Isolated GUI Tools
User Guide

1 Revision History

Version	Date	Comment
Fusion GUI 1.8.130	**October 27, 2011**	**Draft**

2 IMPORTANT NOTICE

Texas Instruments and its subsidiaries (TI) reserve the right to make changes to their products or to discontinue any product or service without notice, and advise customers to obtain the latest version of relevant information to verify, before placing orders, that information being relied on is current and complete. All products are sold subject to the terms and conditions of sale supplied at the time of order acknowledgement, including those pertaining to warranty, patent infringement, and limitation of liability.

TI warrants performance of its semiconductor products to the specifications applicable at the time of sale in accordance with TI's standard warranty. Testing and other quality control techniques are utilized to the extent TI deems necessary to support this warranty. Specific testing of all parameters of each device is not necessarily performed, except those mandated by government requirements.

Customers are responsible for their applications using TI components.

In order to minimize risks associated with the customer's applications, adequate design and operating safeguards must be provided by the customer to minimize inherent or procedural hazards.

TI assumes no liability for applications assistance or customer product design. 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 of TI covering or relating to any combination, machine, or process in which such products or services might be or are used. TI's publication of information regarding any third party's products or services does not constitute TI's approval, license, warranty or endorsement thereof.

Reproduction of 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. Representation or reproduction of this information with alteration voids all warranties provided for an associated TI product or service, is an unfair and deceptive business practice, and TI is not responsible or liable for any such use.

Resale of TI's products or services with *statements different from or beyond the parameters* stated by TI for that products or service voids all express and any implied warranties for the associated TI product or service, is an unfair and deceptive business practice, and TI is not responsible nor liable for any such use.

Also see: Standard Terms and Conditions of Sale for Semiconductor Products.

www.ti.com/sc/docs/stdterms.htm

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

3 Contents

3.1 Table of Contents

3.2 Figures

4 About This User's Guide

4.1 Documented Program Version

This user guide describes Fusion Digital Power Designer and some of the tools that come packaged with it in the Installer. The installer has components that are related to non-isolated and isolated users. This guide is written specifically with isolated users in mind, with emphasis on the UCD31XX.

4.2 Conventions

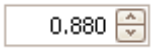
Any hexadecimal number will be prefixed by 0x. For example, 0xFF. Any other number should be assumed to be decimal.

4.3 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.

4.4 Terminology

Designer GUI or GUI – refers to Fusion Digital Power Designer GUI (main tool)

Device GUI or Engineering GUI – refers to UCD3xxx Device GUI

4.5 Providing Feedback on this User's Guide

Please contact your Texas Instruments representative to give feedback on this document.

5 Getting Started

5.1 PC Requirements

The GUI requires the following:

- A PC running Windows XP/Windows 7
- Microsoft.NET Framework version 2.0

Microsoft.NET is the runtime application framework that the GUI uses. The GUI's installer will ensure version 2.0 of .NET is installed, and install if necessary.

5.2 USB Adapter

The EVM is attached to the PC through a Texas Instruments serial bus adapter, part number HPA172[i]. The user should have received this adapter with an EVM. 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."

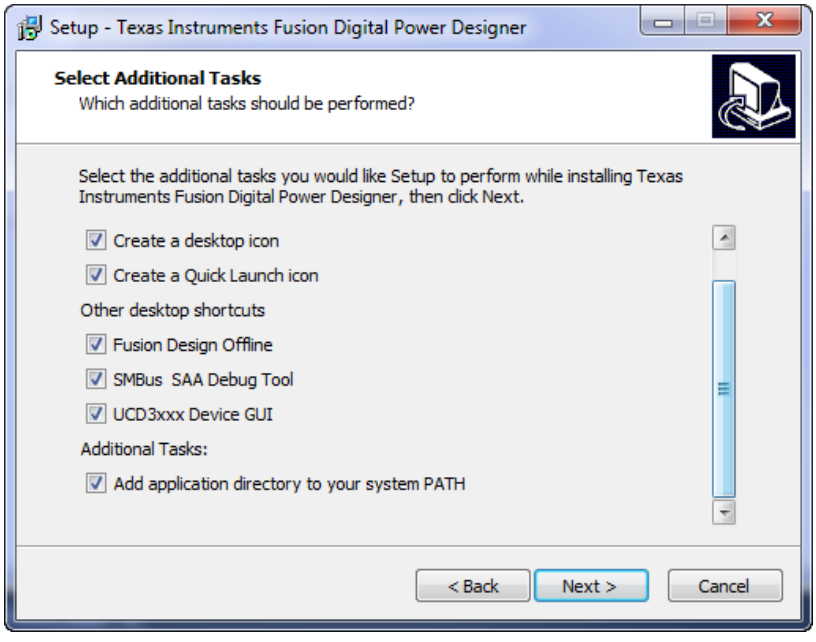
5.3 Download & Installation

The latest public production versions can be found at http://www.ti.com/fusiongui. 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.



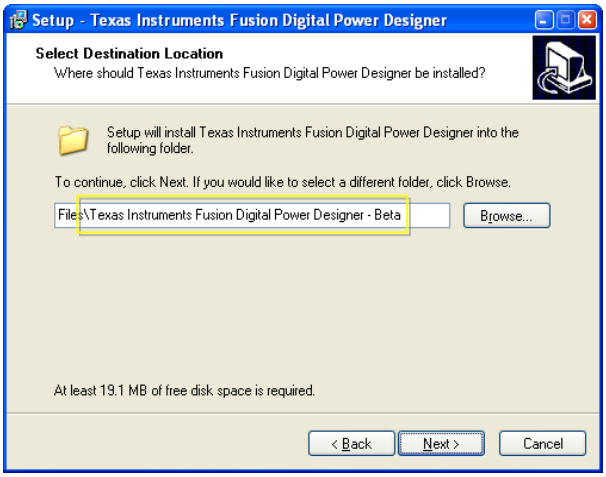
5.4 Upgrading the GUI

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.

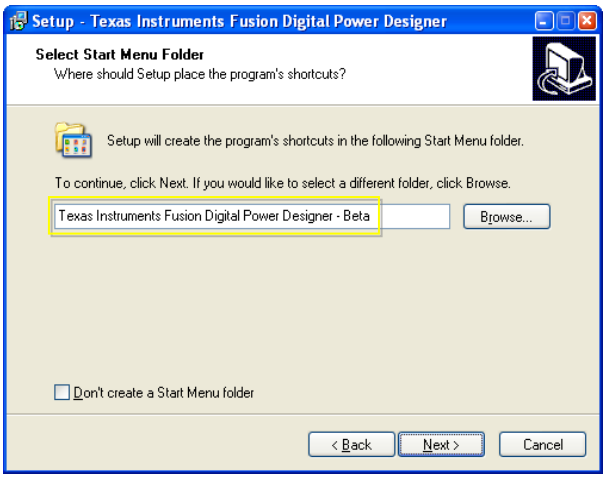
5.5 Multiple Installations of the GUI

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.

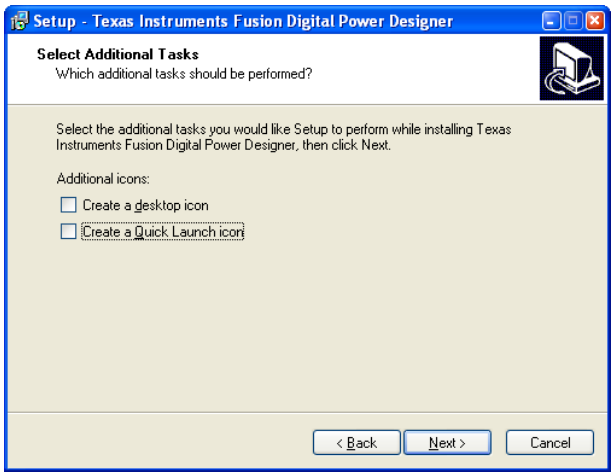
When you install a second copy of the GUI, you need to ensure the name of the folder for the additional copy is named different from the default folder name, “Texas Instruments Fusion Digital Power Designer.” The easiest way to do this is to append something descriptive to the folder name. For example,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



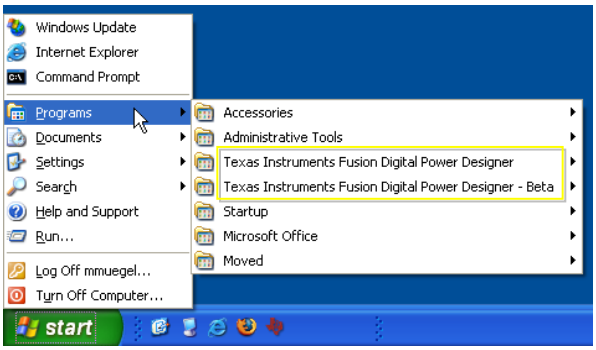
You will also need to rename the Start Menu folder that gets created. Again, “ – Beta” has been appended to the default:



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 GUI from the Start Menu:



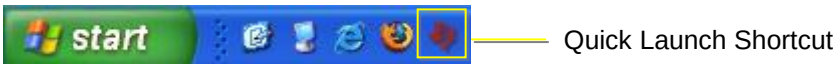
5.6 Overview

The Fusion Digital Power Designer or Designer GUI is the main tool of the package. In addition to the Designer GUI there are a number of tools that are very helpful. Depending on what stage of development, one tool may be more advantageous than another. This user guide aims to cover the most important tools included in the installer.

6 Fusion Digital Power Designer (Designer GUI)

6.1 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.



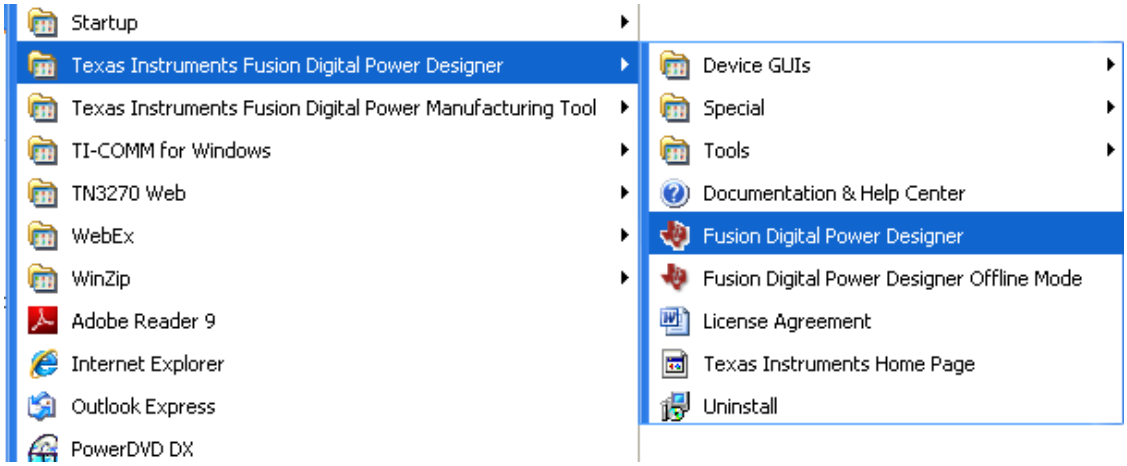
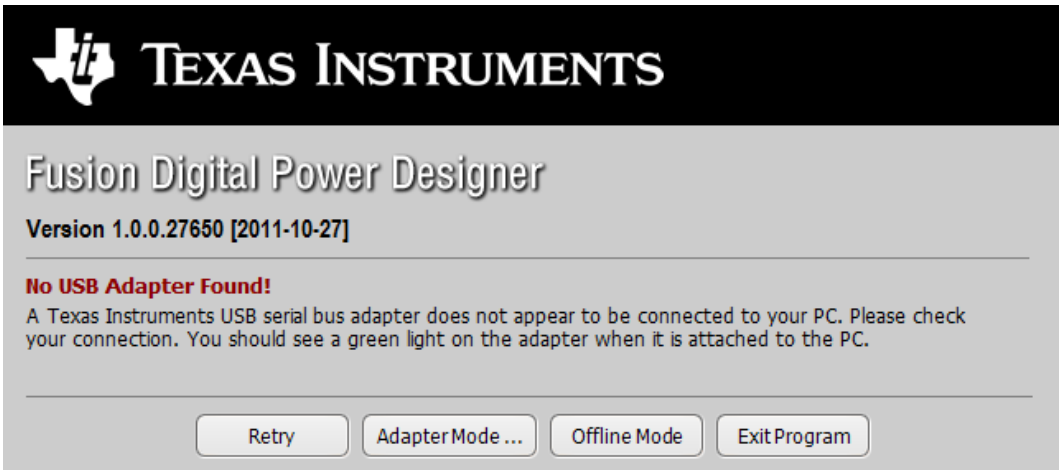


Figure 1 - Accessing Designer GUI from Start Menu

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

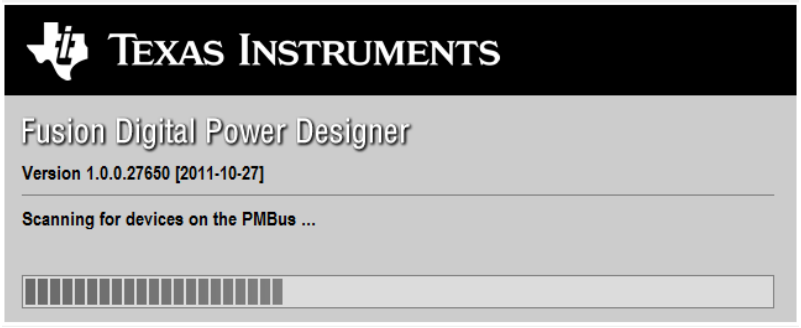
1. The GUI looks for an attached USB serial bus adapter. If it is not found you will see the following figure.



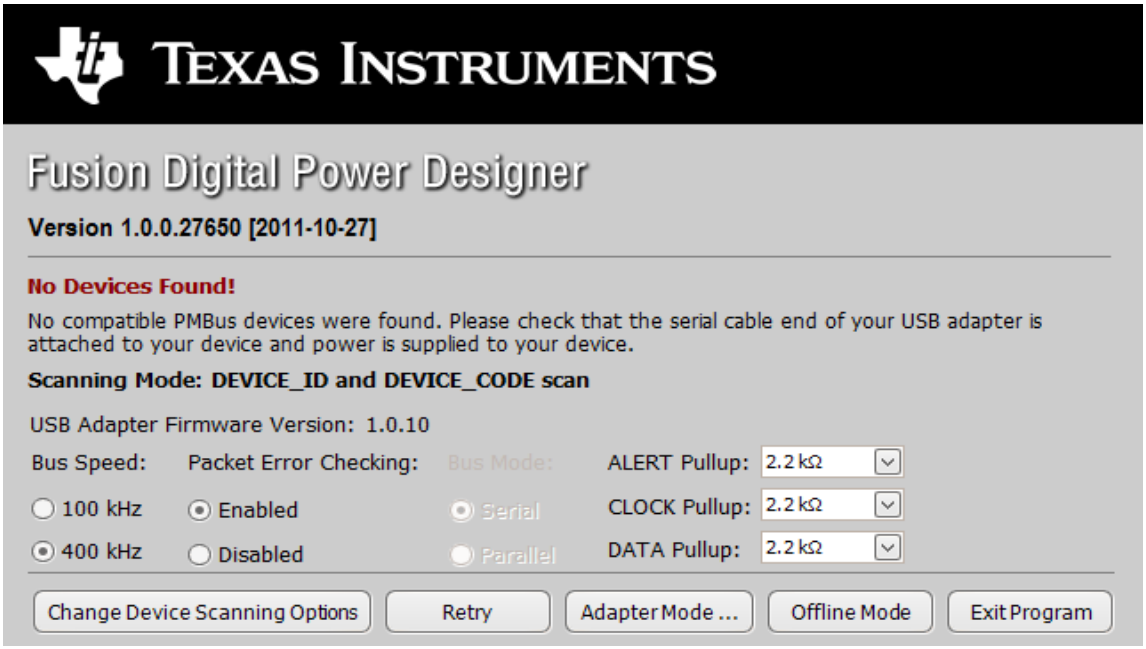
2. The GUI sends SMBus commands to the “broadcast” address 11 telling any devices that are in ROM mode to execute their program (go to flash mode). While this is not necessary for production devices such, it may be necessary for in-development products that are set to boot to ROM mode.

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 analysed. 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:



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



Double check your USB adapter connection and power to your device and click "Retry" if you would like to retry the scan.

If the GUI is still unable to detect the device see the following troubleshooting tips in “Section 6.1.1 Connection Troubleshooting Tips.”

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.

6.1.1 Connection Troubleshooting Tips

Problem	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 can not 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

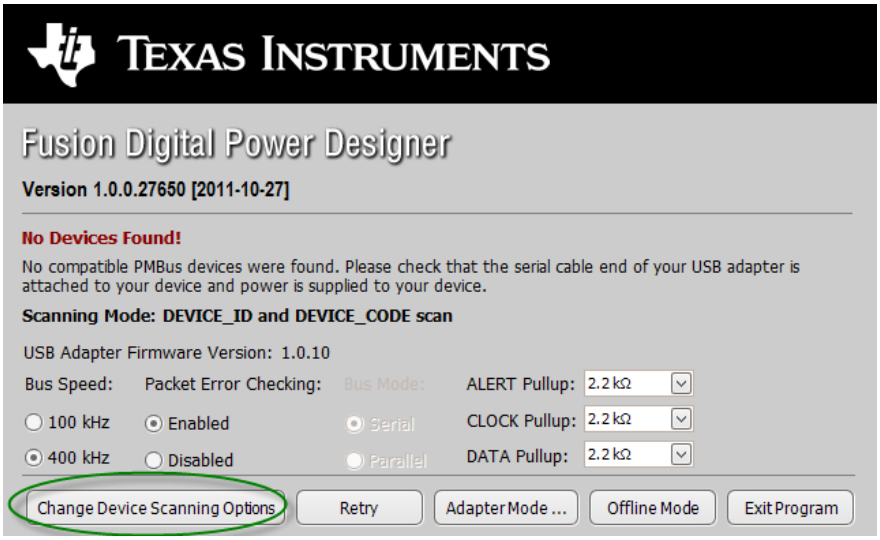


Figure 2 - Select Change Device Scanning Options

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.

