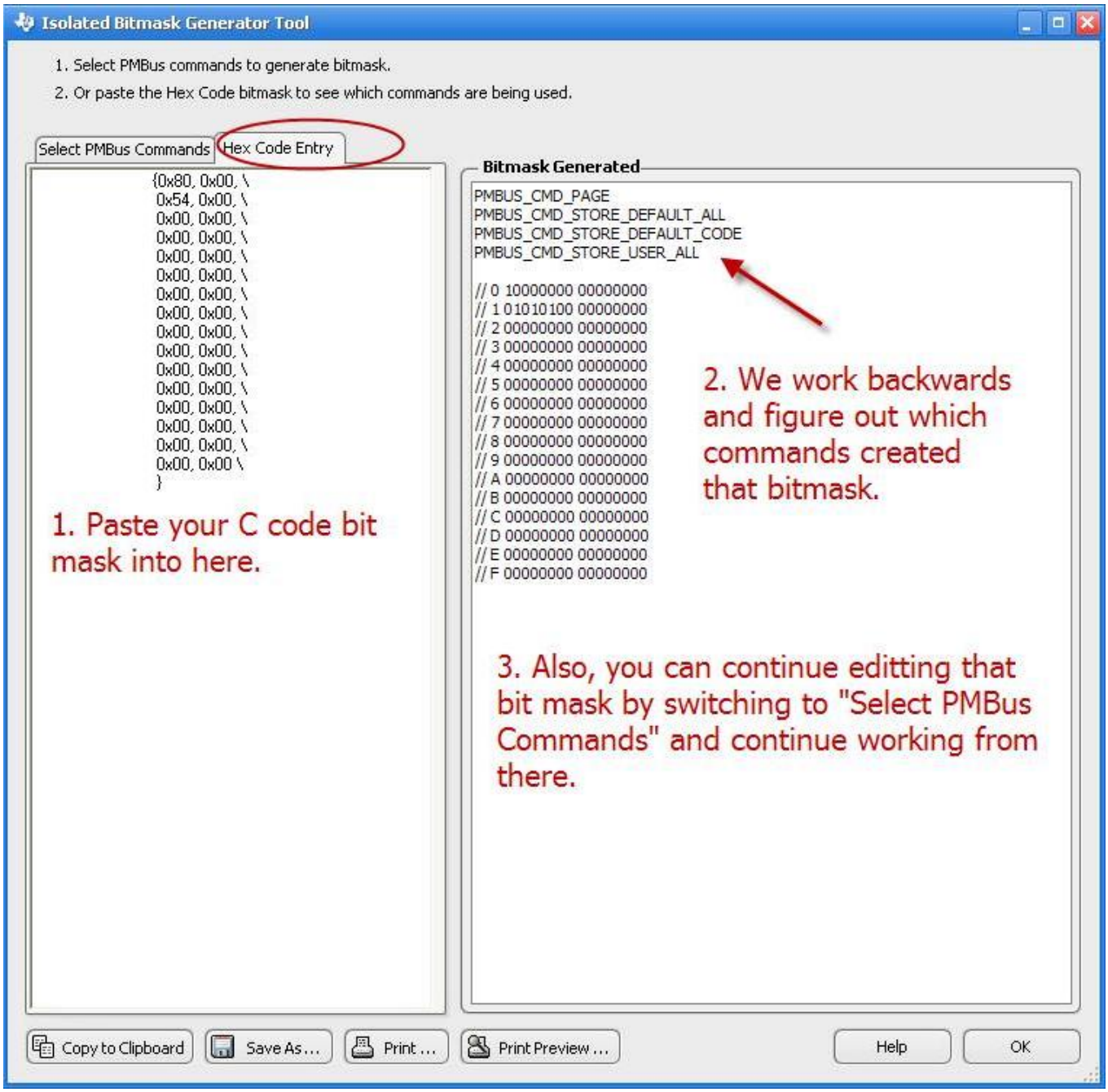


Figure 78 - Bitmask tool

2. Select commands desired in the bitmask and the bitmask code on the right will automatically be generated.



3. The user can also work in reverse by pasting a known bitmask in C code and then see what commands those bitmasks were indicating. They can also go back to the Select PMBus commands tab and all the indicated ones will be checked.



7.7 Firmware Memory Debugger

Included with the Fusion Digital Power Design software suite a powerful low level GUI is available for debug using the PMBus. Click the Debug tab and click Memory Debugger.

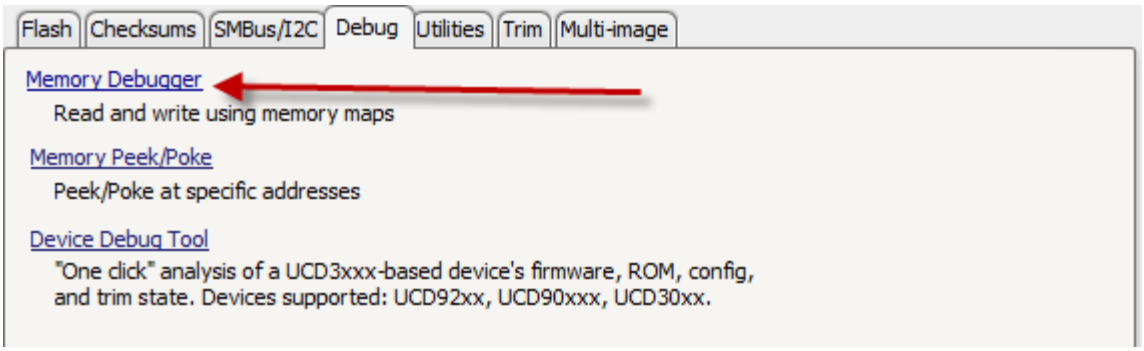


Figure 79 - Memory Debugger

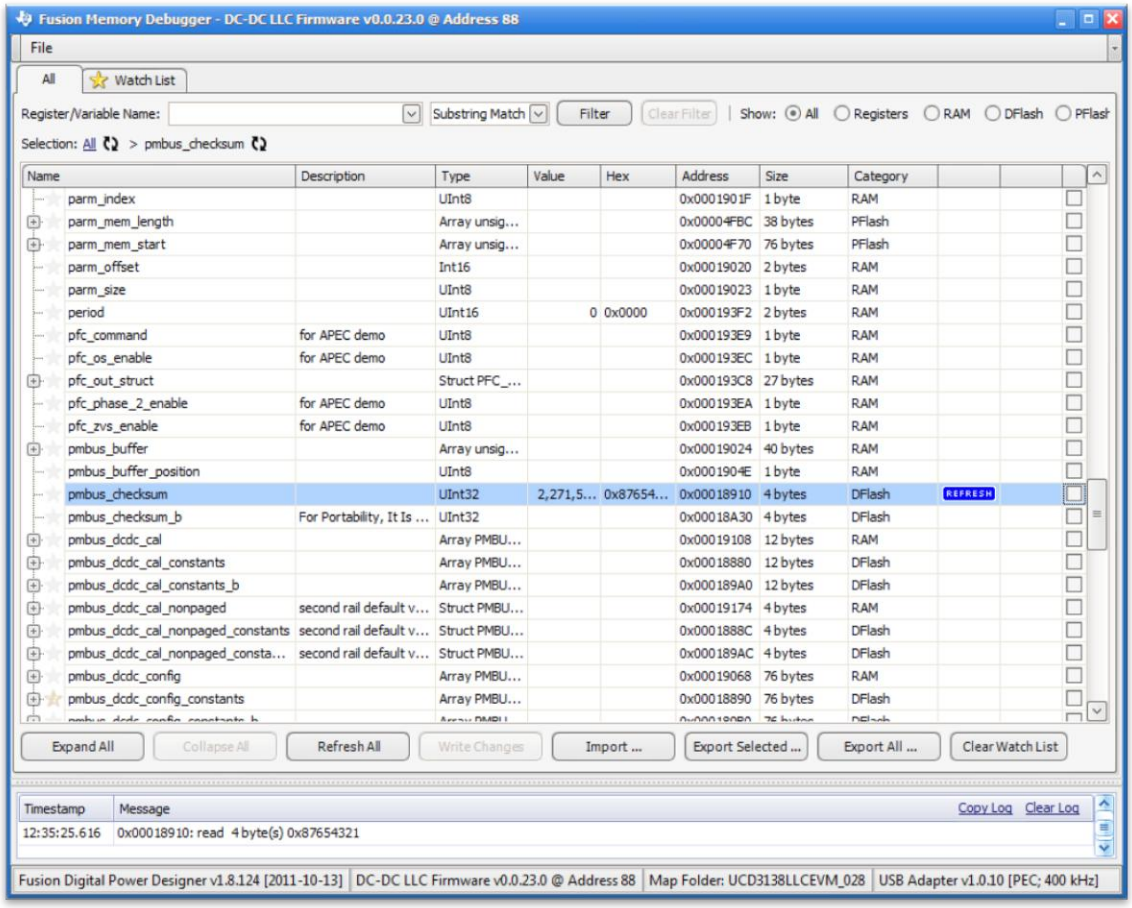


Figure 80 - GUI Debugger

To also access the GUI through the Design GUI click the “Memory Debugger” item under tools, shown in Figure 81.

