Digital Power Insight™
User Manual

1 Table of Contents

2 Introduction

GE Critical Power products such as Digital Point-of-Load (POL) modules, Digital Bus Converters, Digital AC/DC and DC/DC Power Supplies support customer needs for increased control and communications access to DC/DC Power Modules. These devices offer comprehensive control and telemetry capability over a digital bus defined by the industry standard I^{2}C and SMBus transport interfaces. The communications interface to the modules is based on the industry standard PMBus™ protocol. PMBus™ is an open standard for communicating to both DC/DC power modules and other related power equipment such as AC/DC rectifiers. With this standard, devices from various manufacturers can be accessed, controlled and monitored using a common protocol.

Digital Power Insight™ (DPI) is a powerful interactive tool set that can be used to setup, configure, control and read back supported telemetry information from these new digital power modules.

The Digital Power Insight tool set is composed of three tools.

1. A powerful, windows-oriented graphical User Interface, referred to as the DPI ProGUI. Supporting communications up to 64 modules, the tool provides a rich set of functions:
    - Finding all modules on the bus and sequentially presenting by product type and address
    - Supporting product-specific display and changes for module parameters
    - Set up polling of module variables, monitor and display in a table measurements from all modules
    - Flexible plotting of individual or groups of module measurements
    - Saving of monitored data in a user-specified file for additional processing (such as for example through Microsoft Excel)
    - Clearing the status registers of selected modules
    - Monitoring of SMBusAlert# and Responding to the Alert_Responder_Address request
    - Setting up and saving configuration files for groups of modules for copy and replicate capability
2. A simple, graphically-oriented User Interface, referred to here as the DPI GUI. This tool is easy to learn and designed for users who do not want to do the detailed level of programming needed to communicate with one or more modules. It is ideal for the Power Design Engineer who wants to access digital functionality in a limited number of power modules without getting into the details of PMBus commands and programming. Finally, the GUI enables a user to communicate with a board containing multiple modules (up to a total of six) by supporting various functions such as loading commands into the power modules, "on-the-fly" adjustment of module parameters and functions, board power data access and real-time display of module configuration and measured data.
3. The third tool is a low level command line interface tool called the DPI Command Line Interface or DPI-CLI. The DPI-CLI provides a host of capabilities ranging from invoking simple PMBus commands to scripting complex test programs with multiple PMBus commands that can be used to control and acquire data from power modules. By providing a lower level interface to multiple power modules, this tool delivers comprehensive support of programming activities as well as testing/debugging capability for users who are developing software to control the modules or want to examine specific commands or controller-module interactions in detail. The DPI-CLI also provides a means to query and change settings of the USB Interface Adapter.

This User Manual starts with detailed information on installing the configuration software that places all drivers and executable programs into a directory structure consistent with the Windows environment. The manual then provides detailed descriptions of all three DPI tools.

3 Setup

The following components are required to set up a working system

- A desktop or laptop Personal Computer running Microsoft Windows XP, Vista or the Windows 7 operating systems
- An open USB interface slot on the computer
- The USB Interface Adapter provided by GE
- A USB to Mini-USB cable to connect the USB Interface Adapter to the computer
- A 10-conductor ribbon cable with connectors at each end to connect from the USB Interface Adapter to either an evaluation board or a user PWB.
- An evaluation board or customer board containing one or more GE Digital Modules. If a customer evaluation board is used it must have a 10-pin connector that can mate with the 10-conductor ribbon cable
- The DPI Tool Set software provided as an installation file.

The figure below shows an example setup using two evaluation boards along with a Laptop Computer, the USB Interface Adapter and the interconnection cables.

Note: DPI only works with the GE USB Interface Adapter.
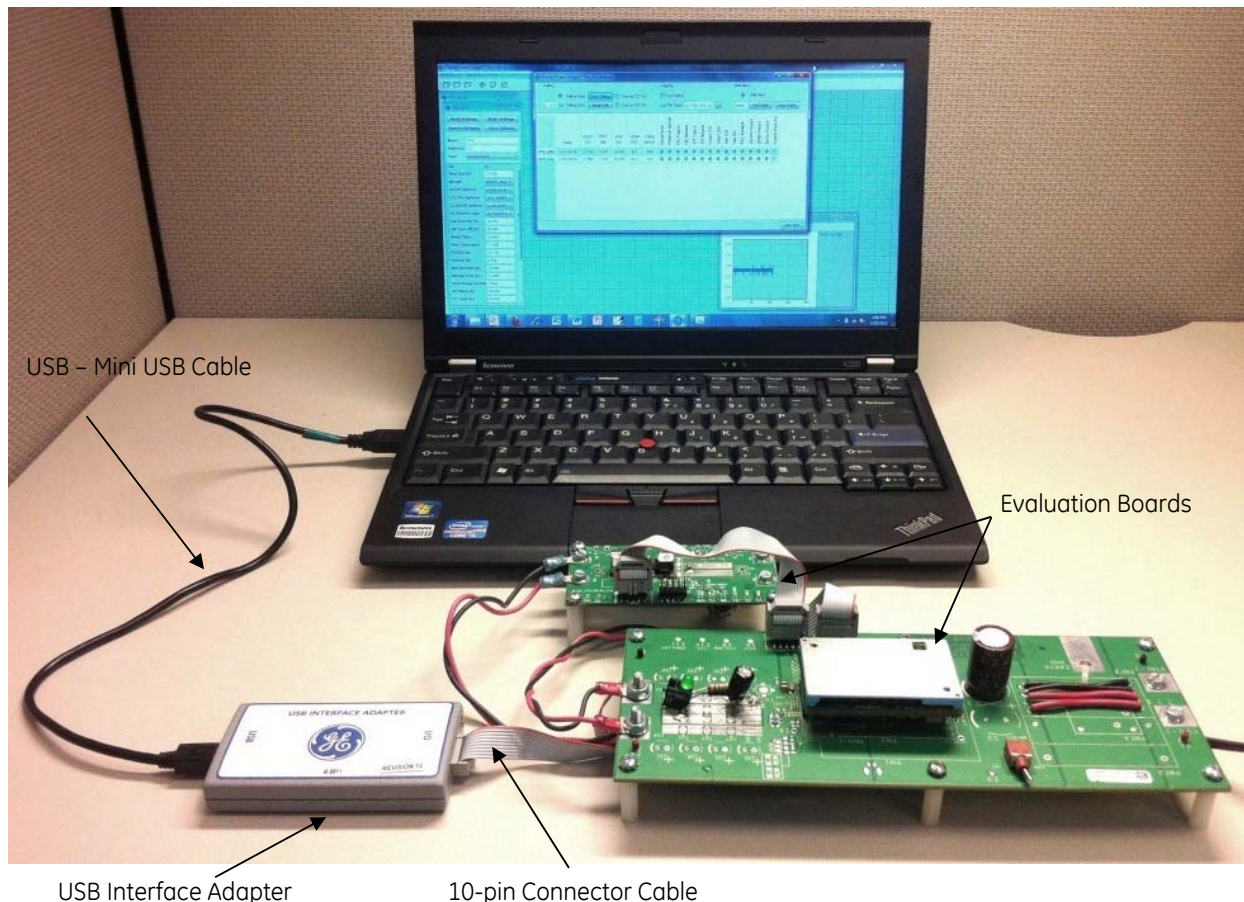
3.1   USB Interface Adapter

This adapter is powered off the USB port on the computer and translates commands from the Personal Computer delivered through the USB port to I2C/SMBus commands and also translates data received over I2C/SMBus back to

the PC. This adapter is designed specifically to work with GE products. Check with the GE Technical Support team for other uses.

3.1.1   Pull-up Resistors

The translator has internal pull-up resistors connected to 3V that source a default value of 3.3mA for the clock and data lines. Other possible values are 0.9mA, or 0.44mA. The SMBAlert# signal is pulled up to 3V via a 7.5kΩ resistor. See 4.2.12 for details on how the pull-up resistors can be changed.

3.1.2   LED Status Indicator

The USB Interface Adapter has a three-color (Green/Orange/Red) LED on one end where the USB interface cable plugs into the adapter. The status of the adapter is displayed as follows:

- Green ON – Normal operation
- Green – Fast Blinking – The translator is communicating on the bus.
- Orange – Blinking or ON – The SMBAlert# line is active. An SMBusASlert# LO state requests service from the system controller because an event change occurred in one of the modules connected to the system. The Alert# signal will remain LO until it gets cleared.
- Red – Internal fault

3.1.3   I²C Clock Speed Settings

The default value of the I²C clock speed is set at 100kHz. It can also be programmed to 400kHz. See 4.2.7 for details on how the clock speed can be changed.

3.2   Installing the DPI Software

The DPI software is installed using a self-executable install file. This install file is downloadable from the **GE – Critical Power** website. To install the software double click on the file setup_dpi.exe. This results in the screen shown below. Follow the installation instructions to install the software.

Note:

The installation locates the DPI Program files in the directory: C:\Program Files\GE Energy\DPI

Since some of the DPI-GUI programs also create log files every time the program is executed in the same directory, the user may choose to have the DPI Program files installed in an alternate directory that can be specified during the installation.

The DPI Software installation automatically places icons for the DPI_ProGUI, DPI-GUI and DPI-CLI programs on the Windows Desktop, in addition to entries in the **Windows – Start - Programs** menu.

After the software installation, a driver file also needs to be installed to allow the computer to communicate to the GE USB Interface Adapter. See 2.2.1 for instructions on installing the driver file.
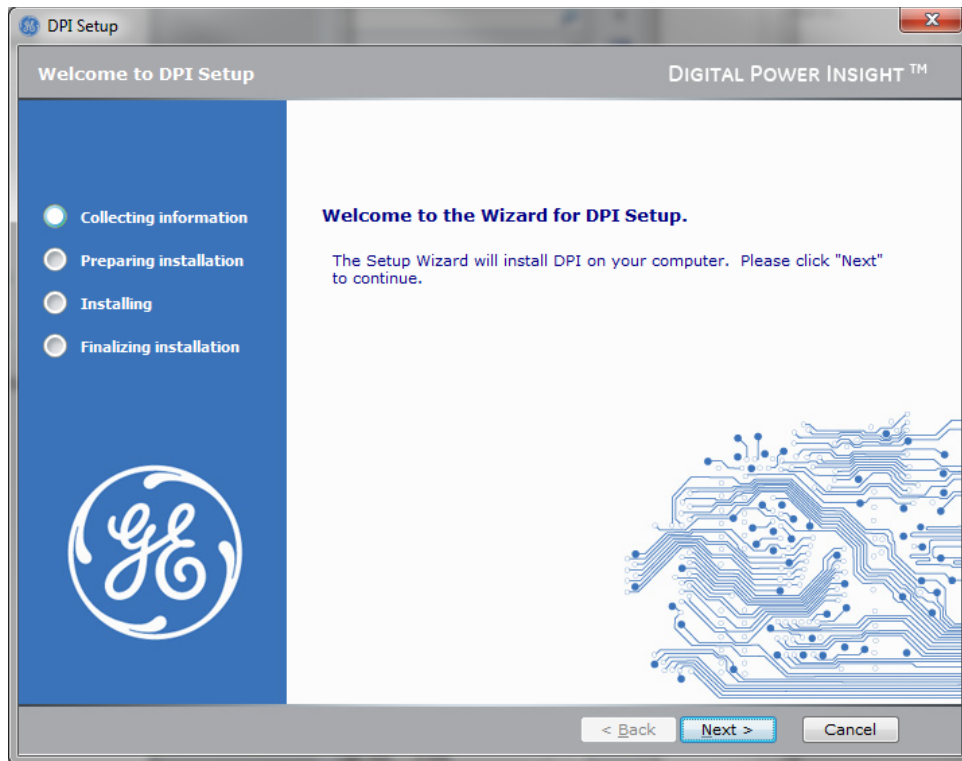
3.2.1   USB Interface Adapter driver setup

The files

GE_Power_Electronics_USB_Interface_Adapter.inf and
GE_Power_Electronics_USB_Interface_Adapter_64bit.inf

contain the Windows driver information, for Windows 32-bit and 64-bit operating systems, respectively. During the DPI software installation, the file

GE_Power_Electronics_USB_Interface_Adapter.inf

Is automatically placed in the C:\Windows\System32 directory.

The file

GE_Power_Electronics_USB_Interface_Adapter_64bit.inf

is placed in the C:\Windows\SysWOW64 directory. Please select the appropriate driver file based on your PC operating system.

Plug the USB Interface Adapter into an open USB port on your computer. The computer should recognize that new hardware has been connected and it will ask you for a location for the driver for this new hardware.

In the screen shown above, select **No, not this time**, and then click on the **Next>** box. This will bring up the screen shown below:
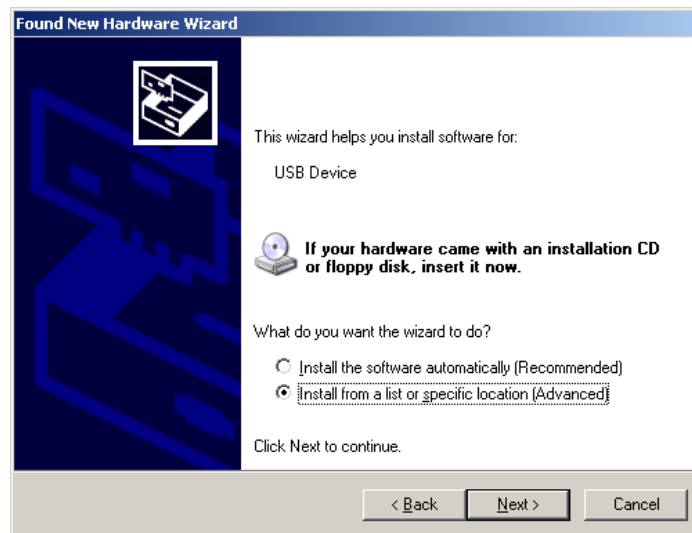
Select **Install from a list or specific location (Advanced)** and click on **Next>** to proceed to the next screen.

The resulting screen is shown below.
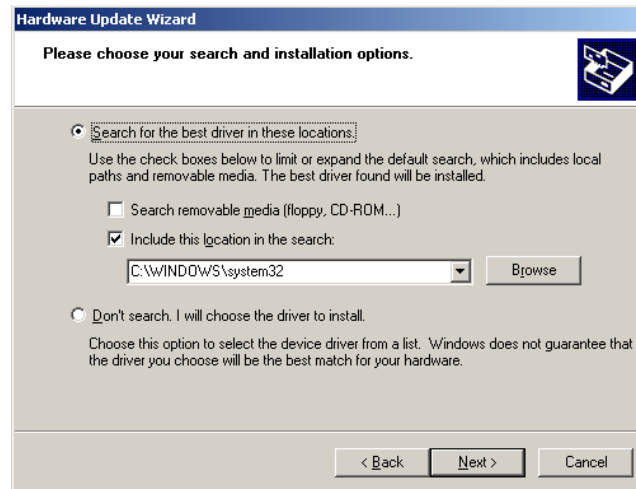
Select **Search for the best driver in these locations,** followed by **Include this location in the search** and enter C:\Windows\System32 or C:\Windows\SysWOW64 as appropriate for your PC Operating System.

Click on the **Next>** button and the computer should now install the driver for the USB Interface Adapter. Wait until the driver installation completes before proceeding to the next step.

3.2.2   Communication Port Setup

In order to improve communication speed, settings of the COM port that is used by the USB Interface Adapter should be changed. This step is not necessary, but may result in faster communication speeds. In Microsoft Windows, from the **Start** button, select **Settings** and then **Control Panel**. Within the **Control Panel** window, double click on **System,** select the **Hardware** tab on the **System Properties** window, and then select **Device Manager**. Under **Device Manager** navigate down to **Ports** and double click on the **USB Interface Adapter** icon. The port properties should be displayed. Select the **Port Settings** tab and change the transmission speed to *115200* and flow control to *Xon/Xoff* by selecting and clicking on the drop down menu. The screen shot for this step is shown below.
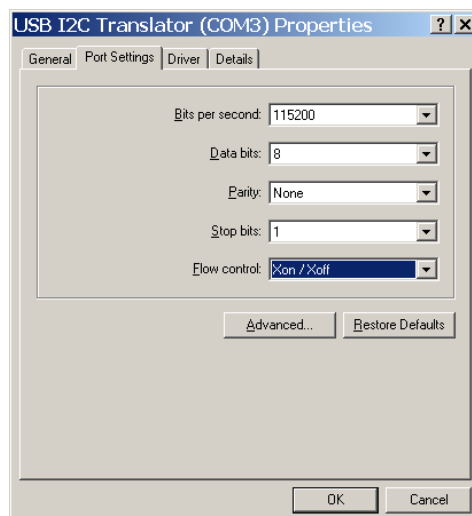
Once these steps are completed, the DPI Tool Set is ready to use.

3.3   Files Created by DPI

The GUI and CLI tools within DPI create files that keep logs of commands, states and readings. These log files are created in the directory where the DPI software is installed (typically C:\Program Files\GE Energy\DPI).

Filenames are automatically created by the tools. Those created by the GUI have the filename

 "dpi-gui-YYYYMMDD-HHMM.txt"

where "YYYYMMDD" refers to the year, month and day when the file was created and "HHMM" refers to the time. These files can be safely deleted if not needed for additional examination.

The configuration of which data is saved by the GUI file cannot be changed.

The log file created by the CLI have the filename

"dpi-cli-YYYYMMDD-HHMM.txt"

with "YY YYMMDD" referring to the year, month and day when the file was created and "HHMM" referring to the time.

The contents of the log file created by the CLI tool are user configurable as outlined in the <SUPPRESS> command description in Section 5.2.18. The user therefore has a choice between keeping the 'longer' style or 'abbreviated' style of records. The <SUPPRESS> command can be invoked at any time within CLI. Commands executed after the <suppress> command are recorded in the style directed by the command.

3.4   Products Supported by Executable Files

Executable	Product support
dpi_cli.exe	All codes, except Power Entry Modules: PIM400
pro_gui.exe	POL, BUS converters, CP3000/ CP3500 rectifier; Dlynx, Dual-Dlynx, SlimLynx POLs and Digital Barracuda
dpi_gui.exe	Simple GUI supporting POL and BUS converters; Dlynx, Dual-Dlynx, SlimLynx POLs and Digital Barracuda
cpgui_l.exe	Simple GUI supporting the second generation of the CP product line; CP1800AC, CP2000AC, CP2000DC, CP2500DC, CP2725AC – all version
dpi_car.exe	Simple GUI supporting PMBus compliant versions of the CAR platform; CAR0812, CAR1212, CAR2012, CAR2024, CAR2512, CAR2548

4 DPI-ProGUI

The DPI-ProGUI is a powerful, graphically-oriented tool for interfacing to up to 64 modules. Tool capabilities include

- A flexible, multi-window display setup that supports interfacing with multiple modules
- The ability to automatically find all modules connected to the PMBus, retrieve all module information along with the address and display the summary module parameters in multiple tiled windows
- Using product-specific windows to read, display, change and write module parameters
- Set up polling of module variables at a specified rate, and monitor and display in a table measurements from all modules
- Flexible plotting in one or more windows of individual or groups of module measurements
- Saving of monitored data in a delimited text format in a user-specified file for additional processing (such as for example through Microsoft Excel)
- Clearing the fault status of one or more selected modules
- Monitoring of SMBusAlert# and Responding to the Alert_Responder_Address request
- Setting up and saving configuration files for groups of modules so that copy and replicate capability can be easily achieved

4.1   Starting the DPI-ProGUI Tool

Before starting the ProGUI tool, make sure that the hardware connections (computer to USB-I^{2}C adapter, adapter to modules on evaluation boards or system boards) are all correct and that power is provided to the modules. You cannot communicate with the modules unless input power to them is present.

Start the ProGUI tool by either double-clicking the ProGUI icon placed on the computer desktop during installation or by clicking on **Windows Start** – **All Programs** – **DPI Suite** – **DPI-ProGUI.** The tool starts up and displays the screen shown below (Note: it may take a few seconds for the program to start up and the screen to appear – this is normal):
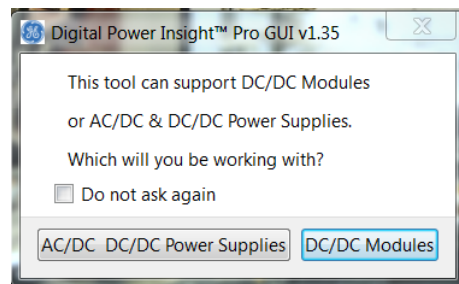
This screen asks the user to select the product family that is going to be connected so the tool can configure the appropriate set of features.. If the **Do not ask again** box is checked, the Pro_GUI tool will start with settings configured in the previous session. The user can reconfigure the product family setting as shown below.

After the product selection is made the tool starts by displaying the basic control window
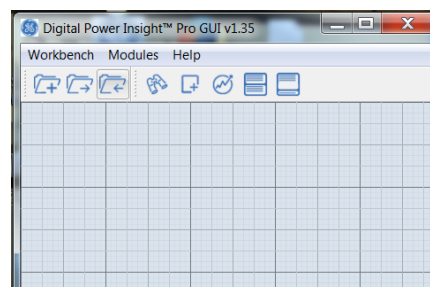
The product selection can be changed in the control window by clicking on **Workbench**, and in the trailing pop-up window click on **Change Type.** The product selection window then appears and the selection can be changed by the user. The user also has the option to either select or de-select the **do not ask again** feature. Once the selection is made the tool starts again by displaying the basic control window.
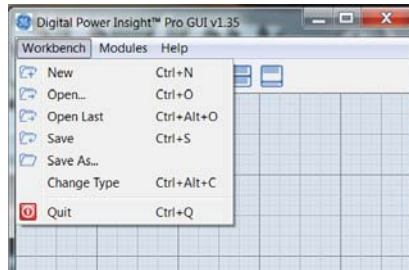
Note: Multiple DPI tools cannot be run at the same time, i.e. you can run one instance of the DPI-ProGUI, DPI_GUI or DPI_CLI at any one time. This is because they share the same COM port to connect to the USB adapter. If you have another program using the same COM port, the warning screen shown below will appear. You will need to close the other application before restarting DPI-ProGUI.
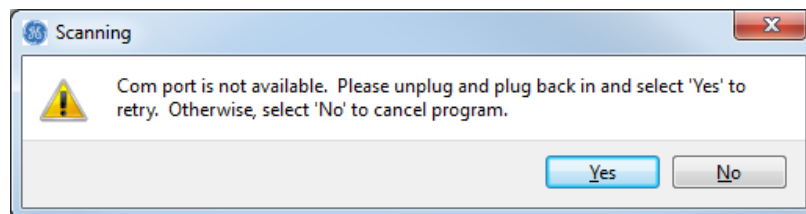
If the DPI-ProGUI program is started without the GE USB Interface Adapter plugged into a USB port on the computer, the program displays the following error message.
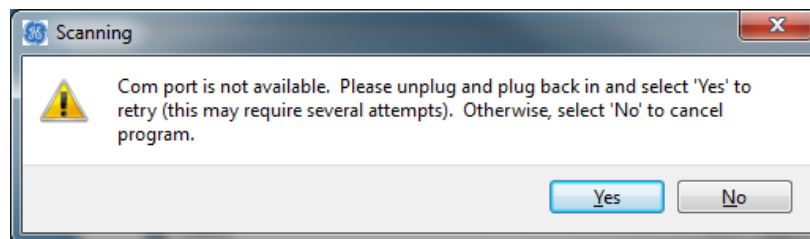
4.2   Main Windows of the DPI-ProGUI Tool

The DPI-ProGUI tool uses multiple windows to display and control multiple functions and capabilities. The screenshot below shows the various windows used in the DPI-ProGUI tool when DC/DC Converters are used.
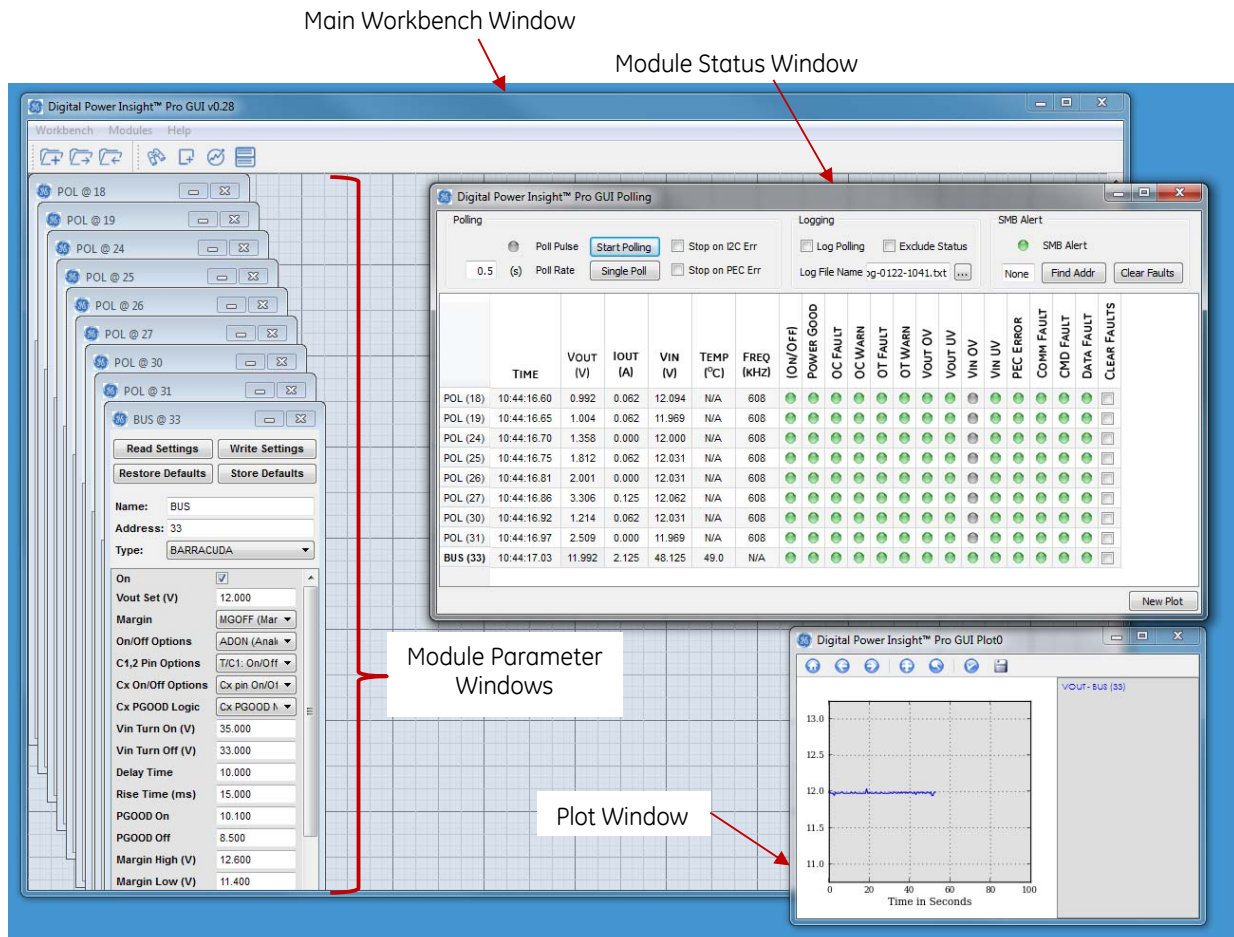
The Workbench window main menu commands and icons are located on the top left portion of the window. Within this window, one or more Module Parameter Windows are located – these allow module parameters to be read from and written to the module as well as entered by the user into the tool.