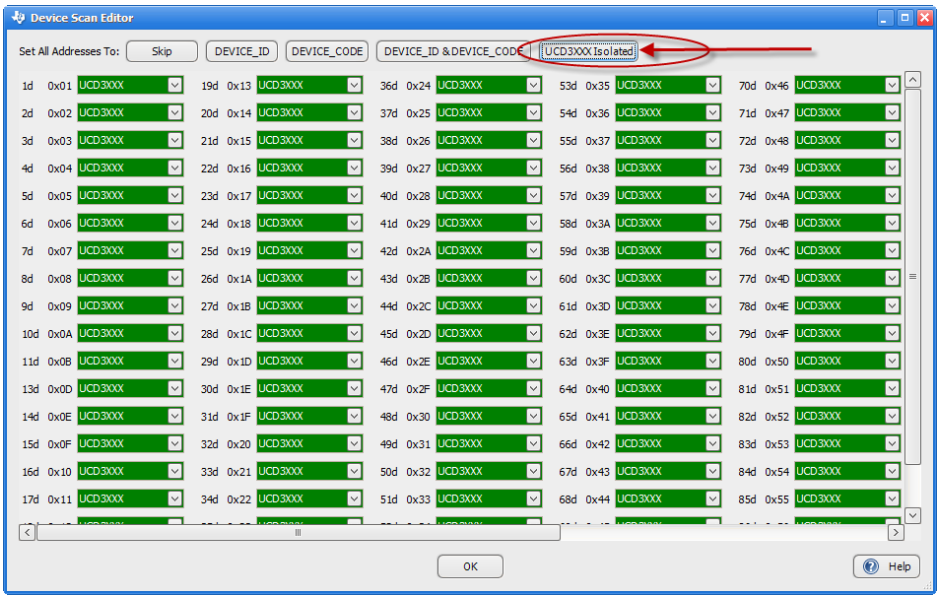


Figure 3 - Click UCD3XXX Isolated at top

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.

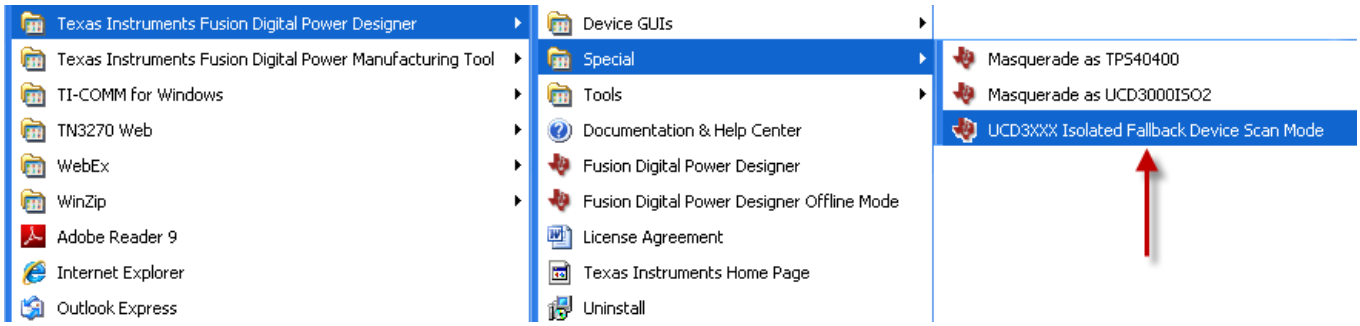


Figure 4 - Start>Texas Instruments...>Special>UCD3XXX Isolated Fallback Device Scan Mode

6.2 Enable GUI Protected Features

Figure 5 shows how to access the configuration screen to enable the GUI protected features. Figure 6 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.

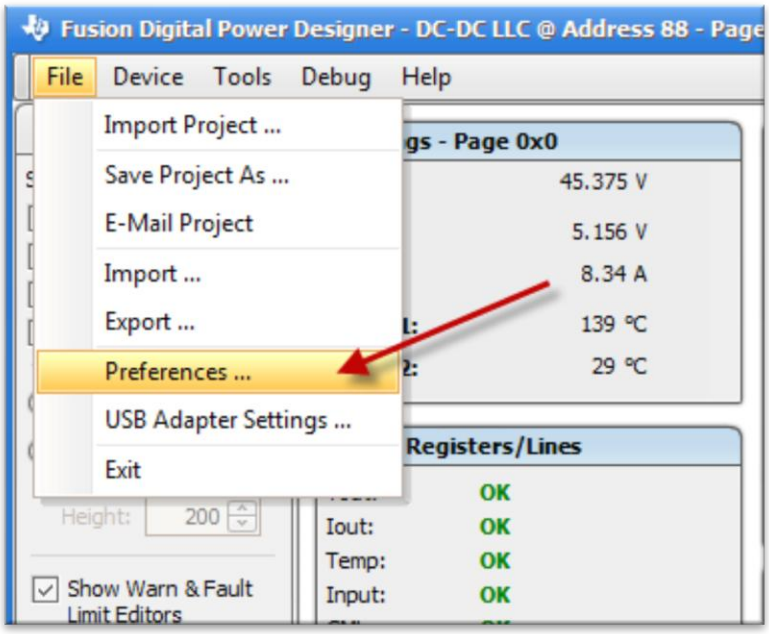


Figure 5: GUI Preferences

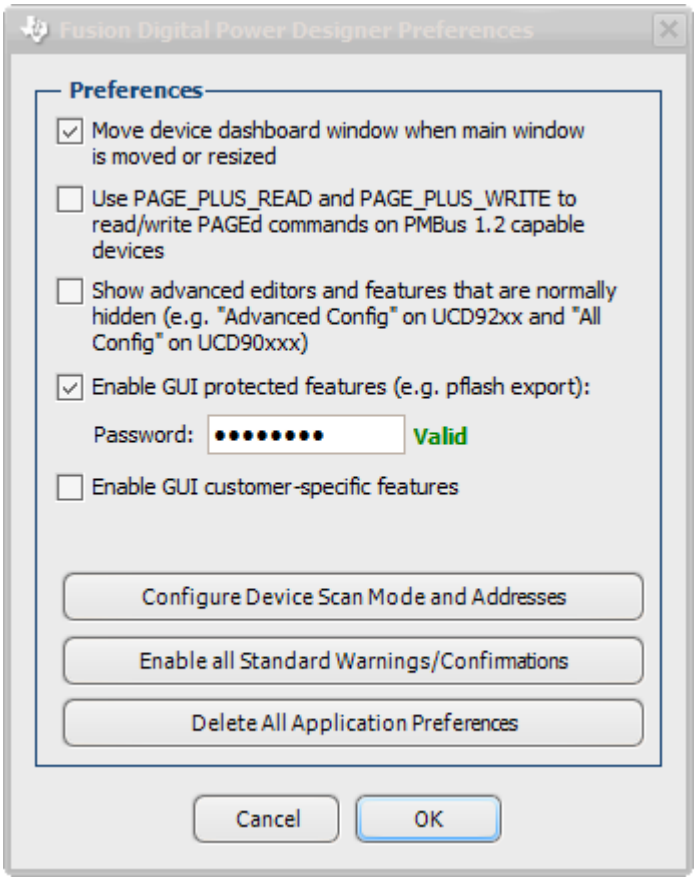


Figure 6: GUI Protected Features

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.

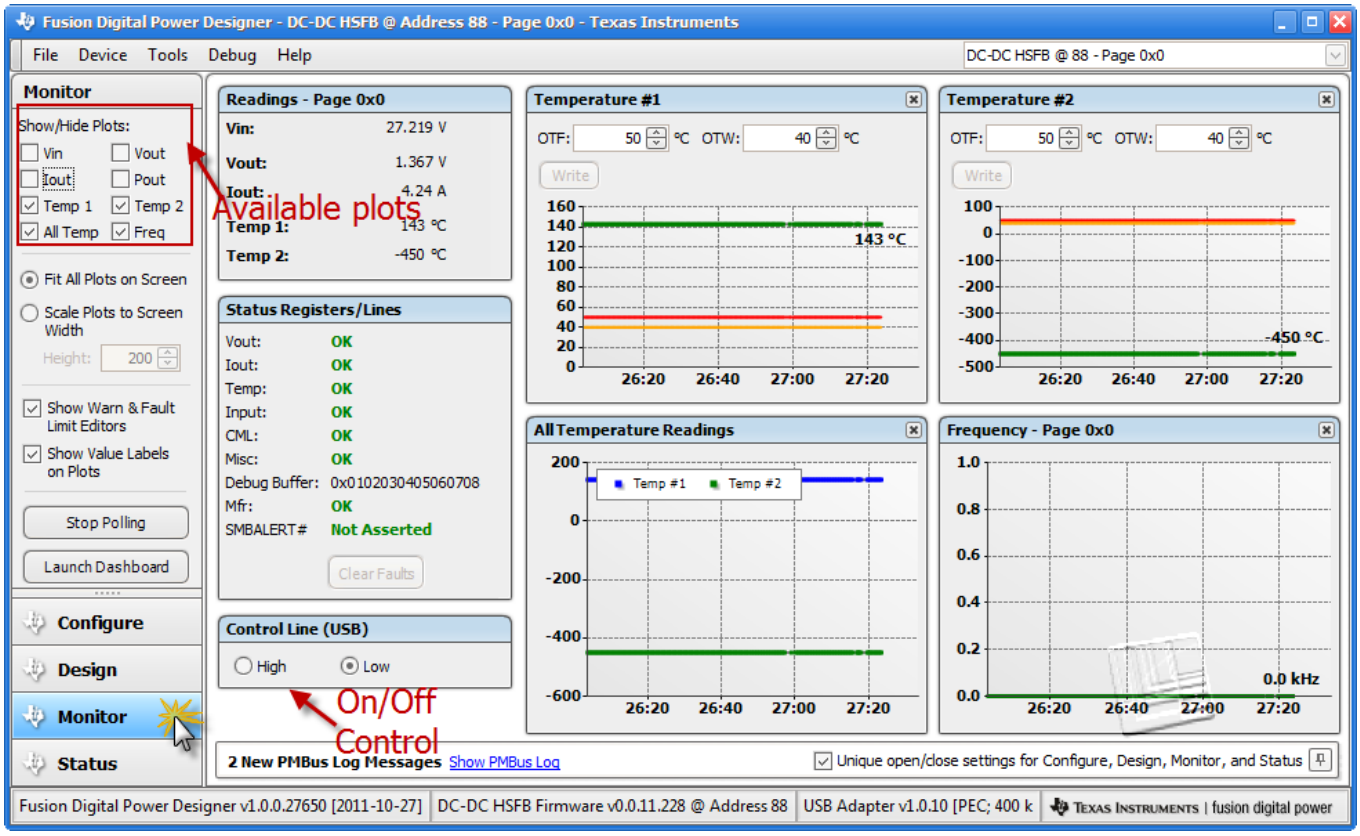
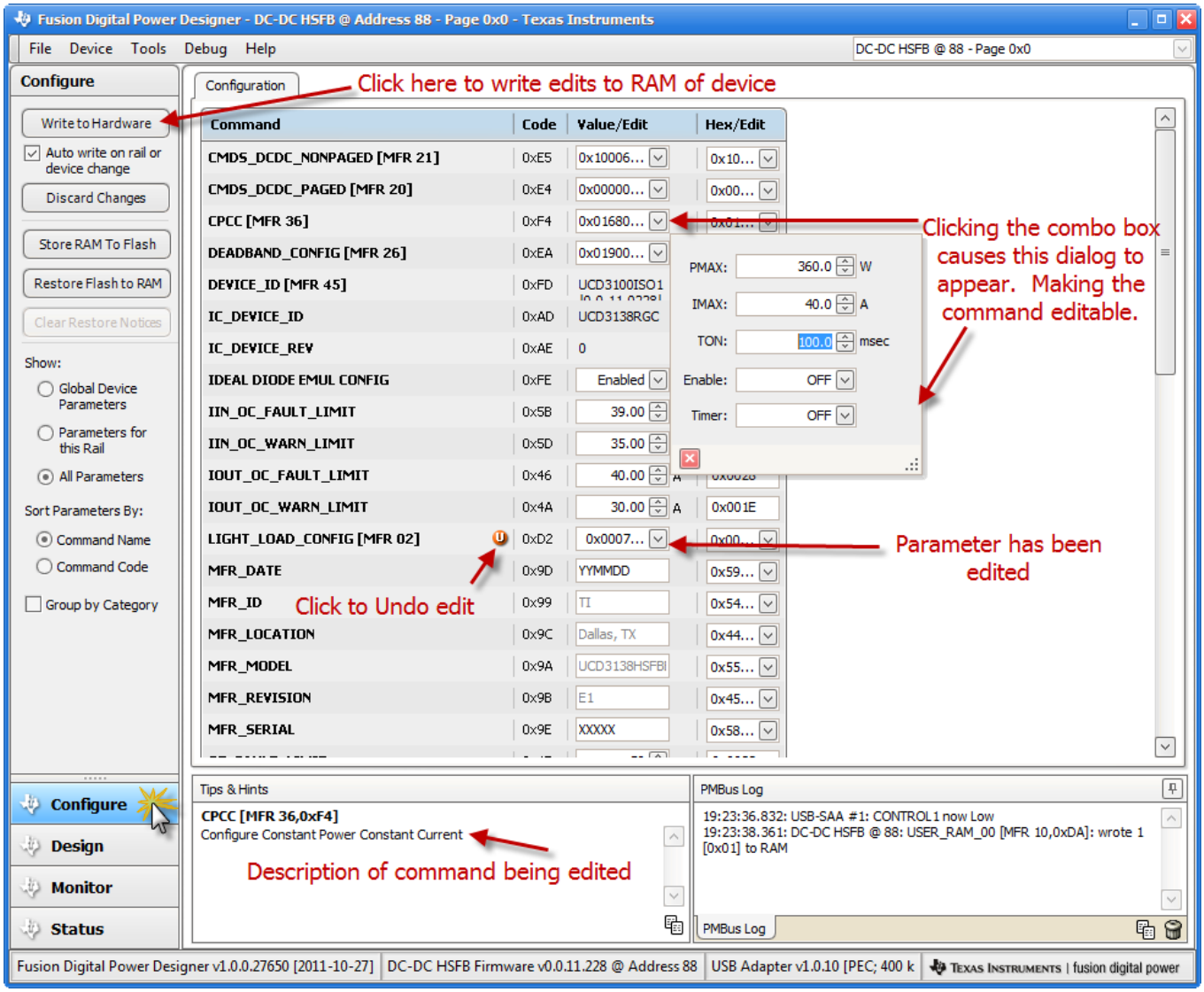


Figure 7 - Monitor mode displays some of the live parameters being read from the device.

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



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 do 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 then “Store RAM To Flash” would need to be clicked. “Section 6.4.3 Saving PMBus command values to local file” discusses saving 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 do 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 relaunch 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. “Section 7.4 Isolated Bitmask Tool” discusses 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”

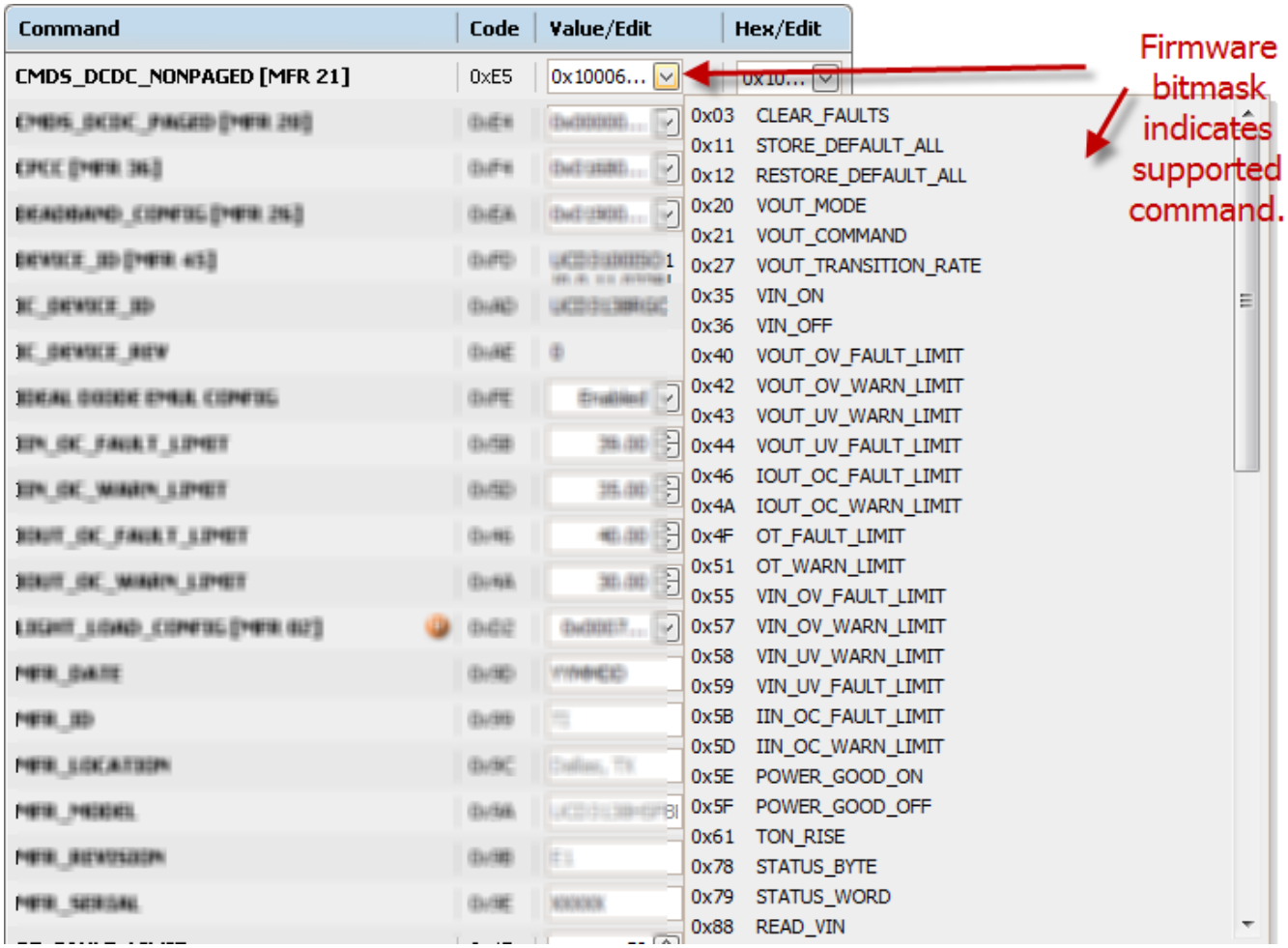


Figure 8 - Displays the list of commands the firmware supports

6.4.3 Saving PMBus command values to local file

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 your topology.