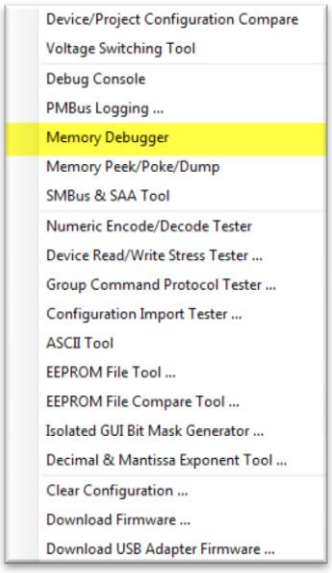


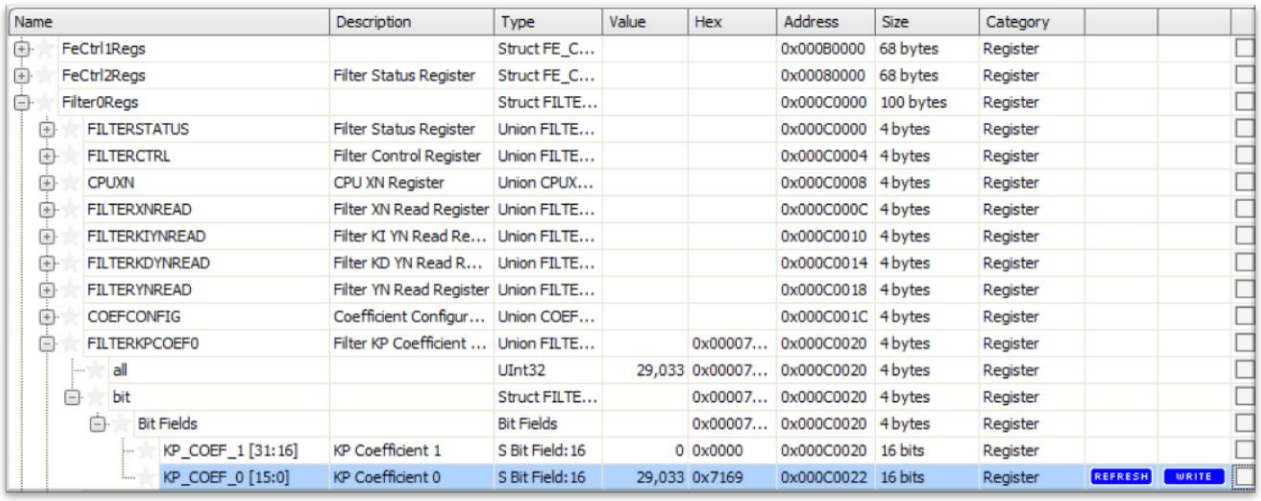
Figure 81 - Fusion Designer GUI Debugger Tool

By default the tool comes up displaying all of the hardware based device registers.

Name	Description	Type	Value	Hex	Address	Size	Category			
AdcRegs	IRQ Index Offset Ve...	Struct ADC_...			0x00040000	152 bytes	Register	REFRESH	WRITE	
CimRegs	Memory Fine Base A...	Struct CIM_...			0xFFFFFF20	24 bytes	Register			
DecRegs	DPWM Individual Reg...	Struct DEC_...			0xFFFFFE00	156 bytes	Register			
Dpwm0Regs		Struct DPWM...			0x000D0000	140 bytes	Register			
Dpwm1Regs		Struct DPWM...			0x000A0000	140 bytes	Register			
Dpwm2Regs		Struct DPWM...			0x00070000	140 bytes	Register			
Dpwm3Regs	Analog Comparator ...	Struct DPWM...			0x00050000	140 bytes	Register			
FaultMuxRegs	Ramp Control Register	Struct FAULT...			0x00030000	128 bytes	Register			
FeCtrl0Regs		Struct FE_CT...			0x000E0000	68 bytes	Register			
FeCtrl1Regs		Struct FE_CT...			0x000B0000	68 bytes	Register			
FeCtrl2Regs	Filter Status Register	Struct FE_CT...			0x00080000	68 bytes	Register			
Filter0Regs		Struct FILTE...			0x000C0000	100 bytes	Register			
Filter1Regs		Struct FILTE...			0x00090000	100 bytes	Register			
Filter2Regs	Fault Port I/O Directi...	Struct FILTE...			0x00060000	100 bytes	Register			
GioRegs	Front End Control 0 ...	Struct GIO_...			0xFFF7FA00	64 bytes	Register			
LoopMuxRegs	Clock Trim Register	Struct LOOP...			0x00020000	120 bytes	Register			
MiscAnalogRegs	Static Memory Contr...	Struct MISC_...			0xFFF7F000	72 bytes	Register			
MmcRegs	PMBus Control Regist...	Struct MMC_...			0xFFFFFD00	60 bytes	Register			
PMBusRegs	Clock Control Registe...	Struct PMBU...			0xFFF7F600	36 bytes	Register			
SysRegs	T24 Counter Data Re...	Struct SYS_R...			0xFFFFFFD0	48 bytes	Register			
TimerRegs	UART Control Regist...	Struct TIMER...			0xFFF7FD00	156 bytes	Register			
Uart0Regs		Struct UART...			0xFFF7EC00	56 bytes	Register			
Uart1Regs	: allow reading const...	Struct UART...			0xFFF7ED00	56 bytes	Register			

Figure 82: GUI UCD3138 Debugger – Defaults

If you expand any item on this list you will have access to every bit field inside the UCD3138 device. This access extends to both reading and writing to these registers.



Name	Description	Type	Value	Hex	Address	Size	Category		
FeCtrl1Regs		Struct FE_C...			0x000B0000	68 bytes	Register		
FeCtrl2Regs	Filter Status Register	Struct FE_C...			0x00080000	68 bytes	Register		
Filter0Regs		Struct FILTE...			0x000C0000	100 bytes	Register		
FILTERSTATUS	Filter Status Register	Union FILTE...			0x000C0000	4 bytes	Register		
FILTERCTRL	Filter Control Register	Union FILTE...			0x000C0004	4 bytes	Register		
CPUXN	CPU XN Register	Union CPUX...			0x000C0008	4 bytes	Register		
FILTERXNREAD	Filter XN Read Register	Union FILTE...			0x000C000C	4 bytes	Register		
FILTERKIYNREAD	Filter KI YN Read Re...	Union FILTE...			0x000C0010	4 bytes	Register		
FILTERKDYNREAD	Filter KD YN Read R...	Union FILTE...			0x000C0014	4 bytes	Register		
FILTERYNREAD	Filter YN Read Register	Union FILTE...			0x000C0018	4 bytes	Register		
COEFCONFIG	Coefficient Configur...	Union COEF...			0x000C001C	4 bytes	Register		
FILTERKPCOEF0	Filter KP Coefficient ...	Union FILTE...		0x00007...	0x000C0020	4 bytes	Register		
all		UInt32	29,033	0x00007...	0x000C0020	4 bytes	Register		
bit		Struct FILTE...		0x00007...	0x000C0020	4 bytes	Register		
Bit Fields		Bit Fields		0x00007...	0x000C0020	4 bytes	Register		
KP_COEF_1 [31:16]	KP Coefficient 1	S Bit Field: 16	0	0x0000	0x000C0020	16 bits	Register		
KP_COEF_0 [15:0]	KP Coefficient 0	S Bit Field: 16	29,033	0x7169	0x000C0022	16 bits	Register	REFRESH	WRITE

Figure 83 - Device Debugger Bit Field Selector

Figure 83 displays one register set fully expanded in the debugger. Clicking the “REFRESH” button on the right will force the debugger to read the corresponding register from the device. Entering a new value in the “Value” or “Hex” fields and then clicking “WRITE” will write the new values to the device. ***Keep in mind that reading and writing to any register in the device is very powerful and also dangerous. Some registers should not be changed and others are cleared on read so care should be used when selecting which registers you want to access. Please see the appropriate programmer’s manual for further details.***

Since there are so many different fields inside of the UCD3xxx devices a “Watch List” is available to create a convenient place to both read and write to the addresses of interest. Clicking one of the stars next to a variable name will turn it gold indicating that it has been added to the watch list. To remove an item from the watch list, simply click the star again. Clicking the “Watch List” tab at the top of the window will now display the selected.