The Module Status window displays various module variables such as output voltage and current, input voltage, status values etc. Although this Module Status window is opened from the Workbench menu, it can be relocated anywhere on the PC Window and thus does not need to remain inside the Workbench windows. There is only one Module Status window that displays the status of all modules connected to the PMBus.

The third type of window is the Plot Window. A Plot Window is opened from the Module Status window by clicking the New Plot menu button located on the bottom right section of the Module Status Window. Plots Windows are used to plot module variables by selecting them in the Module Status window and dragging them to the Plot Window. Each Plot Window can plot multiple module variables and additionally, multiple Plot Windows can also be created by the user.

4.3 The DPI-ProGUI Workbench

The DPI-ProGUI main window can be considered a workbench where multiple windows corresponding to the main DPI-ProGUI functions such as Display/Change Module parameters, Display Module Status and Plot Module Variables can be displayed and organized.

4.3.1 DPI-ProGUI Menus and Icons

The DPI-ProGUI main window, shown below, displays both menu choices and icons. Either can be selected to perform various tasks. By locating the mouse pointer on an icon, a description of the function linked to the icon is also displayed.
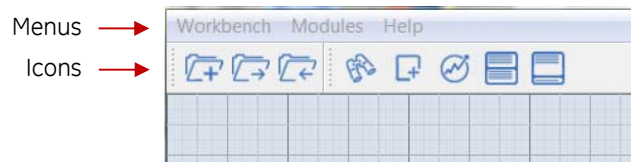
Clicking on the Workbench menu opens a number of menu choices as shown below. If the menu selection is to open an existing workbench, a directory window opens up for the user to tell the program the location and name of the exiting workbench configuration file.
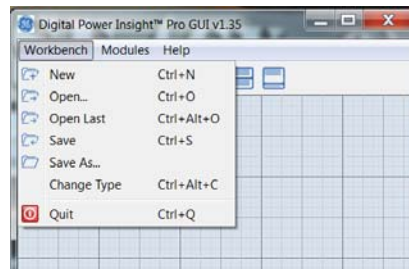
New – allows for a new workbench configuration to be created
Open – used to read in a previously created and saved workbench configuration
Open Last – opens the last created and saved workbench configuration
Save – saves the current workbench configuration to a user-specified file
Save As... - saves the current workbench configuration to a different user-specified file name
Change Type – used to change the product type configurator for the DPI-ProGUI
Quit – exit the DPI_ProGUI program

If a new workbench is selected, the program informs the user that the selection was accepted in a pop-up window. Click on the OK button.
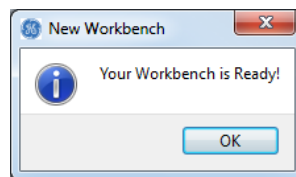
The icons are an alternate way of accessing some of the more commonly used menu choices as shown below.

Save the current workbench

Open an existing workbench

Create a new workbench
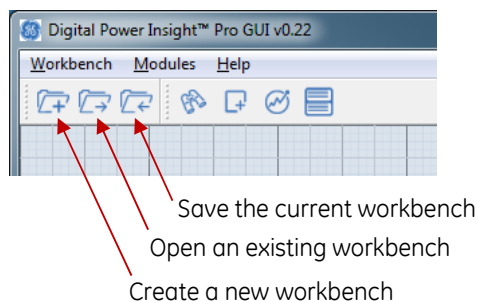
4.3.2 Finding Modules

There are two methods of adding module information to a workbench. Clicking on the Modules menu displays a set of choices as shown below.
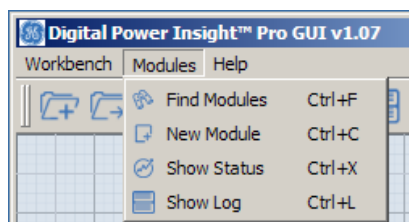
Find Modules – The tool finds all modules connected on the PMBus. For every module that is found, all supported module parameters are read from the module and displayed in a Module Parameter Window (one for each module found).   The icon can also be used to Find modules.

Shown below is an example screenshot where nine modules were found by the DPI_ProGUI tool. The nine Module Parameter Windows, one for each module, are arranged in a cascaded manner on the workbench. These windows can be moved around, resized or each collapsed to an icon within the workbench window using standard Microsoft Windows procedures.
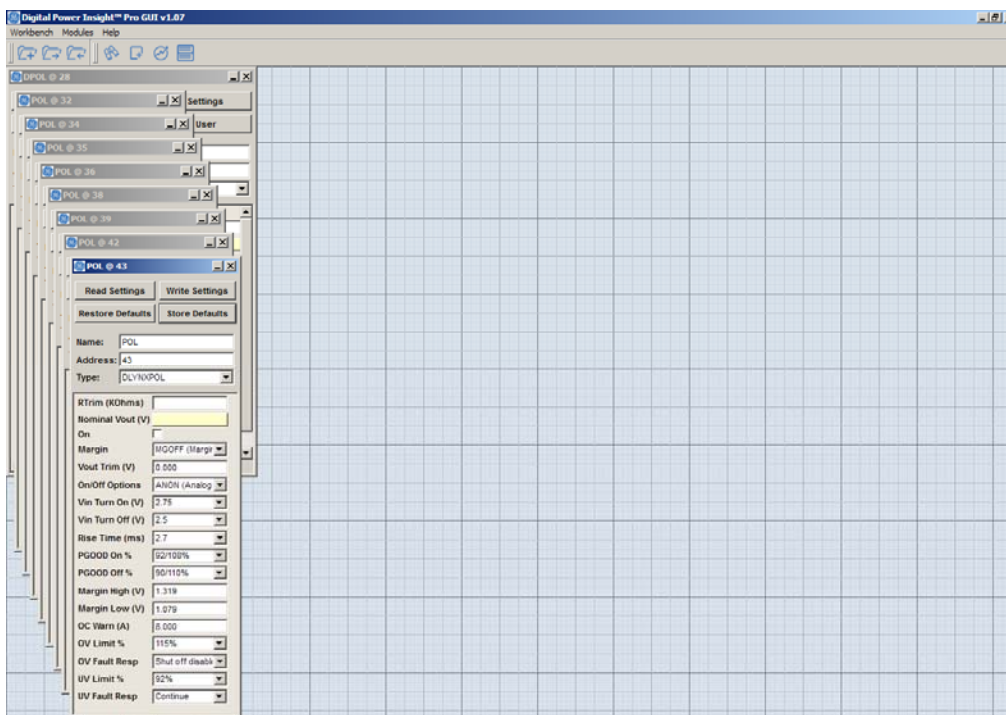
A particular Module Parameter Window can be deleted by clicking on the [X] icon located at top right hand corner of the window. Deleted Module Parameter Windows will be removed from the workbench and may be reintroduced by executing the Find Modules command again or adding a New Module. Note that the Find Modules command may take up to 30 seconds to execute depending on the total number of modules connected to the PMBus.

4.3.3   Adding New Modules

New Module – This allows the user to define a new module. If this command is selected, is it not necessary for that module to be on the Bus. Module parameters, including the address, can be defined prior to actually communicating with the module. The Module Parameter Window initially has blank entries for the module address and parameters, allowing for user entry of this information, as shown in the example below.
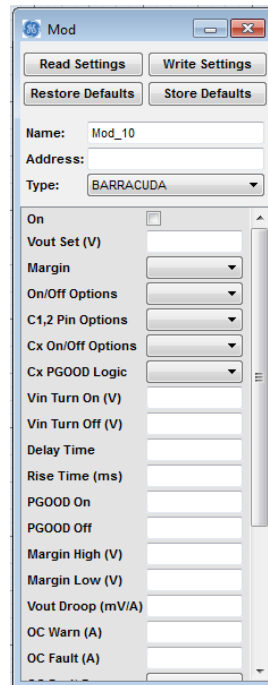
4.3.4   Navigating in the Workbench Window

The navigation tools are available by clicking on the HELP menu of the workbench. These commands allow the user to change the display of the Module Parameter Windows (**Cascade** or **Tile**), reduce or expand to full size (**Restore All** or **Iconize All**) or select a particular Module Parameter Window and move to the next or previous one in sequence.
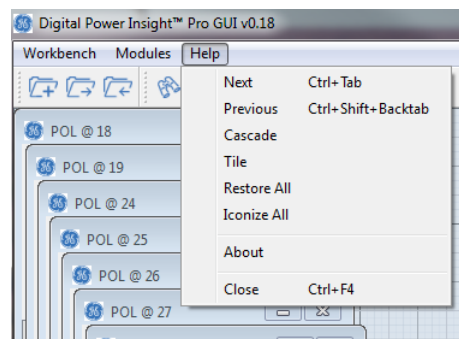
An example of Module Parameter Windows being re-displayed in Tile view is shown below. This view displays the Module Parameter Windows in sequence with increasing module address from left to the right of the screen. If more

Module Parameter Windows are present than can be displayed across the screen, a scroll bar appears on the bottom of the Workbench window.
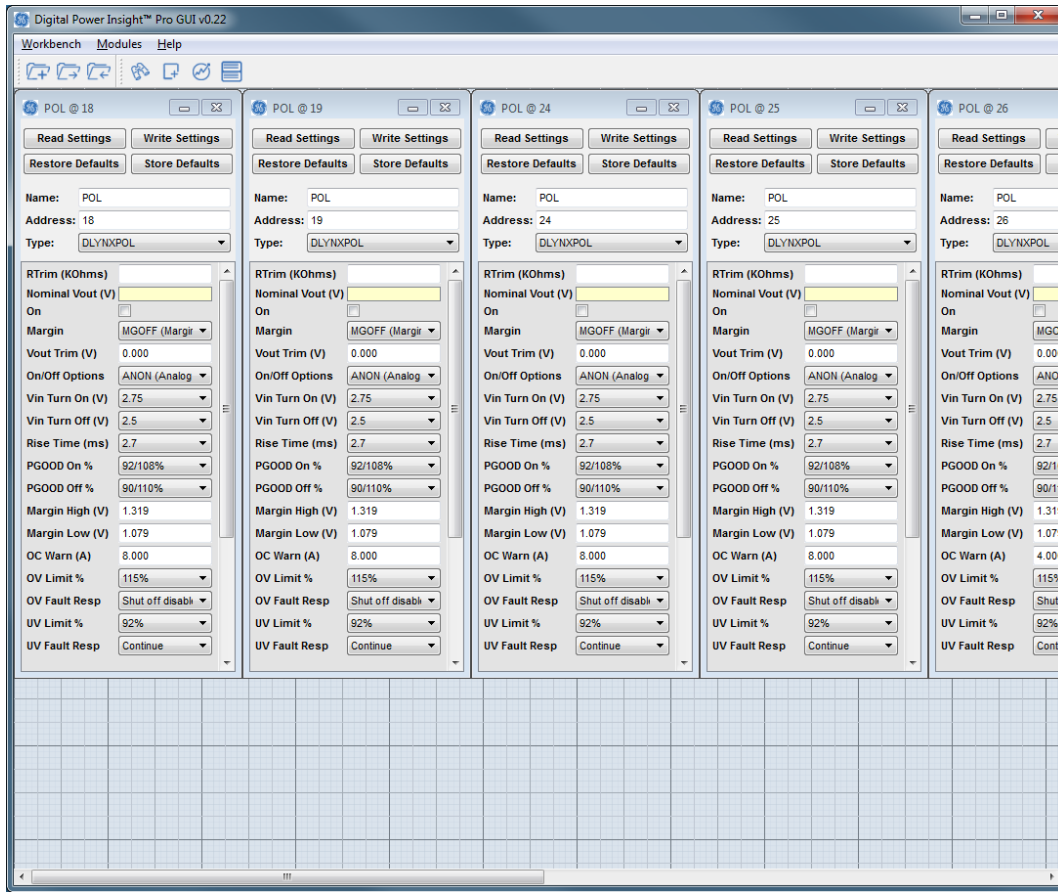
The **Close** command in the Help Menu deletes the selected module from the Workbench. Note that the same function can be accomplished by clicking on the icon at the top right corner of the selected Module Parameter Window.

4.4   Module Parameter Window

The Module Parameter Window is used to display and define module parameters. Shown below is an example of a POL Module Parameter Window with explanations of the various items in the window.

Module Name and PMBus Address

Read settings from module

Restores into the module the non-volatile memory settings

Write settings to the module

Commands the module to write it's settings into non-volatile memory

Module Name

Module Address

Module Type (e.g. POL, Bus Converter etc.)

Trim Resistor Value (only for POLs)

Nominal Output Voltage (corresponding to RTrim)

Various Module Parameters – See Explanation below

4.4.1   Read/Write Settings and Restore/Store Defaults

Both the Barracuda Bus Converter and the DLynx™ POL modules include two sets of memory, one called volatile and the other non-volatile. The Non-volatile memory contains parameters that retain their value even when the modules are unpowered. These are the parameters used during start-up to load values into volatile memory. The values in volatile memory can be changed by communicating to the module when it is powered.

The user cannot directly change values in non-volatile memory. Instead a two-step process is used. First, new values are sent into volatile memory through the Bus, and then a command is issued to the module to copy what is in volatile memory into non-volatile memory. This then overwrites what was previously in non-volatile memory and will be used when the module is powered up again.

Read Settings - Reads into the Module Parameter Window all the settings and values residing in the volatile memory section of the module.
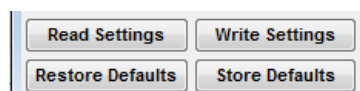
Write Settings - Writes the settings and values that are in the Module Parameter Window into the volatile section of module memory.

Note: If incorrect values are entered in any of the fields and the Write Settings command is selected, the program displays an error message as shown below that indicates the fields that have invalid data and indicating that the data will not be written to the module.
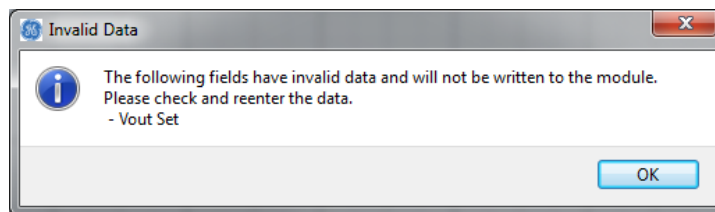
Restore Defaults - Copies module parameters from non-volatile memory in the module into the volatile section of memory in the module and also into the Module Parameter Window.

Store Defaults - Commands the module to copy module parameter values in volatile memory in the module to the non-volattile memory location in the module. Note that to have the module parameter values match those in the corresponding module parameter window values in the DPI_ProGUI, the Write Settings command should be used to transfer the module parameter values from the DPI_ProGUI to volatile memory in the module.

4.4.2 Name, Address and Type

Name - When using a Find Module command, a default name is entered here by the tool. The name of the module can be changed by the user. This information resides in the workbench and is not known to the module.

Address - The PMBus address of the module in decimal format. This value is also pre-populated by the tool when using the Find Modules command. It can also be changed by the user if required.

Type - Identifies the type of module. This value is also prepopulated by the Find Modules command. For POLs the DlynxPOL type should be selected. The remaining portion of the Module Parameter Window will change depending on which module type has been selected.

4.4.3 Other Parameters Stored in the Module – Single Output POL Module Parameter Window

The remaining module parameters will vary depending on the type of module. These parameters are all stored in the non-volatile memory of the module and copied into active memory of the module when it powers up. The Find Modules command copies these over to the Module Parameter Window. If a New Module command is executed, these parameters come up with blank values.

For POLs, the output voltage is set externally through the Rtrim resistor connected between the TRIM pin and SIG_GND of the module. The value of Rtrim (external trim resistor) in kΩ, connected to each module must be entered into the configuration window enabling the program calculation of the Vout_scale_loop constant used to derive the

scaling of the output voltage [note that all entries must be followed by a tab or pressing the <return> key before the entry can be processed].

If the trim resistor value has not been entered, the following pop-up appears when executing the Write Settings command, showing the most common output voltages and their corresponding Rtrim value.

The Rtrim value for output voltages other than those listed above can be calculated either by using the formula for Rtrim in the module datasheet or the POL Programming Tool that can be downloaded from www.gecriticalpower.com and navigating down to the tools download section.

ON - This command is the only one that is automatically executed. Checking the box instructs the module to turn ON via the OPERATION firmware command. The instruction is executed if the on/off options are set to digital ON (DGON) or analog or digital on (ADON) – these options are described further down in this section. Unchecking the box turns OFF the module if DGON or ADON are selected. If the On/Off option is ANON, then the Write Settings command must be issued to actually change the options in the module to DGON or ADON.

Margin - Clicking on the drop-down arrow of the margin box brings up a pop-up window with other margin selection possibilities. The default can be changed by clicking on one of the other options in the drop-down list. The selected option then appears next to the margin command. The selection does not take effect until **Write Settings** is executed.

Vout Trim - Trims the output voltage up or down by the value entered. Click within the box and type in the desired value. A positive value trims the output voltage up while a negative value trims it down. The maximum change in the output voltage is limited to ± 25% of the nominal value. The selection does not take effect until **Write Settings** is executed.

On/Off Options – Controls the module On/Off options. A drop down menu of choices is displayed when the box is selected by clicking on the down arrow. The selection does not take effect until **Write Settings** is executed.

Vin Turn On (V) - Sets the turn ON level of the input voltage from a drop-down menu of choices. The selection does not take effect until **Write Settings** is executed. Note that Vin Turn On must be higher than the Vin Turn Off value. The module will flag a data error if this condition is not met.

Vin Turn Off (V) - Sets the turn OFF level of the input voltage from a drop-down menu of choices. The selection does not take effect until Write Settings is executed. Note that the Vin Turn Off voltage level must be lower than the Vin Turn On level. The module will issue a data error if this condition is not met.

Rise Time - Sets the rise time level of the output voltage from a drop-down menu of choices. The selection does not take effect until **Write Settings** is executed.
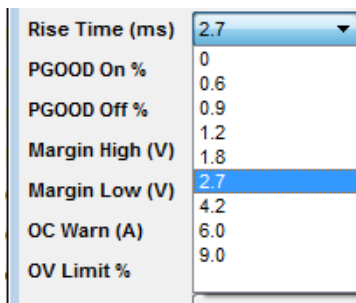
PGOOD On % - Sets the level as a percent of the output voltage that turns the Pgood signal ON. This signal turns ON when the output voltage is between the lower and upper settings as configured by this command. The selection does not take effect until **Write Settings** is executed.
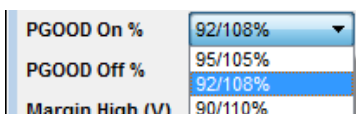
PGOOD Off % - Sets the level, as a percent of the output voltage, that turns the Pgood signal OFF. This signal turns OFF when the output voltage is outside the lower and upper settings as configured by this command. The selection does not take effect until **Write Settings** is executed.

Margin High (V) - The module will raise the output voltage to the value entered in this box when the module is commanded to margin_high. The maximum voltage is the sum of $V_{trim} + V_{margin\ max} \leq 125\%V_{out,nom}$. The selection does not take effect until **Write Settings** is executed.

Margin Low (V) - The module will lower the output voltage to the value entered in this box when the module is commanded to margin_low. $V_{out,}$ The minimum voltage is the sum of $V_{trim} + V_{margin\ min} \geq 75\%V_{out,nom.}$ The selection does not take effect until **Write Settings** is executed.

OC Warn (A) - The output overcurrent warning level can be changed by entering the new value in the box. The selection does not take effect until **Write Settings** is executed.

OV Limit % - The OV shutdown limit can be changed from the drop-down menu of choices. The selection does not take effect until **Write Settings** is executed.

OV Fault Resp - The response to OV shutdown can be changed in the drop-down menu that appears. The selection does not take effect until **Write Settings** is executed.

UV Limit % - The under voltage fault limit can be selected from the drop-down menu choices. The selection does not take effect until Write Settings is executed.

UV Fault Resp - The response to UV shutdown can be changed in the drop-down menu that appears. The selection does not take effect until **Write Settings** is executed.

4.4.4   Other Parameters Stored in the Module – Dual Output POL Parameter Window

The Dual DLynx converter has a slightly different set of module parameters as detailed in the descriptions that follow. An example of the Module Parameter Window for the Dual DLynx™ converter is shown below.

For POLs, the output voltage is set externally through the Rtrim resistor connected between the TRIM pin and SIG_GND of the module. The value of Rtrim (external trim resistor) in kΩ, connected to each module must be entered into the configuration window for each output. This enables the program to calculate the Vout_scale_loop constant used to derive the scaling of the output voltage.

If the trim resistor value has not been entered, the following pop-up appears when executing the Write Settings command, showing the most common output voltages and their corresponding Rtrim values.
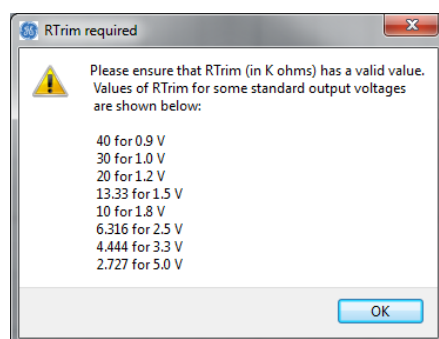
The Rtrim value for output voltages other than those listed above can be calculated either by using the formula for Rtrim in the module datasheet or the POL Programming Tool that can be downloaded from www.gecriticalpower.com and navigating down to the tools download section.

ON - This command is the only one that is automatically executed. Checking the box instructs the selected output of module to turn ON via the OPERATION command. The instruction is executed if the on/off options are set to DGON (digital ON) or ADON (analog or digital on) – these options are described further down in this section. Unchecking the box turns OFF the module if DGON or ADON are selected. If the On/Off option is ANON, then the Write Settings command must be issued to actually change the options in the module to DGON or ADON.