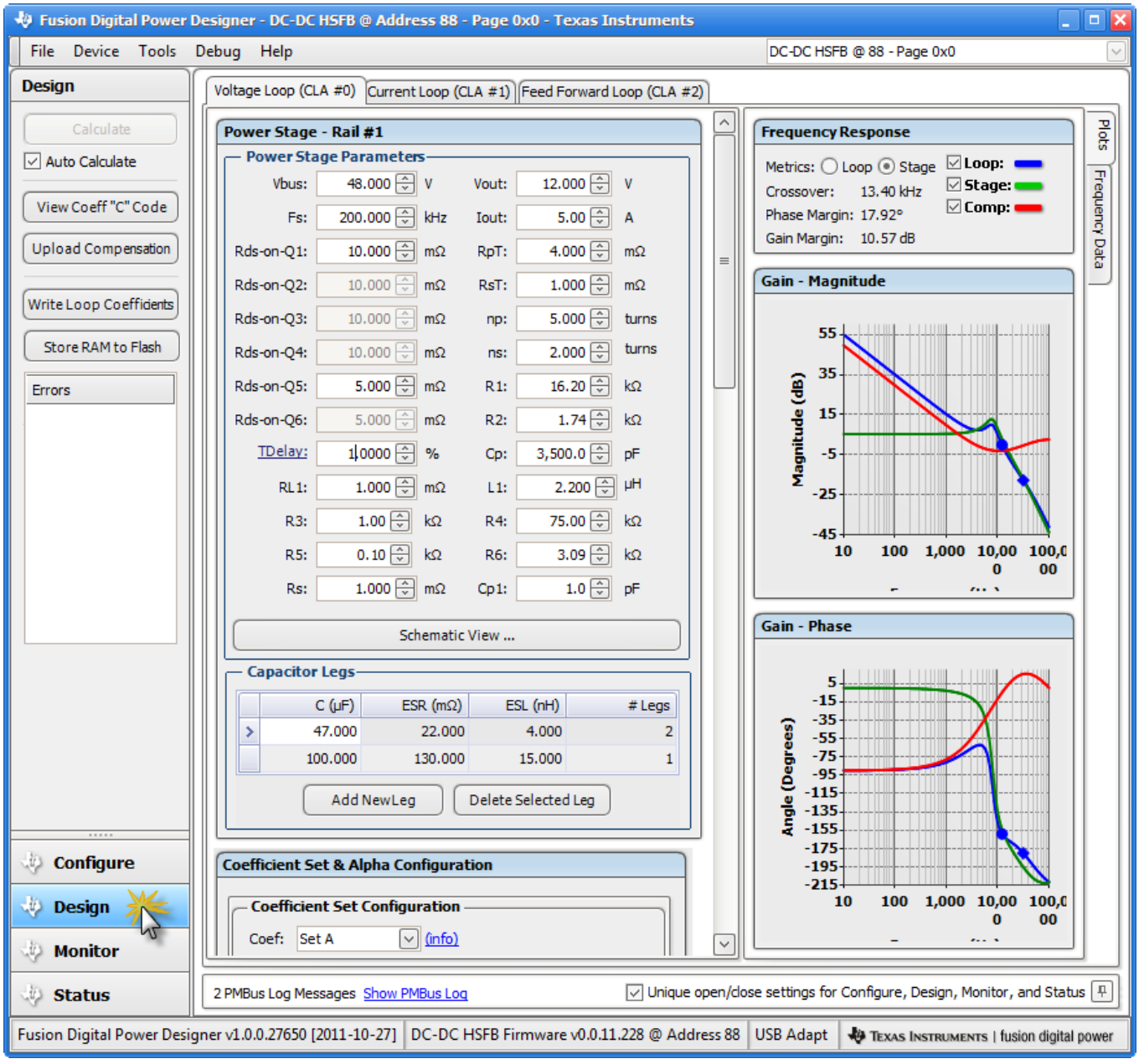


Figure 9 - Design mode selected

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:

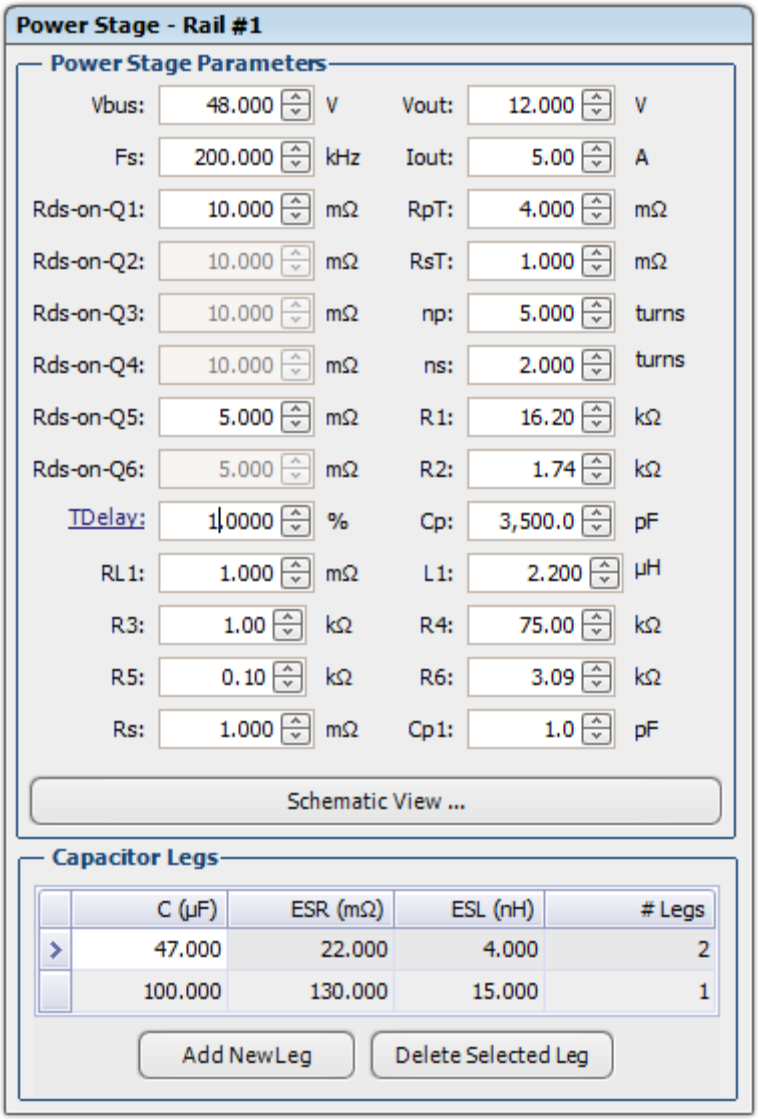


Figure 10 - Stage parameters for HSFB for Voltage Loop (CLA #0)

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. The power stage equation differs from loop to loop. The figure above is part of the voltage loop as shown in Figure 9.

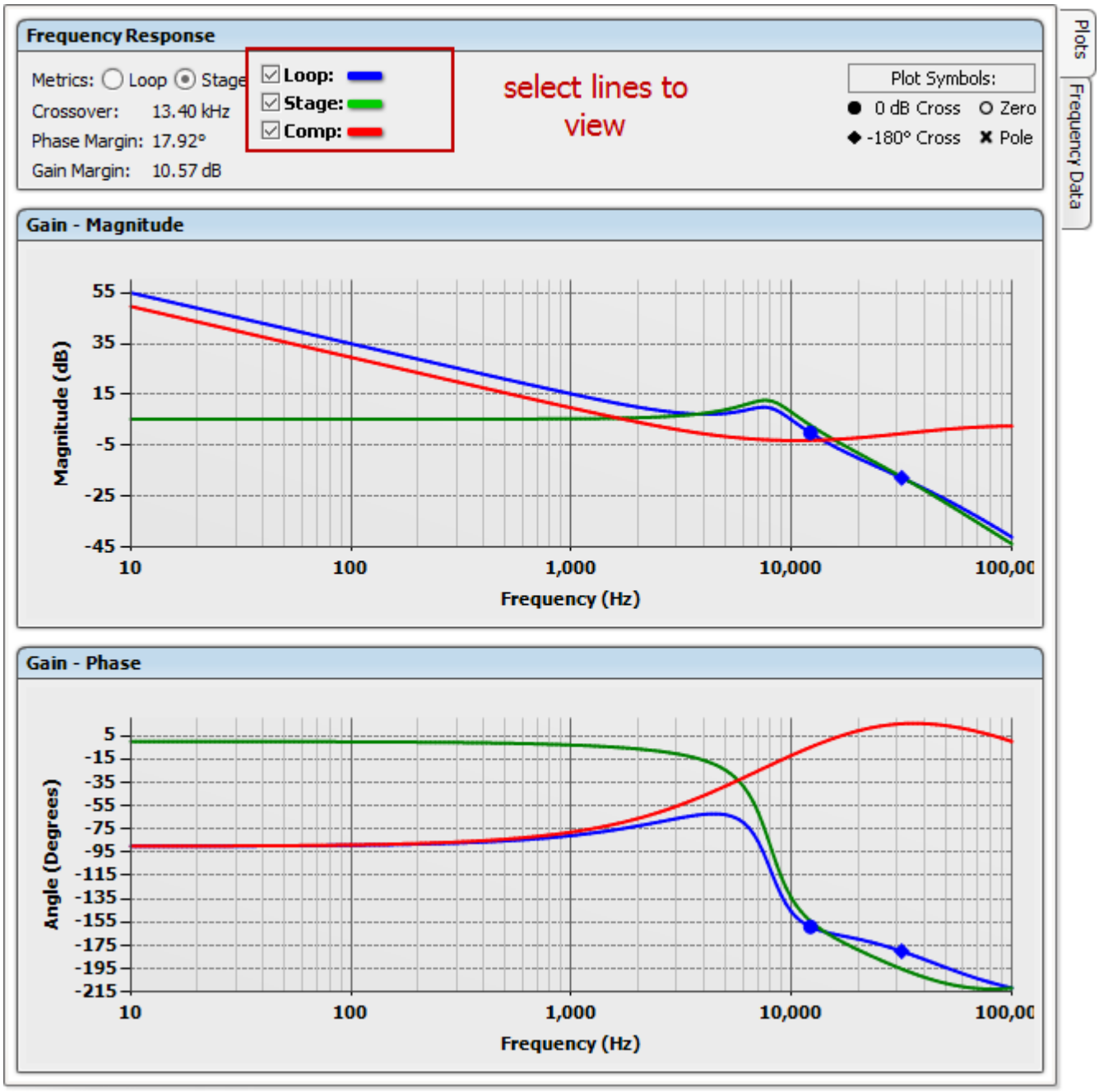


Figure 11 - Bode plots

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 10 will open a dialog with a picture of the schematic. See below.

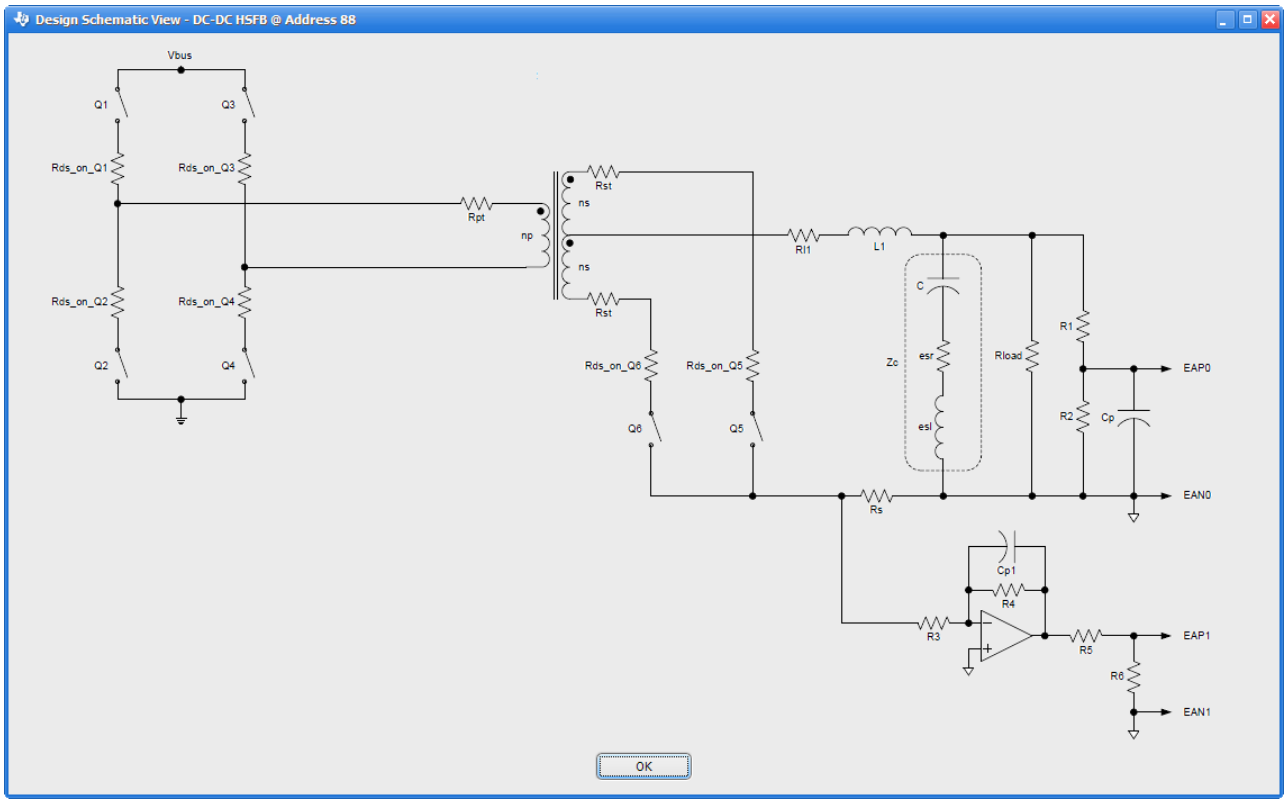


Figure 12 - HSFB schematic being modelled

The Bode plots are updated automatically as you set the values.

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.

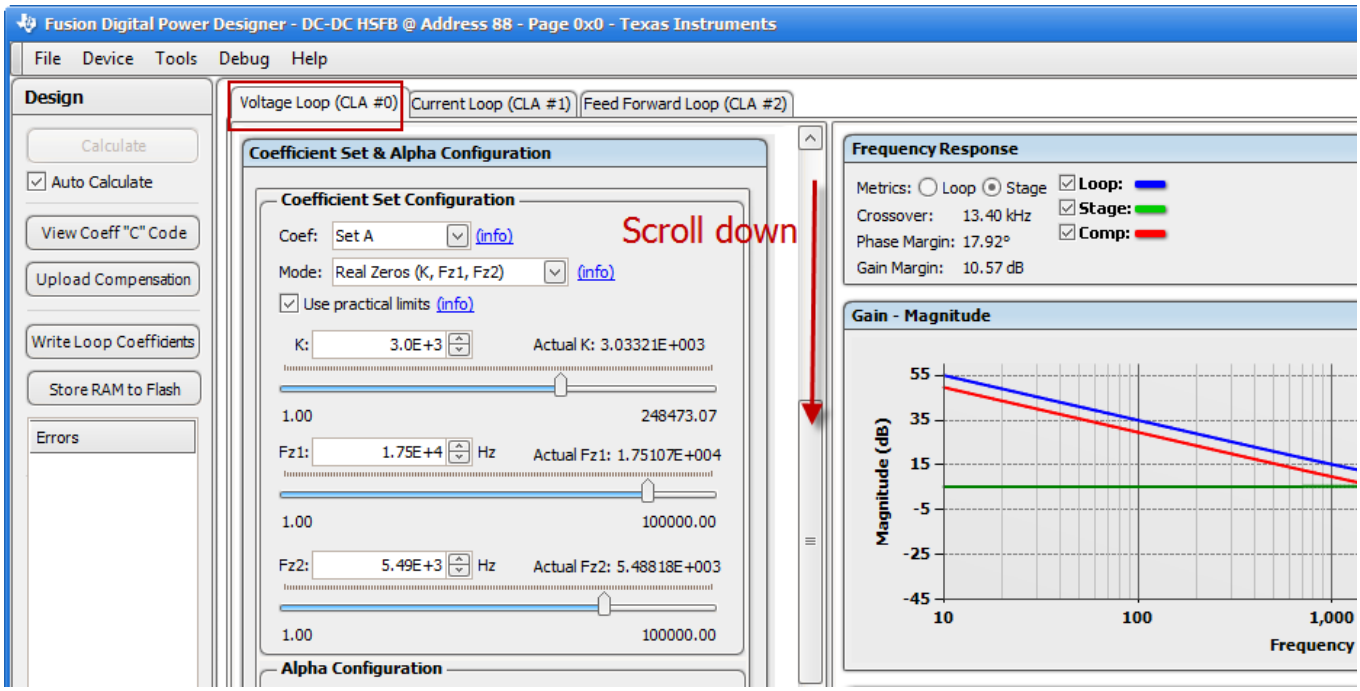


Figure 13 - Scroll down the stage parameters to see the compensator

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 user guide and training materials for details.