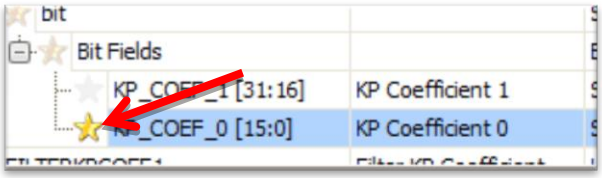


Figure 84 - Watch List Selection Star

The debugger also has the ability to read and write to any global firmware variable. This can be done by providing the GUI with the path to find the “.map” and “.pp” files from the firmware build. Click the item shown in Figure 85.

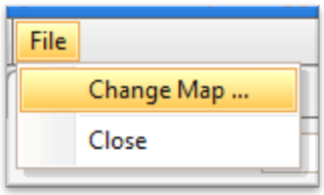


Figure 85 - Map File Selection

After clicking this item, a window will pop up providing detailed instruction on what to do. For an example, see Figure 86.

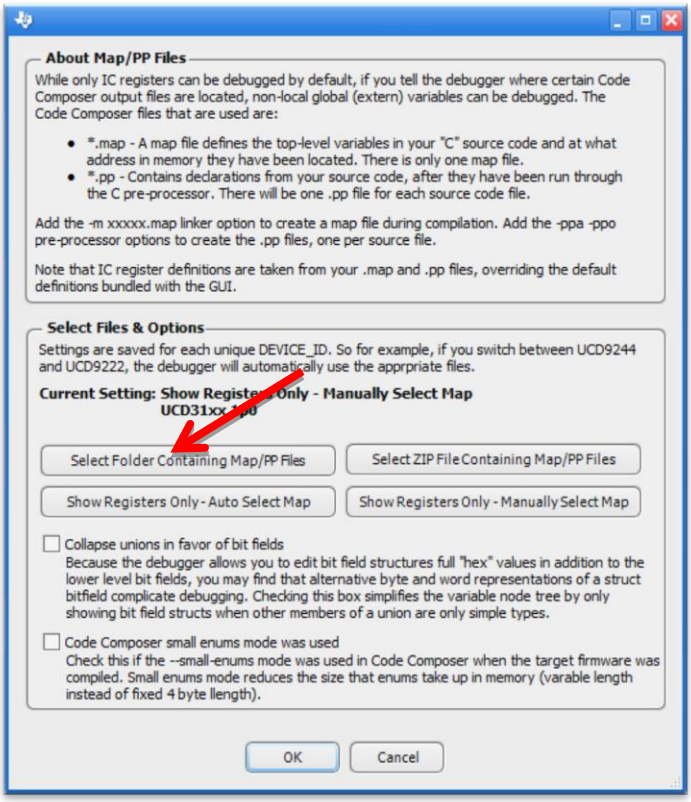


Figure 86 - Debugger Customization Tool

The creation of the ".pp" files can be configured by modifying the Code Composer build options as shown in Figure 87.

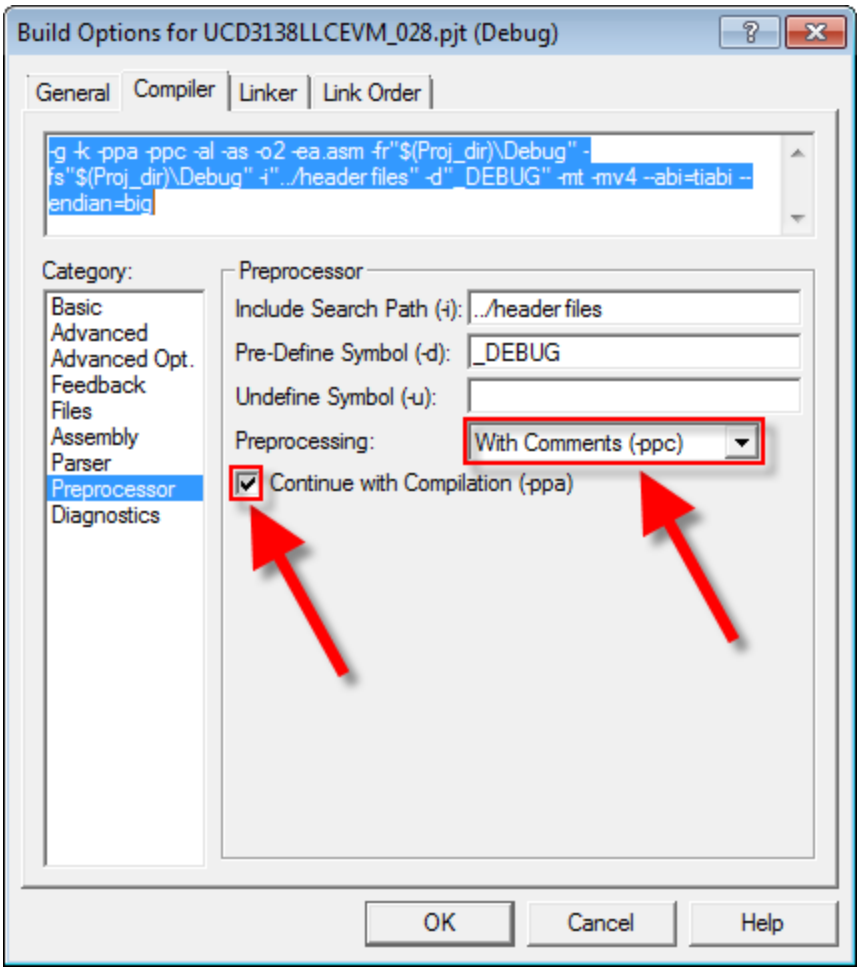


Figure 87 - ".pp" Generation Parameters

The "*.map" file name and location can be specified in the code compose build options as shown in Figure 88.

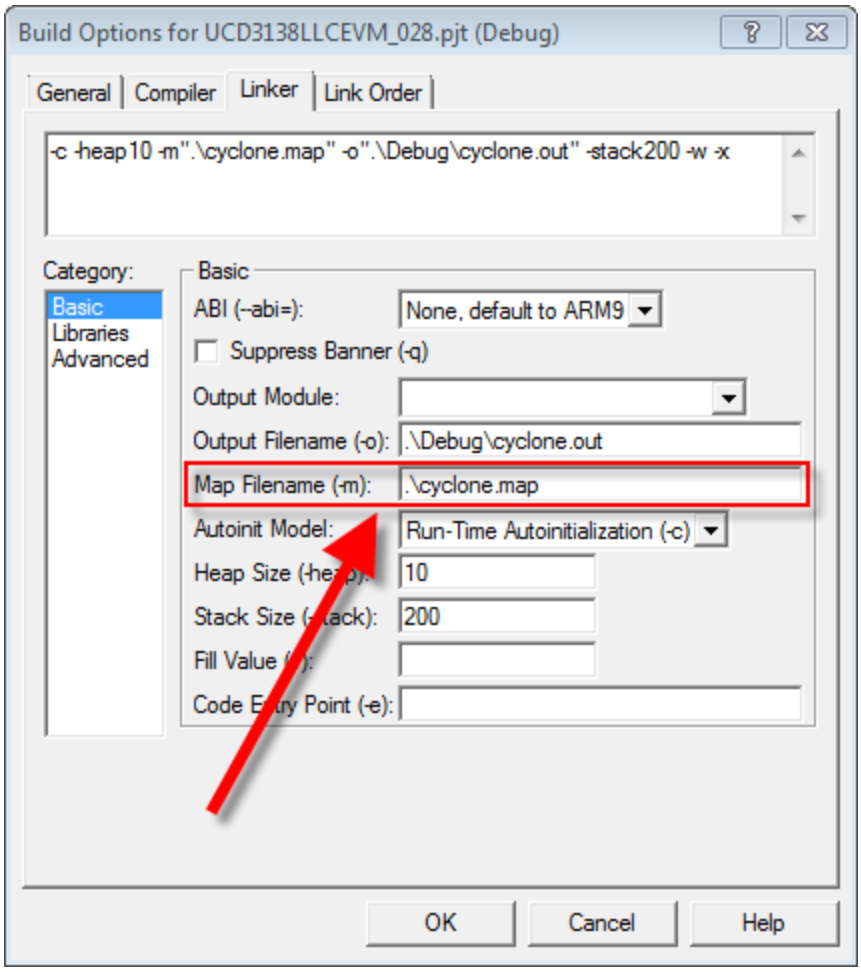


Figure 88 - Map Filename

After selecting the location of the ".map" and ".pp" files the debugger will extract the information it needs to allow read/write access to all global firmware variables. Depending on the speed of the system this can take a few moments. The GUI will create a local cache of the data it extracts. So as long as the files do not change subsequent launches of the debugger will be much faster.