	Output#1	Output#2	
On	☐	☐	← Module is OFF

	Output#1	Output#2	
	☑	☑	← Module is ON

Margin - Clicking on the drop-down arrow of the margin box brings up a pop-up window with other margin selection possibilities for each output. The default can be changed by clicking on one of the other options in the drop-down list. The selected option then appears next to the margin command. The selection does not take effect until **Write Settings** is executed.
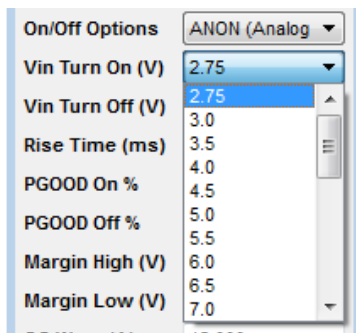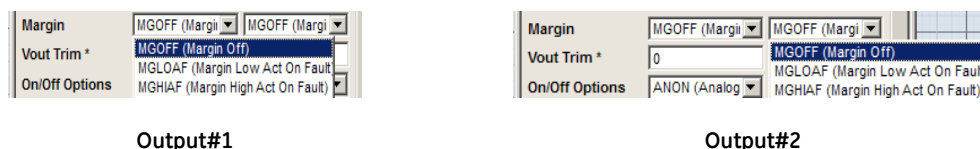
Output#1

Output#2

Vout Trim - Trims the output voltage up or down for each output by the value entered. Click within the box and type in the desired value. A positive value trims the output voltage up while a negative value trims it down. The Vout Trim for the Dual DLynx is a percentage based setting off the reference voltage of 0.6V. The range is from -20% to 10% for Margin High. Permissible values are from -120mV to 60mV in 2mV steps. The selection does not take effect until **Write Settings** is executed.

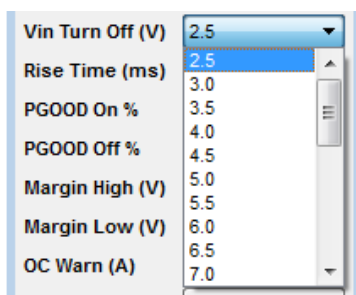|	Output#1	Output#2
Vout Trim *	0	0

On/Off Options – Controls each of the module Outputs. A drop down menu of choices is displayed when the box is selected by clicking on the down arrow with the associated output. The selection does not take effect until **Write Settings** is executed.

	Output#1	Output#2
On/Off Options	ANON (Analog ▾	ANON (Analog ▾
Vin Turn On (V)	4.25 ▾	AAON (Always On)
Vin Turn Off (V)	4.0 ▾	ANON (Analog On)
Rise Time (ms)	2.7 ▾	DGON (Digital On)
		ADON (Anolog or Digital On)

Vin Turn On (V) - Sets the turn ON level of the input voltage from a drop-down menu of choices. The selection does not take effect until **Write Settings** is executed. Note that Vin Turn On must be higher than the Vin Turn Off value. The module will flag a data error if this condition is not met.

| Vin Turn On (V)	4.25 ▾
Vin Turn Off (V)	4.25
Rise Time (ms)	4.5
PGOOD(HiLo)/UV/(	4.75
	5.0
Margin High (V)	5.25
Margin Low (V)	5.5
	5.75
OC Warn (A)	6.0
	6.25
UV/OC Fault Resp	6.5

Vin Turn Off (V) - Sets the turn OFF level of the input voltage from a drop-down menu of choices. The selection does not take effect until Write Settings is executed. Note that the Vin Turn Off voltage level must be lower than the Vin Turn On level. The module will issue a data error if this condition is not met.

Rise Time - Sets the rise time level of the output voltage from a drop-down menu of choices. The selection does not take effect until **Write Settings** is executed.

UV, PGOODLOW (Low/High), PGOODHIGH (High/Low), OV - Sets the Undervoltage, PGOODLOW low, PGOODLOW high, PGOODHIGH high, PGOODHIGH Low level, as a percent of the output voltage. This signal turns OFF when the output voltage is outside the lower and upper settings as configured by this command. The PGOOD pin has hysteresis. When the output trips PGOODLOW (PGL) going low, the output must rise past PGOODLOW high before PGOOD is reset. Likewise when the output trips PGOODHIGH (PGH) going high, the output must lower past PGOODHIGH low before PGOOD is reset. When OV is tripped, the output must go below the PGH low threshold before PGOOD and OV are reset, likewise, when UV is tripped. The selection does not take effect until **Write Settings** is executed.

Margin High (V) - The module will raise the individual output voltage to the value entered in this box when the module is commanded to margin_high. The Margin High for the Dual DLynx is a percentage based setting off the reference voltage of 0.6V for each output. The range is from 0-10% for Margin High. Permissible values are from 0 -60mV in 2mV steps. Including Vout Trim the range translates to -30% to +10%. The selection does not take effect until **Write Settings** is executed.

Margin Low (V) - The module will lower the individual output voltage to the value entered in this box when the module is commanded to margin_low. V_{out}, The Margin Low for the Dual DLynx is a percentage based setting off the reference voltage of 0.6V for each output. The range is from -20 to 0% for Margin Low. Permissible values are from -120mV to 0mV in 2mV steps. Including Vout Trim the range translates to -30% to +10%. The selection does not take effect until **Write Settings** is executed.

OC Warn (A) – The individual output overcurrent warning level can be changed by entering the new value in the box. The selection does not take effect until **Write Settings** is executed.

UV Fault Resp - The response to UV shutdown can be changed in the drop-down menu that appears. The selection does not take effect until **Write Settings** is executed.

4.4.5   Other Parameters Stored in the Module – Barracuda Bus Converter Parameter Window

The Barracuda Bus converter has a slightly different set of module parameters as detailed in the descriptions that follow. Examples of the Module Parameter Windows for the Barracuda bus converters are shown below. Barracuda I, II have identical commands, but Barracuda III which includes all commands from the earlier Barracuda I, II, has additional commands to allow user control of the loop compensation coefficients.

BARRACUDA I and II

BUS @ 18	
Read Settings	Write Settings
Restore Defaults	Store Defaults
Name :	BUS
Address (d) :	18
Type :	BARRACUDA I, II
On	✓
Vout Set (V)	12.00
Margin	MGOFF
On/Off Options	ADON (A
C1,2 Pin Options	T/C1: On
Cx On/Off Options	Cx pin O
Cx PGOOD Logic	Cx PGO(
Vin Turn On (V)	35.00
Vin Turn Off (V)	33.00
Delay Time	20.00
Rise Time (ms)	40.00
PGOOD On	10.10
PGOOD Off	8.50
Margin High (V)	12.60
Margin Low (V)	11.40
Vout Droop (mV/A)	0.00
OC Warn (A)	27.50
OC Fault (A)	30.00
OC Fault Resp	Shut off +
OVin Limit (V)	85.00
OVout Limit (V)	15.99
OVout Fault Resp	Shut off +
OT Warn (C)	125.00
OT Fault (C)	135.00
OT Fault Resp	Shut off +
Alert Response	ARA disa
MFR_FW_REV	0.3.4.4
MFR_MOD_DATE	14P125831174

BARRACUDA III

BUS3 @ 18	
Read Settings	Write Settings
Restore Defaults	Store Defaults
Name :	BUS3
Address (d) :	18
Type :	BARRACUDA III
On	✓
Vout Set (V)	12.00
Margin	MGOFF (Margin
On/Off Options	ADON (Analog a
C1,2 Pin Options	T/C1: On/Off - C2
Cx On/Off Options	Cx pin On/Off igr
Cx PGOOD Logic	Cx PGOOD Pos
Vin Turn On (V)	35.00
Vin Turn Off (V)	33.00
Delay Time	0.00
Rise Time (ms)	14.00
PGOOD On	10.10
PGOOD Off	8.50
Margin High (V)	12.60
Margin Low (V)	11.40
Vout Droop (mV/A)	0.00
OC Warn (A)	45.00
OC Fault (A)	60.00
OC Fault Resp	Shut off + restart
OVin Limit (V)	85.00
OVout Limit (V)	15.00
OVout Fault Resp	Shut off + restart
OT Warn (C)	116.00
OT Fault (C)	126.00
OT Fault Resp	Shut off + restart
Alert Response	ARA disabled
Loop Coeff Kp	1275
Loop Coeff Ki	75
Loop Coeff Kd	-100
Loop Coeff Alpha	-90
MFR_FW_REV	1.1.8.7
MFR_MOD_DATE	YYLLWW123456

Loop Compensation Coefficients –please contact your GE sales representative to discuss procedure to optimize these values

ON - This command is the only one that is automatically executed. Checking the box instructs the module to turn ON via the OPERATION firmware command. The instruction is executed if the on/off options are set to digital ON (DGON) or analog or digital on (ADON) – these options are described further down in this section. Unchecking the box turns OFF the module if DGON or ADON are selected.

| On | ☐ | ← Module is OFF |
| On | ☑ | ← Module is ON |

Vout Set (V) - Changes the set point of the module to the value entered in the box. The selection does not take effect until write_settings is executed.

Margin - Instructs the module to either margin high, or low, or keep margin OFF. Act on Fault is the only supported feature in the bus converter. To change the currently-selected feature, use the drop-down menu and select the desired choice. The selection does not take effect until **Write Settings** is executed.

On/Off Options - Either Analog On, or Analog and Digital On are supported. To change the currently selected feature, use the drop-down menu to select the choice. The selection does not take effect until Write Settings is executed.

C1,2 Pin Options - This command selects the features to be represented by the user configurable C1 and C2 pins. Three combinations of features are supported and are listed in the drop-down menu of choices. The selection does not take effect until **Write Settings** is executed.

Cx On/Off Options - This command further defines the desired ON/OFF features of the C1 and C2 configurable pins. The drop-down menu shows the possible choices. The selection does not take effect until **Write Settings** is executed.

Cx PGOOD Logic - This command further defines the desired PGOOD features of the C1 and C2 configurable pins. The drop-down menu shows the possible choices. The selection does not take effect until **Write Settings** is executed.

Vin Turn Off (V) - Sets the input voltage level below which the output voltage should turn OFF. To change the level, enter the desired voltage level in the box. The selection does not take effect until **Write Settings** is executed.

Delay Time (ms) - To change the displayed level, enter the desired delay time in ms in the box. The selection does not take effect until **Write Settings** is executed.

Rise Time (ms) - To change the displayed level, enter the desired rise time in the box. The selection does not take effect until **Write Settings** is executed.

PGOOD On - To change the displayed level, enter the desired voltage level in the box. The selection does not take effect until **Write Settings** is executed.

| PGOOD On | 10.100 |

PGOOD Off - To change the displayed level, enter the desired voltage level in the box. The selection does not take effect until **Write Settings** is executed.

| PGOOD Off | 8.500 |

Margin High (V) - To change the displayed level, enter the desired voltage level in the box. The selection does not take effect until **Write Settings** is executed.

| Margin High (V) | 12.600 |

Margin Low (V) - To change the displayed level, enter the desired voltage level in the box. The selection does not take effect until **Write Settings** is executed.

| Margin Low (V) | 11.400 |

Vout Droop (mV/A) - To change the displayed level, enter the desired droop level in the box. The selection does not take effect until **Write Settings** is executed.

| Vout Droop (mV/A) | 0.000 |

OC Warn (A) – To change the overcurrent warning level, enter the desired level in the box. The selection does not take effect until **Write Settings** is executed.

| OC Warn (A) | 36.000 |

OC Fault (A) – To change the overcurrent fault level, enter the desired level in the box. The selection does not take effect until **Write Settings** is executed.

| OC Fault (A) | 40.000 |

OC Response - The response to an OC condition is user selectable from a drop-down menu of choices. The selection does not take effect until **Write Settings** is executed.

OC Response	Shut off + re ▼
OVin Limit (V)	Shut off no restart
	Shut off + restart

OVin Limit (V) - To change the displayed level, enter the desired droop level in the box. The selection does not take effect until **Write Settings** is executed.

| OVin Limit (V) | 85.000 |

OVout Limit (V) - To change the displayed level, enter the desired droop level in the box. The selection does not take effect until **Write Settings** is executed.

| OVout Limit (V) | 15.000 |

OVout Response - The response to an OVout condition is user selectable from a drop-down menu of choices. The selection does not take effect until **Write Settings** is executed.

OVout Response	Shut off + re ▼
	Shut off no restart
	Shut off + restart

OT Warn (degC) - To change the displayed level, enter the desired droop level in the box. The selection does not take effect until **Write Settings** is executed.

| OT Warn (degC) | 125.000 |

OT Fault (degC) - To change the displayed level, enter the desired droop level in the box. The selection does not take effect until **Write Settings** is executed.

OT Fault (degC)	140.000

OT Response – The response to an OT Fault condition user selectable from a drop-down menu of choices. The selection does not take effect until **Write Settings** is executed.

OT Response	Shut off no r ▼
Alert Response	Shut off no restart
	Shut off + restart

Alert Response - Selects whether to respond to the host generated Alert Response request. See the device data sheet for further explanations on how this feature works. The selection does not take effect until **Write Settings** is executed.

Alert Response	ARA disable ▼
	ARA enabled
	ARA disabled

Loop Coeff Kp - To change the displayed level, enter the desired droop level in the box. The selection does not take effect until **Write Settings** is executed.

Loop Coeff Kp	1275

Loop Coeff Ki - To change the displayed level, enter the desired droop level in the box. The selection does not take effect until **Write Settings** is executed.

Loop Coeff Ki	75

Loop Coeff Kd - To change the displayed level, enter the desired droop level in the box. The selection does not take effect until **Write Settings** is executed.

Loop Coeff Kd	-100

Loop Coeff Alpha - To change the displayed level, enter the desired droop level in the box. The selection does not take effect until **Write Settings** is executed.

Loop Coeff Alpha	-90

4.4.5.1
Device Identification

The pro-gui automatically reads the manufacturer product information recorded in memory.

MFR_FW_REV – Displays the revision number of the module's DSP firmware
MFR_MOD_DATE– the structure of this information is shown below
> YY – 2 character representation of the year of manufacture.
> LL – 2 character representation of the manufacturing location.
> WW - 2 character representation of the week of manufacturing.
> 123456 – Up to 6 characters of serial number.

MFR_FW_REV	1.1.8.7
MFR_MOD_DATE	YYLLWW123456

4.4.6   Other Parameters – CP3500 Parameter Window

The CP3500 rectifier has a slightly different set of module parameters as detailed in the descriptions that follow. An example of the Module Parameter Window for the CP3500 rectifier is shown below.

4.4.6.1   Memory management

The CP3500 (AC/DC power supply) uses three memory stacks. The first two memory stacks, Factory settings and user settings, are non-volatile. That is, their information is retained if bias power is removed from the processor. The third memory stack is working memory. This is the memory stack the power supply uses for program values. Working memory is volatile. If bias power to the processor is interrupted, the values in working memory are lost.

Factory settings are used to permanently retain the data used at the factory for future customer use. As such these settings cannot be changed by the user.

User settings are initially programmed at the factory with the same values as the factory settings. On start up the unit actually downloads user settings into working memory. The function of the user settings is to enable the user to make permanent fault response changes to the power supply.

Working memory data is used by the power supply for configuration and response behaviour.   Changes written into working memory are immediately used by the power supply program.

These data bases are easily represented by the flow chart below. User memory is transferred into working memory on start-up. The data written into working memory can get uploaded into user memory so that subsequent start-ups use the revised values and responses in user memory.

The user can also reset the power supply back into the factory default levels. This is accomplished by first reading into working memory the levels of factory default. This is then followed by writing into user memory the levels of the current working memory, which is now identical to the factory default levels. Subsequent start-ups still read back the levels of user memory that has been reset into the factory default values.
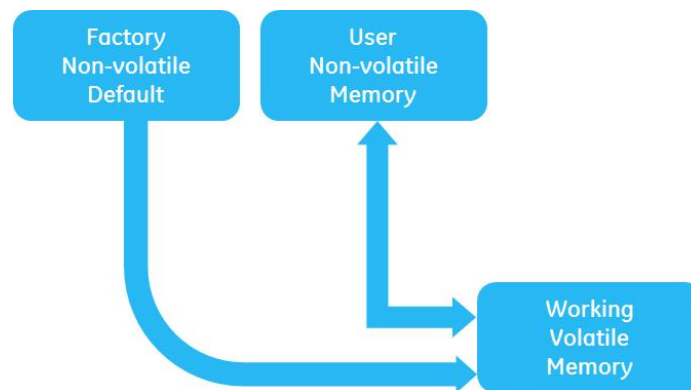
4.4.6.2   Memory command set

Six commands, three read and three write, are used to manage the data banks of the three memory registers.
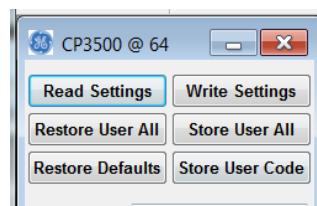
Read settings – This command loads into the pro_gui the values and responses programmed into working memory

Restore user all – This command loads into the pro_gui the values and responses programmed into user memory

Restore defaults – This command loads into the pro_gui the values and responses programmed into default memory

Write settings – This command writes into working memory the values and responses shown in the parameters window of the pro_gui.

Store user all – This command writes into user memory the values and responses stored into working memory

Store User Code - This function enables the user to save into user memory only a selected set of features rather than all of working memory. Clicking on the **Store User Code** button enables the selection table below. Check the features that should get stored permanently in the user memory stack. When finished the desired selections click on **Store**. See the screen capture below,
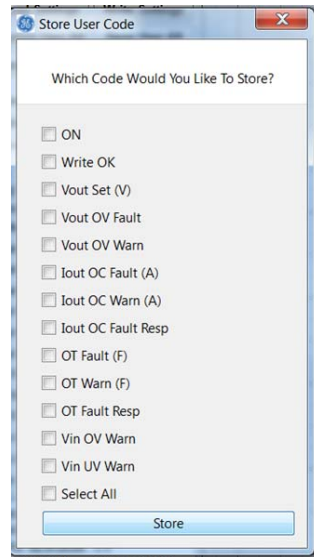
4.4.6.3   Immediate commands

These commands are executed immediately as soon as they are clicked by the user. There are three such commands;

ON - Checking the box instructs the module to turn ON the main output via the OPERATION firmware command. Unchecking the box turns OFF the main output via the OPERATION firmware command.

LED test – Checking the box instructs the module to execute blinking of all LEDs until the box is un-checked.

Service LED - Checking the box instructs the module to execute blinking of the service LED until the box is un-checked.

4.4.6.4   Parameter settings

Changes in these fields are not implemented by the program until **write settings** is executed. All changes can be saved into the user memory registers for permanent execution during power up.

The pro-gui executes a **read settings** automatically when the parameter window is requested. The displayed values are currently residing in working memory. Values can be changed as long as the entry is within reasonable operational levels.

Write OK – Overwriting the features in working memory can be controlled by this command. The default configuration permits all features to be writable as shown by the **ALL** selected in the box. Other selections appear when the drop down box is selected

> **WP** – disables all writes except the **write protect** command

> **WP_OP** – disables all writes except **write protect, OPERATION** and **PAGE** commands

WP_OP_CFG – disables all writes except **write protect, OPERATION, PAGE, ON_OFF_CONFIG, VOUT_COMMAND**
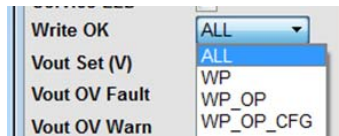
Vout Set(V) – Changes the output voltage setting. The entered value does not get executed until **write settings** is clicked. The power supply only accepts values between 42 and 58Vdc. Out-of-range values are identified by setting the data_fault bit and asserting SMB Alert.

Vout OV Fault -
Vout OV Warn - These registers change the overvoltage warn and fault (shutdown) levels.   The entered value does not get executed until **write settings** is clicked. The power supply only accepts values between 42 and 60Vdc. Out-of-range values are identified by setting the data_fault bit and asserting SMB Alert.
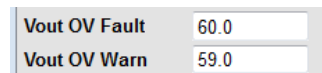
Iout OC Fault -
Iout OC Warn - These registers change the overcurrent warn and fault (shutdown) levels.   The entered value does not get executed until **write settings** is clicked. The power supply only accepts values below the default warning of 67A and the default fault of 68A. Out-of-range values are identified by setting the data_fault bit and asserting SMB Alert. Note: The overcurrent settings can only be changed at high line when maximum output power is available.
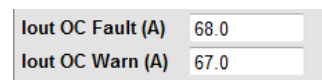
Iout_OC_Fault_Response – The overload response can be changed from the defaulted Hiccup configuration to Latched. The entered value does not get executed until **write settings** is clicked.

OT Fault -
OT Warn - These registers change the overtemperature warn and fault (shutdown) levels.   The entered value does not get executed until **write settings** is clicked. The power supply only accepts values below the default warning of 105°C and the default fault of 110°C. Out-of-range values are identified by setting the data_fault bit and asserting SMB Alert. Note: Note: The reported temperature is the highest of the various measured temperature levels.
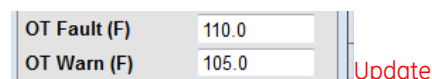
OT_Fault_Response - The over temperature response can be changed from the defaulted Restart configuration to Latched. The entered value does not get executed until **write settings** is clicked.

Vin_OV_Warn_limit

Vin_UV_Warn_limit – The OV_warn and UV_warn limits can be changed from the default values. The entered values do not get executed until **write settings** is clicked.

Vin OV Warn	295.0
Vin UV Warn	87.5

Fan_command_1 – The fans can be instructed to spin faster than needed by power supply control. The instruction enters the fan speed in integers of percent of full load. The entered value does not get executed until **write settings** is clicked.

Fan Speed (%)	0.0

4.4.6.5   Device Identification

The pro-gui automatically reads the manufacturer product information recorded in memory and the run time of the power supply measured from the time input power is applied.

MFR ID – the abbreviation is for GE – Critical Power
MFR MODEL – 12 character representation of the part identification
MFR REVISION – 7 character representation of the series number shown on the label.
MFR SERIAL – Up to 15 characters of serial number.
FIRMWARE REVISION – Displays the revision number of the three on-board processors
RUN TIMER – accumulated on time of the power supply in hours

MFR ID	GE-CP
MFR MODEL	CP3500AC54T
MFR REVISION	0:0
MFR SERIAL	14KZ40008364
FW Rev : Pri	1.23
Sec	1.61
I2C	0.45
Run Timer	13 hrs

4.5   Monitoring Module Status

The DPI-ProGUI can monitor analog readings, digital status and alarm information of either all or a selected set of modules. Both continuous polling, at a pre-defined interval, or a single poll capability exists. The Module Status Window is opened from the Workbench window by either using the menu choices (Modules – followed by Show Status) or using the Show Status Icon, as shown below.

Opens Module Status Window
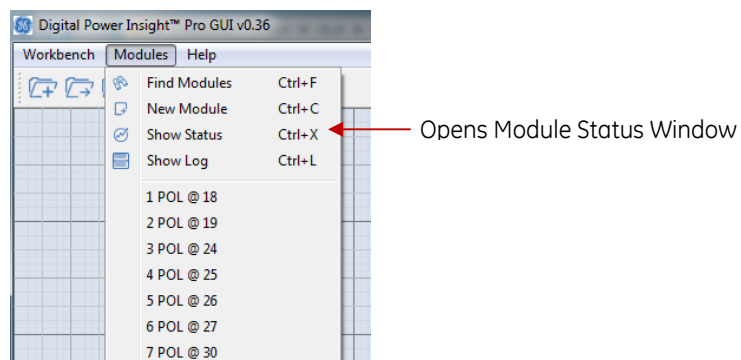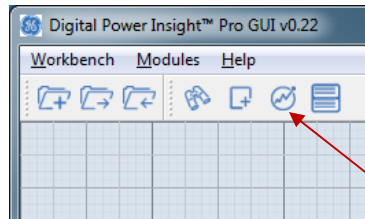
Clicking Icon opens Module Status Window

4.5.1 Module Status Window

An example of a Module Status Window is shown below highlighting the key user commands and functions provided.

Polling Section

Logging Section

SMB Alerts Section

Status Summary Section

New Plot Button
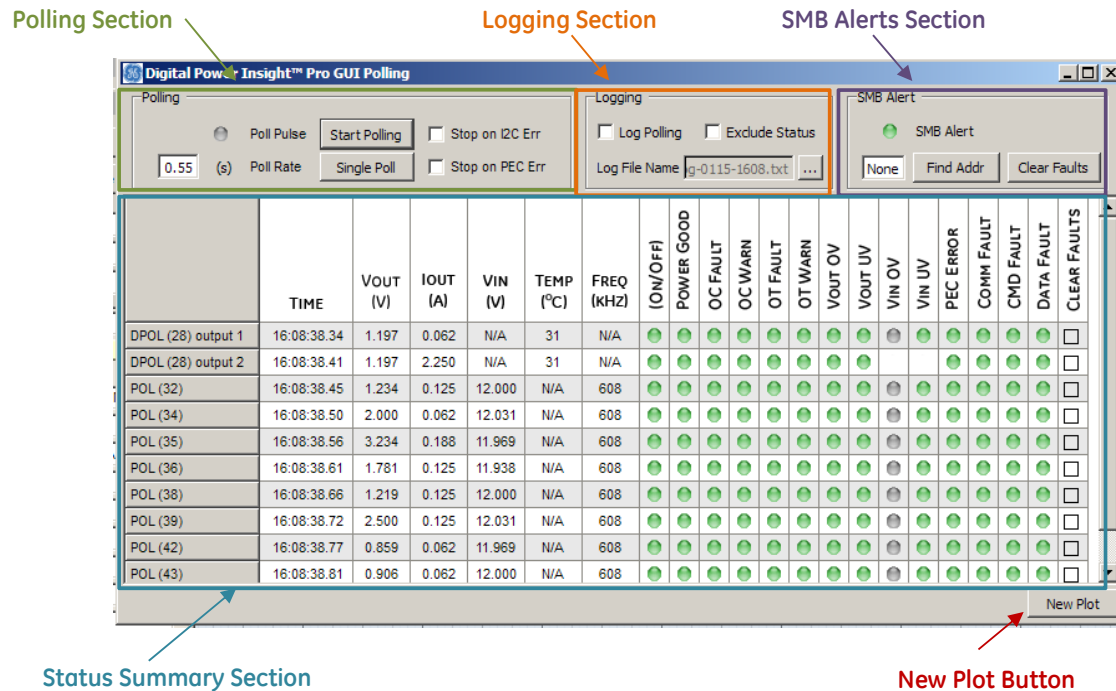
4.5.2 Polling Section

Polling refers to the program reading variables from the module and updating the Module Status window. Both a single poll or repetitive polling are supported. For repetitive polling, the Polling Rate in seconds, which specifies how often the group of all modules is polled, is entered in the box. The lowest polling rate possible is 0.3s. In a system with a large number of modules, a fast polling rate may not be possible.