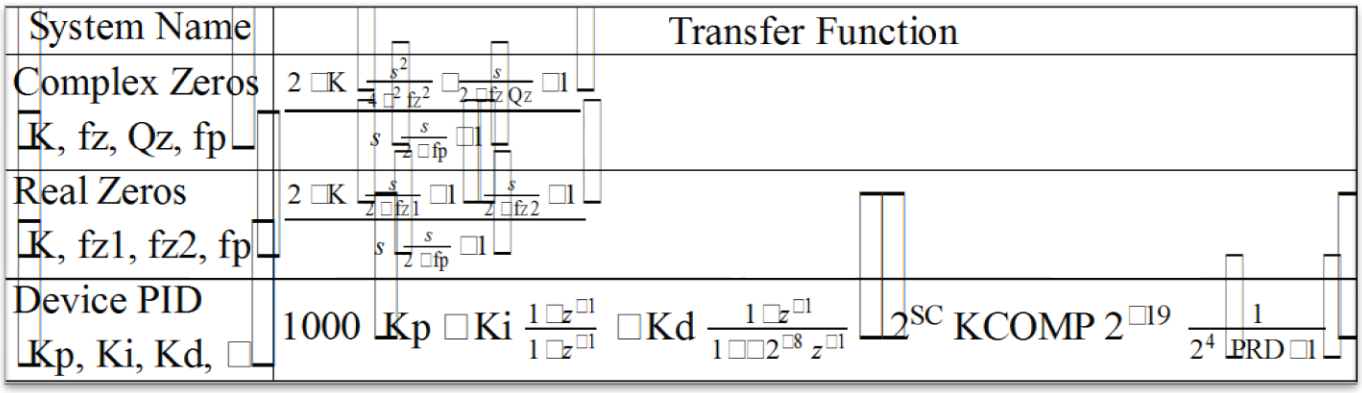


Figure 14 - Three ways to program compensator

6.5.2.1 Coefficient sets and Alpha

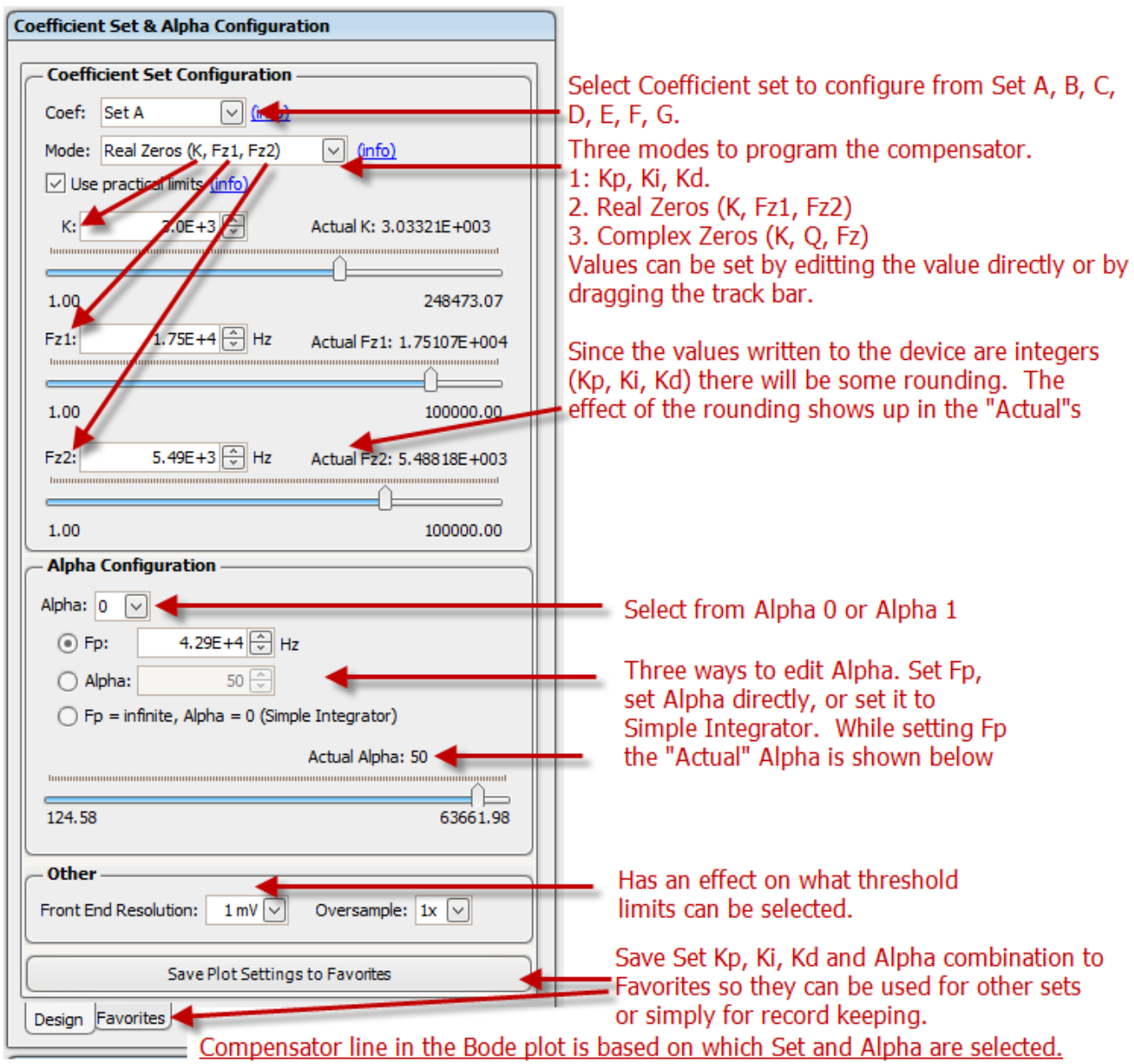


Figure 15 - Coefficient Set and Alpha Configuration

6.5.2.2 Bode Plot

The Bode plot located on the right of Figure 13 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 15.

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 16.

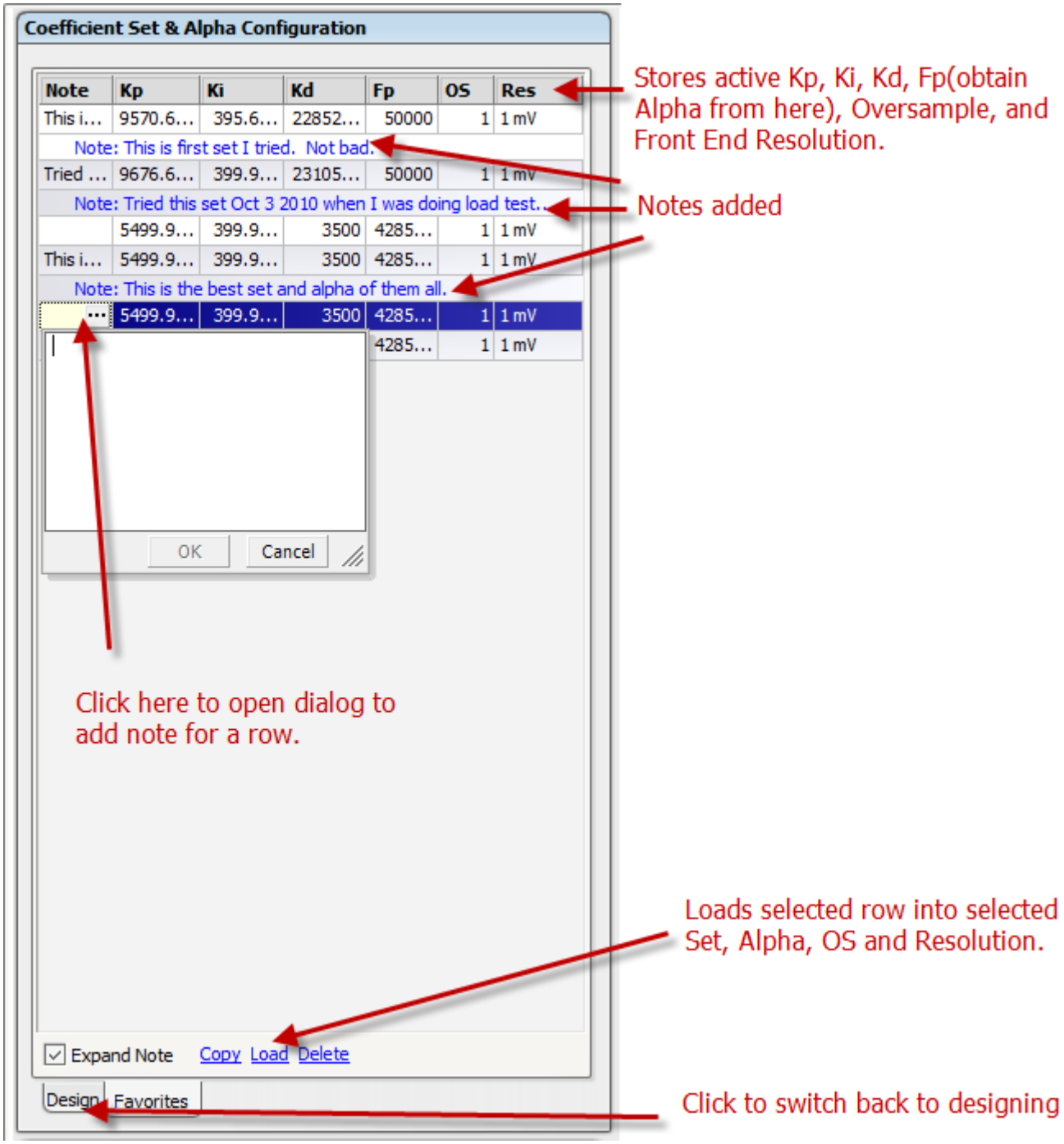


Figure 16 - Favorites

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.

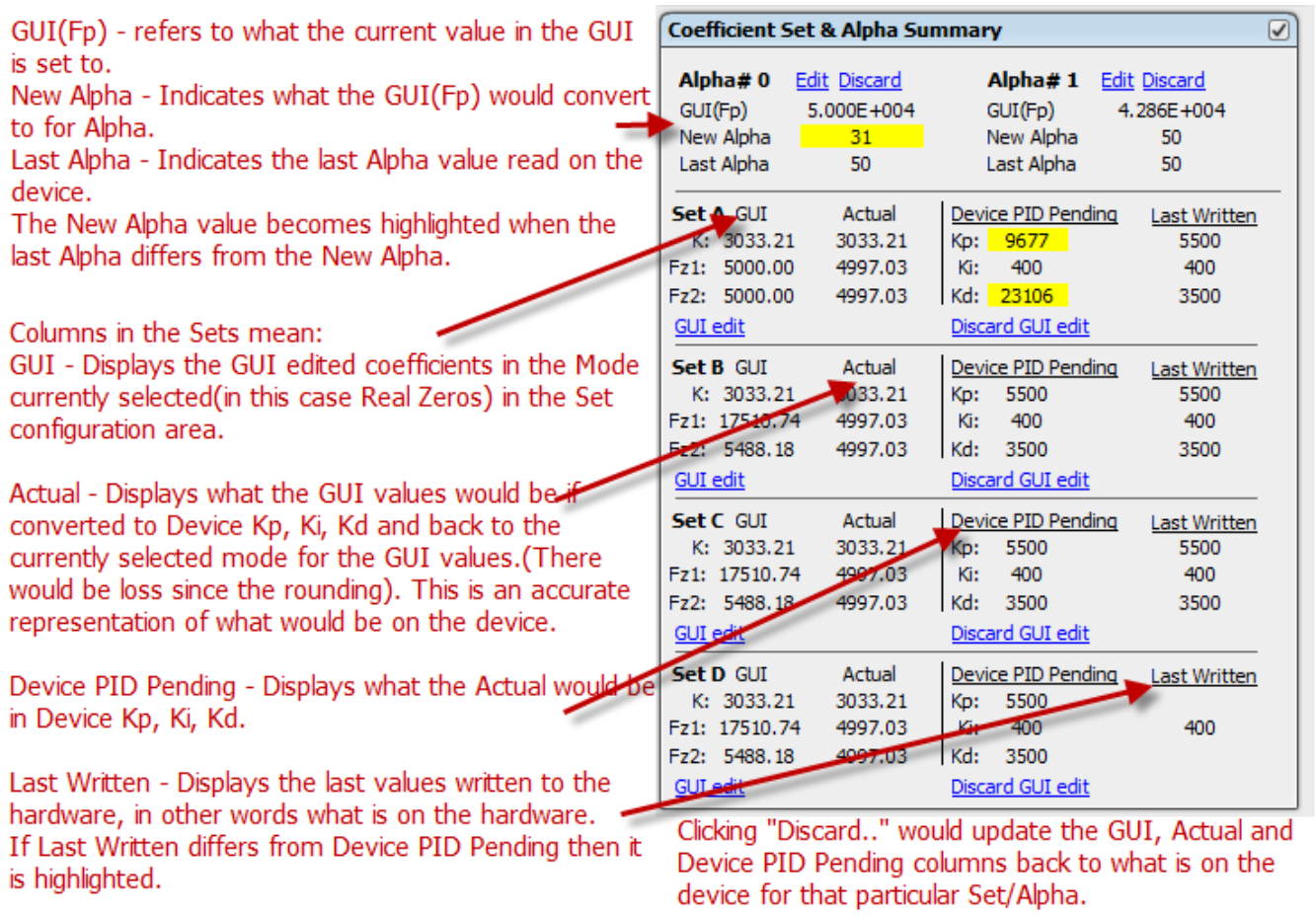


Figure 17 - Coefficient Set & Alpha Summary

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.

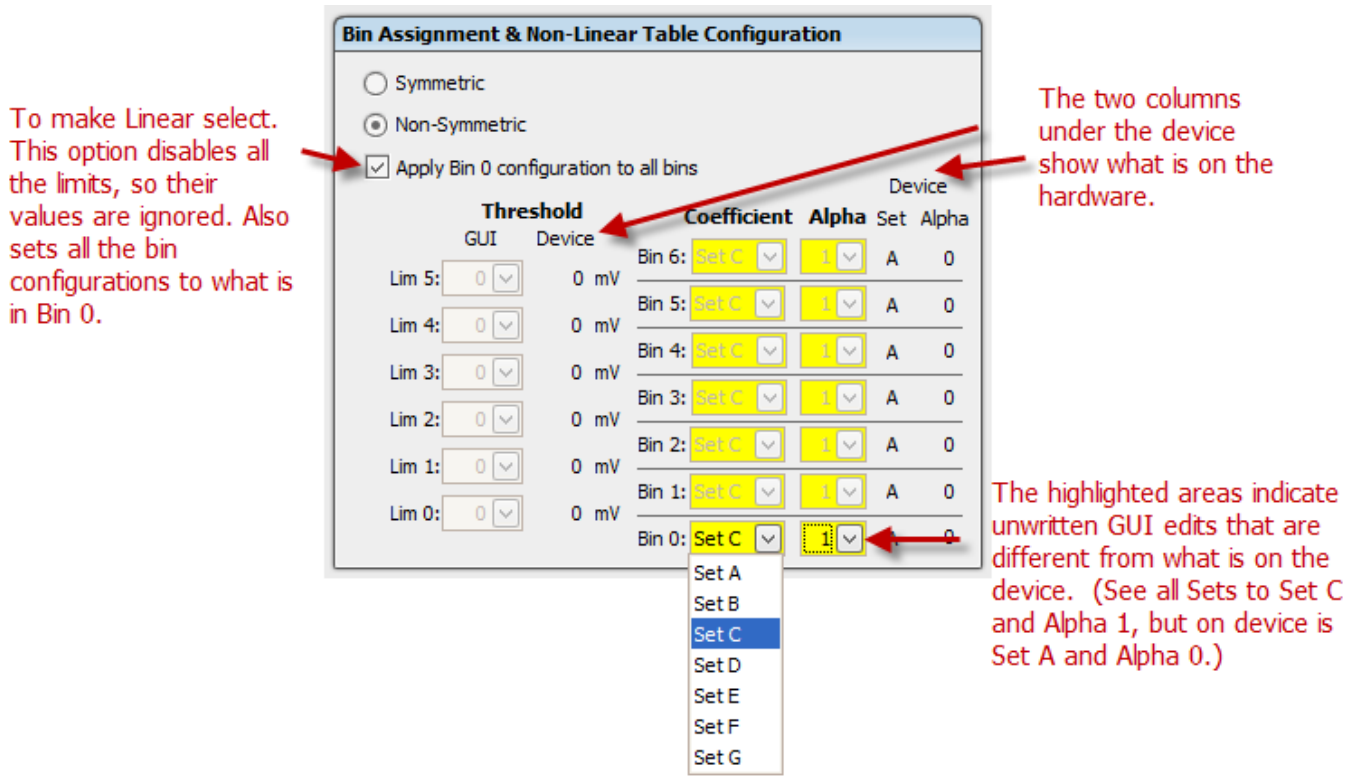


Figure 18 - Apply Bin 0 to all bins (Linear)

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.