You now can interact with RAM, DFLASH or PFLASH variables in the same way described above for device registers. Figure 89 shows an example where variables from RAM and DFLASH have been added to the watch list. "vout_cmd" is the mantissa of a linear16 variable and "supply_state" is a variable indicating the state of the IRQ state machine. Notice that the debugger picks up comments as well as the details of enumerated data types. These variables can be read or written to just like any other variable in the system.

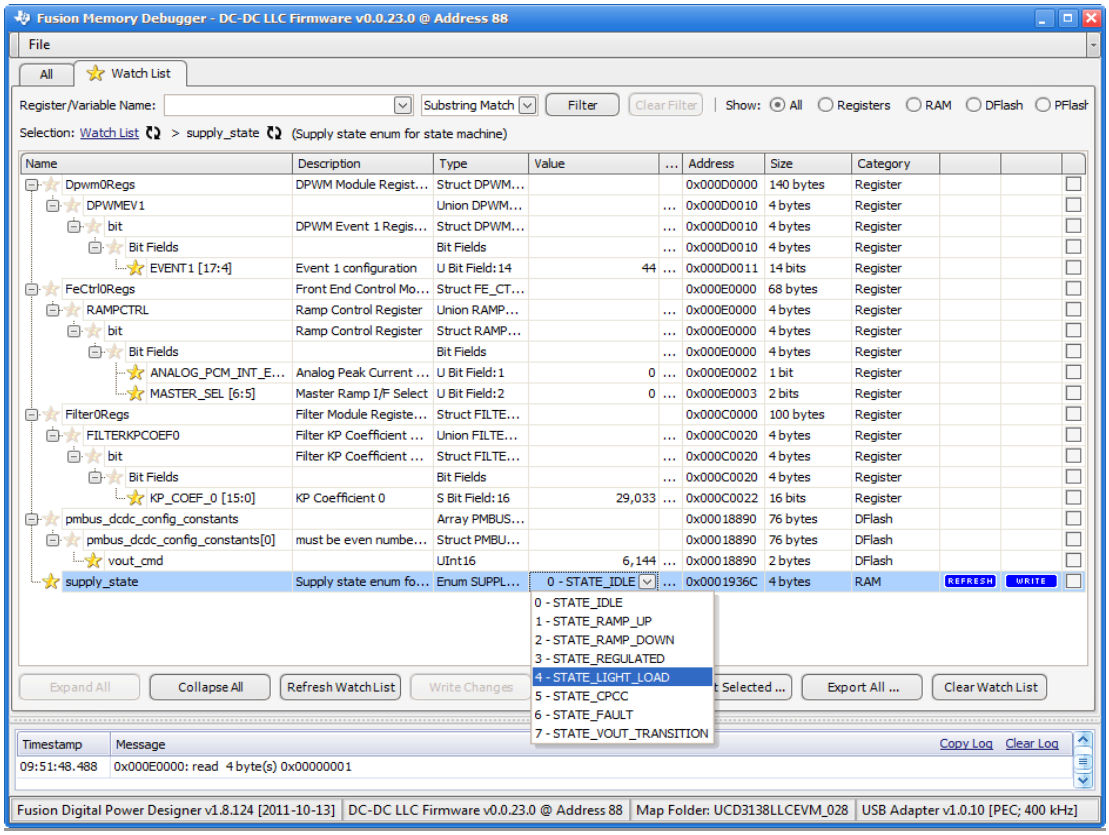
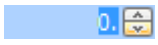


Figure 89 - Watch List with Firmware Variables

For the editable values there are up and down arrows.



The increment is normally 1. However, the firmware developer has the ability to specify how large the increments are and what the max and min of the variable is. They do this by specifying it in the comments. See highlights in comments below,

```
extern Uint16 my_uint16; // test root node [min=5, max=200, step=5]

typedef struct
```

```
{
        Uint8  a;      // [step=10]
        Uint8 b;      // [min=0, max=100, res=5]
        Uint8 c;      // [min=100, res=5]
        float   d;      // [min=-1e-3, max=1e3] step/res do not make sense with floats
        Int8    e;      // [min=-100, max=100]
} struct1;
```

Order within the brackets does not matter. White space also does not matter.

Note there are two different ways to change how the up/down arrows work in the decimal editor:

- step: simple increment/decrement. If the current value is 2 and the step is 5, clicking up, will change the value to 7.
- res: modulo oriented resolution. If the current value is 2 and the res is 5, clicking up, will change the value to 5.

7.8 SMBus Debug

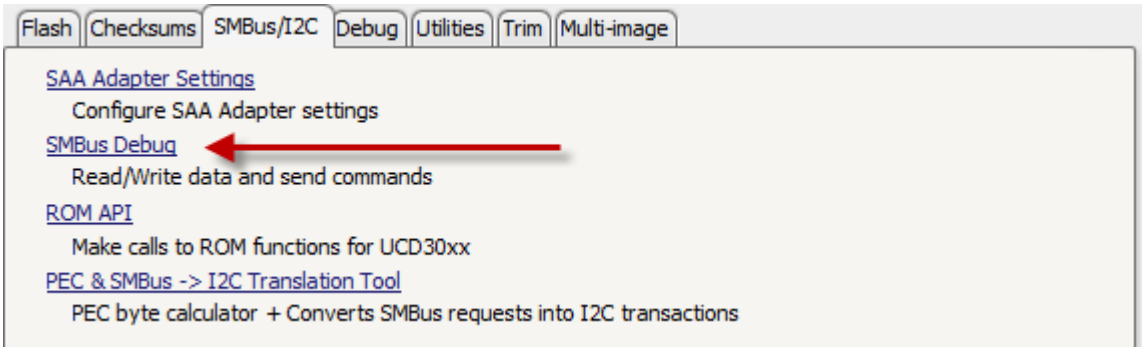


Figure 90 - SMBus Debug Link

The tool looks as follows when launched,

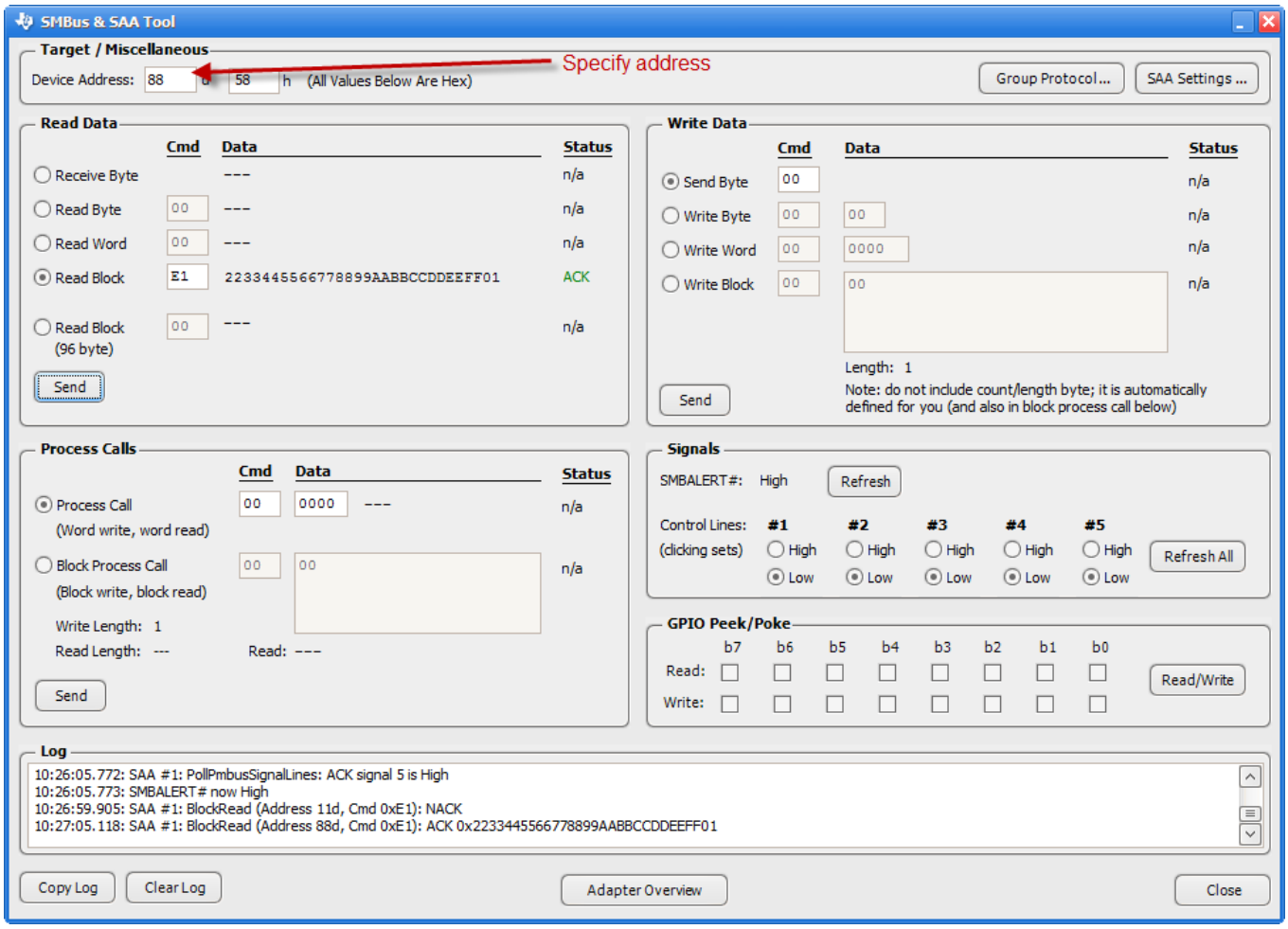


Figure 91 - SMBus Debug

In order to use this tool the user needs to specify the device address. This tool can be used to interact with PMBus commands. It can be used to Read commands by specifying the hex command and it can be used to write to commands specifying the command and the data.