By clicking the Start Polling button, the tool starts polling the modules at the rate set in the Polling Rate box. The LED-like oval to the left of the Polling Pulse also flashes blue at the polling rate to visually alert the user that polling is occurring. When the Start Polling button is clicked, it changes to color to blue and toggles to become the Stop Polling button. Clicking it will stop the polling process.
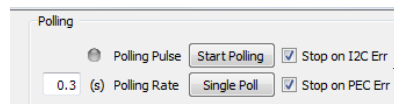
Polling will automatically stop on I²C or PEC errors if these options are selected

4.5.3 Logging Section

Logging saves the polled data to a text file in a delimited format that can read by programs such as Microsoft Excel. This section allows the user to choose whether the polled data needs to be logged as well in a user-specified file name. The user can choose whether to exclude Status data in the logged file – this will reduce the size of the log file

and make it easier to manipulate. Note that the filename of the logged file cannot be change while data is being polled from modules. The user must stop polling and enter a new logging file name, and can then resume polling.

Logging
☐ Log Polling   ☐ Exclude Status
Log File Name og-0122-1041.txt [...]
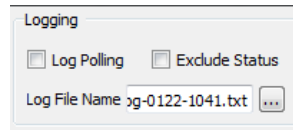
Logging will only occur when the tool is polling modules. All module variables are saved in the file at the same rate as specified for polling. An example of a logging file is shown below

Date   Module 1 Name and variables   Time

```
MODULE NAME 1 TIME    VOUT_1 TIME    IOUT_1 TIME    VIN_1  TIME    TEMP_1 TIME    FREQ_1 13:41:22.65
    01-22-2013      doc version 1.01
POL (18)     0.04   0.992  0.05   0.062  0.11   12.094 0.13   N/A    0.13   608
POL (18)     0.38   1.008  0.4    0.062  0.41   12.062 0.42   N/A    0.42   608
POL (18)     0.66   0.992  0.68   0.062  0.69   12.062 0.7    N/A    0.71   608
```

Data from
Scan 1

The logging file that is created uses the Tab character as a delimiter. The first line of the logging file has header information with Module Name and variables associated with that module. When multiple modules are present, this information is repeated on the first line for each module. The end of the first line has the time and date that the file was created followed by the version number of the file format.

Each line that follows then has the data from one scan where the module name, time/variable value are written for each of the modules. Subsequent lines contain similar information for each succeeding scan.

The logging file can be read into Microsoft Excel. By using choosing the Delimited file option in the Text Import wizard in Excel, and Tab as the delimiter, the data can be converted to an Excel file for further processing or plotting. An example of a screenshot from a conversion into Excel format is shown below. Each line contains the summary of one polling scan and has the variables of all modules collected during that scan.

4.5.4   SMB Alerts Section

In a system with PMBus-compatible modules, the controller monitors the SMBALERT line which each module is connected to. The SMBALERT signal is asserted by any of the modules pulling the line low. Therefore, a high on this line is indicated by a GREEN LED in this section indicating all modules are operating properly. If any of the modules asserts SMBALERT by pulling the line low, the SMB Alert LED display in this section turns RED.

Clicking on the **Find Addr** button identifies the module with the lowest address that has triggered the Alert line. By clicking the **Find Addr** button again, the next module with the lowest address that is still pulling down the Alert line can be identified. Note that once a module has sent out its address it is required to clear the Alert line).

When the final module that pulled down the Alert line has been identified the SMB Alert LED changes its color from RED to GREEN indicating that all modules have been identified.

For Bus Converters, a more complicated Alert handling process is implemented – please see the Bus Converter data sheets for additional information.

For AC/DC and DC/DC Power Supplies the Alert handling process is not supported.
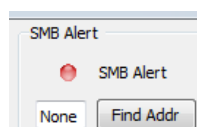
4.5.5   Clearing Status Indicator and Faults

Prior to clearing the status indicators the modules to be cleared have to be selected. Clicking on the Clear Faults header selects all modules automatically and checks the Clear Faults box of each module. Then all module faults can be cleared using the Clear Faults button. If only select module faults are to be cleared, the user can select only the boxes corresponding to the modules that need to be cleared of faults.

Clear Faults Button. Clicking this clears all module faults

Clear Faults Header. Clicking this selects (or deselects) all modules for clearing faults.

AC/DC and DC/DC Power Supplies assert the SMBAlert line on power-up to notify the host controller that they are available on the bus. This is the mechanism by which power supplies that have been hot plugged into a system declare their presence to the controller.

4.5.6 Global commands for power supplies

Power supplies can be commanded simultaneously using the Global commands section of the status window. Setting the output voltage or changing the 'host controller' in control are executed in the pro_gui using the Global commands section.
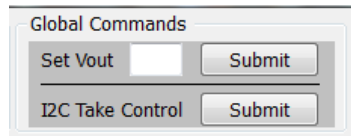
Global instructions, such as changing the output voltage, make sure that the power supplies simultaneously change their outputs so that they could continue to current share and provide their share of output power during the transition in output voltage setting of paralleled power supplies.

Every command can be issued using either the Global or individual addressing schemes. In the pro-gui the intent of the two Global commands is to demonstrate the practicality of the Global function. That does not imply that other commands cannot be supported using the Global function. For example, to turn the output bus power OFF, a global OFF command via the OPERATION feature can be issued to turn OFF the main outputs of all power supplies simultaneously.

4.6 Plotting

Plotting of module variables in a separate window is initiated by clicking on the New Plot button located at the bottom right hand corner of the Module Status window.

Multiple plot windows can be opened by repeatedly clicking on the New Plot button. These plot windows can be resized and relocated using standard Windows procedures. The Plot Windows can also be iconized and restored as needed.
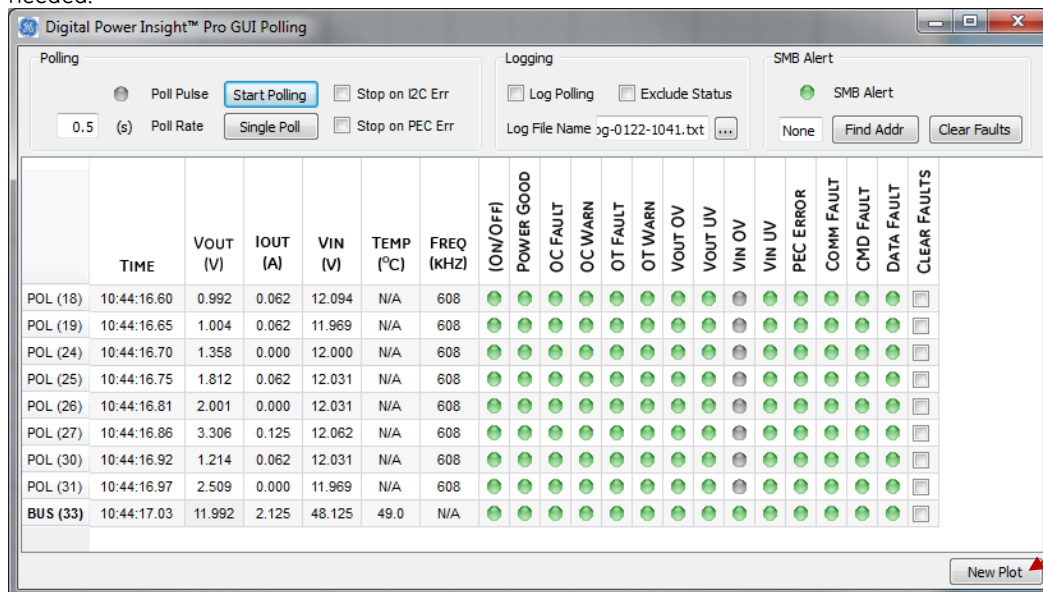
New Plot Button

An example of a blank Plot Window is shown below. Various icons control different functions related to plotting as described below.

Reset Original View
Previous and Next Views
Pan Plot
Zoom Plot
Edit Plot Parameters
Save Plot as a Graphic File

Variable Name Display Area

Plot Display Area
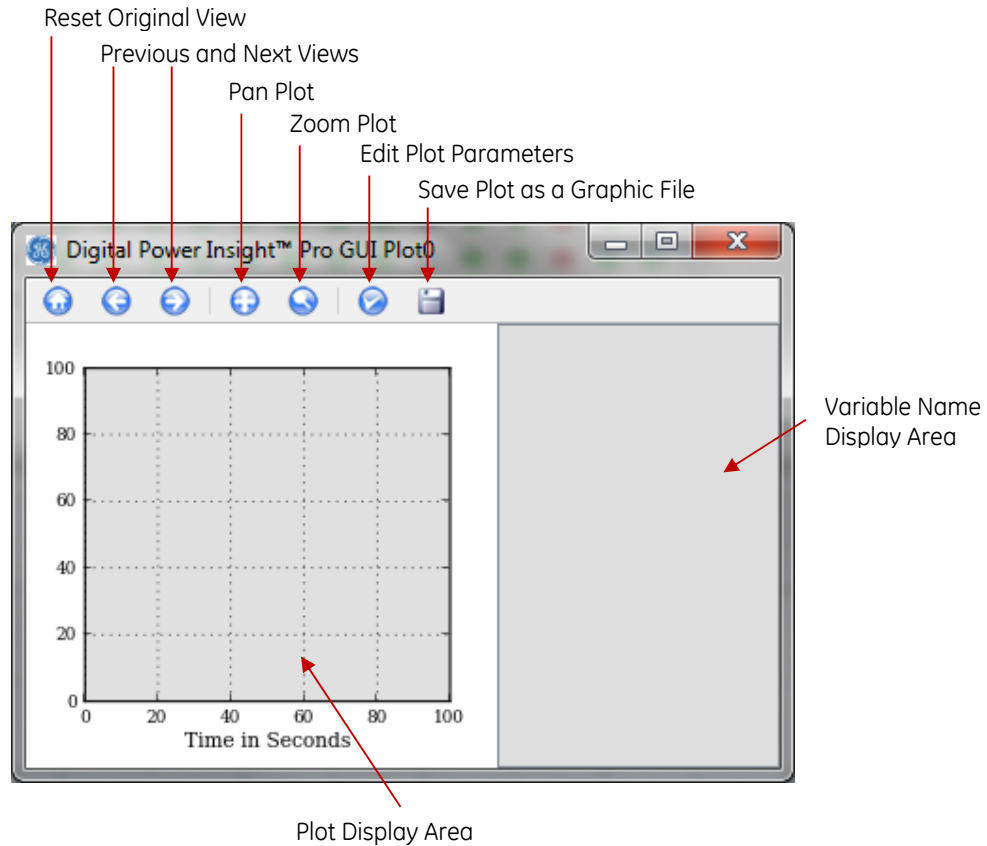
4.6.1   Selecting and Plotting Module Variables

To plot a particular module variable, first identify the function in the Module Status window by positioning the mouse pointer over the variable and clicking the mouse button. When the variable is identified, the variable within the Module Status window turns blue as shown below.
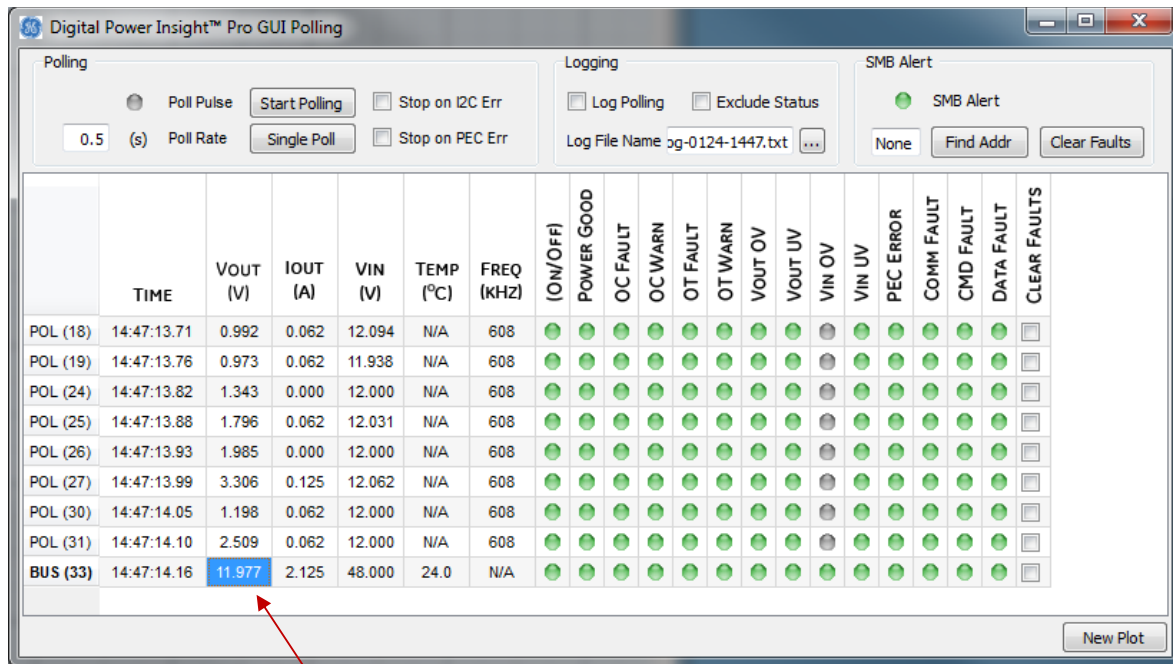
Module variable selected within Module Status Window

Next, the highlighted area is dragged using the mouse (left click, hold and move mouse pointer) to the Plot Window Variable Name Display Area. As the mouse pointer is moved into the Plot Window Variable Name Area, it changes to a + sign. At that point the mouse button can be released to drop the variable name to the Plot Window. In the example below, the variable Vout of the module Bus_1 (17) has been selected for plotting.
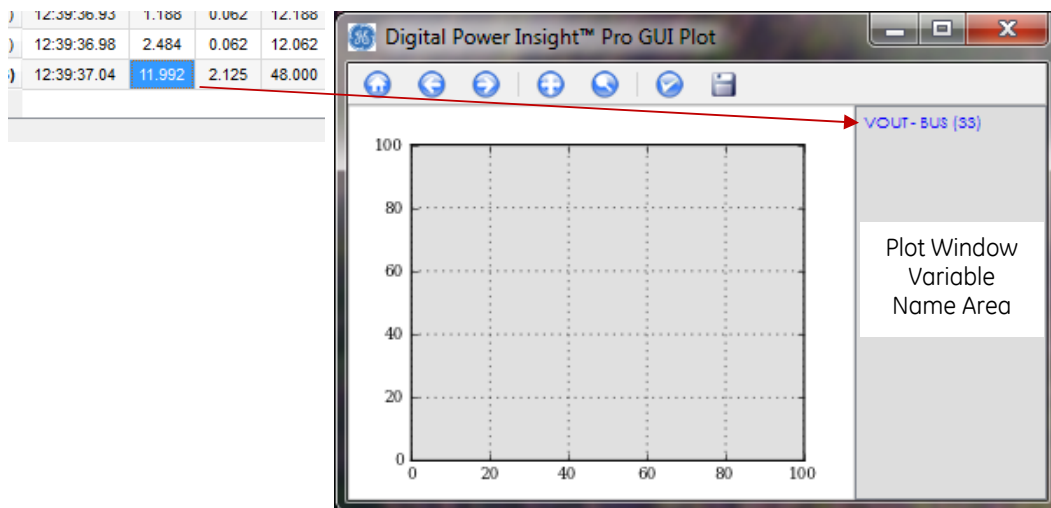
To select multiple module variables for plotting, repeat the process of selecting and dragging module variables to the Plot Window Variable Name Display area. Note that multiple module names can be selected in the Module Status window by using the <CTRL>+Click (for a sequence of names) or <SHIFT>+Click to select an area of module names for plotting. Examples of the Module status window when multiple module names are selected are shown below.

TIME	VOUT (V)	IOUT (A)	VIN (V)	TEMP (°C)	FREQ (KHZ)
:12:54.73	0.891	0.000	12.031	N/A	608
:12:54.79	0.922	0.062	11.719	N/A	608
:12:54.85	1.250	0.125	11.938	N/A	608
:12:54.90	1.703	0.125	12.000	N/A	608
:12:54.95	2.000	0.062	12.000	N/A	608
:12:55.00	3.250	0.188	11.969	N/A	608
:12:55.05	1.188	0.062	12.031	N/A	608
:12:55.11	2.453	0.062	12.062	N/A	608
:12:55.17	11.992	2.125	48.000	48.0	N/A

TIME	VOUT (V)	IOUT (A)	VIN (V)	TEMP (°C)	FRI (kH
14:12:54.73	0.891	0.000	12.031	N/A	60
14:12:54.79	0.922	0.062	11.719	N/A	60
14:12:54.85	1.250	0.125	11.938	N/A	60
14:12:54.90	1.703	0.125	12.000	N/A	60
14:12:54.95	2.000	0.062	12.000	N/A	60
14:12:55.00	3.250	0.188	11.969	N/A	60
14:12:55.05	1.188	0.062	12.031	N/A	60
14:12:55.11	2.453	0.062	12.062	N/A	60
14:12:55.17	11.992	2.125	48.000	48.0	N/

Multiple module names selected using <CTRL>+Click

Multiple module names selected using <SHIFT>+Click

The actual plotting of the selected variables in the Plot Window occurs when the Start Polling button is clicked in the Module Status window. Plotting continues until polling is stopped. The Plot Window will automatically adjust the vertical and horizontal scales as the plotting proceeds. If variables with widely different values are being plotted on the same Plot Window, the vertical scale will adjust to accommodate all variables. Similarly, the horizontal scale will automatically adjust as the total plot time increases. The Plot horizontal axis start time is always zero, unless adjusted by the user using the Edit Plot Parameters function. Note that if a plot is started, stopped and then started again, the Plot Window erases the previous plots and restarts plotting with time equal to zero.

A Plot Window can be opened after the Start Polling button has been clicked. In that case, the time=0 reference point is when the Start Polling button was pressed and the plot will start at some value of time > 0. If multiple Plot windows are opened at different times, they maintain time synchronization, i.e. time=0 is the same for all plots and is when the Start Polling button was clicked.

4.6.2   Changing Plotting Window Parameters

Various plotting parameters can be changed using the icons in the Plotting Window.

Zoom Function - By clicking on this icon, the user can define a zoom area rectangle and created a plot expanded around the selected area. The Previous View icon will restore the previous plot view if desired.

Pan Function - The pan function can be used to relocate the plots within the plot window. Again the Previous View icon can be used to restore the previous plot view.

Edit Plot Parameters - Allows the user to enter a title for the plot as well as custom X- and Y-axis minimum and maximum scale values and labels. Parameters of the curves being plotted such as type of line (solid, dotted or dashed), line width and a label for the line can also be entered. The Edit Plot Parameters brings up the two Figure Options windows shown below. The one on the left shows the options for adjusting Axis parameters, while the one on the right shows the choices for plot curve parameters.
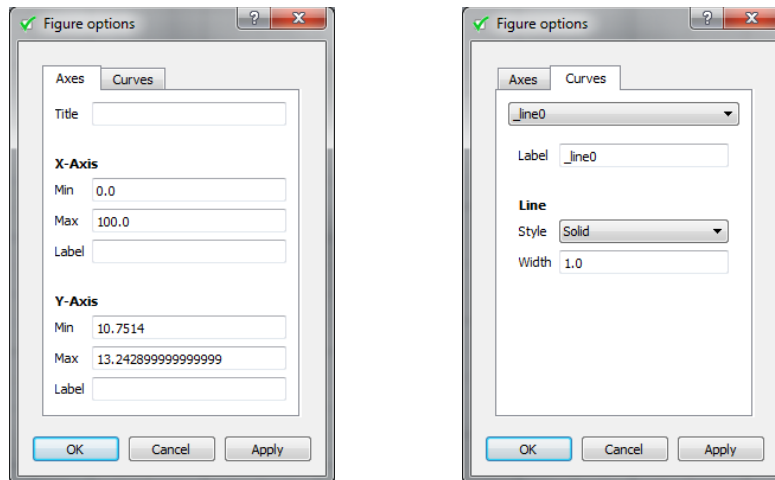
 Reset Original View - This function restores the original plot view (before the Pan, Zoom or Edit plot parameters were invoked by the user).

 Back and Forward Plot Views – Allows the user to go back to the previous view or forward to the next view (if in a series of views).

 Save the Figure – Allows the user to save the plot as a graphics image file in a user-specified file name using the .png format.

4.7   Transaction Logging

The DPI-ProGUI tool supports transaction logging of all communication (I²C) traffic as well as commands and data issued/read by the tool. This information is by default logged and stored in a file called "Transactions_Logs.txt" which is located in the directory where the DPI-ProGUI executable is located. This file can be opened and viewed by a text editor (such as Windows Notepad) any time during or after the DPI-ProGUI tool has been opened.

In addition to saving transaction data from the tool in a file, these can also be viewed in a Transaction Log Window within the tool. This window can be activated either by using a menu choice in the DPI-ProGUI Main window or clicking on the icon to open the Transaction Log Interface. These choices are shown below.
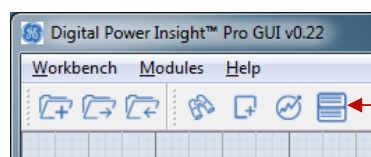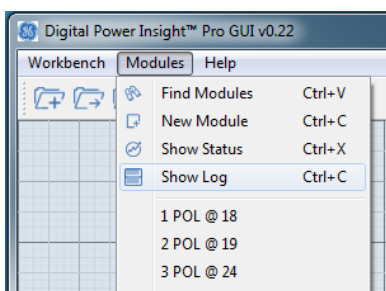
The Transaction Log Window is split into the Communications Log and Command/Data Log sections as shown below.
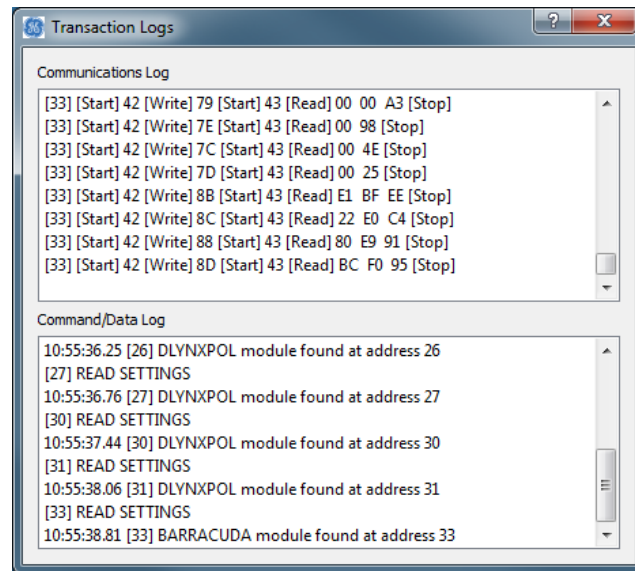
The Communications Log section details the I^{2}C traffic by showing the traffic on the bus. These commands are displayed as hex letters or [Start], [Write], [Read] and [Stop]. The commands are listed in the sequence they are sent across the bus and a scroll bar on the right allows examination of earlier commands. A time stamp is also displayed periodically in this window.

The Command/Data Log displays a summary of the commands sent to the modules and data received back. Note that not every command to the module is displayed, just a summary of key commands.

Please note that while the Transaction Log file is always created, displaying the Transaction Log Window within the DPI-ProGUI tool can be done at any time the tool is running. When the Window is opened, all of the transaction log data is displayed in the Transaction Log Window, with the more recent data visible.

5 DPI-GUI

The DPI-GUI is a graphically-oriented tool for interfacing to up to six modules. The tool capabilities include

- Identifying and configuring the addresses of up to six modules connected to the I²C bus interfaced by the GE USB Interface Adapter
- Setting Trim Resistor value and numerous parameters associated with the modules
- Display of status and input/output voltage and output current information from the modules
- Reading and writing settings information to the registers in the modules
- Storing and restoring parameters to the non-volatile memory storage locations in the modules
- Setting up periodic polling and display of key information from the modules
- Saving and loading GUI configuration information to and from a user-specified file
- Communications log displaying commands on the I²C bus
- Automatically storing all commands and status changes with associated time stamps in a log file

5.1 Starting the DPI-GUI Tool

Once the USB Interface Adapter is plugged into the USB port on the Personal Computer, the Green LED of the Adapter should be ON. Apply input power to the module and start the DPI-GUI application. Note that DPI-CLI and DPI-GUI applications cannot run simultaneously as they share the same COM port connection to the USB Interface Adapter on the personal computer.