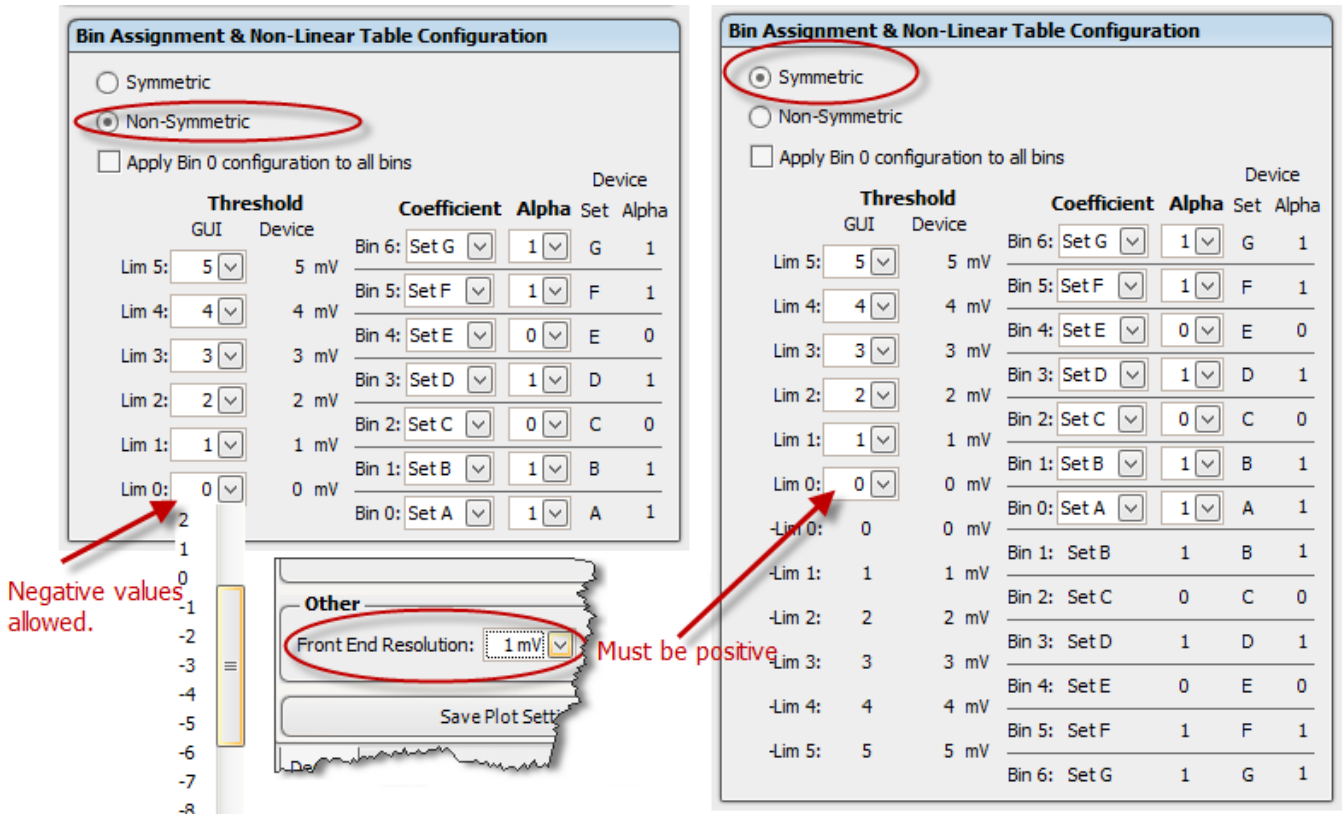


Figure 19 - Symmetric and Non-Symmetric

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.

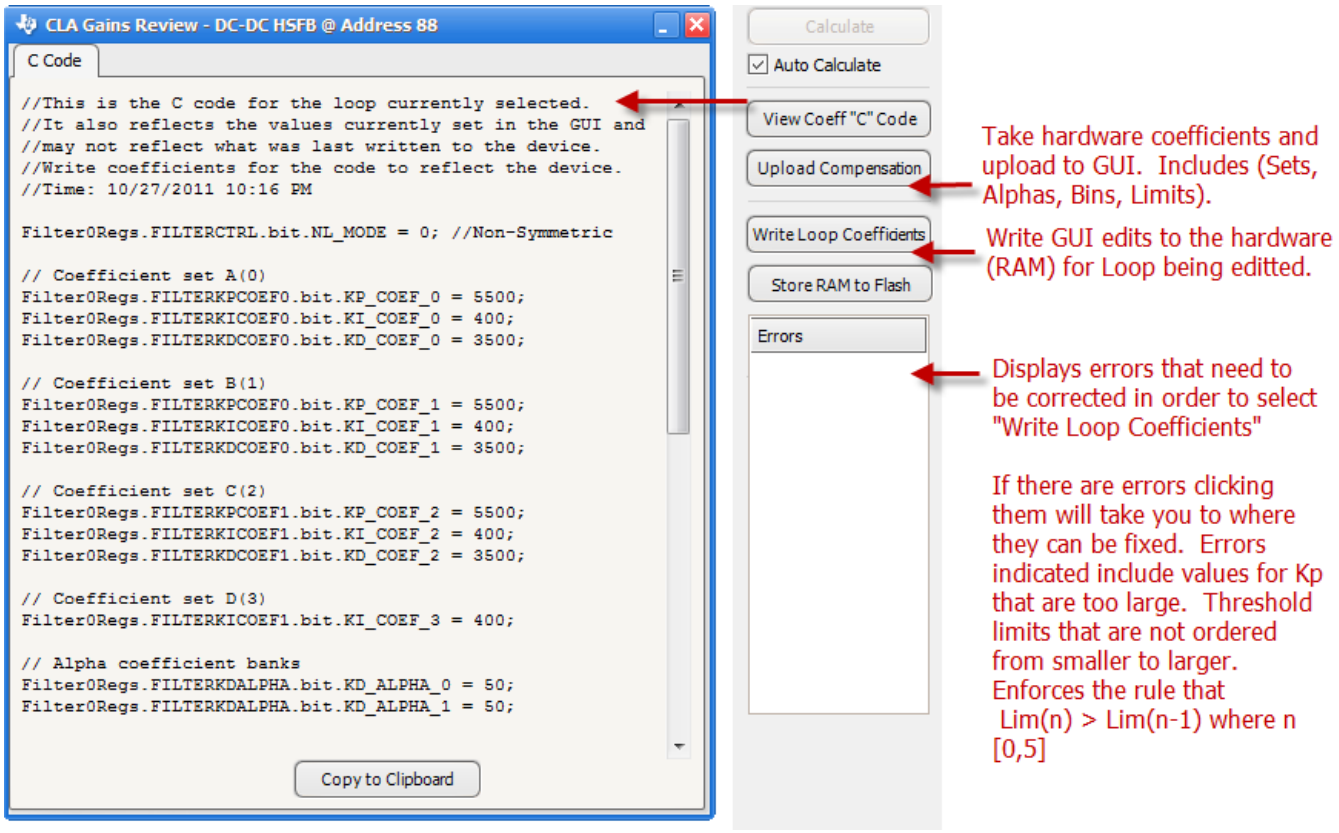


Figure 20 - Writing Loop Coefficients, and global reset of GUI edits to hardware coefficients

6.6 Status

The final mode is the status tab. It provides additional details on the type of fault or warning. Figure 21 - Status Mode shows a screen shot of this tab.

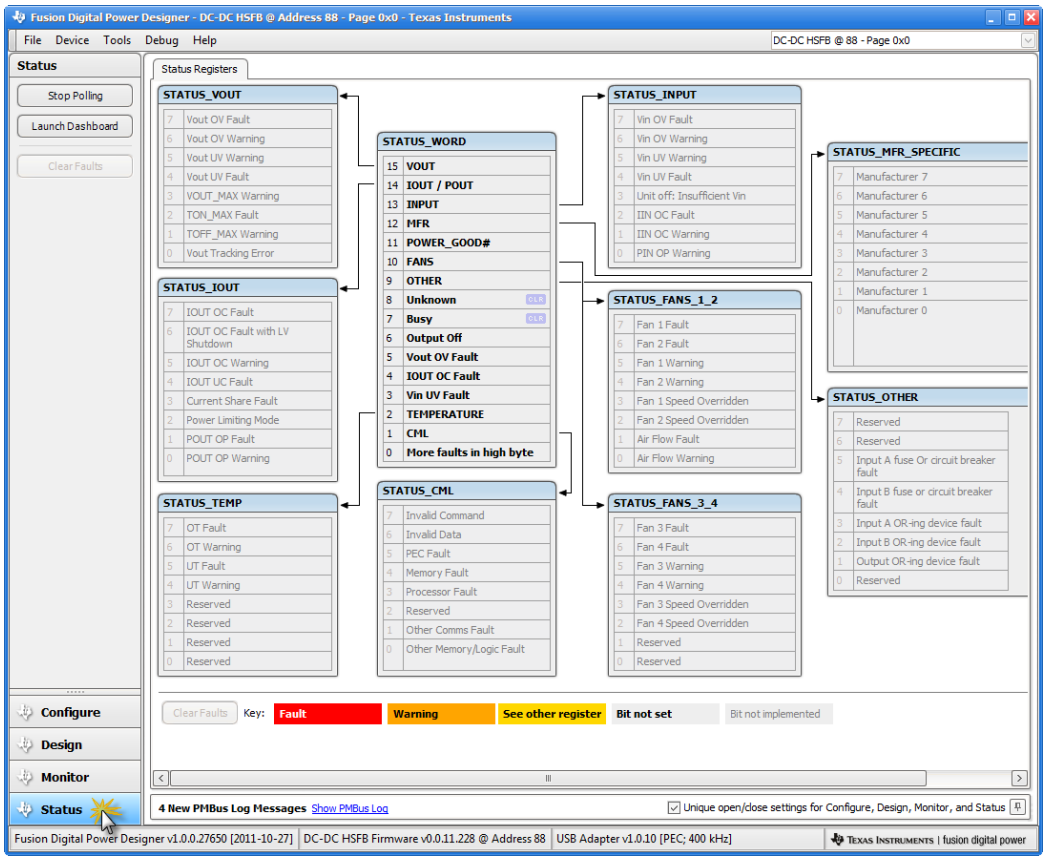


Figure 21 - Status Mode

6.7 Capture the State of the Device - Saving 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 commit 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 …

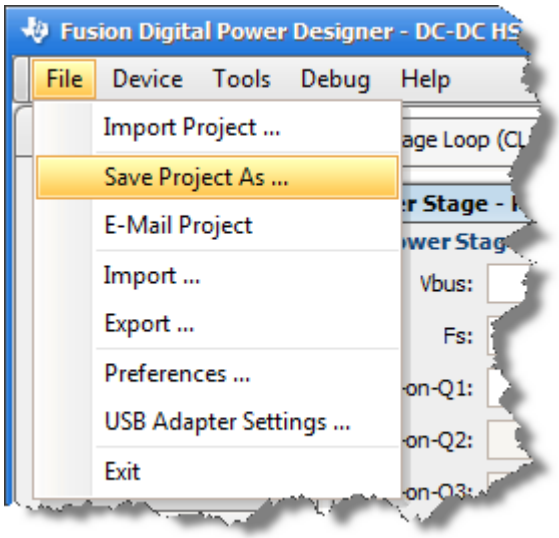


Figure 22 - Save Project File

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 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 modelling 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.

6.8.1 Starting in Offline mode

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

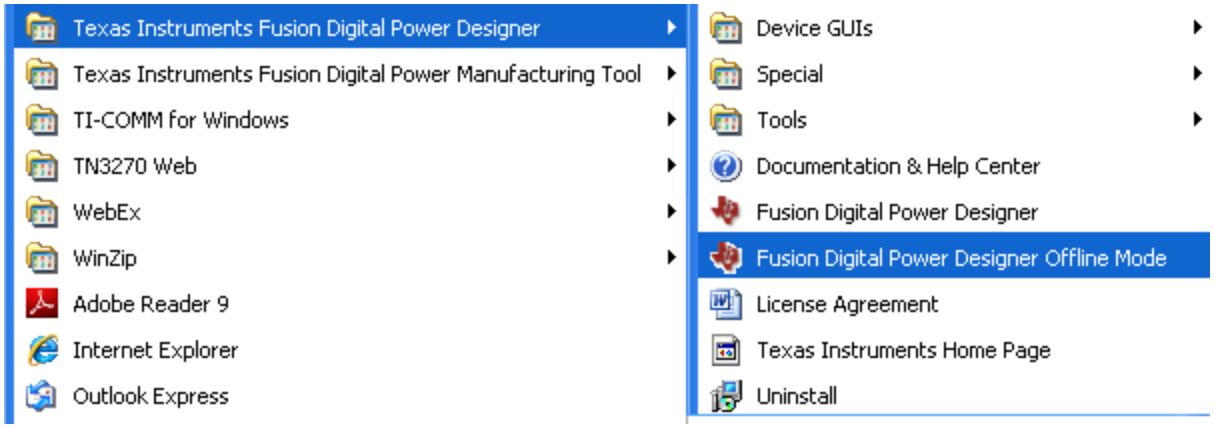


Figure 23 - Starting in offline mode

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.

6.8.2 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.

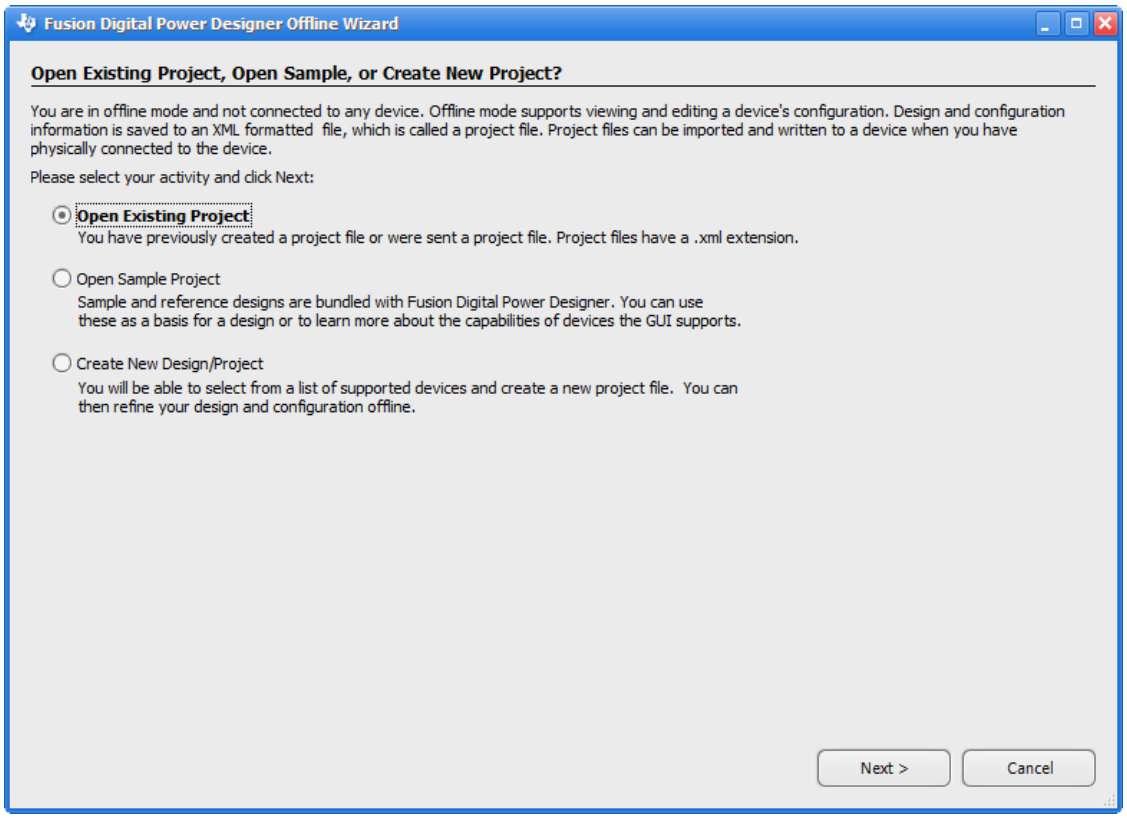
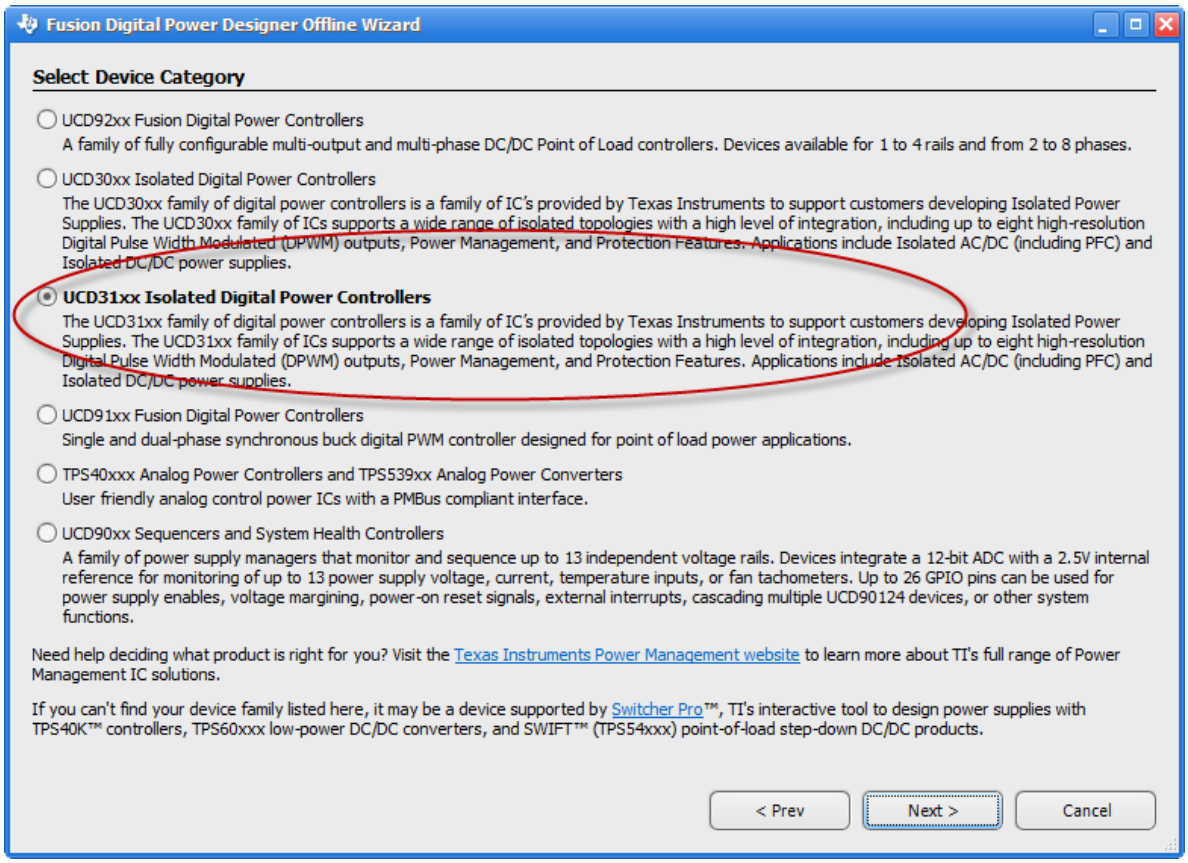