7.9 CCS conversion

This tool converts UCD31XX device projects from CCS 3.3 to CCS 5.5. NOTE: Although the project should compile after conversion, in rear cases some manual steps may be required. All files in the original folder will end up in the new folder. Only relevant files will be updated or used. All files used, updated or simply copied will be reported in the log.

7.9.1 How to access

To access from the “Start” menu, click Texas Instruments Fusion Digital Power Designer->Tools->Isolated CCS Conversion Tool.

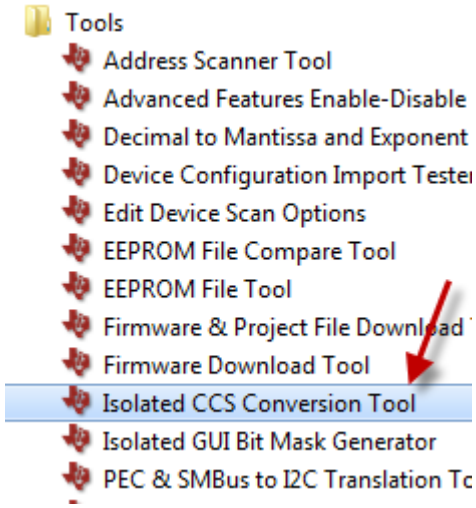


Figure 92 - CCS conversion tool access from Start Menu

From the “UCD3XXX/UCD9XXX Device GUI” it can be found in the “Utilities” tab.

Figure 93 - Conversion tool access from Device GUI

7.9.2 Usage

1) Browse to the location of the CCS 3.3 UCD31XX project file (*.pjt).

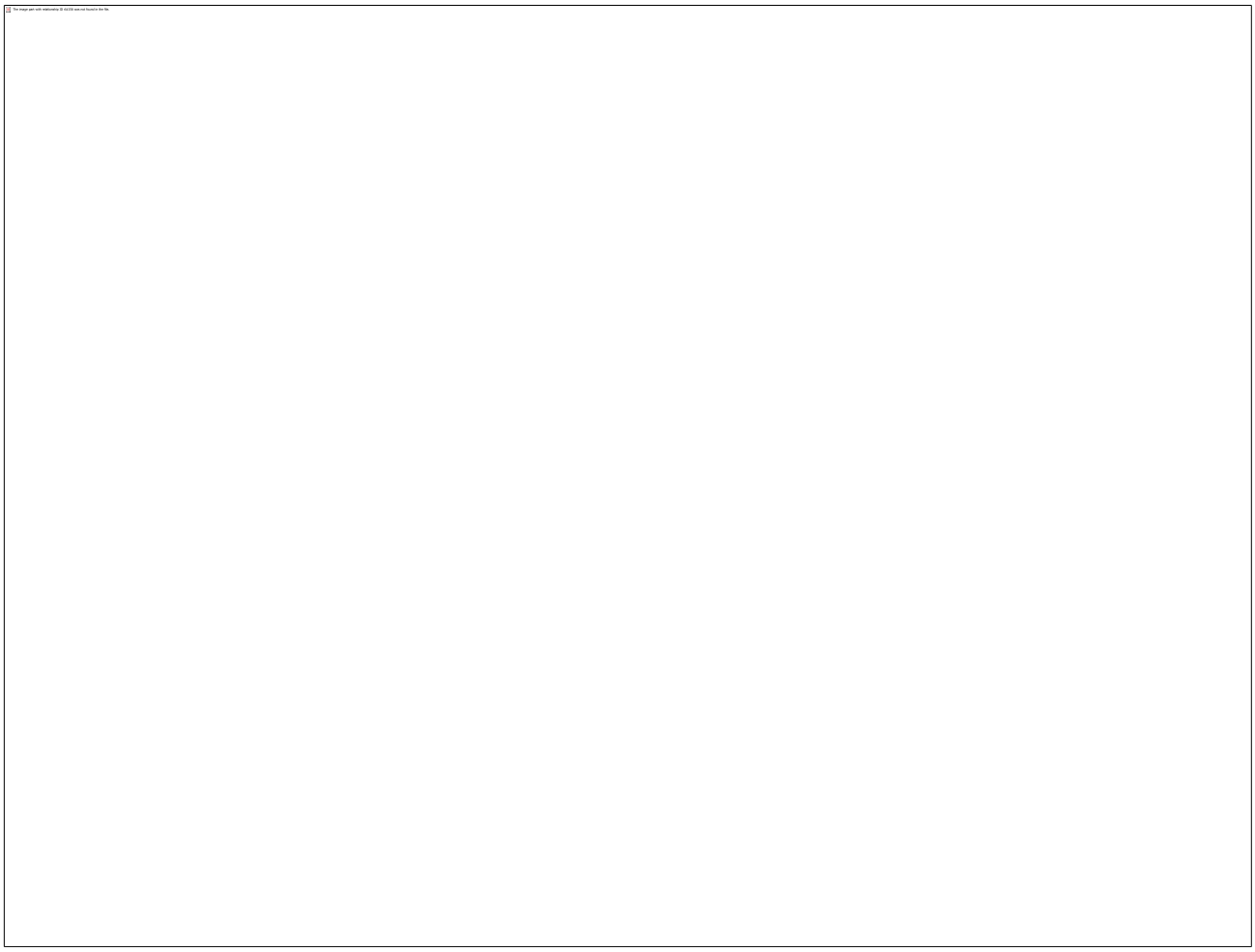


Figure 94 - Browse to CCS 3.3 project

2) Browse to the location of where the new CCS 5.5 project will be stored. By clicking the browse button it will suggest the new “Project name” based on the project name from CCS 3.3. The newly created project will create a folder with the “Project name” and a timestamp appended (for example “UCD3138LLCEVM_028_25-12-2014-12-05-22”).