The DPI-GUI tool can be started by double clicking on the **DPI GUI** icon located on the desktop, or using the standard **Windows Start – Programs – DPI – DPI GUI** option. The tool starts up and displays the following screen where four POLs and two Bus Converters are shown, (Note: it may take up a few seconds for this screen to appear. This is normal)
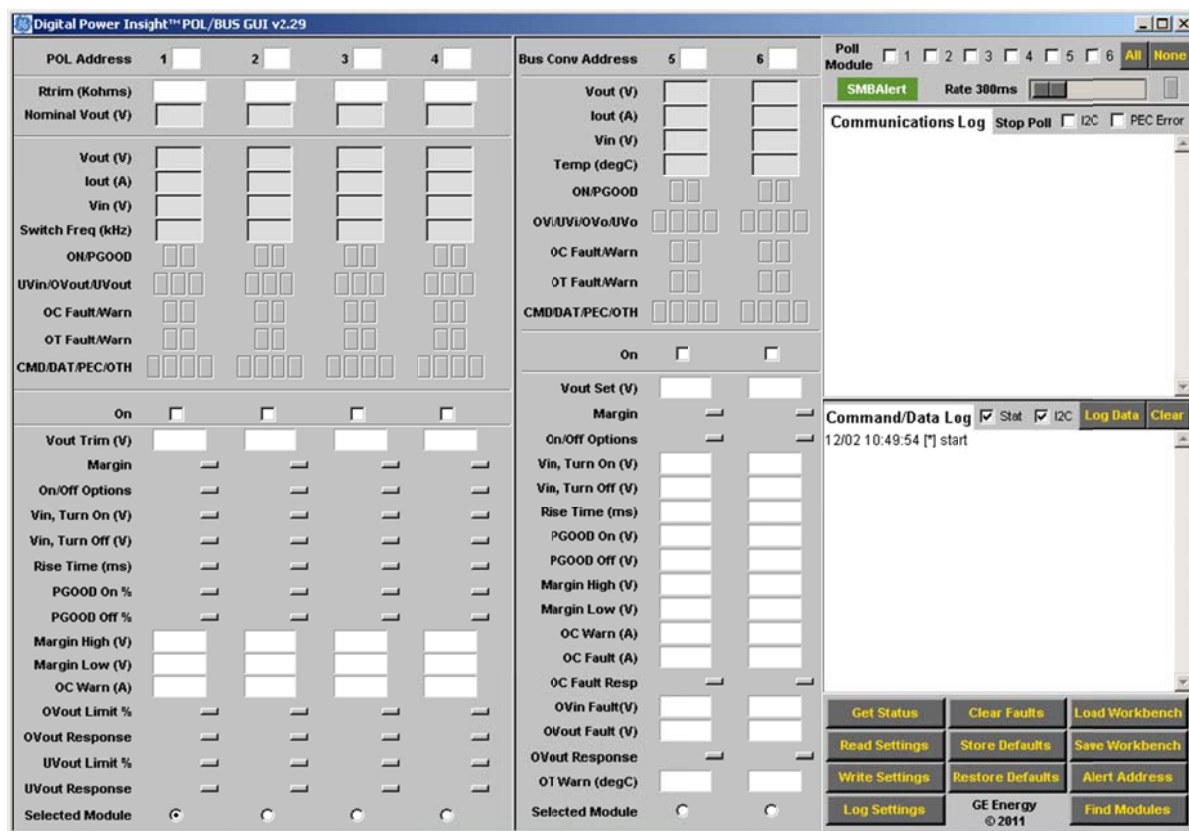
This tool has only the one screen shown above. The various functions of the tool will be explained in the following sections.

Note that the GE USB Interface adapter needs to be plugged into the computer for the DPI-GUI program to work properly. If the GE USB Interface adapter cannot be accessed by the DPI-GUI software for any reason, it will display the following window during startup.

In some cases, this can be corrected by removing and re-inserting the USB cable connected to the GE USB Interface adapter, and restarting the DPI-GUI program. Correcting this condition may require at least two reinsertions .To exit the DPI-GUI program, just click on the X in the top right corner of the tool display window.

5.1.1 Custom Module Configuration

The default startup configuration of the GUI displays four POLs and two Isolated Digital Bus converters. If a different combinations of POLs and Bus Converters is desired, the user can customize the number of POLs and Bus converters configured on the GUI screen. To start with a custom module configuration, the GUI must be started using a DOS Window.

To do this, click on the Windows Start button

, select <Run...> and enter "cmd" in the box that opens, and click on the OK button.
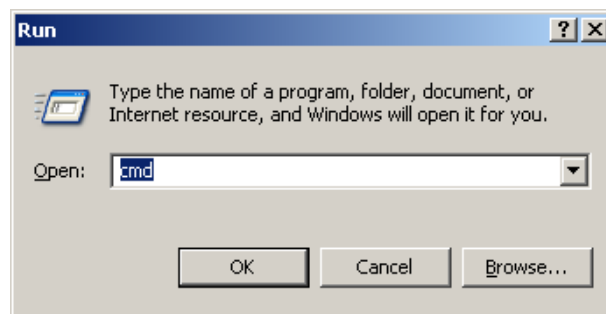
This opens up a DOS command window.

Navigate to the directory where the DPI files are stored (typically C:\Program Files\GE Energy\DPI

The GUI can be started with a custom module configuration by using the following syntax:

```
C:\Program Files\GE Energy\DPI> dpi_gui [POL_#]
```

Here POL_# would be the number of POLs you want to display in the GUI with the custom module configuration. POL # needs to between 1 and 6. So a total of six modules are displayed, ranging from 1 POL and 5 Bus Converters to all 6 being POLs.

For example, typing the command

```
C:\Program Files\GE Energy\DPI> dpi_gui 2
```

starts the GUI with a screen displaying two POLs and four Bus Converters.

5.2   DPI-GUI Description

The DPI-GUI tool display may be divided into the areas shown in the figure below as an aid to describing its functionality.
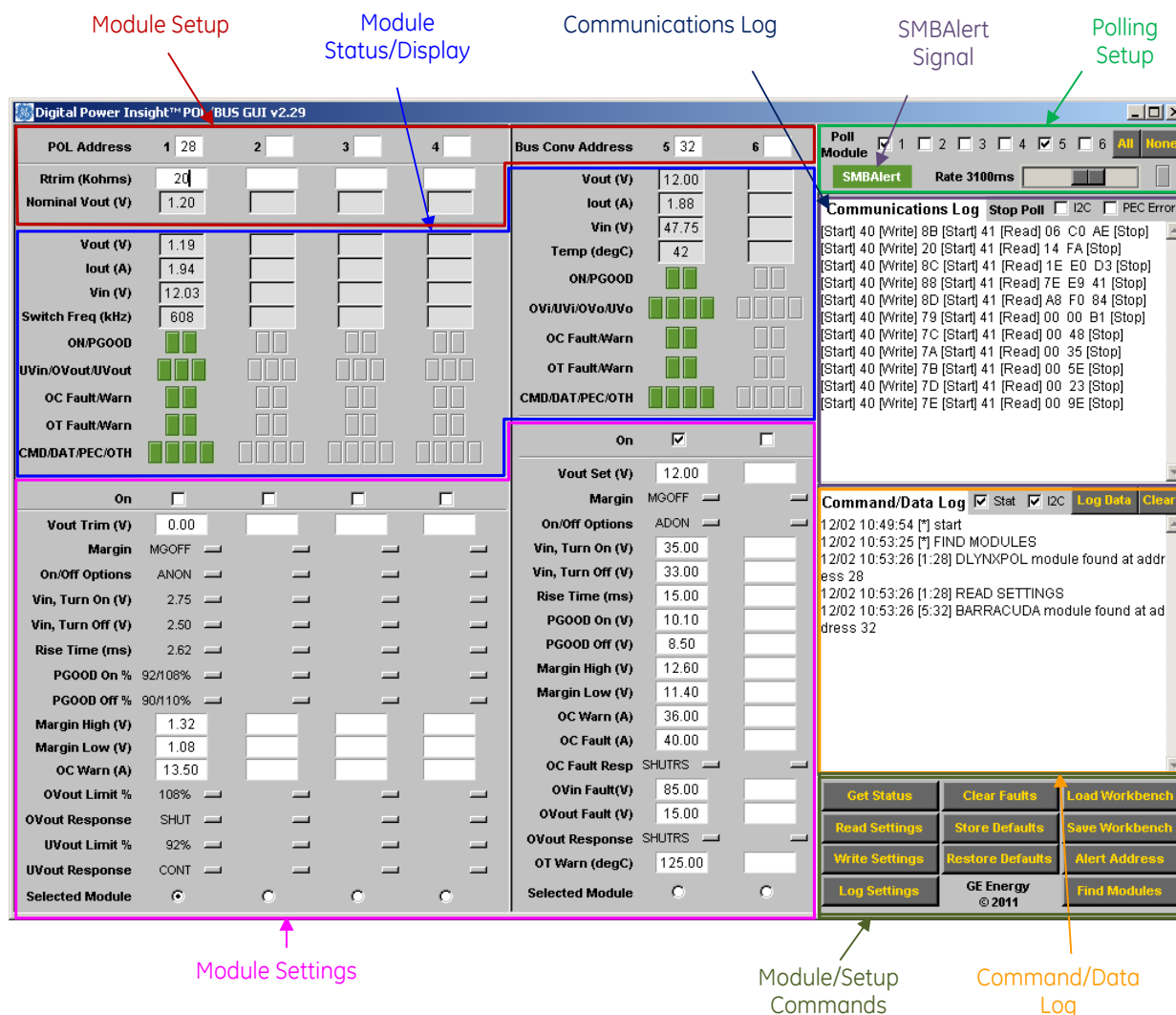
5.2.1   Module Setup

The DPI-GUI can simultaneously display up to six modules. In the default configuration, four POL modules(Dual DLynx not supported) are displayed on the left portion of the screen and two Isolated Digital Bus Converters on the right. In the POL section two important control parameters set external to the module need to be entered. The first is the module address which can range from 1d to 63d. The FIND MODULES function in the DPI-GUI tool (see 3.2.4) automatically finds the first four POLs and the first two bus converters with the lowest addresses connected to the PMBus. When this function is exercised, the module addresses are automatically filled in by the tool. In addition, for each module found the module settings are also obtained and filled in by the tool. If there are more than four POL modules connected to the PMBus, the user may need to manually overwrite the addresses selected automatically by the tool, and then use the Read Settings command to read in the specific module settings. All address information can be overwritten manually by the user if desired.

The second control parameter that must be entered by the user in the POL section is the value of RTrim (external Trim resistor) connected to each module. The value of the resistor is used to set the module parameter VOUT_SCALE_LOOP

which is used by each module to calculate the scaling of the output voltage. The Trim Resistor value in KΩ for each module is entered in the appropriate box. When the value is entered the nominal output voltage of the module is automatically displayed.

Please refer to the module data sheet for additional information on how the module address and RTrim are specified for the particular module.

Note: If an attempt is made to write settings from the GUI to the module without specifying the value of RTrim, the GUI displays a pop-up window, as shown below, asking for RTrim to be entered.

In the Isolated Digital Bus Converter section, the only control parameter set external to the module is the module address. Again, the FIND MODULES command finds the two Bus Converter modules with the lowest addresses and the module addresses are automatically filled in by the tool. If more than two Bus Converter modules are found, the user may need to manually overwrite the addresses selected automatically by the tool. the addresses of the first two are filled in automatically. Again all address information can be overwritten manually by the user if desired. Note that if a module address is overwritten by the user, the Read Settings command will need to be executed to get updated settings for the module.

5.2.2   Module Status and Display Area

This area of the tool displays the status of key module parameters as a combination of numerical values and LED-like status displays. Again this display area is divided into a section for POL converters and another for Isolated Digital Bus Converters. Most of the status display parameters are the same for both types of converters. The common numerical displays are for Vout (module output voltage), Iout (module output current), and Vin (module input voltage). POLs display the Switch Freq (module switching frequency) and Bus Converters display Temp (degC) (module temperature).

The module status is shown using groups of LED-like displays with green indicating normal operation and red indicating a fault or output OFF condition. The status displays are grouped as follows:

- ON/PGOOD : Shows if the module output is ON or OFF and the status of the PGOOD signal
- UVin/OVout/UVout : Shows if Input Undervoltage/Output Overvoltage/Output Undervoltage faults have occurred
- OC Fault/Warn : Shows if Output Current Fault/Warning levels have been exceeded
- OT Fault/Warn : Shows if Module Overtemperature Fault/Warning levels have been exceeded
- CMD/DAT/PEC/OTH : Shows if various communication faults have occurred.
  - CMD – invalid or unsupported command has been received.

- DATA – invalid or unsupported data has been received.
- PEC – Packet Error Check error. The instruction has not been executed.
- OTH – other communications fault.

5.2.3   Module Setup/Commands Area

This area of the tool provides a number of useful commands to communicate with the module as well as save and restore tool configurations and functions.

Get Status : Used to read a single set of parameter and status values from the selected module (module can be selected by clicking on one of the **Selected Module** buttons at the bottom of each column in the Module Settings area). When this command is activated by clicking on the button, the module status is refreshed in the tool, and the last display of readings is maintained.

Clear Faults : Module faults/warning status can be cleared by clicking on this button. Note that modules status will not be updated on the screen until Get Status is clicked to refresh the module status in the tool. An alternate approach is to poll the status of all modules periodically to maintain a continuously refreshed status display (see 3.2.5). If the fault/warning condition in the module persists the refreshed display will continue to indicate the fault/warning.

Load Workbench : Used to retrieve module configuration values from a user-specified file.

Read Settings : When this button is clicked, the tool obtains the settings from working memory (not non-volatile memory) of the selected module and updates the display in the Module Settings area. This can be convenient to pre-populate the selected module settings prior to making changes.

Store Defaults : Used to copy settings from the module's working memory to non-volatile memory.

Save Workbench : Used to save the module values (those entered in the Module Setup and Module Settings areas of the tool) to a user-specified file. This command along with the Load Config File command can be used to save module settings to a file and retrieve them from the file for later use.

Write Settings : Used to "write" settings from the DPI-GUI tool into the working memory of the module. All settings of the selected module in the GUI are then written to the selected module. Note that Write Settings does not update the non-volatile memory of the module.

Restore Defaults: Used to copy settings from non-volatile memory to working memory of the module.

Alert Address: Returns the address of the module that asserted the SMBAlert signal. If multiple modules assert the Alert line simultaneously, then the device with the lowest address is identified first. The module that responded to the request must clear its Alert line. Please note that the Barracuda Bus Converter's factory default configuration does not support the Alert Response Address (ARA). Please refer to the Barracuda data sheet for instructions on using the CLI tool and the MFR_CPIN_ARA_CONFIG command to change the ARA bit.

Log Settings : Obtains all settings of the working memory of the selected module, displays them in the Command/Data Log area of the tool and writes the settings into the data file created for this session. Note that **these** may not be the same as those displayed on the screen of the DPI-GUI. Settings in the DPI-GUI may be different from those in the module if they have not yet been written to the module by the user using DPI-GUI.

Find Modules: When this button is clicked, the tool will automatically find the POLs and Bus Converters with the lowest addresses, corresponding to the module configuration set up during startup of the DPI-GUI. If more modules are present on the bus they will be ignored. The command also retrieves the settings from each of the modules and populates the GUI with those values. An alternate to using this command is to enter module addresses manually by entering the address (in decimal) into the **Module (address)** display area. This allows the user to select any combination of up to four POLs and two Bus Converters they desire to communicate with.

5.2.4 Module Settings Area (POL)

This Module Settings area of the tool is divided into a POL and Bus Converter section. This section describes the key POL module parameters that can be viewed and changed. The module parameters to be changed are either entered as values or as choices in a drop-down menu activated by clicking the button shown in line with the parameter and selecting the desired option. Check the data sheet for module parameters that are entered as values to ensure that they are valid. Note that module parameters entered here must be "written" to the module using the Write Settings button described in 3.2.3 in order to take effect. A detailed description of how each module parameter can be set is given below.

On : Clicking the white box will insert a √ indicating that the module is to be turned ON. For this feature to actively control the ON/OFF of the module, the On/Off Options should be set to either DGON (Digital ON) or ADON (Analog or Digital ON). Note that clicking the module ON or OFF will immediately turn the module ON or OFF, i.e. there is no need to Write settings to the module. All other module settings become active in the module only when the Write Settings command is activated. This parameter is common to both POLs and Bus Converters.

Vout Trim (V) : The offset to the nominal output voltage value can be entered here. The module output voltage will then be changed the new value (Vout,nom + Vout Trim) when the parameters are "written" to the module. This parameter applies only to POL modules.

Margin : Clicking the button in line with this parameter opens a drop down menu as shown below
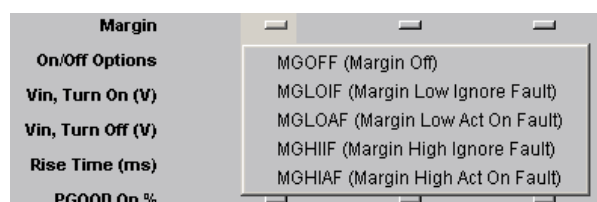
The drop down menu has five choices. For all dropdown menus, the pointer location is highlighted by the background. To select a particular choice, click on the highlighted area within the drop down menu. Both the mnemonic label and its meaning are shown (e.g. MGOFF for Margin Off).

On/Off Options : When this button is clicked, the parameter choices shown below are displayed.
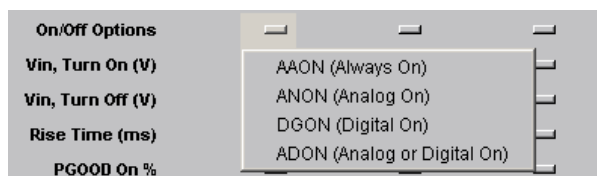
Vin, Turn On (V) and Vin,Turn Off (V): Clicking these buttons show the drop down menus with the choices shown below for POL Modules.

Rise Time (ms) : Clicking this button shows a drop down menu with the choices shown below for POL Modules.

PGOOD On % : Clicking this button shows a drop down menu with the choices shown below for POL Modules. The choices indicate combinations of two percentage values of the nominal output voltage – the first being the lower threshold while the second is the upper threshold.

PGOOD Off % : Clicking this button shows a drop down menu with the choices shown below for POL Modules. The choices indicate combinations of two percentage values of the nominal output voltage – the first being the lower threshold while the second is the upper threshold.

Margin High (V) : The margin high voltage of the module can be entered here. The module goes to the commanded margin high voltage when Margin High is enabled and the new settings are written to the module.

Margin Low (V) : The margin low voltage of the module can be entered here. The module goes to the commanded margin low voltage when Margin Low is enabled and the new settings are written to the module.

OC Warn (A) : Allows the Overcurrent Warning level to be specified for both POLs.

OVout Limit % : Clicking this button shows a drop down menu with the choices shown below for POL Modules. The user can select the Output Overvoltage protection threshold from the available choices expressed as a percentage of the set output voltage.

OVout Response : Clicking this button shows a drop down menu with the choices shown below for POL Modules. These choices control the module response in the event of an output overvoltage condition.

UVout Limit % : Clicking this button shows a drop down menu with the choices shown below for POL Modules. The user can select the Output Undervoltage protection threshold from the available choices expressed as a percentage of the set output voltage.

UVout Response : Clicking this button shows a drop down menu with the choices shown below for POL Modules. These choices control the module response in the event of an output undervoltage condition.

Selected Module : Allows the user to select the module that the tool communicates with using the commands in the Module Setup/Confuguration area. Note that these commands apply to only the selected module,, i.e. the tool communicates with only one module at a time.

5.2.5   Module Settings Area (Bus Converter)

Check the data sheet for all module parameters that are entered as values to ensure that they are valid.

On : Clicking the white box will insert a √ indicating that the module is to be turned ON. For this feature to actively control the ON/OFF of the module, the On/Off Options should be set to ADON (Analog and Digital ON). Note that clicking the module ON or OFF will immediately turn the module ON or OFF, i.e. there is no need to Write settings to the module. All other module settings become active in the module only when the Write Settings command is activated. This parameter is common to both POLs and Bus Converters.

Vout Set (V) : The output voltage value can be entered here. The module output voltage will then be changed to the new value when the parameters are "written" to the module. This parameter applies only to Bus Converters.

Margin : Clicking the button in line with this parameter opens a drop down menu as shown below
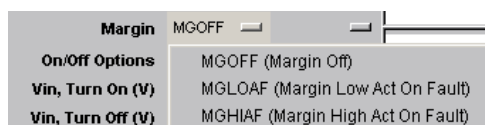
The drop down menu has three choices. For all dropdown menus the selected choice is identified by highlighting the background. Both the mnemonic label and its meaning are shown (e.g. MGOFF for Margin Off).

On/Off Options : When this button is clicked, the parameter choices shown below are displayed.
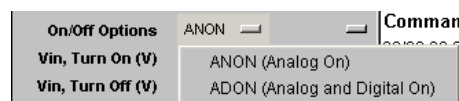
Vin, Turn On (V) and Vin,Turn Off (V): For Bus Converters, the Turn On and Turn Off input voltages are entered directly.

Rise Time (ms): For Bus Converters, the Rise Time in ms is entered directly.

PGOOD On (V) : For Bus Converters, the PGOOD On lower threshold in V is entered directly.

PGOOD Off (V) : For Bus Converters, the PGOOD Off lower threshold in V is entered directly.

Margin High (V) : The margin high voltage of the module can be entered here. The module goes to the commanded margin high voltage when Margin High is enabled and the new settings are written to the module.

Margin Low (V) : The margin low voltage of the module can be entered here. The module goes to the commanded margin low voltage when Margin Low is enabled and the new settings are written to the module.

OC Warn (A) : Allows the Overcurrent Warning level to be specified for Bus Converters.

OC Fault (A) : Allows the Overcurrent Shutdown level in A to be specified for Bus Converter Modules.

OC Fault Resp : Clicking this button displays the drop down menu with the choices shown below for Bus Converters.
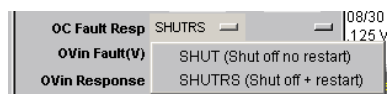
OVin Fault (V) : Allows the Input Overvoltage fault threshold in V to be specified for Bus Converter modules.

OVout Fault (V) : Allows the Output Overvoltage fault threshold in V to be specified for Bus Converter modules.

OVout Response : Clicking this button shows a drop down menu with the choices shown below for Bus Converters. These choices control the module response in the event of an output overvoltage condition.
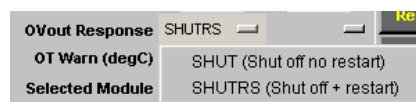
OT Warn (degC) : Allows the Module Temperature Warning threshold in °C to be specified for Bus Converter modules.

Selected Module : Allows the user to select the module that the tool communicates with using the commands in the Module Setup/Configuration area. Note that these commands apply to only the selected module,, i.e. the tool communicates with only one module at a time.

5.2.6   Polling Setup and Alert State Area

The DPI-GUI tool can be set up to repetitively poll up to six modules to retrieve module parameters and status information. The data is used to periodically update the Module Status/Display area.
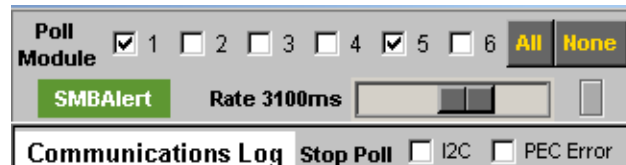
The modules to be polled can be selected along with the desired polling rate (from roughly once every 200ms up to once every 5000ms or 5s). The polling can also be set to stop on I2C or PEC (Packet Error Checking) errors. The LED button to the right of the polling rate is an indicator that blinks for every executed module poll.

The SMBAlert_State displays the status of the SMBAlert signal. The color of the display changes to RED when the SMBAlert signal is asserted and is GREEN when normal (not asserted).

5.2.7   Communications Log Area

This area of the DPI-GUI tool shows the I2C communications traffic in a simplified hex format. Note that what is displayed in this area is only a representation of the I2C traffic because the DPI-GUI tool has no direct access to the bus and therefore can only estimate what would be sent across the bus based on what is sent from the tool to and from the USB Interface Adapter. Each message string, between [Start] and [Stop], typically consists of the complete 'address[write]' byte containing the 'write' or 'read' LSB, followed by the 'write' command/instruction byte, the 'address[read]', one or two data bytes, and terminated by the Packet Error Checking (PEC) byte. If the command is only an instruction, without a read-back trailer, then the [Stop] would be placed after sending out the command/instruction byte followed by the PEC byte.