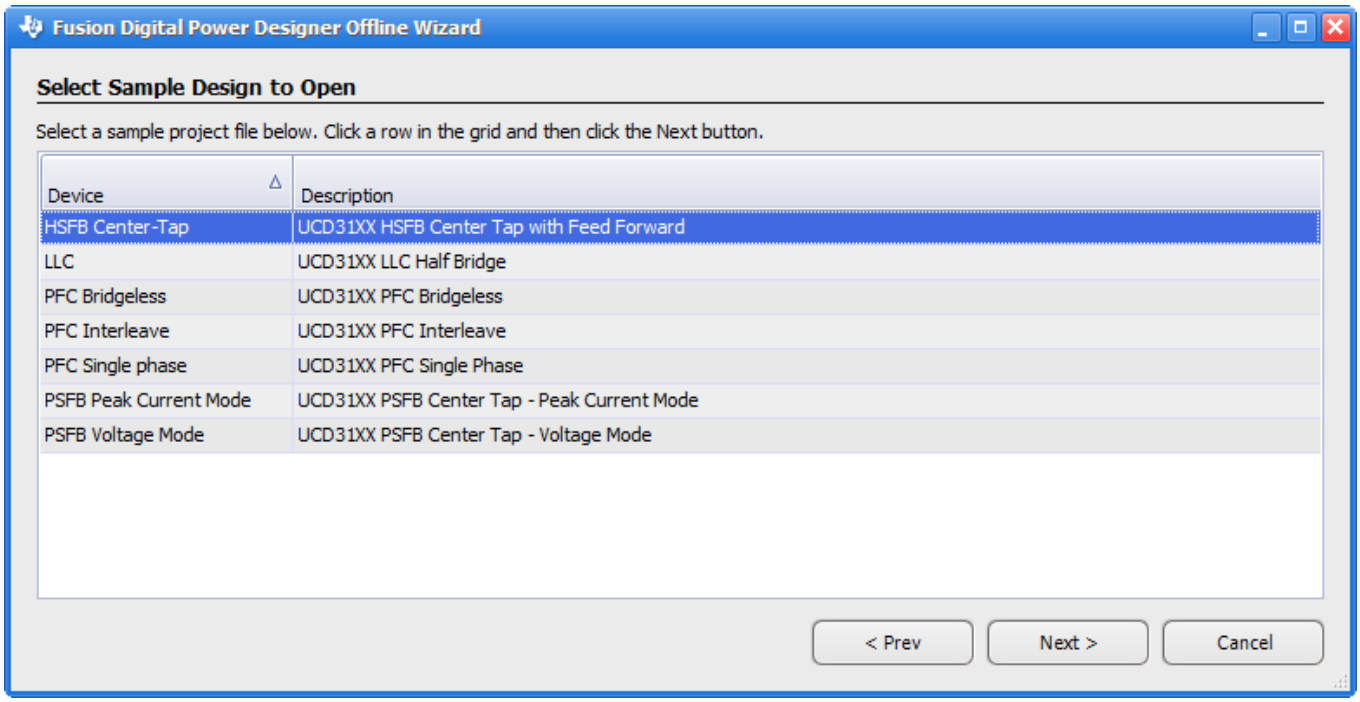
Figure 24 - Offline options

6.8.3 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 UCD31XX users should click “UCD31xx Isolated Digital Power Controllers” See figure below.



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



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 tools. 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 password.

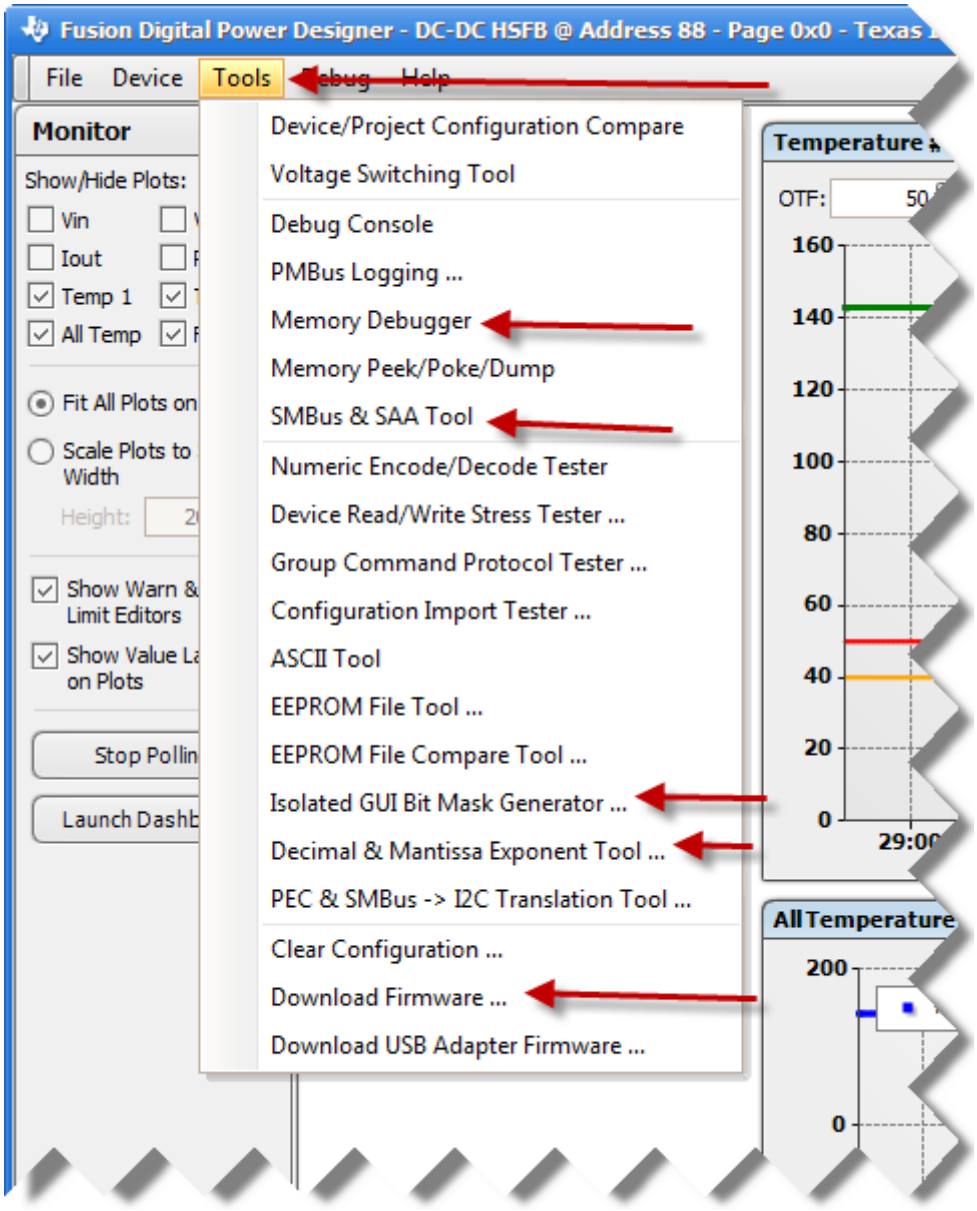


Figure 25 - Designer GUI Tools menu

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 also be accessed from the Start Menu.

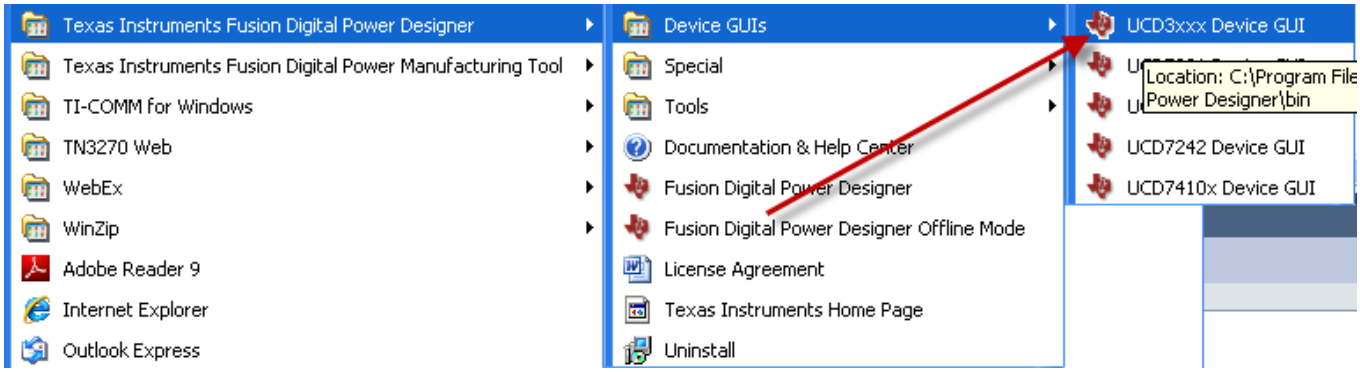


Figure 26 - Opening UCD3xxx Device GUI

The Device GUI looks as follows,

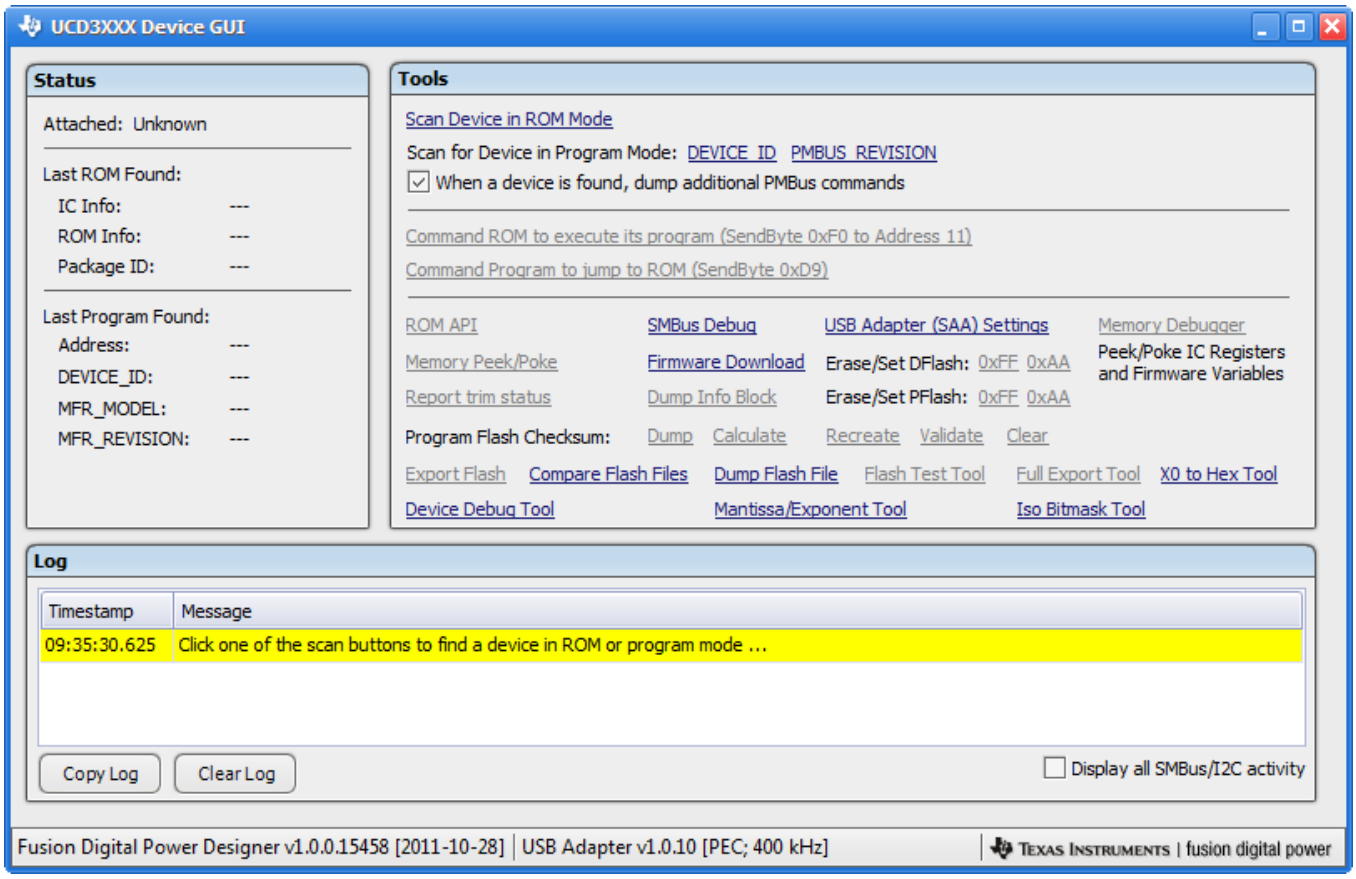
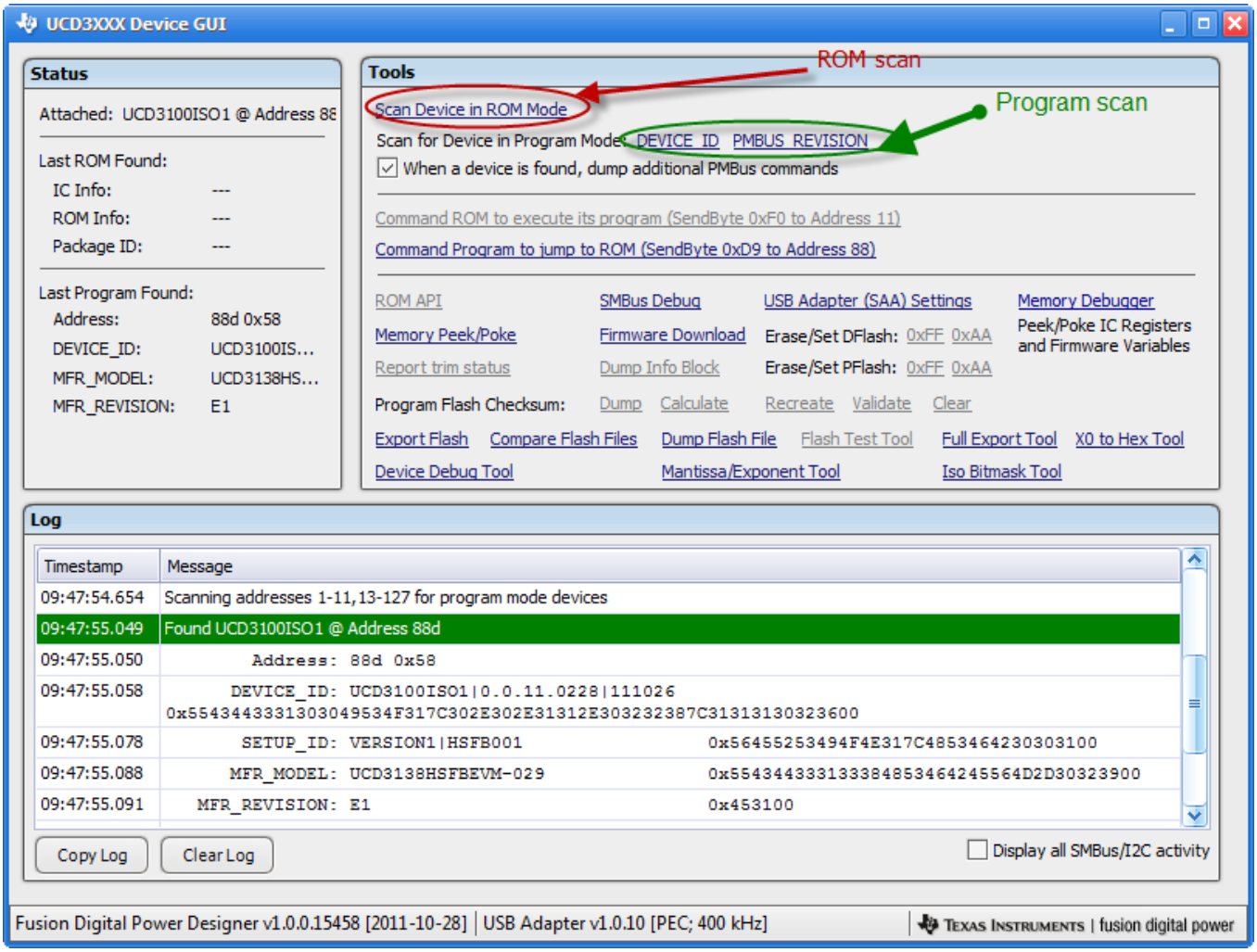


Figure 27 - UCD3xxx Device GUI

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 No ROM detected. If the device is in Program mode then the user should select "Device ID" or "PMBus REVISION".