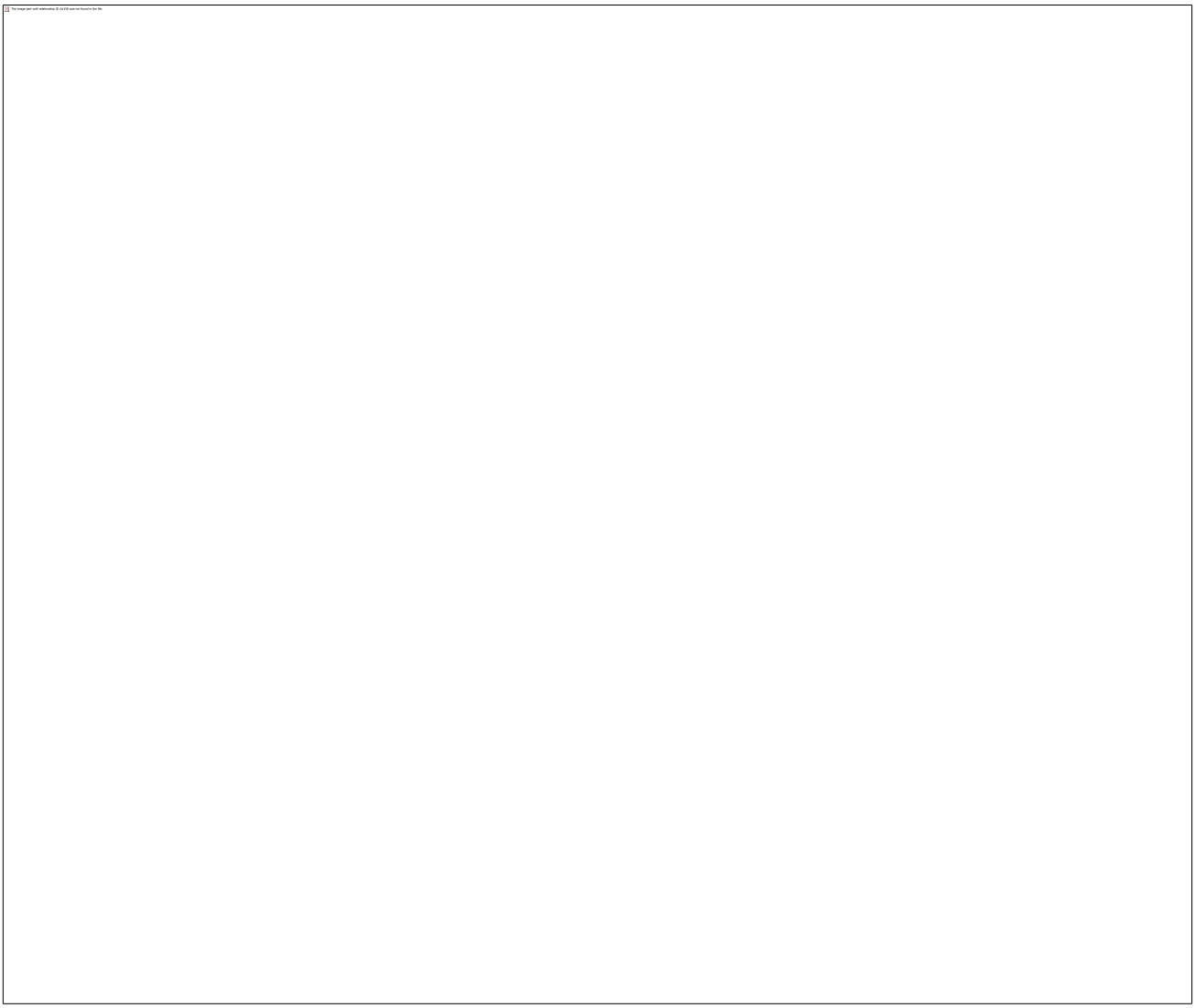


Figure 95 - Browse to new location of converted project

3) Click “Convert.”

Figure 96 - Click to convert

4) After the project has completed (usually after a couple seconds) the log will be updated with the results. Sometimes it will indicate warnings in yellow (for example “Stale file zoiw.asm” may appear if the file was in the old folder but was not referenced in the project file, i.e. the original project was not even using this file). All updates made by the tool will be displayed in the log. Code changes will be displayed in a light green. The old and new versions of the code will both be shown. Below are some snapshots of the log. A copy of the log is automatically stored in the converted project folder with a timestamp (for example “Conversion-Log-2014-04-17-16-13-54.html”).

   a. Conversion completed

Figure 97 - Conversion completed

b. Code changes made including filename, line number, old and new code

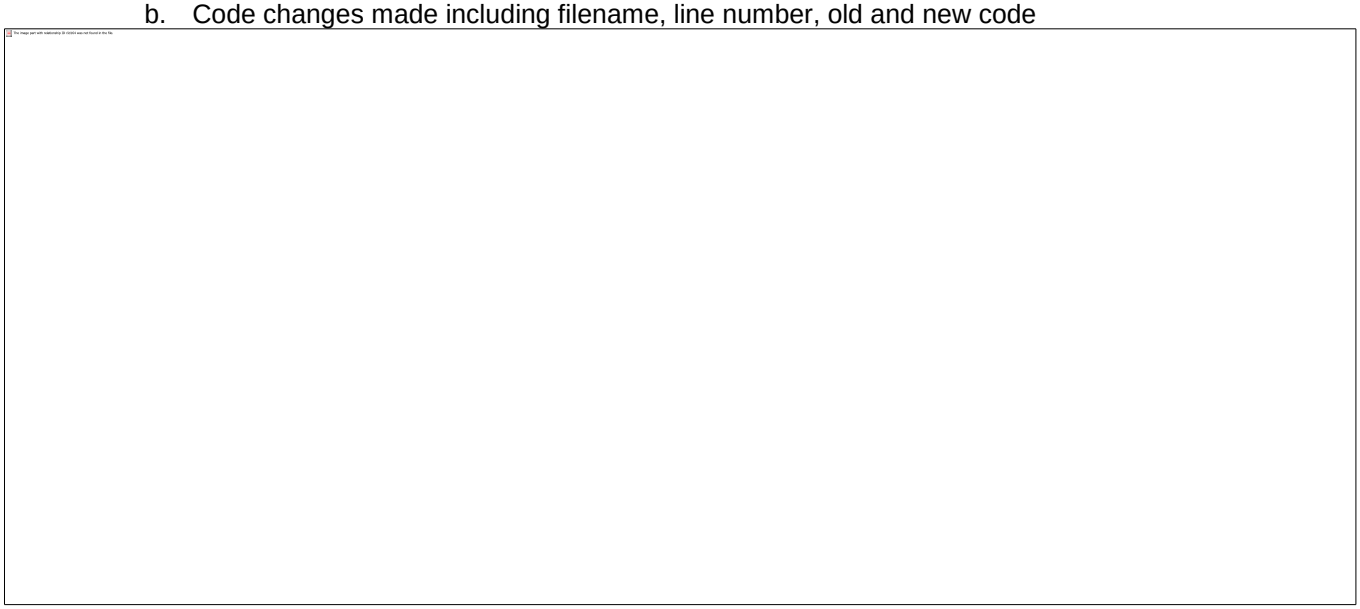


Figure 98 - Displays file, line # and change made

c. The log can be copied or opened in a web browser. The log opened from the button is the same one stored in the converted project folder.