The log area clears automatically after a group of commands are executed.
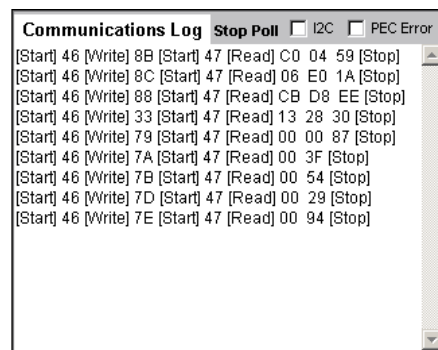
5.2.8   Command/Data Log area

This area displays the most recent commands and key data received or sent by the GUI. The DPI-GUI also creates a text file that automatically captures the information recorded in the Command/Data Log area. This file is located in the same directory that the executable program resides in. A new text file is created every time that the GUI is executed. The file name (in the form dpi-yyyymmdd-hhmm.txt) includes the date (year, month, day) and time (hours and minutes) when the executable started (and the log file was created). This file name can be changed later by the user just like any other file name in Windows. The intent of this Log file is to capture and time stamp the occurrence of all instructions and state changes of the modules being monitored or controlled by the GUI. Each individual message of instructions and state changes was carefully designed to allow the user to easily step through the executed set of events at a later date. These files remain in the computer until they get deleted by the user.

Within the Command/Data Log area the user may 'poll' modules and monitor their states automatically for days or even longer time durations. If the 'state' of any of the monitored devices should change, this change will be recorded and time stamped automatically in the text file for future review. The user may also perform a 'qualification verification test' on each module and use the log area for record keeping of the executed commands and of the behaviour of the module to the stimulated event. These are just two of the many potential uses for these records.

A detailed description of each executable command is listed below:

Log Data: Records a single set of data values that reside in the Module Status/Display registers

Clear: Erases the content in the Log window but does not erase the contents of the text file.

Status and I2C check boxes: Checking these boxes instructs the program which events should get recorded by the Command/Data Log routine. Normally both boxes should always be checked to get the most out of the automatic data recording capability of the GUI.

The Log area displays not only the commands in upper case letters but also all the data that has been changed or stored. In the example screen capture shown below the user starts with issuing the command FIND MODULES. The Log area shows the executed command in upper case and then displays the found modules, and their decimal addresses.

Two commands are executed next, READ SETTINGS and GET STATUS. Although not shown here, these two commands populated the Module Status/Display and Module Configuration areas for the module in location 1 with address 28. Next, LOG SETTINGS was instructed and the result displays all the settings that are in 'working memory' of the module. Some of the reported settings do not appear in the Module Configuration Settings area because they should not be changed by the user. Similarly, WRITE SETTINGS would display those configuration settings that have been changed followed by a single read of the input and output voltages and output current of the module.

In addition to normal events logging, the data log section captures all errors or problems and time stamps when they occurred if polling is executed. The tool also time stamps if a recovery to normal also occurred. This information is obtained by reading the status registers of the module and comparing the present read to the previous state of the status registers. Any difference is automatically recorded.
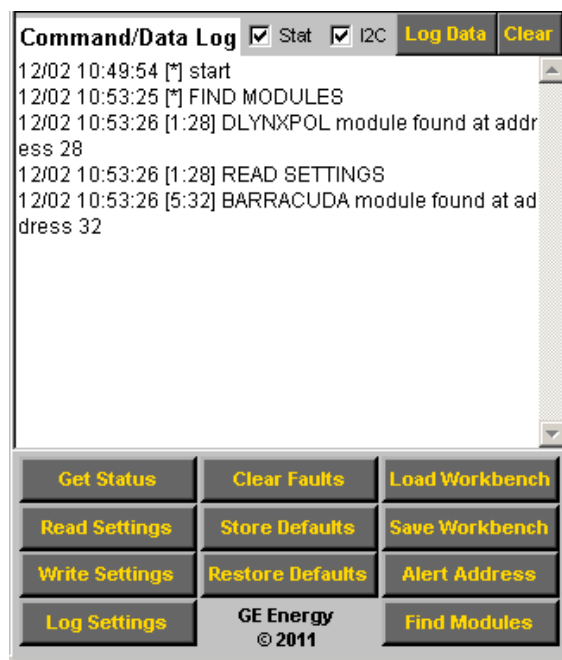
6 DPI-CLI (Command Line Interface)

The DPI-CLI is a low-level command-line based tool that is useful for decoding and analyzing basic communications with the module. The low level capability facilitates debugging when specific command interactions need to be examined in detail. The DPI-CLI also has a powerful scripting and data capture/logging capability that supports diagnostics-type debugging of a system with one or more modules.

6.1 Starting the DPI-CLI Tool

Once the USB Interface Adapter is plugged into the USB port on the Personal Computer, the Green LED of the Adapter should be ON. Apply input power to the module.

The DPI-CLI tool can now be started by double clicking on the DPI-CLI icon placed by the installation program on the desktop. The tool can also be started using the standard Windows *START – Programs – DPI – DPI CLI*.

The tool starts up and displays the following

```
Shortcut to dpi_cli                                                    _ □ ×
+---------+--------------------------+----------------------------------~
¦   GE    ¦ DPI Command Line Interface ¦ USB Port (4)  Module (28d 1Ch 34o)
¦ Energy  ¦ Ver 11/08/2011 01:05:00 PM ¦ Auto-finding module address.
+---------+--------------------------+----------------------------------~
1. 11:03:41 CMD [DATA] or HELP -->
```
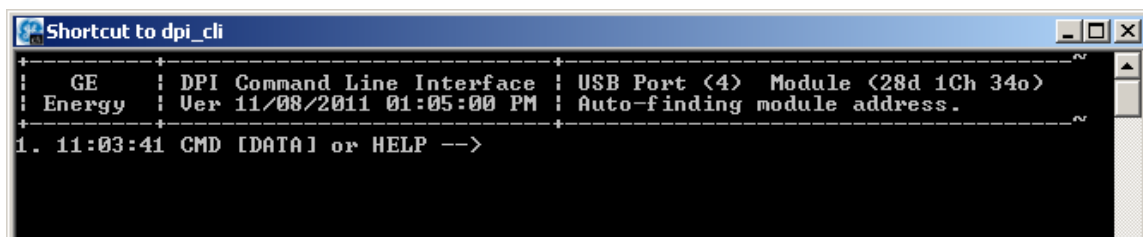
The tool automatically finds all modules with valid addresses connected to the I²C bus connected to the USB Interface adapter. When multiple modules are found, the tool will indicate that and ask the user to select the module to communicate with (only one module can be addressed at a time).

Once the module address is selected, the CLI command prompt is displayed and the tool is ready to communicate with the selected module.

6.2 DPI-CLI Functions

The DPI-CLI tool has a number of command line functions. The syntax of each function is shown in specific sub-sections of the manual. In general, the syntax consists of the function name and one or more optional arguments as follows:

FUNCTION <argument1> <argument2> ...

Below is a list of supported functions and their brief description:

- H or HELP – provides help within the program with a detailed list of functions, brief descriptions and examples of usage
- A or ALERT – reads the status of the SMBAlert Line
- C or COMMAND – a special function that is used to implement certain specific commands with no arguments
- D or DELAY – allows a specified delay to be inserted before the next function is executed
- G or GROUP – allows for the execution of multiple read commands in a comma separated list. Works only with text commands (cannot use the hex value of the command)
- I or INPUT – used to load a text file (.txt) containing a series of commands to be executed by DPI-CLI
- K – for displaying or changing the I²C clock rate between 100kHz (default) and 400kHz in the USB Interface Adapter

- L or LIVE – allows continuous, repetitive running of one or more commands
- M or MODULE – changes the module being addressed
- N or NOTE – supports insertion of notes for logging
- O or OUTPUT – allows for saving all results to a file as well as the screen
- P – for displaying or changing pull-up resistor values in the USB Interface Adapter
- Q or QUIT – exit the DPI-CLI tool
- R or READ – allows data to be extracted from the power module
- REGINFO – tabulates the supported functions and commands of DPI-CLI
- S or STOP – supports stopping the saving of results to a file
- SHOWALL – displays a summary of all commands within DPI-CLI
- SUPPRESS_X – used to abbreviate the display returned by the tool, X is either Y (for yes}, or N (for No)
- V – shows version of firmware in the USB Interface Adapter
- W or WRITE – allows data to be written to the module

All functions are case-insensitive, i.e. they can be entered as lower case, upper case or a mix of the two. Also note that except for the REGINFO and SHOWALL functions, it is not necessary to type the entire function name, just the first letter is sufficient.

REGINFO provides additional information such as the hex command byte corresponding to each command and the default value of each command. For example, trimming the output voltage up by 0.25 volts can be executed in a number of ways, each valid and equivalent (from REGINFO or SHOWALL the hex command equivalent of Vout_trim is 22h).

For example:

```
>WRITE VOUT_TRIM 0.25

>W VOUT_TRIM 0.25

>w vout_trim 0.25

>w 22 0.25
```

are all equivalent. Detailed instructions and examples for each of the functions follow.

6.2.1   H or HELP Function

This function can be used in two forms, with and without an argument. If no argument is used, the function returns a screen with the following:

- Examples of how the DPI-CLI tool can be invoked and the purpose of each command line argument
- Description of how module addresses can be specified (in hex, octal or decimal format)
- Examples of how to use functions
- How to exit the tool

The HELP function can also be used with an argument, where the argument is the specific function on which help is needed. For example,

```
>HELP ON_OFF_CONFIG or
>H 02
```

results in the DPI-CLI tool displaying help information specific to the ON_OFF_CONFIG PMBus command which also has the hex value 02.

6.2.2   A or ALERT Function

The ALERT or A function will read the SMBALERT line status. If the line is asserted it will return the value 'asserted' and if the line is not asserted the value 'normal' will be returned.

Example:

```
>ALERT
```

6.2.3   D or Delay Function

This function is used to insert a specified delay in seconds and is commonly used in a list with multiple functions. If the Delay function is used without any specified delay, the tool will pause until the user presses the <ENTER> key to continue.

Example:

```
>DELAY 1.2
```

results in a 1.2 second delay, before the next function is executed.

6.2.4   G or GROUP Function

This function allows multiple commands to be placed in a comma separated list for execution. Note that there should be no spaces between multiple commands.

Example:

```
>GROUP VOUT,VIN,IOUT
```

reads the output voltage, input voltage and output current and return all three values.

Comparison capability: The group function can also be used to compare analog data to specific limits and digital data to specific values. These comparison entries are entered in brackets starting with the nominal expected value followed by the upper permitted percent deviation and then the lower permitted percent deviation, all separated by commas. The comparison takes the form g [parameter](nominal,%high,%low)

EXAMPLE

```
>g vout(12,3,4),status_word(0000)
```

the output voltage is compared to +3% and -4% of 12V and the status_word register is compared to 0000h. If the register comparison fails the program identifies the discrepancy. Notice that there are no spaces between the arguments and the comparisons. The output of the command either simply returns the obtained reading or returns the reading within estruses' if the reading falls outside of the comparison window (i.e   * 11.40 *)

6.2.5   I or INPUT Function

This function is used to load a text file containing one or more functions. Only one function may appear in each line of the text file. This function supports a scripting capability where a sequence of multiple functions with numerous commands can be loaded and run using the DPI-CLI tool.

Example:

```
>INPUT test_sequence.txt
```

reads in the file **test_sequence.txt** and execute functions contained within the file.

6.2.6   K - Clock Setting Function

This function is used to read or change the I²C clock setting in the USB Interface Adapter. The default value is 100kHz but the clock rate can be changed to 400kHz. The desired clock rate can be specified as 100 or 100kHz or 400 or 400kHz. If this function is used without an argument, the current I²C clock rate in the USB Interface adapter is returned.

Examples:

```
>K 400Khz
```
(sets the USB Interface Adapter Clock speed to 400kHz)

```
>K 100
```
(sets the USB Interface Adapter Clock speed to 100kHz)

```
>K
```
(reads the USB Interface Adapter Clock speed setting)

6.2.7   L or LIVE Function

This function supports the continuous and repetitive running of one or more commands. The delay between the execution of a set of commands can also be specified in seconds, as well as the number of repetitions desired.

Example:

```
>L vin,vout 2.5 30
```

reads Vin and Vout from the module 30 times with a 2.5 second delay between each set of read operations (read Vin and Vout). Note that there should be no spaces between the entered commands (e.g. between vin and vout in the example above). but a space must be entered between the delay and the number of repetitions. If the number of repititions is not specified, the comparison continues indefinitely until it is stopped by using the DOS command (<CNTL>C).

6.2.8   M or MODULE Function

This function allows a different module to be addressed by specifying the new module address along with the function. The default address is assumed to be decimal, but hex or octal addresses are also supported if the letters 'h' or 'o' are placed before or after the address numbers.

Examples:

```
>module 25
```

```
>module 19h
```

```
>module 31o
```

are all equivalent and change the module addressed to 25 (decimal), 19 (hex) or 31 (octal).

6.2.9   N or NOTE Function

This function allows a note to be put out by the tool – primarily useful for logging purposes. If spaces are to be included, enclose the enter text string within single quotes.

Example:

```
>N 'this is a test'
```

places the text string "this is a text" in the output from the tool.

6.2.10 O or OUTPUT Function

This function supports saving results from the tool to a specified file as well as being displayed on the screen. The file using the specified name is saved in the same directory from where the DPI-CLI tool is executed. The STOP function is

used to stop saving results into the specified file, so the file can be used to capture everything put out by the tool from when the OUTPUT function is entered to when the STOP function is entered.

Example:

```
>OUTPUT testing_results.txt
```

places results from the DPI-CLI tool into the file named ***testing_results.txt*** located in the directory from where the DPI-CLI tool is run.

6.2.11 P Function

This function is used to set the clock and data line pull-up resistors in the USB Interface Adapter. Possible options are 3.3mA (default), 0.9mA, 0.44mA, or 0 (none). The desired pull-up current capability is specified in mA as an argument for the function. If no argument is specified, the current value is returned.

Examples:

```
>P 0.9
```

```
>P 3.3mA
```

```
>P
```

6.2.12 Q or QUIT Function

This function is used to exit the DPI-CLI tool.

6.2.13 R or READ Function

This function is used with a single argument which can be the PMBus command entered as a command string or a command byte in hex format.

Examples:

```
>READ ON_OFF_CONFIG
```

```
>R 02
```

The CLI tool responds to the READ function with a detailed response, an example of which is shown below with explanations.

```
>r status_word
```

returns with the following response from the tool:

```
17. 09:05:42 CMD [DATA] or HELP --> r status_word
{  'BINARY': '0000100001000010',
   'CMDADDR': '79',
   'CMDNAME': 'STATUS_WORD',
   'ERRORS': [],
   'FINAL': {  'CML': 'Y',
               'IOUT_OC': 'N',
               'IOUT_POUT': 'N',
               'ONOFF': 'OFF',
               'OTHER': 'N',
               'POWER_GOOD': 'N',
               'TEMPERATURE': 'N',
               'VIN_UV': 'N',
               'VOUT': 'N',
               'VOUT_OV': 'N'},
   'LEDS': {  'IOUT_OC': 'GREEN',
              'ONOFF': 'RED',
              'POWER_GOOD': 'RED',
              'STATUS_OTHER': 'GREEN',
              'VIN_UV': 'GREEN',
              'VOUT_OV': 'GREEN'},
   'MODADDR': 35,
   'RAW': /O=/OCC\r /D23=* /*T~79=/MTC\r /R03=/MRC~42~08~ce\r /C=/CCC\r '}
```

Digital bit representation of the data read back – high byte first

The command: hex byte representation and its meaning

Lists any syntax or transmission errors

Interpretation of all status

The device address in binary

The RAW data shows the communication between the CLI and the USB Translator. The very first transmission has the following instructions detailed: /O – open , /OCC – open command complete;, /H1 – hex bytes transmitted /D23 – device address in hex, /*T~ command is followed by a restart, do not issue a STOP followed by command 79 , /MTC –master transmit complete, /R03 – read back three bytes, /MRC – master receive complete, followed by the three data bytes. (~ separation between bytes). The last instruction is /C – close followed by /CCC – close command completed by the translator.

The PEC Packet Error Checking byte is the last byte following all addresses, commands and all returned data

The READ function also has enhanced features supporting testing operations that indicate whether the data returned is within specified limits (for analog variables) or matches expected values (for digital data). Note that in the examples that follow SUPPRESS_Y has been invoked to limit the display to data returned.

For reading back analog variables from the module, the READ command enhanced functionality can be invoked by using the following format:

READ [parameter] (nominal,%high,%low)

The command must have a space between the command and the parameter and an additional space between the parameter and the values being compared to in parenthesis. If the data is within the prescribed limits then the data is tabulated as it is read. If the data is not within limits than the data is going to appear as ** value **, with the asterisk (*) identifying that the data is out of limits.

Example:

```
>READ VOUT (12,5,5)
```

will compare the data being read back to ensure that it is between 12.6V [12 x 1.05] and 11.4 [12 x .95] . If the data is within limits it will print as 12.00. If the data is out of limits it will print for example as ** 12.65 **.

For reading back digital variables from the module, the READ command enhanced functionality is invoked using the following format:

READ [parameter] (XXXXXX)

The command must have a space between the command and the parameter and another space between the parameter and the contents in parenthesis. The contents in parenthesis (XXXXXX) contain the expected value of the binary data string being read back in hex format. The data entry must be in most significant byte – least significant byte (MSB-LSB) format. If the binary data string does not match the anticipated value the program will display results placing them within asterisks (**).

Example;

```
>READ STATUS_WORD (0000)
```

compares the register data being returned to the value 0000h. If the returned value is 0002h, the error is highlighted within the asterisks and will look as follows:

```
"STATUS_WORD | {'': '**  EXPECTED :  0x0000 – RETURNED : 0x0800  **',
'POWER_GOOD': 'N', 'BUSY': 'N', 'TEMPERATURE': 'N', 'VOUT': 'N', 'MFR_SPECIFIC':
'N', 'UNKNOWN': 'N', 'OTHER': 'N', 'IOUT_OC': 'N', 'FANS': 'N', 'CML': 'N',
'INPUT': 'N', 'VIN_UV': 'N', 'IOUT_POUT': 'N', 'VOUT_OV': 'N', 'ONOFF': 'ON'}"
```

A special READ function queries the module whether it asserted the SMBAlert line. Address 12 decimal is used for this function. This address should not be used as the module ID Address because that would conflict with this query function. The READ takes the following form:

Example;

```
>READ ALERT_ADDRESS
>R A
>r a
```

6.2.14 REGINFO Function

This command is used with no arguments and provides a compact listing of all supported PMBus commands.

Example:

```
>REGINFO
```

6.2.15 S or STOP Function

This command is used along with the OUTPUT command to stop saving results from DPI-CLI to a file.

Example:

```
>STOP
```

6.2.16 SHOWALL Function

This function displays all commands with initial default values, the default bit string and the interpreted meaning of the command. An example of the command with a partial view of the response is shown here.

```
05/18/2010 12:38:54 --- R/W/C/H CMD [WDATA] --> showall
| HX | PARAMETER        | CURRENT BINARY   | INTERPRETED RESULT
| 01 | OPERATION        | 00000000         | {'On': 'N', 'Margin': 'Off'}
```

```
| 02 | ON_OFF_CONFIG     | 00010111         | {'CPA': 'Y', 'CPR': 'Y', 'CMD': 'N',
'POL': 'Y', 'PU': 'Y'}
| 10 | WRITE_PROTECT     | 00000000         | {'ALLOW': 'ALL'}
| 20 | VOUT_MODE         | 00010110         | {'VOUTEXP': -10, 'VOUTMODE': 0}
| 22 | VOUT_TRIM         | 0000000000000000 | 0.0
```

6.2.17 SUPPRESS_Y or SUPPRESS_N Function

Instructs DPI-CLI to suppress or turn back display of some of the message content in order to reduce the amount of information being displayed and saved. The SUPPRESS_Y function will suppress part of the displayed content while SUPPRESS_N will turn back on the full display.

For example a live scan instruction to read the output voltage will display the following:

```
{  'ALIAS': 'VOUT',
   'BINARY': '1100000000010111',
   'CMDBYTE': '8B',
   'CMDNAME': 'READ_VOUT',
   'DATA': '12.006',
   'ERRORS': [],
   'MODADDR': '28d',
   'RAW': '/O=/OCC\r   /D1C=*   /*T~8B=/MTC\r   /R03=/MRC~17~c0~5d\r   /C=/CCC\r   '}
>SUPPRESS_Y
```

When the SUPPRESS_Y function has been exercised, the same instruction will display as follows:

```
VOUT | 12.006
```

The user can invoke this command multiple times if desired during a scripting program.

6.2.18 V or Version Function

This function reads back the software (firmware) revision of the USB Interface Adapter.

Example:

```
>V              returns the firmware revision of the software in the USB Interface Adapter.
```

6.2.19 W or WRITE Function

The WRITE function can have multiple syntaxes. The basic syntax consists of a single argument and data value, as follows:

WRITE <argument> <value>

For example,

```
>WRITE TON_RISE 2.4        commands the module to set the Rise Time to 2.4ms.
```

The WRITE function can also execute successive arguments in a single command line with the following syntax:

WRITE <argument> <name1:value1> <name2:value2> <name3:value3> ...

For example,

```
>WRITE OPERATION ON:Y,Margin:ActHigh
```

```
>W 01 on:y,margin:acthigh
```

commands the module to turn ON and follow with the output margined high.