7.2 Moving between ROM and Program mode

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

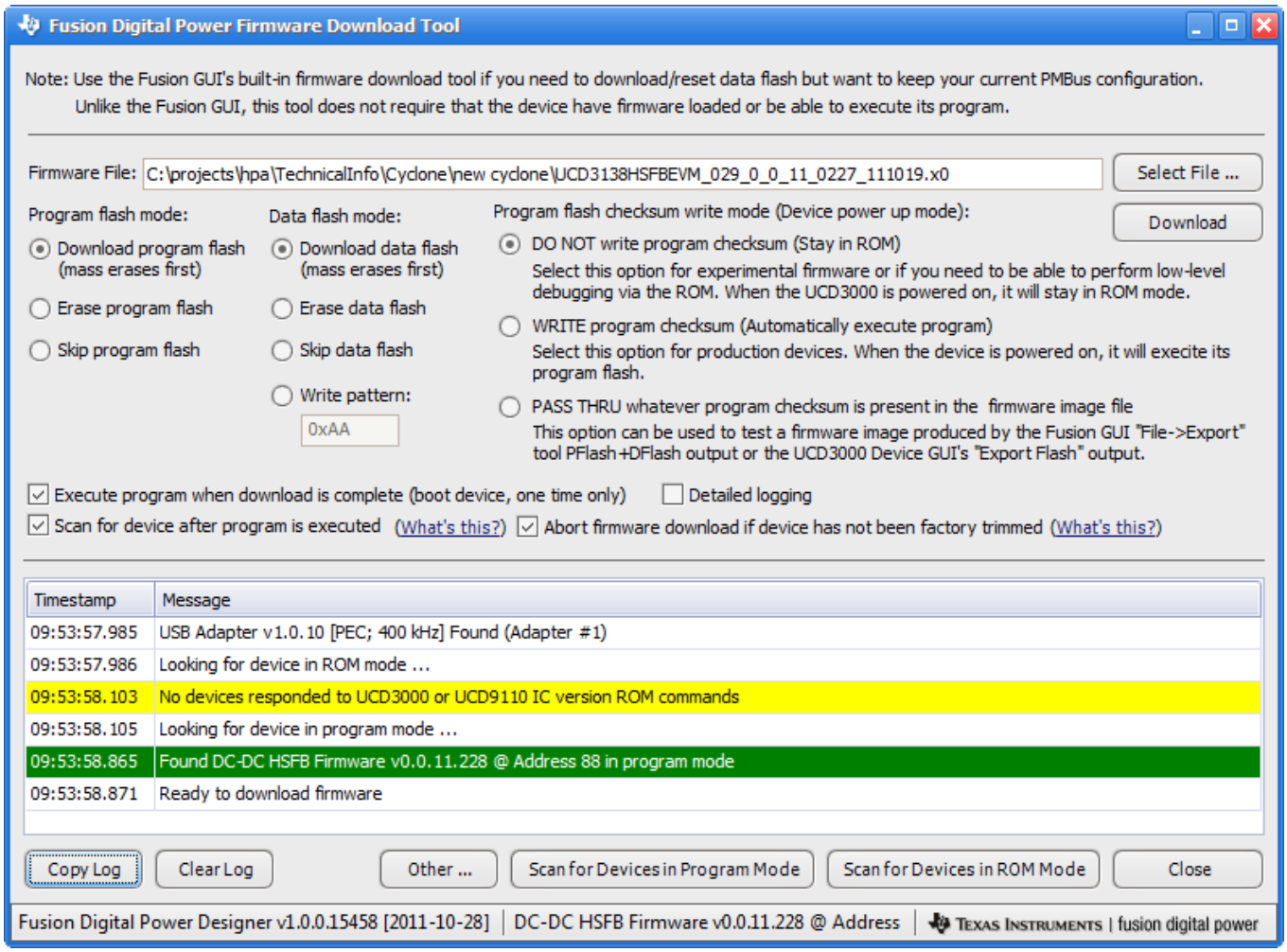
Figure 28 - Moving from ROM mode to Program mode

7.3 Firmware Download Tool

To open the Firmware Download tool click “Firmware Download”

The tool launched should look as follows:

Figure 29 - Scanning for device in ROM or Program mode



The user can choose what they would like to download with regards to the Program Flash, and Data Flash. Regarding the Program flash checksum (last column of options) it is important to note that if the program checksum is written then on a device reset, the Program mode will always be executed. This is a potential problem for firmware that hasn't reached production level and the commands to jump back to ROM have not been fully implemented. The problem that can occur is for the device to get locked in program mode. So it is preferred not to write the program checksum for firmware in initial stages.

The user picks the firmware file to download and clicks download. 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.

7.4 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 do Implemented Commands on the Firmware Appear in the GUI?"

GUI when first opened

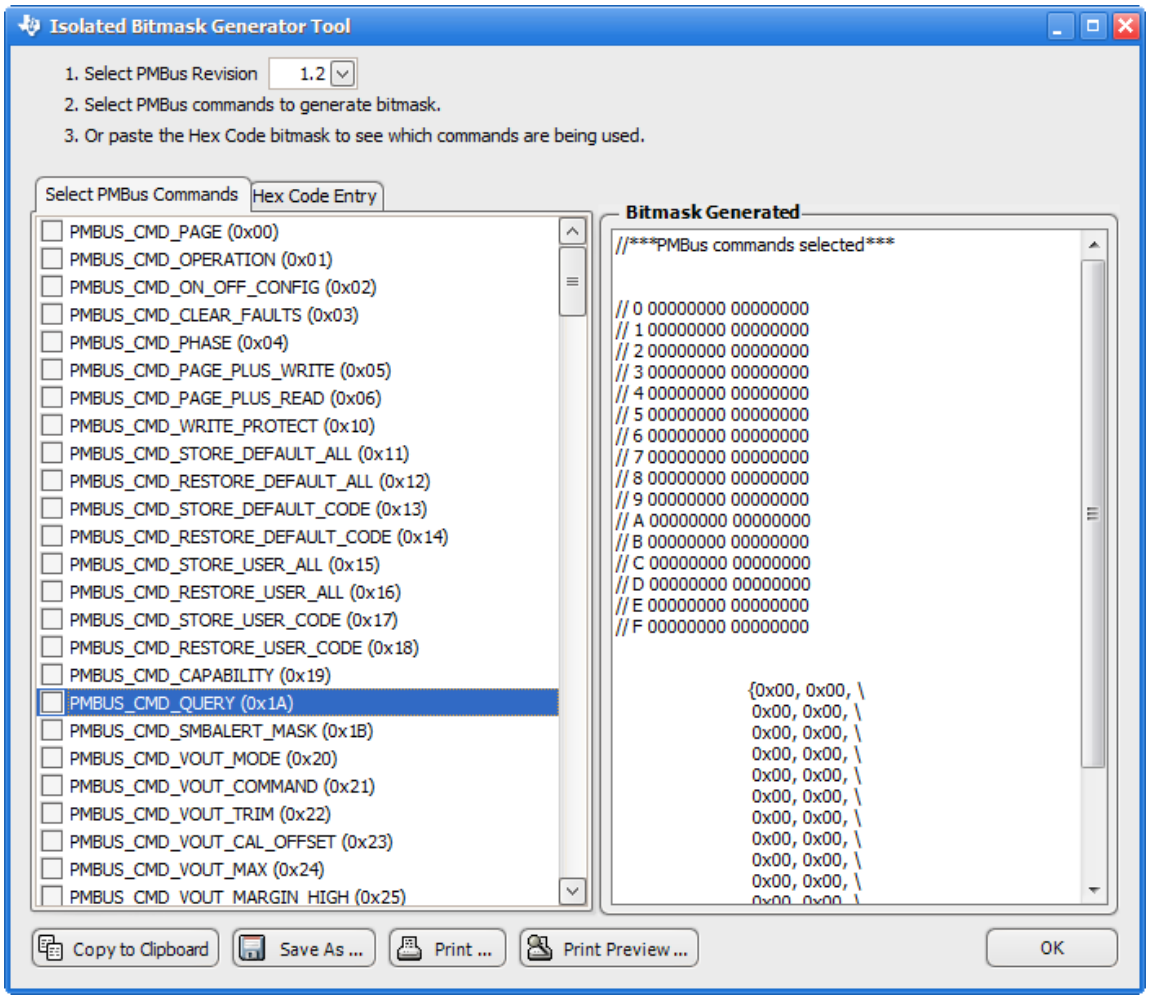
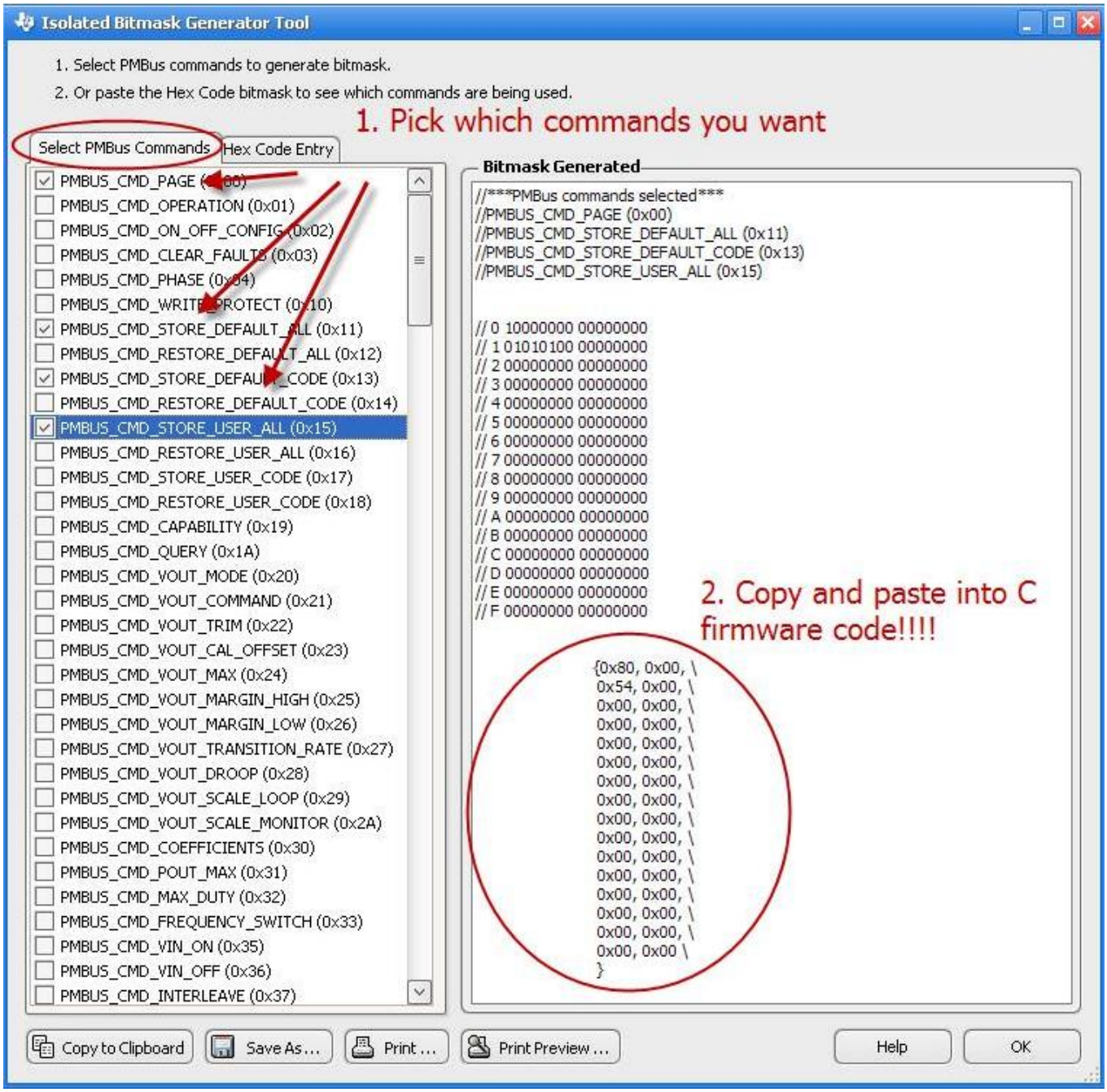
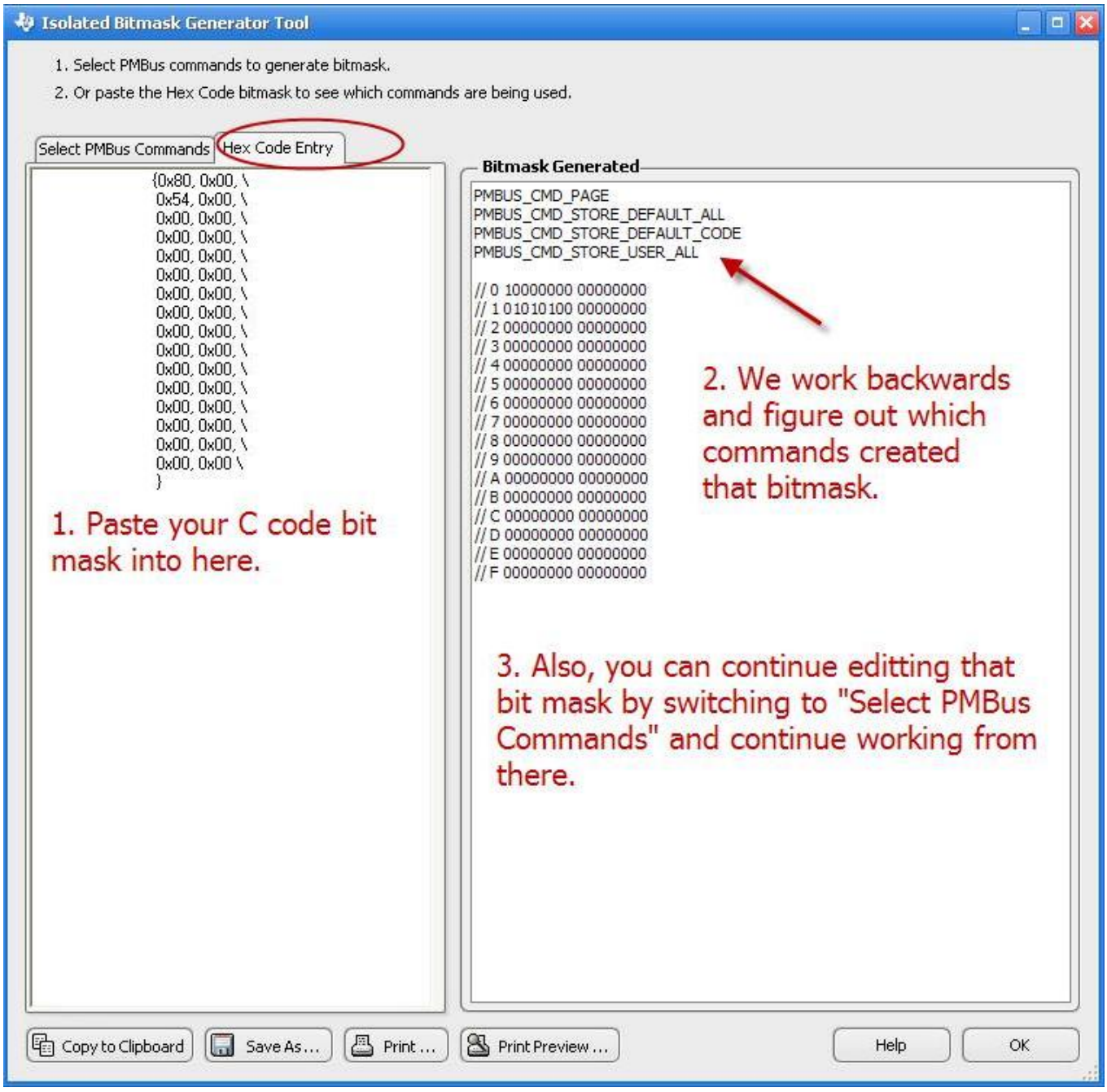


Figure 30 - Bitmask tool

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.



7.5 Firmware Memory Debugger

Included with the Fusion Digital Power Design software suite a powerful low level GUI is available for debug using the PMBus.

Figure 31 - Memory Debugger

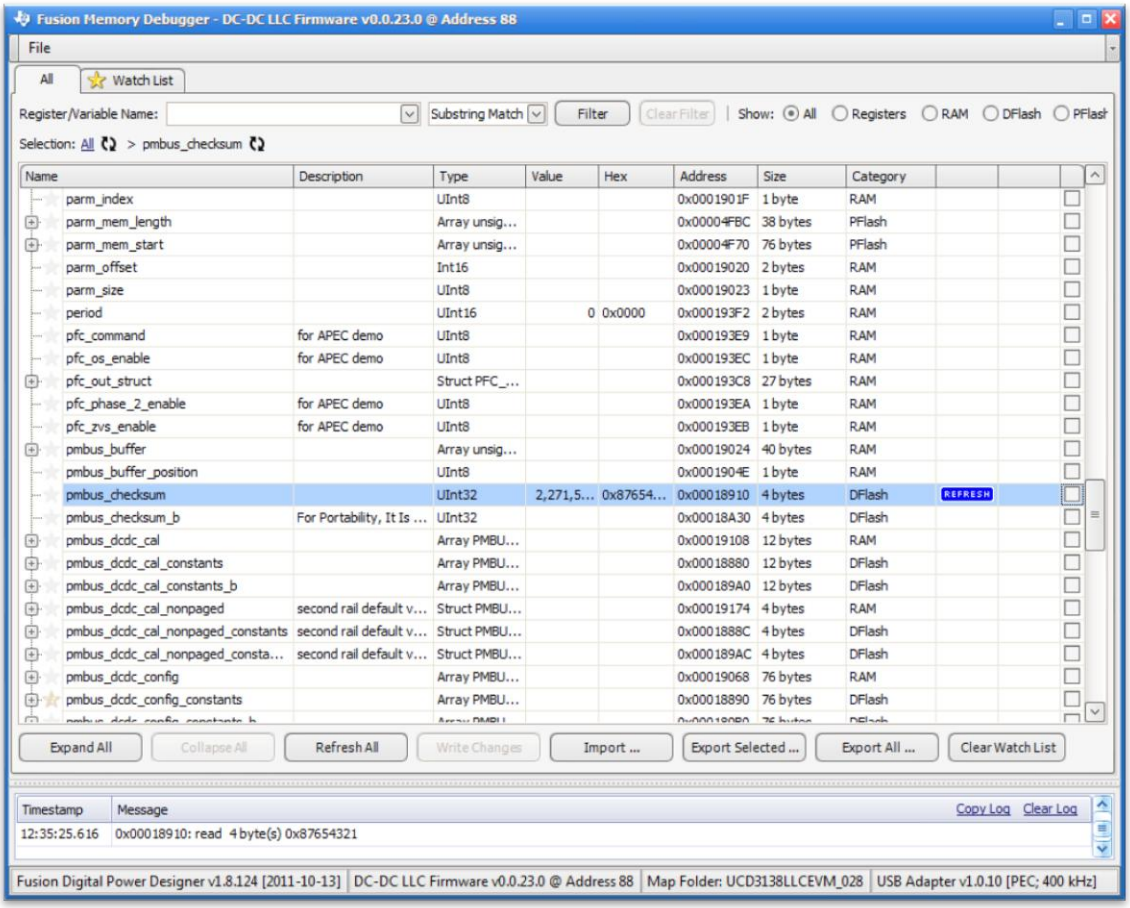


Figure 32: GUI Debugger

To also access the GUI through the Design GUI click the “Memory Debugger” item under tools, shown in Figure 33.

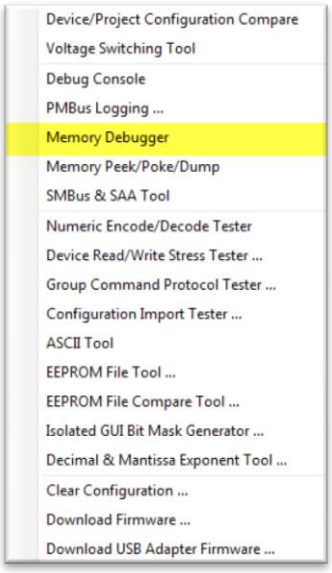


Figure 33: Fusion Designer GUI Debugger Tool

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 34: 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.