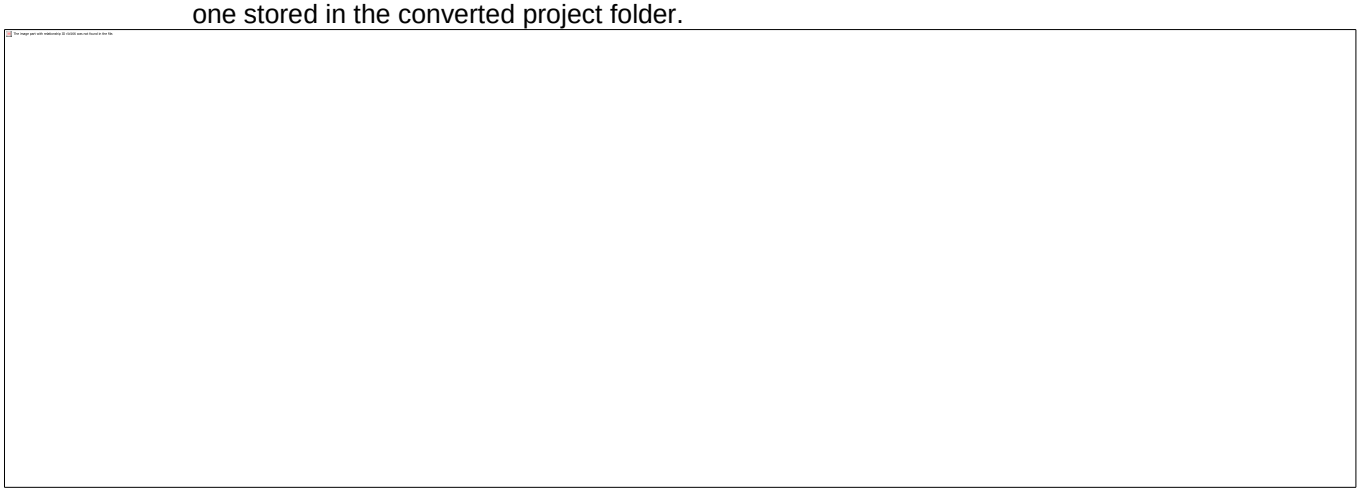


Figure 99 - Buttons to access the log generated

d. Display Log in web browser

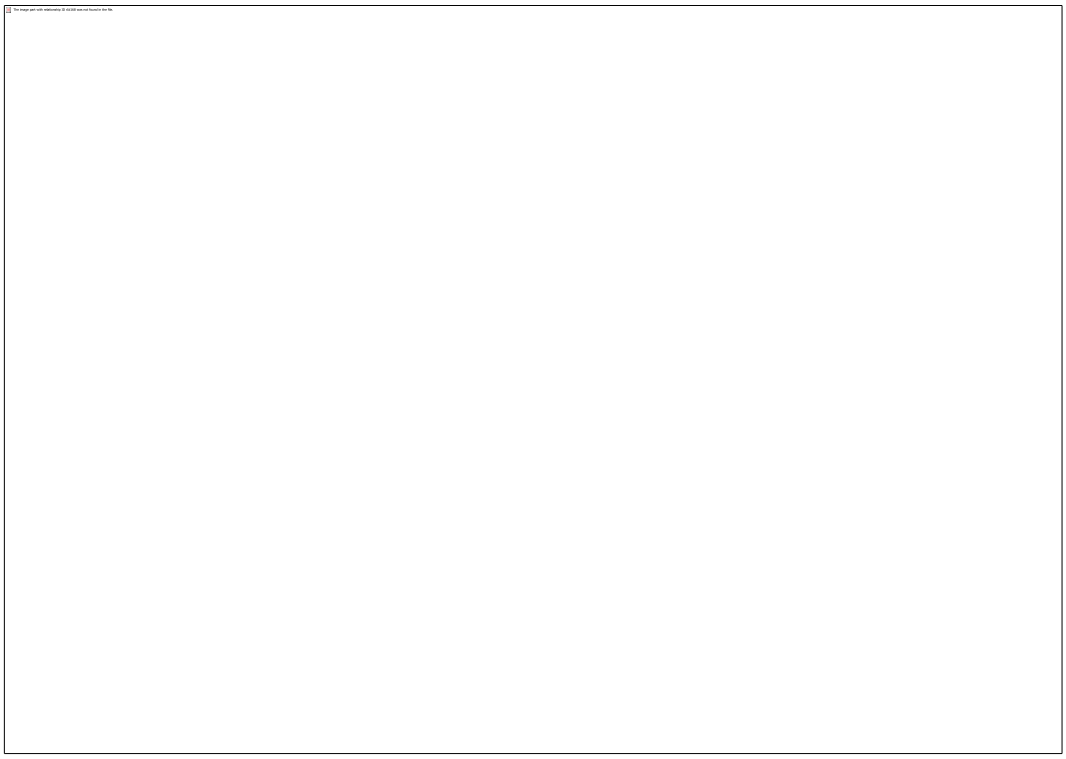


Figure 100 - Conversion log opened in browser

5) To quickly access the newly converted project click “Open Folder” in the “New CCS 5 Project Location” area.

Figure 101 - Open folder location of converted project

The project can now be imported to CCS 5.5!

7.10 Function Command Summary

The following table lists the ROM/Program commands called for some of the common functions used in the Device GUI.

Device GUI Function	Mode: ROM/ Program	Commands				Description
		Code (hex)	Command	Trans. type	Data Format	
Scan Device in ROM Mode	ROM	0xEC	Read Version			Scans for the device in ROM mode and reads PKGID at address 0xFFF7F010 using 0xFD and 0xFA.
		0xFD	Configure Read Address			
		0xFA	Read 4 Bytes			
Scan for Device in Program Mode: DEVICE ID	Program	0xFD	DEVICE_ID	Read Block (up to 32 bytes)	String	MFR command supported by UCD devices. Ex. "UCD310128V1|0.1.0.0010|131009"
	Program	0xE4	CMDS_DCDC_PAGED	Read only	Bitmask	Contains bitmask of paged DCDC supported commands in the firmware
	Program	0xE5	CMDS_DCDC_NONPAGED	Read only	Bitmask	Contains bitmask of non paged DCDC supported commands in the firmware
	Program	0xE6	CMDS_PFC	Read only	Bitmask	Contains bitmask of PFC supported commands in the firmware.
	Program	0xE7	SETUP_ID	Read only	String	Special value that maps to a topology and how it is compensated within the Fusion Designer GUI. Ex. "VERSION1|LLC001"
Scan for Device in Program Mode: DEVICE CODE	Program	0xFC	-	-	-	Not applicable for UCD devices
Scan for Device in Program	Program	0xAD	-	-	-	Not applicable for UCD

Mode: IC DEVICE ID						devices	
Scan for Device in Program Mode: PMBUS REVISION	Program	0x98	PMBUS_REVISION	Read Byte	Byte	Defined by PMBus spec. All PMBus devices support it.	
[Check] When a device is found, dump additional PMBus commands	Program	0x9A	MFR_MODEL	R/W Block	String	Ex. "UCD3138LLC EVM-028"	If this box is checked then after any of the scans above are completed these commands will be read.
		0X9B	MFR_REVISION	R/W Block	String	Ex. "E3"	
		0X9E	MFR_SERIAL	R/W Block	String	Ex. "SV001" – unique identifier	
		0X99	MFR_ID	R/W Block	String	Ex. "TI"	
		0X9D	MFR_DATE	R/W Block	String	Ex. "14033" YYMMDD	
		0X9C	MFR_LOCATION	R/W Block	String	Ex. "Dallas, TX"	
Command ROM to execute its program (SendByte 0xF0 to Address 11) First Block	ROM	0xF0		Send Byte		Executes the First Block of program flash	
Command ROM to execute its program (SendByte 0xF7 to Address 11) Second executable block	ROM	0xF7		Send Byte		For devices that support multiple program flashes this command executes the flash from the second block (or the third block for UCD3138 128).	
Command Program to jump to ROM (SendByte	Program	0xD9	ENABLE_ROM	-	-	Send device to ROM mode and clears the checksum. 0xF9 when	

0xD9)						password protected.
Command Program to jump to ROM (SendByte 0xD9) **with 0xF9 implemented**	Program	0xF9	ENABLE_ROM2	W	String	When pressing the command to jump to ROM if 0xF9 is implemented the user must enter a password and it will be sent with 0xF9
Memory Debugger/Peek Poke	Program	0xE2	PARM_INFO	W	Block	PARM_INFO and PARM_VALUE are both used. The first sets the address to be read and the second returns the value at that location.
	Program	0xE3	PARM_VALUE	R	Block	

7.11 Override commands

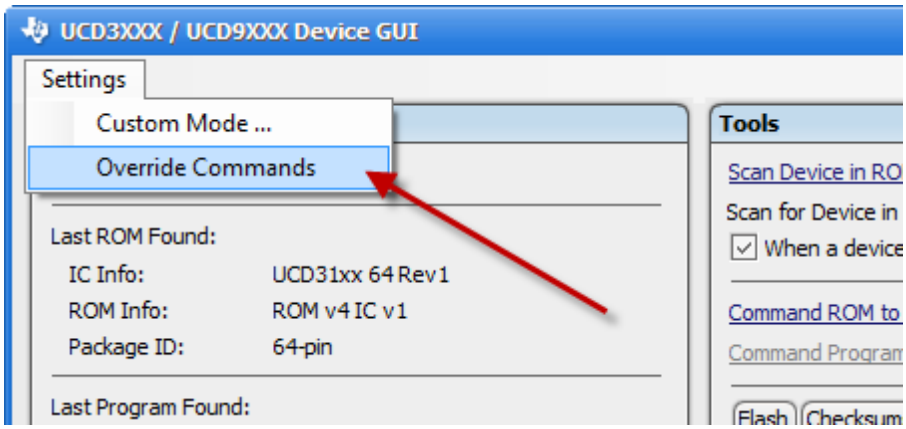


Figure 102 - Access to Override Commands

In the previous “Section 7.10 Function Command Summary”, a description was provided for the various commands used by the Device GUI. By default, the Device GUI assumes that certain MFR commands use a default hex code and are implemented a certain way. Sometimes this assumption is not valid and the user needs to override which command codes are used due to a conflict with another command having the same hex code. Assuming the implementation of the command is the same, the Device GUI provides a way to override or change the command code that the Device GUI assumes so that users can still benefit from the Device GUI. The following figure displays the available MFR commands that the user can override, assuming implementation has remained the same.

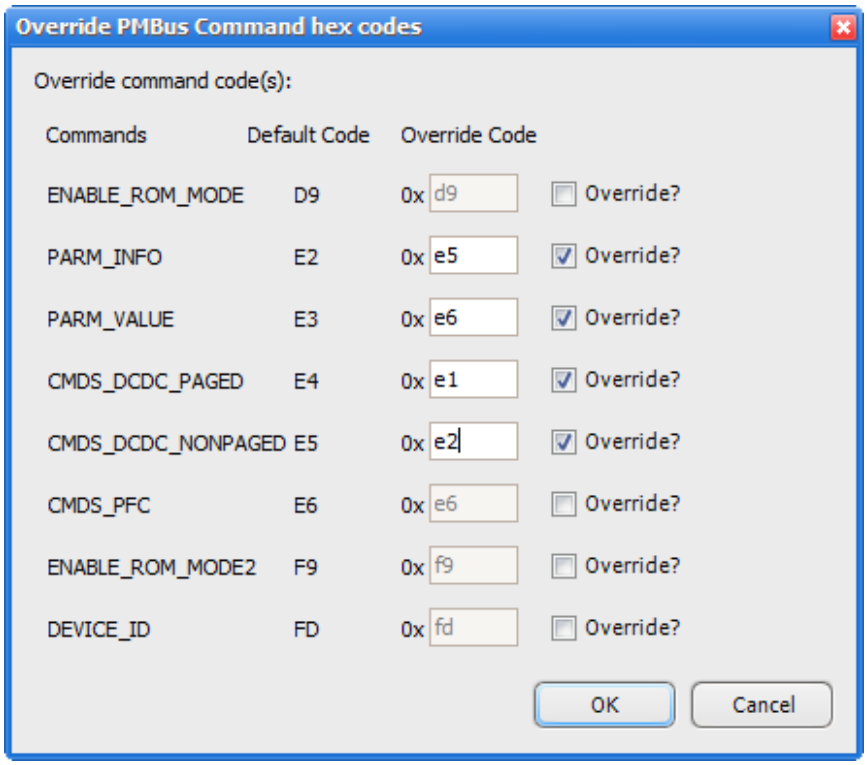


Figure 103 - four commands have been overridden

After overriding a link will appear on the top right saying “Override(4)” indicating the number of commands overridden. Clicking the link will launch the override screen again.