6.3   Summary of Supported PMBus Commands for Single Output DLynx POL Converters

The first command to be implemented should be to change the VOUT_SCALE_LOOP (29h) value to match the output voltage set using the trim resistor. Failure to do so will result in incorrect responses from the module

Command	Code	Capability	Name	Value	Function
OPERATION	01h	r/w	On	Y	Turns Module ON
Example: `> r OPERATION or >r 01`				N	Turns Module OFF
`> w 01 On:Y,Margin:OFF`			Margin	OFF	Output Margining is OFF
				IgnoreLow	Margin Low Ignoring any Faults
Note: entry required only for the changing function				ActLow	Margin Low but act on any Faults
				IgnoreHigh	Margin High Ignoring any Faults
				ActHigh	Margin High but act on any Faults
ON_OFF_CONFIG	02h	r/w	Pu	Y	Power up is contingent on the settings below
Example: `> r 02`				N	Unit powers up irrespective of the settings below
`> w 02 Pu:Y,Cmd:Y,Cpr:N`			Cmd	Y	Unit powers up with the ON bit of OPERATION
Note: entry required only for the changing function				N	Unit ignores the ON/OFF state of OPERATION
			Cpr	Y	Unit responds to the state of the CNTL pin
				N	Unit ignores the state of the CNTL pin
			Pol	Y	Unit responds to CNTL active HI
				N	Unit responds to CNTL active LO
			Cpa	Y	No turn-OFF delay, not user programmable
CLEAR_FAULTS **Example:** `>w 03`	03h	w			Clears the Status registers and resets the SMBAlert# signal.
WRITE_PROTECT	10h	r/w	Allow	All	Enable all writes
Example: `> r 10`				WP	Enable only WRITE_PROTECT command
`> w 10 Allow:WP_OP_CFG`				WP_OP	Enable only WRITE_PROTECT and OPERATION commands
				WP_OP_CFG	Enable only WRITE_PROTECT, OPERATION and ON_OFF_CONFIG commands
STORE_DEFAULT_ALL **Example:** `> w 11`	11h	w	.		Store all the configuration settings into non-volatile memory
RESTORE_DEFAULT_ALL **Example:** `> w 12`	12h	w			Get all configuration settings from non-volatile memory
STORE_DEFAULT_CODE **Example:** `> w 13 29`	13h	w			Store parameters associated with specified command into non-volatile memory
RESTORE_DEFAULT_CODE **Example:** `> w 14 29`	14h	w	.		Restore from non-volatile memory parameters associated with specified command
VOUT_MODE **Example:** `> r 20`	20h	r	.	Byte	Returns mode and exponent for output voltage related commands
VOUT_TRIM **Example:** `> r 22` `> w 22 0.15`	22h	r/w		Real Number	Read/write value of the desired offset in output voltage
VOUT_MARGIN_HIGH **Example:** `> r 25` `> w 25 1.33`	25h	r/w		Real Number	Read/write value of the desired output voltage margin high level
VOUT_MARGIN_LOW **Example:** `> w 26 1.10`	26h	r/w		Real Number	Read/write value of the desired output voltage margin low level
VOUT_SCALE_LOOP **Example:** `> r 29` `> w 29 0.6`	29h	r/w		Real Number	Read/write value of the scaling factor (divider ratio) for output voltage. Calculate using 0.6/Vout

Command	Code	Capability	Name	Value	Function
VIN_ON **Example:** `> w 35 3.0` `> r 35`	35h	r/w		Real Number	Read/write value of the input voltage turn-ON level. Values are 2.75V, 3V, 3.5V, in increments of 0.5V until 18.0
VIN_OFF **Example:** `> r 36` `> w 36 2.8`	36h	r/w		Real Number	Read/write value of the input voltage turn-OFF level. Values are 2.5V, 3V, 3.5V, in increments of 0.5V until 17.5
IOUT_CAL_GAIN **Example:** `> r 38`	38h	r		Real Number	Read scaling factor for output current sense signal
IOUT_CAL_OFFSET **Example:** `> r 39`	39h	r		Real Number	Read offset for output current sense signal
VOUT_OV_FAULT_LIMIT **Example:** `> r 40` `> w 40 1.35`	40h	r/w		Real Number	Read/write value of the output overvoltage fault level. There are only 4 acceptable values which are 108%, 110%, 112% and 115% of Vout
VOUT_OV_FAULT_RESPONSE	41h	r/w	RSP.	Continue	Set module to continue without interruption on output voltage OV fault
Example: `> r 41` `> w 41 RSP:Shutdown RS:TryRestart` Note: entry required only for the changing function				Shutafter4	Set module to continue for four cycles on output overvoltage fault
				Shutdown	Set module to shutdown on output overvoltage fault
				Cutoutput	Set module to cut off output on output overvoltage fault and remain OFF until fault is cleared
			RS	Norestart	Set module to not restart on output overvoltage fault
				Tryrestart	Set module to restart on output overvoltage fault
VOUT_UV_FAULT_LIMIT **Example:** `> r 44` `> w 44 1.12`	44h	r/w		Real Number	Read/write value of the output undervoltage fault level. There are only 4 acceptable values which are 92%, 90%, 88% and 85% of Vout
VOUT_UV_FAULT_RESPONSE	45h	r/w	RSP.	Continue	Set module to continue without interruption on UV fault
Example: `> r 45` `> w 45 RSP:Shutdown RS:TryRestart` Note: entry required only for the changing function				Shutafter4	Set module to continue for four cycles on output voltage UV fault
				Shutdown	Set module to shutdown on output voltage UV fault
				Cutoutput	Set module to cut off output on output voltage UV fault and remain OFF until fault is cleared
			RS	Norestart	Set module to not restart on output voltage UV fault
				Tryrestart	Set module to restart on output voltage UV fault
IOUT_OC_FAULT_LIMIT **Example:** `> r 46`	46h	r	.	Real Number	Read value of the output overcurrent fault level
IOUT_OC_WARN_LIMIT **Example:** `> r 4A`	4Ah	r/w		Real Number	Read/write value of the output overcurrent warning level. The range of acceptable values is 0A to 63.5A in 0.5A steps.
OT_FAULT_RESPONSE	50h	r/w	OTF_RS	NoRestart	Set module to not restart on overtemperature fault
Example: `> r 50` `> w 50 OTF_RS:Restart`				Restart	Set module to restart on overtemperature fault

Command	Code	Capability	Name	Value	Function
POWER_GOOD_ON(PGON) **Example:** >r 5e 　　　　>w 5E 2.5	5Eh	r/w		Real Number	Value of the Power Good On threshold. There are only 3 acceptable low side values; 95%, 92%, 90% of Vout. Once the low side value is set, the corresponding high side value of either 105%, 108% or 110% is automatically set by the module. PGON should be higher than PGOFF
POWER_GOOD_OFF (PGOFF) **Example:** >r 5F 　　　　>w 5f 2.5	5Fh	r/w		Real Number	Value of the Power Good OFF threshold. There are only 3 acceptable low side values; 92%, 90%, 88% of Vout. Once the low side value is set, the corresponding high side value of either 108%, 110% or 112% is automatically set by the module PGOFF should be lower than PGON
TON_RISE **Example:** >r 61 　　　　>w 61 2.4	61h	r/w		Real Number	Value of the rise time of the output. The values available are 0.6, 0.9, 1.2, 1.8, 2.7, 4.2, 6.0 and 9.0msec
STATUS_BYTE **Example:** >r 78	78h	r			Returns byte with module status
STATUS_WORD **Example:** >r 79	79h	r		Word	Returns two bytes with module status
STATUS_VOUT **Example:** >r 7A	7Ah	r			Returns byte with module output voltage related faults
STATUS_IOUT **Example:** >r 7A	7Bh	r			Returns byte with module output current related faults
STATUS_TEMPERATURE **Example:** >r 7D	7Dh	r			Returns byte with module temperature related faults
STATUS_CML **Example:** >r 7E	7Eh	r			Returns byte with module communication related faults
READ_VIN **Example:** >r 88	88h	r		Real Number	Returns value of the input voltage
READ_VOUT **Example:** >r 8B	8Bh	r		Real Number	Returns value of the output voltage
READ_IOUT **Example:** >r 8C	8Ch	r		Real Number	Returns value of the output current
PMBUS_REVISION **Example:** >r 98	98h	r			Returns byte with PMBus version number that module complies with
MFR_VIN_MIN **Example:** >r A0	A0h	r		Real Number	Returns value of the minimum input voltage module supports
MFR_VIN_MAX **Example:** >r A1	A1h	r		Real Number	Returns value of the maximum input voltage module supports
MFR_VOUT_MIN **Example:** >r A4	A4h	r		Real Number	Returns value of the minimum output voltage module supports
MFR_VOUT_MAX **Example:** >r A5	A5h	r		Real Number	Returns value of the maximum output voltage module supports
MFR_SPECIFIC_00 **Example:** >r D0	D0h	r		Real Number	Returns two bytes of module specific information
VOUT_CAL_OFFSET **Example:** >r D4 　　　　>w D4 0.15	D4h	r/w		Real Number	Read/write offset correction for output voltage measurement
VOUT_CAL_GAIN **Example:** >r D5 　　　　>w D5 0.02	D5h	r/w		Real Number	Read/write gain factor correction for output voltage measurement
VIN_CAL_OFFSET **Example:** >r D6 　　　　>w D6 -0.2	D6h	r/w		Real Number	Read/write offset correction for input voltage measurement

Command	Code	Capability	Name	Value	Function
VIN_CAL_GAIN **Example:** `>r D7` `>w D7 -0.03`	D7h	r/w		Real Number	Read/write gain factor correction for input voltage measurement
ALERT_STATUS **Example:** `>a`	n/a	r		Word	Returns the SMBALERT status of the module(s). If the SMBALERT is active then the CLI reads back the status as 'Asserted".
ALERT_ADDRESS **Example:** `>a s`	n/a	r		Word	Returns the address of the module that pulled down the SMBAlert line. If there are multiple PMBus modules with alerts on the bus, the lowest address that has triggered the Alert line is returned. By using the ALERT_STATUS command again the first module releases the alert line. When the ALERT_ADDRESS command is executed again the next module with the lowest address that is still pulling down the Alert line is identified. This process is repeated to find all modules with alerts till the ALERT_STATUS is 'Normal'

6.4   Summary of Supported PMBus Commands for Dual Output DLynx POL Converters

Command	Code	Capability	Name	Value	Function
PAGE	00h	r/w	Output	First	All commands address the first channel
Example: > w 00 Output:Both				Second	All commands address the second channel
				Both	All commands address both the channels
OPERATION	01h	r/w	On	Y	Turns Module ON
Example: > r OPERATION or >r 01				N	Turns Module OFF
> w 01 On:Y,Margin:OFF			Margin	OFF	Output Margining is OFF
				IgnoreLow	Margin Low Ignoring any Faults
Note: entry required only for the changing function				IgnoreHigh	Margin High Ignoring any Faults
ON_OFF_CONFIG	02h	r/w	Pu	Y	Power up is contingent on the settings below
Example: > r 02				N	Unit powers up irrespective of the settings below
> w 02 Pu:Y,Cmd:Y,Cpr:N			Cmd	Y	Unit powers up with the ON bit of OPERATION
				N	Unit ignores the ON/OFF state of OPERATION
Note: entry required only for the			Cpr	Y	Unit responds to the state of the CNTL pin
changing function				N	Unit ignores the state of the CNTL pin
			Pol	Y	Unit responds to CNTL active HI
				N	Unit responds to CNTL active LO
			Cpa	Y	No turn-OFF delay, not user programmable
CLEAR_FAULTS **Example:** >w 03	03h	w			Clears the Status registers and resets the SMBAlert# signal.
WRITE_PROTECT	10h	r/w	Allow	All	Enable all writes
Example: > r 10				WP	Enable only WRITE_PROTECT command
> w 10 Allow:WP_OP_CFG				WP_OP	Enable only WRITE_PROTECT, PAGE and OPERATION commands
				WP_OP_CFG	Enable only WRITE_PROTECT, PAGE, OPERATION and ON_OFF_CONFIG commands
STORE_USER_ALL **Example:** > w 15	15h	w	.		Store all the configuration settings into non-volatile memory
RESTORE_USER_ALL **Example:** > w 16	16h	w			Get all configuration settings from non-volatile memory
CAPABILITY	19h	r	MAXBUS	400kHz	400 kHz PMBus Speed
Example: > r 19				100kHz	100 kHz PMBus Speed
			PEC	Supported	Packet Error Checking Supported
				Not_Supported	Packet Error Checking Not Supported
			SMB	SMB_PIN	Device has SMBALERT# Pin and supports SMBus Alert Response Protocol
				NoSMB_PIN	Device does not have SMBALERT# Pin and does not support SMBus Alert Response Protocol
VOUT_MODE **Example:** > r 20	20h	r	.	Byte	Returns mode and exponent for output voltage related commands
VIN_ON **Example:** > w 35 5.0 > r 35	35h	r/w		Real Number	Read/write value of the input voltage turn-ON level
VIN_OFF **Example:** > r 36 > w 36 4.75	36h	r/w		Real Number	Read/write value of the input voltage turn-OFF level
IOUT_CAL_GAIN **Example:** > r 38	38h	r		Real Number	Read scaling factor for output current sense signal

Command	Code	Capability	Name	Value	Function
IOUT_CAL_OFFSET **Example:** `> r 39`	39h	r		Real Number	Read offset for output current sense signal
IOUT_OC_FAULT_LIMIT **Example:** `> r 46`	46h	r	.	Real Number	Read value of the output overcurrent fault level
VOUT_UV_FAULT_RESPONSE	47h	r/w	RS	Norestart	Set module to not restart on output voltage UV fault
				Tryrestart	Set module to restart on output voltage UV fault continuously
IOUT_OC_WARN_LIMIT **Example:** `> r 4A`	4Ah	r/w		Real Number	Read/write value of the output overcurrent warning level
OT_FAULT_LIMIT **Example:** `> r 4A`	4Fh	r/w		Real Number	Read/write value of the output overtemperature fault level
OT_WARN_LIMIT **Example:** `> r 4A`	51h	r/w		Real Number	Read/write value of the output overcurrent warning level
TON_RISE **Example:** `>r 61` `>w 61 2.4`	61h	r/w		Real Number	Value of the rise time of the output
STATUS_BYTE **Example:** `>r 78`	78h	r			Returns byte with module status
STATUS_WORD **Example:** `>r 79`	79h	r		Word	Returns two bytes with module status
STATUS_VOUT **Example:** `>r 7A`	7Ah	r			Returns byte with module output voltage related faults
STATUS_TEMPERATURE **Example:** `>r 7D`	7Dh	r			Returns byte with module temperature related faults
STATUS_CML **Example:** `>r 7E`	7Eh	r			Returns byte with module communication related faults
STATUS_MFR_SPECIFIC_00 **Example:** `>r 80`	80h	r			Returns byte with manufacturer specific information
READ_VOUT **Example:** `>r 8B`	8Bh	r		Real Number	Returns value of the output voltage
READ_IOUT **Example:** `>r 8C`	8Ch	r		Real Number	Returns value of the output current
READ_TEMPERATURE_2 **Example:** `>r 8E`	8Eh	r		Real Number	Returns value of the module external temperature
PMBUS_REVISION **Example:** `>r 98`	98h	r			Returns byte with PMBus version number that module complies with
MFR_SPECIFIC_00 **Example:** `>r D0`	D0h	r		Real Number	Returns two bytes of module specific information
VREF_TRIM **Example:** `>r D4` `>w D4 -20`	D4h	r/w		Real Number	Read/write offset correction for output voltage measurement. Values range from -20 to 10
STEP_VREF_MARGIN_HIGH **Example:** `>r D5` `>w D5 10`	D5h	r/w		Real Number	Read/write offset correction for output voltage measurement. Values range from 0 to 10
STEP_VREF_MARGIN_LOW **Example:** `>r D6` `>w D6 -20`	D6h	r/w		Real Number	Read/write offset correction for output voltage measurement. Values range from -20 to 0

Command	Code	Capability	Name	Value	Function
PCT_VOUT_FAULT_PG_LIMIT	D7h	r/w	SETTING	1	UV -16.67%, PGL Low -12.5%, PGL High -8.33%, PGH High 12.5%, PGH Low 8.33%, OV 16.67%
Example: >r D7 >w D7 SETTING:1				2	UV -12.5%, PGL Low -8.33%, PGL High -4.17%, PGH High 8.33%, PGH Low 4.17%, OV 12.50%
				3	UV -29.17%, PGL Low -20.83%, PGL High -16.67%, PGH High 8.33%, PGH Low 4.17%, OV 12.50%
				4	UV -41.67%, PGL Low -37.5%, PGL High -33.33%, PGH High 8.33%, PGH Low 4.17%, OV 12.50%
SEQUENCE_TON_TOFF_DELAY	D8h	r/w	**TOFF_DE LAY**	0 to 7	Delay Time from when the module is disabled until it stops switching
Example: >r D8 >w D8 TON_DELAY:6			**TON_DE LAY**	O to7	Delay Time from when the module is enable until it soft-start begins
DEVICE_CODE **Example:** >r FC	FCh	r			Returns one byte of module specific information
ALERT_STATUS **Example:** >a	n/a	r		Integer	Returns the SMBALERT status of the module. If the SMBALERT is active then the CLI reads back the status as 'Asserted".
ALERT_ADDRESS **Example:** >a s	n/a	r		Integer	Returns the address of the module that pulled down the SMBAlert line. If there are multiple PMBus modules with alerts on the bus, the lowest address that has triggered the Alert line is returned. By using the ALERT_STATUS command again the first module releases the alert line. When the ALERT_ADDRESS command is executed again, the next module with the lowest address that is still pulling down the Alert line is identified. This process is repeated to find all modules with alerts till the ALERT_STATUS is 'Normal'

6.5   Summary of Supported PMBus Commands for GigaDLynx (GDT080) POL Converters

Command	Code	Capability	Name	Value	Function
OPERATION **Example:** `> r OPERATION or >r 01` `> w 01 On:Y,Margin:OFF` Note: entry required only for the changing function	01h	r/w	On	Y	Turns Module ON
				N	Turns Module OFF
			Margin	OFF	Output Margining is OFF
				IgnoreLow	Margin Low Ignoring any Faults
				ActLow	Margin Low but act on any Faults
				IgnoreHigh	Margin High Ignoring any Faults
				ActHigh	Margin High but act on any Faults
ON_OFF_CONFIG **Example:** `> r 02` `> w 02 Pu:Y,Cmd:Y,Cpr:N` Note: entry required only for the changing function	02h	r/w	Pu	Y	Power up is contingent on the settings below
				N	Unit powers up irrespective of the settings below
			Cmd	Y	Unit powers up with the ON bit of OPERATION
				N	Unit ignores the ON/OFF state of OPERATION
			Cpr	Y	Unit responds to the state of the CNTL pin
				N	Unit ignores the state of the CNTL pin
			Pol	Y	Unit responds to CNTL active HI
				N	Unit responds to CNTL active LO
			Cpa	Y	No turn-OFF delay, not user programmable
CLEAR_FAULTS **Example:** `>w 03`	03h	w			Clears the Status registers and resets the SMBAlert# signal.
WRITE_PROTECT **Example:** `> r 10` `> w 10 Allow:WP_OP_CFG`	10h	r/w	Allow	All	Enable all writes
				WP	Enable only WRITE_PROTECT command
				WP_OP	Enable only WRITE_PROTECT and OPERATION commands
				WP_OP_CFG	Enable only WRITE_PROTECT, OPERATION and ON_OFF_CONFIG commands
STORE_DEFAULT_ALL **Example:** `> w 11`	11h	w	.		Store all the configuration settings into non-volatile memory
RESTORE_DEFAULT_ALL **Example:** `> w 12`	12h	w			Get all configuration settings from non-volatile memory
STORE_DEFAULT_CODE **Example:** `> w 13 29`	13h	w			Store parameters associated with specified command into non-volatile memory
RESTORE_DEFAULT_CODE **Example:** `> w 14 29`	14h	w	.		Restore from non-volatile memory parameters associated with specified command
VOUT_MODE **Example:** `> r 20`	20h	r	.	Byte	Returns mode and exponent for output voltage related commands
VOUT_TRIM **Example:** `> r 22` `> w 22 0.15`	22h	r/w		Real Number	Read/write value of the desired offset in output voltage
VOUT_MARGIN_HIGH **Example:** `> r 25` `> w 25 1.33`	25h	r/w		Real Number	Read/write value of the desired output voltage margin high level
VOUT_MARGIN_LOW **Example:** `> w 26 1.10`	26h	r/w		Real Number	Read/write value of the desired output voltage margin low level
VOUT_SCALE_MONITOR **Example:** `> r 2A` `> w 2A`	2Ah	r/w		Real Number	Read/write value of the scaling factor (divider ratio) for output voltage. Calculate using 0.6/Vout.

Command	Code	Capability	Name	Value	Function
VIN_ON **Example:** `> w 35 3.0` `> r 35`	35h	r/w		Real Number	Read/write value of the input voltage turn-ON level. Values are in increments of 0.7mV until 14.8V
VIN_OFF **Example:** `> r 36` `> w 36 2.8`	36h	r/w		Real Number	Read/write value of the input voltage turn-OFF level. Values are 3.5V, in increments of 0.7mV until 14.8V
IOUT_CAL_GAIN **Example:** `> r 38`	38h	r		Real Number	Read scaling factor for output current sense signal. Factory Calibrated and not allowed to change
IOUT_CAL_OFFSET **Example:** `> r 39`	39h	r		Real Number	Read offset for output current sense signal. Factory calibrated and not allowed to change.
VOUT_OV_FAULT_LIMIT **Example:** `> r 40` `> w 40 125`	40h	r/w		Real Number	Read/write value of the output overvoltage fault level. Depends on VOUT_SCALE_MONITOR Value. Set in percentages, in increments of 0.1%.

VOUT_OV_FAULT_RESPONSE
 	
Example: `> r 41`
`> w 41 RSP:Shutdown RS:TryRestart`
 	
`Note: entry required only for the changing function`
	41h	r/w	RSP.	Continue	Set module to continue without interruption on output voltage OV fault

				Shutafter4	Set module to continue for four cycles on output overvoltage fault
				Shutdown	Set module to shut down on output overvoltage fault
				Cutoutput	Set module to cut off output on output overvoltage fault and remain OFF until fault is cleared
			RS	Norestart	Set module to not restart on output overvoltage fault
				Tryrestart	Set module to restart on output overvoltage fault
VOUT_OV_WARN_LIMIT	42h	r/w		Real Number	Read/write value of the output over-voltage warning level. Depends on VOUT_SCALE_MONITOR value. Set in percentages, in increments of 0.1%.
VOUT_UV_WARN_LIMIT	43h	r/w		Real Number	Read/write value of the output under-voltage warning level. Depends on VOUT_SCALE_MONITOR value. Set in percentages, in increments of 0.1%.
VOUT_UV_FAULT_LIMIT **Example:** `> r 44` `> w 44 1.12`	44h	r/w		Real Number	Read/write value of the output undervoltage fault level. Depends on VOUT_SCALE_MONITOR Value. Set in percentages, in increments of 0.1%.

VOUT_UV_FAULT_RESPONSE
 	
Example: `> r 45`
`> w 45 RSP:Shutdown RS:TryRestart`
 	
`Note: entry required only for the changing function`
	45h	r/w	RSP.	Continue	Set module to continue without interruption on UV fault

				Shutafter4	Set module to continue for four cycles on output voltage UV fault
				Shutdown	Set module to shut down on output voltage UV fault
				Cutoutput	Set module to cut off output on output voltage UV fault and remain OFF until fault is cleared
			RS	Norestart	Set module to not restart on output voltage UV fault
				Tryrestart	Set module to restart on output voltage UV fault
IOUT_OC_FAULT_LIMIT **Example:** `> r 46`	46h	r	.	Real Number	Read value of the output overcurrent fault level. Minimum programmable value is IOUT_OC_WARN_LIMIT.

IOUT_OC_WARN_LIMIT **Example:** > r 4A	4Ah	r/w			Real Number	Read/write value of the output overcurrent warning level. The range of acceptable values is 0A to 80A in 0.2A step
OT_FAULT_LIMIT	4Fh	r/w			Real Number	Read/write value of the temperature threshold for over-temperature protection.
OT_FAULT_RESPONSE **Example:** > r 50 > w 50 OTF_RS:Restart	50h	r/w	OTF_RS	NoRestart	Set module to not restart on overtemperature fault
				Restart	Set module to restart on overtemperature fault
OT_FAULT_LIMIT	51h	r/w			Real Number	Read/write value of the temperature threshold for over-temperature
VIN_OV_FAULT_LIMIT	55h	r/w			Real Number	Read/write value of the voltage threshold for input over-voltage fault
VIN_OV_FAULT_RESPONSE **Example:** > r 45 > w 45 RSP:Shutdown RS:TryRestart Note: entry required only for the changing function	56h	r/w	RSP	Continue	Set module to continue without interruption on input OV fault
				Shutafter4	Set module to continue for four cycles on input OV fault
				Shutdown	Set module to shut down on input OV fault
				Cutoutput	Set module to cut off output on input OV fault and remain OFF until fault is cleared
			RS	Norestart	Set module to not restart on input OV fault
				Tryrestart	Set module to restart on input OV fault
VIN_OV_WARN_LIMIT	57h	r/w			Real Number	Read/write value of the voltage threshold for input over-voltage warning
VIN_UV_WARN_LIMIT	58h	r/w			Real Number	Read/write value of the voltage threshold for input under-voltage warning
VIN_UV_FAULT_LIMIT	59h	r/w			Real Number	Read/write value of the voltage threshold for input under-voltage fault
VIN_UV_FAULT_RESPONSE **Example:** > r 45 > w 45 RSP:Shutdown RS:TryRestart Note: entry required only for the changing function	5Ah	r/w	RSP	Continue	Set module to continue without interruption on input UV fault
				Shutafter4	Set module to continue for four cycles on input UV fault
				Shutdown	Set module to shut down on input UV fault
				Cutoutput	Set module to cut off output on input UV fault and remain OFF until fault is cleared
			RS	Norestart	Set module to not restart on input UV fault
				Tryrestart	Set module to restart on input UV fault
POWER_GOOD_ON (PGON) **Example:** >r 5e >w 5E 1.15	5Eh	r/w			Real Number	Value of the Power Good On threshold. Depends on the value of VOUT_SCALE_MONITOR. The Power Good on thresholds are programmable with increments of 0.1%. PGON threshold must be more than PGOFF threshold.
POWER_GOOD_OFF (PGOFF) **Example:** >r 5F >w 5f 1.10	5Fh	r/w			Real Number	Value of the Power Good OFF threshold. Depends on the value of VOUT_SCALE_MONITOR. PGOFF threshold must be less than PGON threshold.
TON_DELAY **Example:** >r 60 >W 60 2.4	60h	r/w			Real Number	Value of the delay time of the output during turn-on. The values are programmable from 1 to 500msec
TON_RISE **Example:** >r 61 >w 61 2.4	61h	r/w			Real Number	Value of the rise time of the output. The values are programmable from 1 to 100msec

Command	Code	Capability	Name	Value	Function
TOFF_DELAY **Example:** `>r 64` `>w 64 2.4`	64h	r/w		Real Number	Value of the delay time of the output during turn-off. The values are programmable from 1 to 500msec
TOFF_FALL **Example:** `>r 65` `>w 65 2.4`	65h	r/w		Real Number	Value of the fall time of the output during turn-off. The values are programmable from 1 to 100msec
STATUS_BYTE **Example:** `>r 78`	78h	r			Returns byte with module status
STATUS_WORD **Example:** `>r 79`	79h	r		Word	Returns two bytes with module status
STATUS_VOUT **Example:** `>r 7A`	7Ah	r			Returns byte with module output voltage related faults
STATUS_IOUT **Example:** `>r 7B`	7Bh	r			Returns byte with module output current related faults
STATUS_INPUT **Example:** `>r 7C`	7Ch	r			Returns byte with module input related faults
STATUS_TEMPERATURE **Example:** `>r 7D`	7Dh	r			Returns byte with module temperature related faults
STATUS_CML **Example:** `>r 7E`	7Eh	r			Returns byte with module communication related faults
READ_VIN **Example:** `>r 88`	88h	r		Real Number	Returns value of the input voltage
READ_VOUT **Example:** `>r 8B`	8Bh	r		Real Number	Returns value of the output voltage
READ_IOUT **Example:** `>r 8C`	8Ch	r		Real Number	Returns value of the output current
READ_TEMPERATURE_1	8Dh	r		Real Number	Returns the module inductor temperature in °C
READ_TEMPERATURE_2	8Eh	r		Real Number	Returns the module PWM controller temperature in °C
READ_FREQUENCY	95h	r		Real Number	Returns the value of switching frequency of the module. The frequency is in Kilohertz.
PMBUS_REVISION **Example:** `>r 98`	98h	r			Returns byte with PMBus version number that module complies with
MFR_VIN_MIN **Example:** `>r A0`	A0h	r		Real Number	Returns value of the minimum input voltage module supports
MFR_VIN_MAX **Example:** `>r A1`	A1h	r		Real Number	Returns value of the maximum input voltage module supports
MFR_VOUT_MIN **Example:** `>r A4`	A4h	r		Real Number	Returns value of the minimum output voltage module supports
MFR_VOUT_MAX **Example:** `>r A5`	A5h	r		Real Number	Returns value of the maximum output voltage module supports
MFR_SPECIFIC_00 **Example:** `>r D0`	D0h	r		Real Number	Returns two bytes of module specific information
VOUT_CAL_OFFSET **Example:** `>r D4` `>w D4 0.15`	D4h	r/w		Real Number	Read/write offset correction for output voltage measurement
VOUT_CAL_GAIN **Example:** `>r D5` `>w D5 0.02`	D5h	r/w		Real Number	Read/write gain factor correction for output voltage measurement
VIN_CAL_OFFSET **Example:** `>r D6` `>w D6 -0.2`	D6h	r/w		Real Number	Read/write offset correction for input voltage measurement