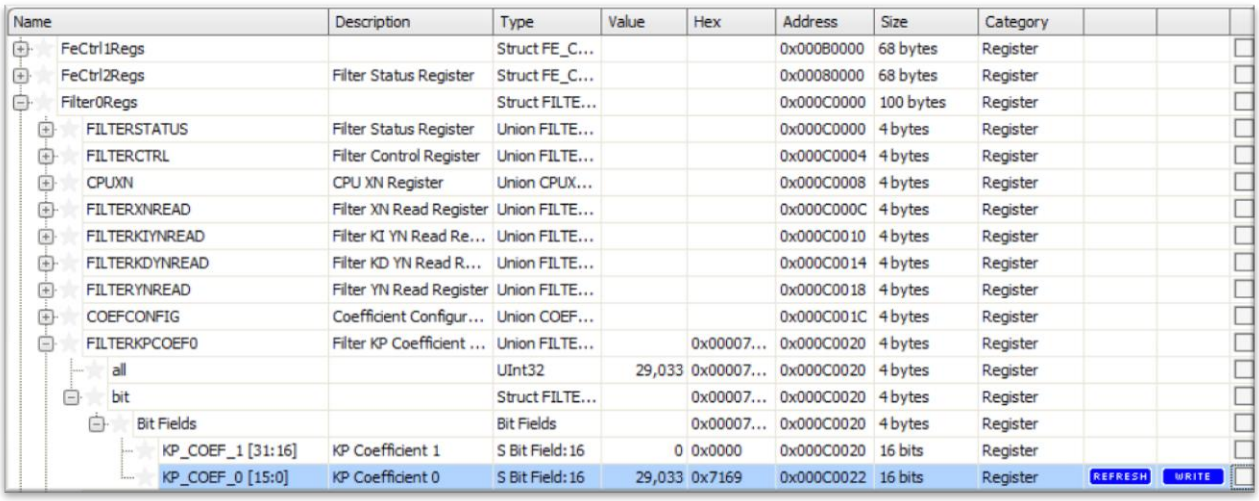


Figure 35: Device Debugger Bit Field Selector

Figure 35 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. If you click one of the stars next to a variable name it will turn gold indicating that it has been added to the watch list. To remove an item from the watch list, simply click the star again. If you click the “Watch List” tab at the top of the window you will now see the items you selected.

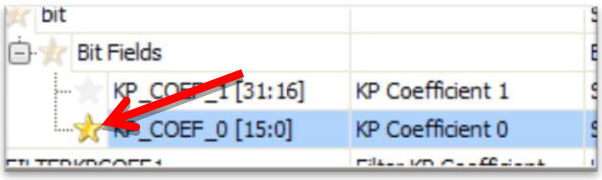


Figure 36: Watch List Selection Star

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 37.

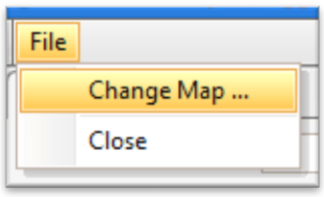


Figure 37: Map File Selection

After clicking this item, a window will pop up providing detailed instruction on what to do. For an example, see Figure 38.

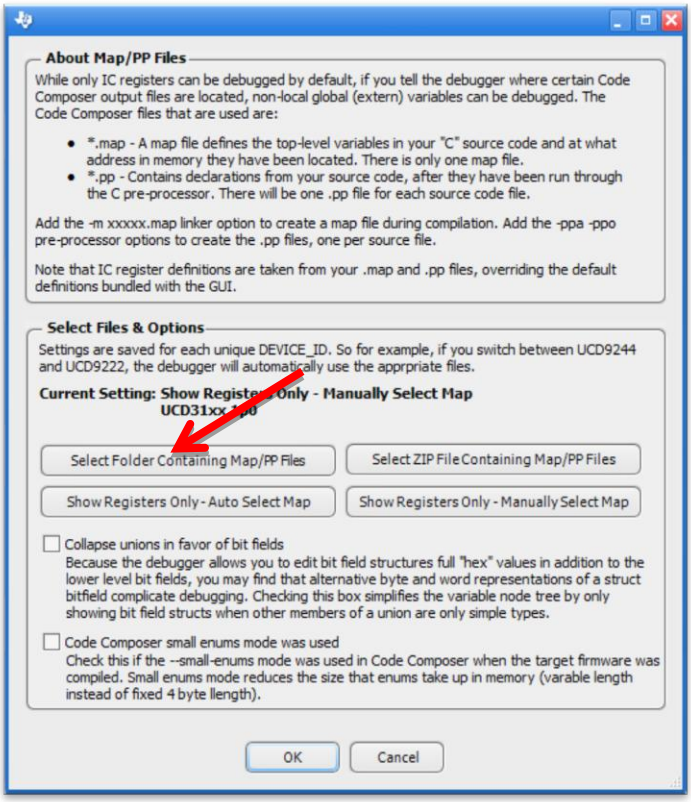


Figure 38: Debugger Customization Tool

The generation of the ".pp" files can be configured by modifying the Code Composer build options as shown in Figure 39.

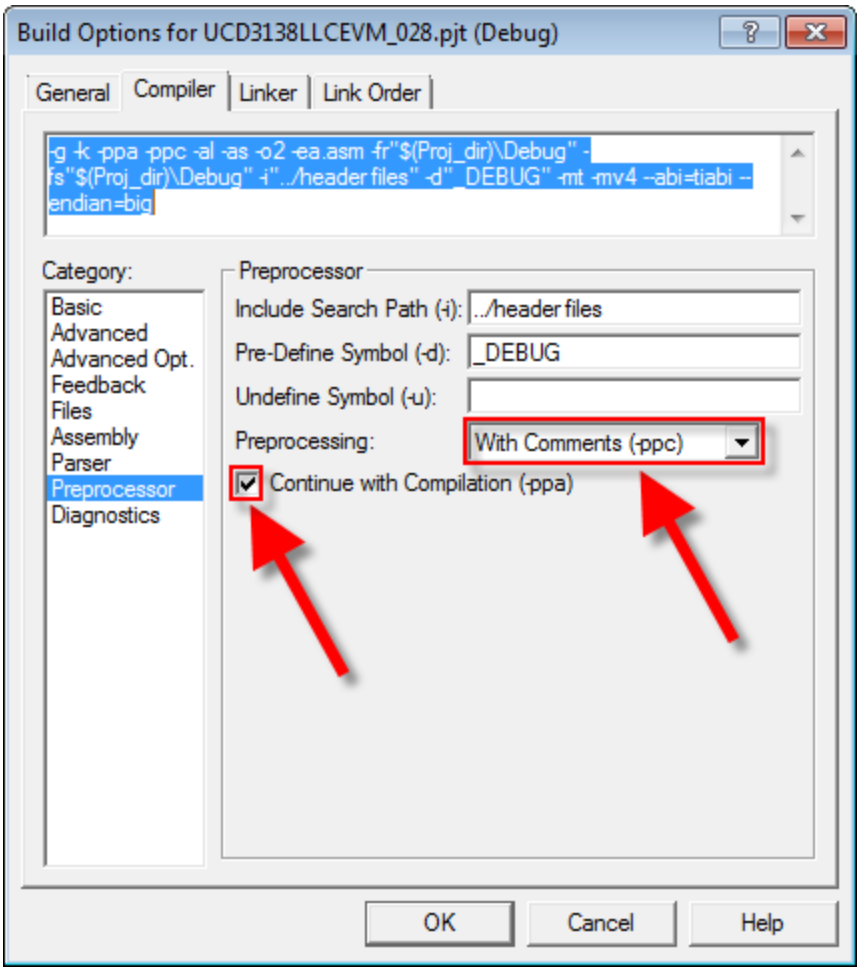


Figure 39: ".pp" Generation Parameters

The "*.map" file name and location can be specified in the code compose build options as shown in Figure 40.

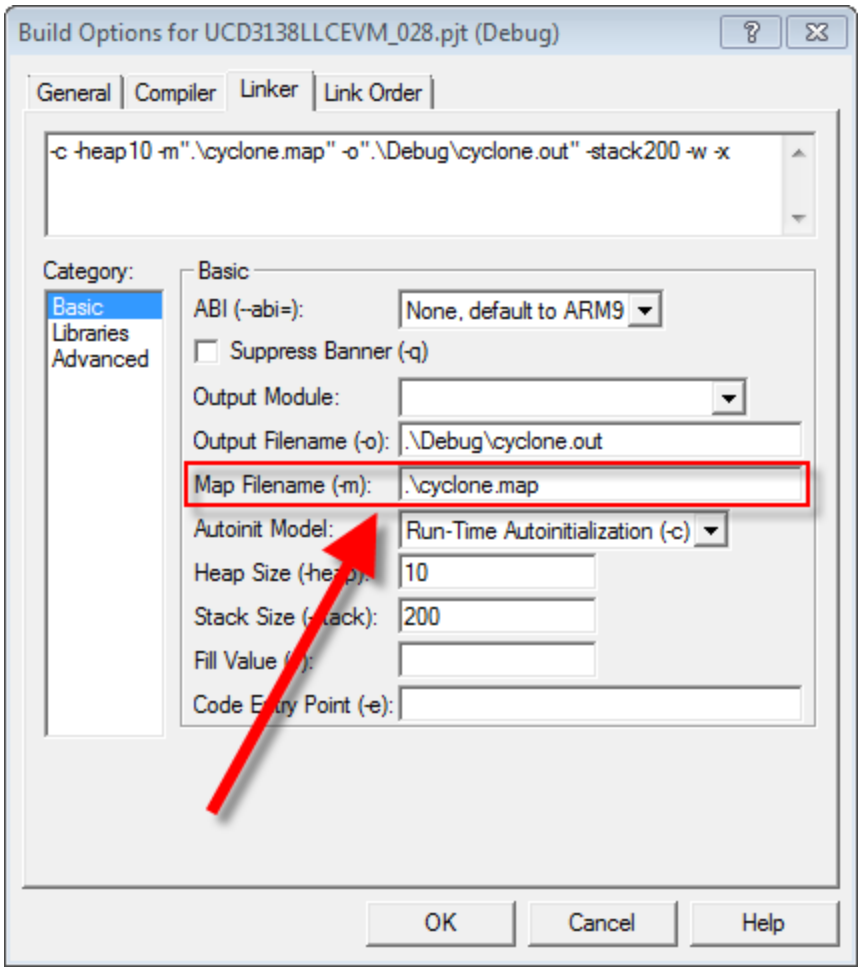


Figure 40: Map Filename

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 41 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.

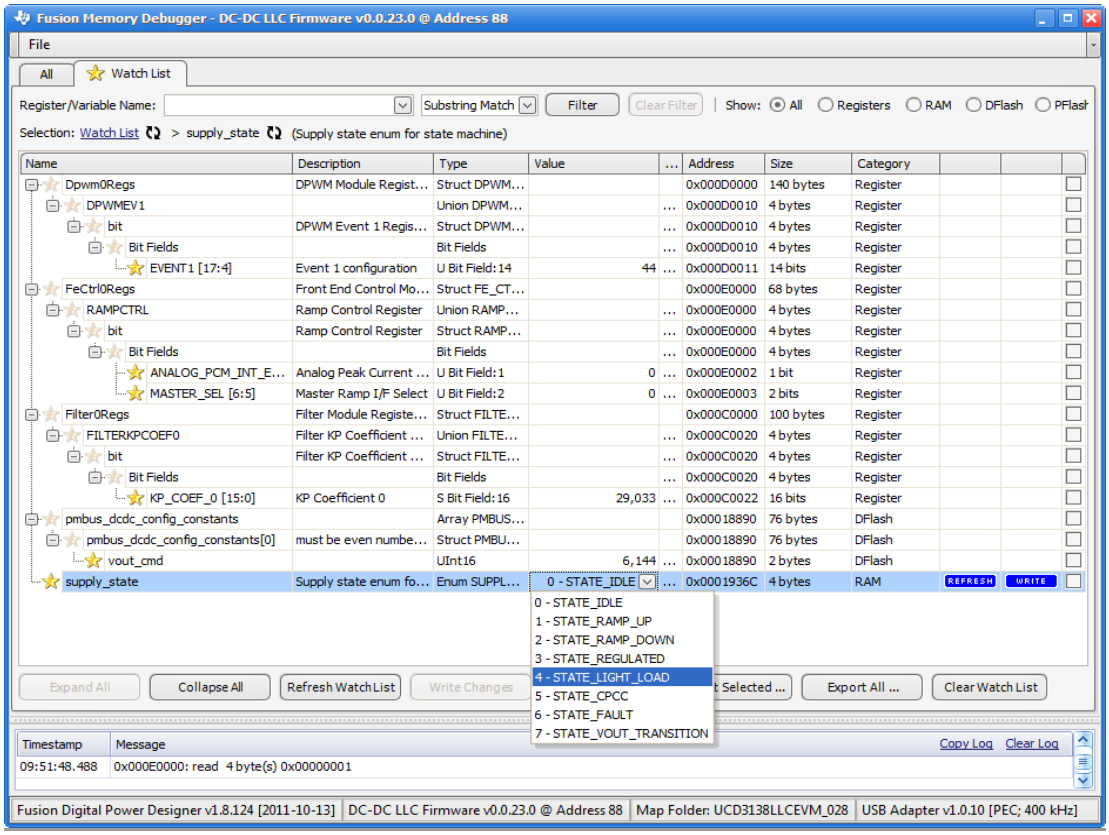
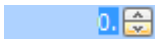


Figure 41: Watch List with Firmware Variables

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 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 current value is 2 and step is 5 and you click up, it will go to 7
- res: modulo oriented resolution. If current value is 2 and res is 5 and you click up, it will go to 5.

7.6 SMBus Debug

The tool looks as follows when launched,

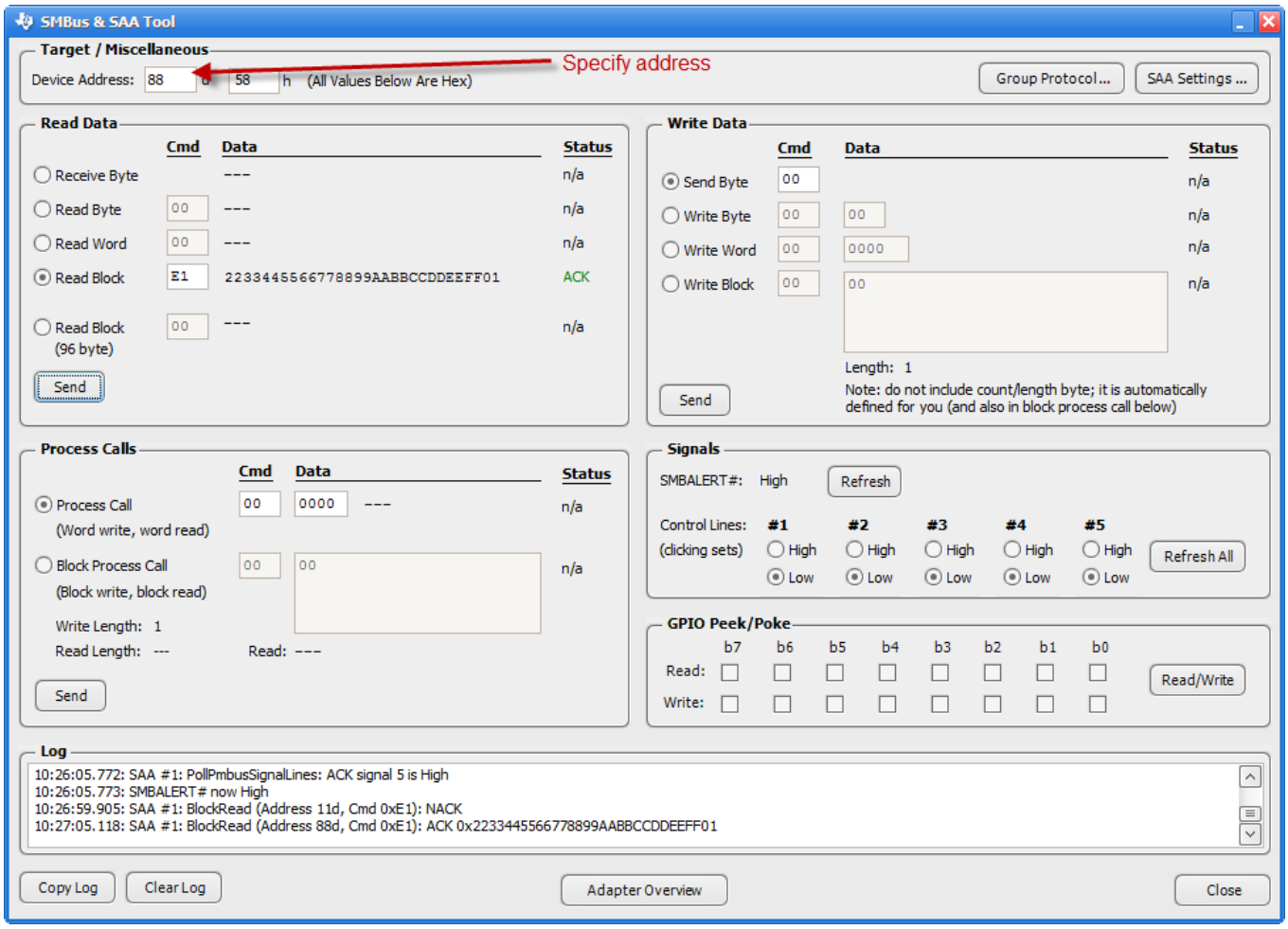


Figure 42 - SMBus Debug

In order to use this tool the user needs to specify the device address. This tool can be used to intereact 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.

8 API – Application Programming Interface

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. See your TI representative for more information regarding this.

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. Some of the functions included are importing a project file on to a device, write serial numbers and MFR date, calibrate devices using instrumentation (GPIB, SCPI, and USB) or manual measurements, test the device's output and various other functions. Users can also develop their own functions to include in the manufacturing scripts. See your TI representative for more information regarding this tool.

10 Documentation and References

10.1 References

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

[i] USB Interface Adapter EVM (HPA172)