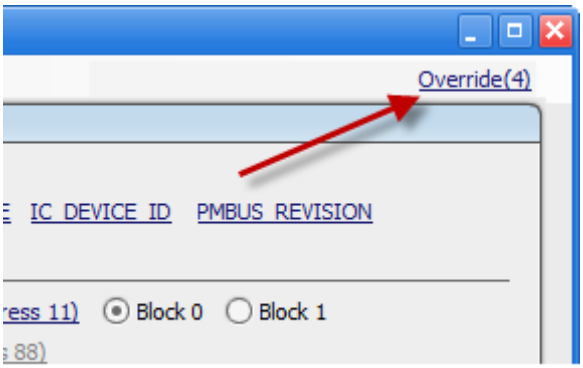


Figure 104 - Override in top right shown

8 API – Application Programming Interface

www.ti.com/tool/fusion_digital_power_api

There is a reusable API behind most of the functionality covered. It can be used via .NET: VB or C#. This can be used to automate tests or even create new custom GUIs. TI will provide binary libraries, source code for examples, and documentation.

9 Manufacturing Tool

www.ti.com/fusion-mfr-gui

When it is time for production there is another tool that has been used to speed up the process of configuring devices. It is called the Manufacturing GUI. This graphical tool can be used to run scripts on the devices and provide a pass/fail result. All functions done through the device GUI can be automated through the MFR GUI. Some of the functions included are downloading or updating firmware, importing a project file on to a device, writing serial numbers and MFR date, calibrating devices using instrumentation(GPIB, SCPI, USB) or manual measurements, testing the device's output and various other functions. Users can also develop their own functions to include in the manufacturing scripts.

10 Documentation and References

10.1 References

[1] PMBus™ specification http://PMBus.org/specs.html.

[i] USB Interface Adapter EVM (HPA172)

Revision History

Version	Date	Comment
SLUA676	**26 November 2013**	**Initial Document**
SLUA676A	**2 July 2014**	Updated Device GUI screenshots Updated Firmware download section Added CCS conversion section Added Override command section Added Function command summary Added device GUI checksum section Added link for Fusion-API
SLUA676A	**January 2015**	Include devices UCD3138128 and UCD3138A Added section on device respins and program mode detection Added section on partial dataflash download

IMPORTANT NOTICE

Texas Instruments Incorporated and its subsidiaries (TI) reserve the right to make corrections, modifications, enhancements, improvements, and other changes to its products and services at any time and to discontinue any product or service without notice. Customers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. All products are sold subject to TI's terms and conditions of sale supplied at the time of order acknowledgment. TI warrants performance of its hardware products to the specifications applicable at the time of sale in accordance with TI's standard warranty. Testing and other quality control techniques are used to the extent TI deems necessary to support this warranty. Except where mandated by government requirements, testing of all parameters of each product is not necessarily performed. TI assumes no liability for applications assistance or customer product design. Customers are responsible for their products and applications using TI components. To minimize the risks associated with customer products and applications, customers should provide adequate design and operating safeguards. TI does not warrant or represent that any license, either express or implied, is granted under any TI patent right, copyright, mask work right, or other TI intellectual property right relating to any combination, machine, or process in which TI products or services are used. Information published by TI regarding third-party products or services does not constitute a license from TI 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. Reproduction of TI information in TI data books or data sheets is permissible only if reproduction is without alteration and is accompanied by all associated warranties, conditions, limitations, and notices. Reproduction of this information with alteration is an unfair and deceptive business practice. TI is not responsible or liable for such altered documentation. Information of third parties may be subject to additional restrictions. Resale of TI products or services with statements different from or beyond the parameters stated by TI for that product or service voids all express and any implied warranties for the associated TI product or service and is an unfair and deceptive business practice. TI is not responsible or liable for any such statements. TI products are not authorized for use in safety-critical applications (such as life support) where a failure of the TI product would reasonably be expected to cause severe personal injury or death, unless officers of the parties have executed an agreement specifically governing such use. Buyers represent that they have all necessary expertise in the safety and regulatory ramifications of their applications, and acknowledge and agree that they are solely responsible for all legal, regulatory and safety-related requirements concerning their products and any use of TI products in such safety-critical applications, notwithstanding any applications-related information or support that may be provided by TI. Further, Buyers must fully indemnify TI and its representatives against any damages arising out of the use of TI products in such safety-critical applications. TI products are neither designed nor intended for use in military/aerospace applications or environments unless the TI products are specifically designated by TI as military-grade or "enhanced plastic." Only products designated by TI as military-grade meet military specifications. Buyers acknowledge and agree that any such use of TI products which TI has not designated as military-grade is solely at the Buyer's risk, and that they are solely responsible for compliance with all legal and regulatory requirements in connection with such use. TI products are neither designed nor intended for use in automotive applications or environments unless the specific TI products are designated by TI as compliant with ISO/TS 16949 requirements. Buyers acknowledge and agree that, if they use any non-designated products in automotive applications, TI will not be responsible for any failure to meet such requirements. Following are URLs where you can obtain information on other Texas Instruments products and application solutions:

Products	**Applications**
Audio www.ti.com/audio	Automotive and Transportation www.ti.com/automotive
Amplifiers amplifier.ti.com	Communications and Telecom www.ti.com/communications
Data Converters dataconverter.ti.com	Computers and Peripherals www.ti.com/computers
DLP® Products www.dlp.com	Consumer Electronics www.ti.com/consumer-apps
DSP dsp.ti.com	Energy and Lighting www.ti.com/energy
Clocks and Timers www.ti.com/clocks	Industrial www.ti.com/industrial
Interface interface.ti.com	Medical www.ti.com/medical
Logic logic.ti.com	Security www.ti.com/security
Power Mgmt power.ti.com	Space, Avionics and Defense www.ti.com/space-avionics-defense
Microcontrollers microcontroller.ti.com	Video and Imaging www.ti.com/video
RFID www.ti-rfid.com	OMAP Mobile Processors www.ti.com/omap
Wireless Connectivity www.ti.com/wirelessconnectivity	

TI E2E Community Home Page e2e.ti.com

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 7526

IMPORTANT NOTICE

Texas Instruments Incorporated and its subsidiaries (TI) reserve 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 (also referred to herein as “components”) 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 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.

TI does not warrant or represent that any license, either express or implied, is granted under 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.

Reproduction of significant portions of TI information in TI data books or data sheets 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.

Resale of TI components or services with statements different from or beyond the parameters stated by TI for that component or service voids all express and any implied warranties for the associated TI component or service and is an unfair and deceptive business practice. TI is not responsible or liable for any such statements.

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 which anticipate dangerous consequences of failures, monitor failures and their consequences, lessen the likelihood of failures that might cause harm 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 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 a special agreement specifically governing such use.

Only those TI components which 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 which have ***not*** been so designated is solely at the Buyer's risk, and that 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.

Products		**Applications**	
Audio	www.ti.com/audio	Automotive and Transportation	www.ti.com/automotive
Amplifiers	amplifier.ti.com	Communications and Telecom	www.ti.com/communications
Data Converters	dataconverter.ti.com	Computers and Peripherals	www.ti.com/computers
DLP® Products	www.dlp.com	Consumer Electronics	www.ti.com/consumer-apps
DSP	dsp.ti.com	Energy and Lighting	www.ti.com/energy
Clocks and Timers	www.ti.com/clocks	Industrial	www.ti.com/industrial
Interface	interface.ti.com	Medical	www.ti.com/medical
Logic	logic.ti.com	Security	www.ti.com/security
Power Mgmt	power.ti.com	Space, Avionics and Defense	www.ti.com/space-avionics-defense
Microcontrollers	microcontroller.ti.com	Video and Imaging	www.ti.com/video
RFID	www.ti-rfid.com		
OMAP Applications Processors	www.ti.com/omap	**TI E2E Community**	e2e.ti.com
Wireless Connectivity	www.ti.com/wirelessconnectivity		

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