Command	Code	Capability	Name	Value	Function
ALERT_STATUS **Example:** >a	n/a	r		Word	Returns the SMBALERT status of the module(s). If the SMBALERT is active then the CLI reads back the status as 'Asserted".
ALERT_ADDRESS **Example:** >a s	n/a	r		Word	Returns the address of the module that pulled down the SMBAlert line. If there are multiple PMBus modules with alerts on the bus, the lowest address that has triggered the Alert line is returned. By using the ALERT_STATUS command again the first module releases the alert line. When the ALERT_ADDRESS command is executed again the next module with the lowest address that is still pulling down the Alert line is identified. This process is repeated to find all modules with alerts till the ALERT_STATUS is 'Normal'

6.6   Summary of Supported PMBus Commands for Bus Converters

Command	Code	Capability	Name	Value	Function
OPERATION	01h	r/w	On	Y	Turns Module ON
Example: `> r OPERATION or >r 01` `> w 01 On:Y,Margin:OFF` `Note: entry required only for the changing function`				N	Turns Module OFF
			Margin	OFF	Output Margining is OFF
				ActLow	Margin Low but act on any Faults
				ActHigh	Margin High but act on any Faults
ON_OFF_CONFIG	02h	r/w	Pu	Y	Power up is contingent on the settings below
Example: `> r 02` `> w 02 Pu:Y,Cmd:Y,Cpr:N` `Note: entry required only for the changing function`			Cmd	Y	Unit powers up with the ON bit of OPERATION
				N	Unit ignores the ON/OFF state of OPERATION
			Cpr	Y	Unit responds to the state of the CNTL pin
			Pol	Y	Read only, factory set, Y=Positive Logic on/off
				N	Read only, factory set, N=Negative Logic on/off
			Cpa	Y	No turn-OFF delay, not user programmable
CLEAR_FAULTS **Example:** `>w 03`	03h	w			Clears the Status registers and resets the SMBAlert# signal.
STORE_DEFAULT_ALL **Example:** `> w 11`	11h	w			Store all the configuration settings from working memory into non-volatile memory
RESTORE_DEFAULT_ALL **Example:** `> w 12`	12h	w			Store all the configuration settings from non-volatile memory into working memory
VOUT_MODE **Example:** `> r 20`	20h	r/w		Word	Returns mode and exponent for output voltage related commands
VOUT_COMMAND **Example:** `> w 21 12.1` `> r 21`	21h	r/w		Real Number	Read/write the output voltage setting of the module
VOUT_CAL_OFFSET **Example:** `> r 23` `> w 23 0.2`	23h	r/w		Real Number	Read/write the calibration offset for precise setting of the output voltage of the module
VOUT_MARGIN_HIGH **Example:** `> r 25` `> w 25 1.33`	25h	r/w		Real Number	Read/write value of the desired output voltage margin high level
VOUT_MARGIN_LOW **Example:** `> w 26 1.10`	26h	r/w		Real Number	Read/write value of the desired output voltage margin low level
VOUT_DROOP **Example:** `> r 28` `> w 28 2.0`	28h	r/w		Real Number	Read/write value of the desired output voltage droop in mV/A
VIN_ON **Example:** `> r 33` `> w 33 34.5`	35h	r/w		Real Number	Read/write value of the input voltage turn-ON level
VIN_OFF **Example:** `> r 33` `> w 33 34.5`	36h	r/w		Real Number	Read/write value of the input voltage turn-OFF level
VOUT_OV_FAULT_LIMIT **Example:** `> r 40` `> w 40 1.35`	40h	r/w		Real Number	Read/write value of the output overvoltage fault level
VOUT_OV_FAULT_RESPONSE	41h	r/w	RSP	Shutdown	Set module to shutdown on output voltage OV fault
Example: `> r 41` `> w 41 RSP:Shutdown,RS:TryRestart` `Note: entry required only for the changing function`			RS	Norestart	Set module to no restart on output voltage OV fault
				Tryrestart	Set module to restart on output voltage OV fault

Command	Code	Capability	Name	Value	Function
IOUT_OC_FAULT_LIMIT **Example:** > r 46 > w 46 25.2	46h	r/w		Real Number	Read/write value of the output overcurrent fault level
IOUT_OC_FAULT_RESPONSE	47h	r/w	RSP	Shutdown	Set module to shutdown on output current OC fault
Example: > r 47 > w 47 RSP:Shutdown,RS:TryRestart Note: entry required only for the changing function			RS	Norestart	Set module to latch off after an output current OC fault
				Tryrestart	Set module to restart on output OC fault
IOUT_OC_WARN_LIMIT **Example:** > r 4A > w 4A 25.2	4Ah	r/w		Real Number	Read/write value of the output overcurrent warning level
OT_FAULT_LIMIT **Example:** > r 4F > w 4F 130	4Fh	r/w	RSP	Real Number	Read/write value of the module overtemperature fault level
OT_FAULT_RESPONSE	50h	r/w	RSP	Shutdown	Set module to shutdown on Overtemperature fault
Example: > r 50 > w 50 RSP:Shutdown,RS:TryRestart Note: entry required only for the changing function			RS	Norestart	Set module to not restart on Overtemperature fault
				Tryrestart	Set module to restart on Overtemperature fault
OT_WARN_LIMIT **Example:** > r 51 > w 51 125	51h	r/w		Real Number	Read/write value of the module overtemperature warning level
VIN_OV_FAULT_LIMIT **Example:** > r 55 > w 55 78.5	55h	r/w		Real Number	Read/write value of the module input overvoltage fault limit
POWER_GOOD_ON **Example:** >r 5e >w 5E 2.5	5Eh	r/w		Real Number	Value of the Power Good On threshold
POWER_GOOD_OFF **Example:** >r 5F >w 5f 2.5	5Fh	r/w		Real Number	Value of the Power Good OFF threshold
TON_DELAY **Example:** >r 60 >w 60 2.4	60h	r/w		Real Number	Value of the output delay time
TON_RISE **Example:** >r 61 >w 61 2.4	61h	r/w		Real Number	Value of the output rise time
STATUS_WORD **Example:** >r 79	79h	r		Word	Returns two bytes with module status
STATUS_VOUT **Example:** >r 7A	7Ah	r		Byte	Returns byte with module output voltage related faults
STATUS_IOUT **Example:** >r 7A	7Bh	r		Byte	Returns byte with module output current related faults
STATUS_INPUT **Example:** >r 7B	7Ch	r		Byte	Returns byte with module input related faults
STATUS_TEMPERATURE **Example:** >r 7D	7Dh	r			Returns byte with module temperature related faults
STATUS_CML **Example:** >r 7E	7Eh	r			Returns byte with module communication related faults
READ_VIN **Example:** >r 88	88h	r		Real Number	Returns value of the input voltage

Command	Code	Capability	Name	Value	Function
READ_VOUT **Example:** >r 8B	8Bh	r		Real Number	Returns value of the output voltage
READ_IOUT **Example:** >r 8C	8Ch	r		Real Number	Returns value of the output current
READ_TEMPERATURE_1 **Example:** >r 8D	8Dh	r		Real Number	Returns value of the output temperature
PMBUS_REVISION **Example:** >r 98	98h	r			Returns byte with PMBus version number that module complies with
MFR_DEVICE_TYPE **Example:** >r D0	D0h	r		Real Number	Returns two bytes of module specific information
MFR_VOUT_READ_CAL_GAIN **Example:** >r D1	D1h	r		Real Number	Read/write gain factor correction for output voltage measurement
MFR_VOUT_READ_CAL_OFFSET **Example:** >r D2	D2h	r		Real Number	Read/write offset correction for output voltage measurement
MFR_VIN_READ_CAL_GAIN **Example:** >r D3	D3h	r		Real Number	Read/write gain factor correction for input voltage measurement
MFR_VIN_READ_CAL_OFFSET **Example:** >r D4	D4h	r		Real Number	Read/write offset correction for input voltage measurement
MFR_IOUT_CAL_GAIN **Example:** >r D6	D6h	r		Real Number	Read/write gain factor correction for output current measurement
MFR_IOUT_CAL_OFFSET **Example:** >r D7	D7h	r		Real Number	Read/write offset correction for output current measurement
MFR_FW_REV **Example:** >r DB	DBh	r		Byte	Returns one byte with module firmware revision level
MFR_C1_C2_ARA_CONFIG **Example:** > r E0 > w E0 ARA:Y,PIN:TRIM_PGOOD	E0h	r/w	ARA	OFF	Does not responds to Alert Response Address 12
				ON	Responds to Alert Response Address 12
			PIN	ON_OFF_PGOOD	Pin C1 set to ON_OFF control and pin C2 to PGOOD
				TRIM_PGOOD	Pin C1 set to TRIM and pin C2 to PGOOD
				TRIM_ON_OFF	Pin C1 set to TRIM and pin C2 to ON_OFF control
MFR_C2_LOGIC **Example:** > r E1 > w E1 SEC:ON,LOGIC:POS	E1h	r/w	SEC	OFF	Secondary side on/off pin ignored
				ON	Secondary side on/off pin enabled
			LOGIC	NEG	ON/OFF Control set to negative logic
				POS	ON/OFF Control set to positive logic
MFR_PGOOD_POLARITY **Example:** > r E2 > w E2 PGOOD_LOGIC:POS	E2h	r/w	PGOOD_LOGIC	NEG	Negative signalling polarity of the PGOOD pin
				POS	Positive signalling polarity of the PGOOD pin

BARRACUDA III Only

Command	Code	Capability	Name	Value	Function
MFR_SPECIFIC_KP **Example:** >r E3 >w E3 1275	E3h	r/w		Real Number	Read/write value for loop proportional coefficient
MFR_SPECIFIC_KI **Example:** >r E4 >w E4 75	E4h	r/w		Real Number	Read/write value for loop integral coefficient
MFR_SPECIFIC_KD **Example:** >r E5 >w E5 –100	E5h	r/w		Real Number	Read/write value for loop derivative coefficient
MFR_SPECIFIC_ALPHA **Example:** >r E6 >w E6 422	E6h	r/w		Real Number	Read/write value for loop alpha coefficient

ALL BARRACUDA

Command	Code	Capability	Name	Value	Function
MFR_MODULE_DATE_LOC_SN **Example:** `>r F0`	F0h	r		12 Bytes	Block read of module manufacturing location/date/serial number
ALERT_ADDRESS **Example:** `> r alert_address`	n/a	r		Integer	Returns the address of the module that pulled down the SMBAlert line. Address 12h is used for this inquiry. This address should not be used as a device address for any module. See module data sheet for support of this function.
ALERT_STATUS **Example:** `>a`	n/a			Integer	Returns the SMBALERT status of the module. If the SMBALERT is active then the CLI reads back the status as 'Asserted".
ALERT_ADDRESS Example `>a s`	n/a			Integer	Returns the address of the module that pulled down the SMBAlert line. If there are multiple PMBus modules with alerts on the bus, the lowest address that has triggered the Alert line is returned. By using the ALERT_STATUS command again the first module releases the alert line. When the ALERT_ADDRESS command is executed again the next module with the lowest address that is still pulling down the Alert line is identified. This process is repeated to find all modules with alerts till the ALERT_STATUS is 'Normal'.

6.7   Summary of Supported PMBus Commands for the CP Platform

Command	Code	Capability	Name	Value	Function
OPERATION **Example:** > w 01 On:Y	01h	w	On	Y	Turns Module ON
				N	Turns Module OFF
CLEAR_FAULTS **Example:** > w 03	03h	w			Clears the Status registers and resets the SMBAlert# signal.
VOUT_COMMAND **Example:** > w 21 52 > write vout_... 52	21h	w		2 Bytes	Changes the main output voltage . The command example instructs the module output to be set to 52Vdc,
VOUT_OV_FAULT_LIMIT **Example:** > w 40 56	40h	w		2 Bytes	Changes the overvoltage shutdown level of the main output voltage. The example changes to OV shutdown level to 56Vdc.
READ_STATUS **Example:** > r D0 > read read_status	D0h	r		10 bytes	Returns the state of the power supply (STATUS and ALARM registers, output voltage, output current, and internal temperature levels).
FLASH_LEDS_ON **Example:** > w D2 > write flash_LEDs...	D2h	w		none	ALL LED test, 0.7sec ON, 0.2sec OFF
FLASH_LEDS_OFF **Example:** > w D3 > write flash_LEDs...	D3h	w		none	Turn OFF LED testing
SERVICE_LED_ON **Example:** > w D4 > write service_...	D4h	w		none	Turn ON service LED, 0.5s ON and 0.5sec OFF
SERVICE_LED_OFF **Example:** > w D5 > write service_...	D5h	w		none	Turn OFF service LED
EEPROM_WRITE_ON **Example:** > w D6 > write EEPROM_.....	D6h	w		none	Enables writing into the upper ¼ of EEPROM
EEPROM_WRITE_OFF **Example:** > w D7 > write EEPROM_...	D7h	w		none	Disables writing into the upper ¼ of EEPROM
INHIBIT_RESTART **Example:** > w D8 > write inhibit_...	D8h	w		none	Sets fault levels into latched mode.
AUTO_RESTART **Example:** > w D2 > write auto_...	D9h	w		none	Sets fault levels into restart mode.
ISOLATION TEST **Example:** > w DA > write isolation...	DAh	w		none	Tests the Or'ing function in paralleled modules
INHIBIT_VOUT_DROOP **Example:** > w D2 > write auto_...	DBh	w		none	Turns OFF the 'droop' feature, if supported
READ_INPUT_STRING **Example:** > r D2 > read auto_...	DCh	r		3 Bytes	Reads input voltage and input power
READ_FIRMWARE_REV **Example:** > r DD > read read_...	DDh	r		3 Bytes	Reads firmware revision of the 3 processors
READ_RUN_TIMER **Example:** > r DE > read read_run_...	DEh	r		3 Bytes	Reads the accumulated ON time of the PS

Command	Code	Capability	Name	Value	Function
FAN_HI_SPEED **Example:** > w DF 75 > write fan_HI_... 75	DFh	w		1 Byte	Increases speed to the specified % of maximum speed capability of the fan. The example sets the duty cycle of the fans to 75%.
FAN_NORMAL_SPEED **Example:** > w E0 > write fan_...	E0h	w		none	Reverts fan speed back to power supply control
READ_FAN_SPEED **Example:** > r E1 > read read_...	E1h	r		4 Bytes	Returns % control, fan1, fan2, fan3 speed in RPM

6.8   Summary of Supported PMBus Commands for the CP3500

Command	Code	Capability	Name	Value	Function
OPERATION **Example:** `> w 01 On:Y`	01h	r/w	On	Y	Turns Module ON
				N	Turns Module OFF
CLEAR_FAULTS **Example:** `> w 03`	03h	w			Clears the Status registers and resets the SMBAlert# signal.
WRITE_PROTECT `> W 10 80`	10h	r/w	Allow	ALL WP WP_OP	Default; enables all writes Disable all writes except write_protect Disable all writes except write_protect, OPERATION
RESTORE_DEFAULT_ALL **Example:** `> w 12`	12h	w	.		Restore all the default configuration settings into non-volatile memory
STORE_USER_ALL **Example:** `> w 15`	15h	w	.		Store all the configuration settings into non-volatile memory
RESTORE_USER_ALL **Example:** `> w 16`	16h	w			Get all configuration settings from non-volatile memory
STORE_USER_CODE **Example:** `> w 17 10`	17h	w	.		Store only the identified configuration setting into non-volatile memory
RESTORE_USER_CODE **Example:** `> w 18`	18h	w	.		Restore only the identified configuration setting into non-volatile memory
VOUT_MODE **Example:** `> r 20`	20h	r	.	Byte	Returns mode and exponent for output voltage related commands

VOUT_COMMAND
Example: `> w 21 52`
`> write vout 52`
`> r 21`
`> r vout`
	21h	r/w		Word	Changes the main output voltage . The command example instructs the module output to be set to 52Vdc,

VIN_ON **Example:** `> w 35 120` `> r 35`	35h	r/w		Real Number	Read/write value of the input voltage turn-ON level
VIN_OFF **Example:** `> r 36` `> w 36 85`	36h	r/w		Real Number	Read/write value of the input voltage turn-OFF level
FAN_CONFIG_1_2 **Example:** `> r 3A`	3Ah	r		Byte	Returned byte with fan configuration

FAN_COMMAND_1
Example: `> r 3B`
`> w 3B 85 (%)`
`> w 3B 8500 (RPM)`
	3Bh	r/w		Real Number	Commands the fans to a specific speed based on RPM or % of full speed as defined by Fan_Config_1_2

VOUT_OV_FAULT_LIMIT **Example:** `> r 40` `> w 40 60`	40h	r/w		Real Number	Changes the output overvoltage fault level. Entry in actual voltage level.

Command	Code	Capability	Name	Value	Function
VOUT_OV_FAULT_RESPONSE	41h	r/w	RSP.	Continue	Not applicable
Example: `> r 41` `  > w 41 RSP:Shutdown RS:TryRestart` Note: entry required only for the changing function				Shutafter4	Set module to continue for four cycles on output overvoltage fault
				Shutdown	Set module to shutdown on output overvoltage fault
				Autorecovers	Set module to cut off output on output overvoltage fault and remain OFF until fault is cleared
			RS	Ignored	Not applicable
				Tryrestart	Set module to restart on output overvoltage fault
VOUT_OV_WARN_LIMIT **Example:** `> r 42` `      > w 42 55`	42h	r/w		Real Number	Changes the output overvoltage warn level. Entry in actual voltage level.
VOUT_UV_WARN_LIMIT **Example:** `> r 43` `      > w 43 40`	43h	r/w		Real Number	Changes the output undervoltage warn level. Entry in actual voltage level.
VOUT_UV_FAULT_LIMIT **Example:** `> r 44` `      > w 44 38`	44h	r/w		Real Number	Changes the output undervoltage fault level. Entry in actual voltage level.
VOUT_UV_FAULT_RESPONSE	45h	r/w	RSP.	Continue	Set module to continue without interruption on UV fault
Example： `  > r 45` `  > w 45 RSP:Autorestart RS:TryRestart` Note: entry required only for the changing function				Shutafterdelay	Set module to continue for 'delay' cycles on output voltage UV fault
				Shutdown	Set module to shutdown on output voltage UV fault
				Autorestart	Set module to autorestart on output voltage UV fault
			RS	Ignored	Not applicable
				Tryrestart	Set module to restart on output voltage UV fault
IOUT_OC_FAULT_LIMIT **Example:** `>r 46` `      >w 46 30`	46h	r/w	.	Real Number	Read/write value of the output overcurrent fault level
IOUT_UV_FAULT_RESPONSE	47h	r/w	RSP.	Continue	Set module to continue without interruption on OC fault
Example: `  > r 47` `  > w 47 RSP:Respond to_rs RS:TryRestart` Note: entry required only for the changing function				Respond to_rs	Respond with rs setting
				Shutdown	Continue to operate for 4 cycles then shutdown
				Cutoutput	Set the module to shutdown upon OC
			RS	Ignored	Output remains disabled
				Tryrestart	Set module to restart on output OC fault
IOUT_OC_LV_FAULT_LIMIT **Example:** `> r 48` `      > w 48 50`	48h	r/w	.	Real Number	Set the output voltage level where an OC limit should cause a shutdown
IOUT_OC_WARN_LIMIT **Example:** `> r 4A` `      > w 4A 48`	4Ah	r/w		Real Number	Set the output overcurrent warning level.
OT_FAULT_LIMIT **Example:** `> r 4F` `      > w 4F 95`	4Fh	r/w		Real Number	Set the internal temperature fault level

Command	Code	Capability	Name	Value	Function
OT_FAULT_RESPONSE	50h	r/w	RSP.	Continue	Set module to continue without interruption on UV fault
				Shutafter4	Set module to continue for four cycles on output voltage UV fault
Example： > r 50 > w 50 RSP:Shutdown RS:TryRestart Note: entry required only for the changing function				Shutdown	Set module to shutdown on output voltage UV fault
				Autorecovers	Set module to recover the output when the temperature reduces below thresholds
			RS	Ignored	Does not attempt a restart
				Tryrestart	Set module to restart on output voltage UV fault
OT_WARN_LIMIT **Example:** > r 51 > w 51 90	51h	r/w		Real Number	Set the internal OT warn level
VIN_OV_FAULT_LIMIT **Example：** > r 55 > w 55 305	55h	r/w		Real Number	Set the Input OV fault level
VIN_OV_FAULT_RESPONSE	56h	r/w	RSP.	Continue	Set module to continue without interruption on input overvoltage fault
				Shutafter4	Set module to continue for four cycles on input overvoltage fault
Example: > r 56 > w 56 RSP:Shutdown RS:TryRestart Note: entry required only for the changing function				Shutdown	Set module to shutdown on input overvoltage fault
				Autorecovers	Set module to recover once the input voltage is above the limit threshold
			RS	Ignored	Output remains disabled until a restart executed
				Tryrestart	Set module to restart on input overvoltage fault
VIN_OV_WARN_LIMIT **Example:** > r 57 > w 57 290	57h	r/w		Real Number	Set the Input OV warn level
VIN_UV_WARN_LIMIT **Example:** > r 58 > w 58 90	58h	r/w		Real Number	Set the Input UV warn level
VIN_UV_FAULT_LIMIT **Example:** > r 59 > w 59 85	59h	r/w		Real Number	Set the Input UV fault level
VIN_UV_FAULT_RESPONSE	5Ah	r/w	RSP.	Continue	Set module to continue without interruption on input undervoltage fault
				Shutafter4	Set module to continue for four cycles on input undervoltage fault
Example: > r 5A > w 5A RSP:Shutdown RS:TryRestart Note: entry required only for the changing function				Shutdown	Set module to shutdown on input undervoltage fault
				Autorecovers	Set module to recover once the input voltage is above the limit threshold
			RS	Ignored	Output remains disabled until a restart executed
				Tryrestart	Set module to restart on input undervoltage fault
STATUS_BYTE **Example:** >r 78	78h	r			Returns byte with module status
STATUS_WORD **Example:** >r 79	79h	r		Word	Returns two bytes with module status
STATUS_VOUT **Example:** >r 7A	7Ah	r			Returns byte with module output voltage related faults

Command	Code	Capability	Name	Value	Function
STATUS_IOUT Example: >r 7B	7Bh	r			Returns byte with module output current related faults
STATUS_INPUT Example: >r 7C	7Ch	r			Returns value with module INPUT related faults
STATUS_TEMPERATURE Example: >r 7D	7Dh	r			Returns byte with module temperature related faults
STATUS_CML Example: >r 7E	7Eh	r			Returns byte with module communication related faults
STATUS_FANS_1_2 Example： >r 81	81h	r			Returns byte with module fan related faults
READ_VIN Example: >r 88	88h	r		Real Number	Returns value of the input voltage in linear format
READ_IIN Example: >r 89	89h	r		Real Number	Returns value of the input current in linear format
READ_VOUT Example: >r 8B	8Bh	r		Real Number	Returns value of the output voltage in linear format
READ_IOUT Example: >r 8C	8Ch	r		Real Number	Returns value of the output current in linear format
READ_TEMP_PFC Example: >r 8D	8Dh	r		Real Number	Returns value of the pfc temperature in linear format
READ_TEMP_DC_PRI Example: >r 8E	8Eh	r		Real Number	Returns value of the dc_pri temperature in linear format
READ_TEMP_DC_SEC Example: >r FD	8Fh	r		Real Number	Returns value of the dc_sec temperature in linear format
READ_FAN_SPEED_1 Example: >r 90	90h	r		Real Number	Returns value of the fan-1 speed in linear format
READ_FAN_SPEED_2 Example: >r 91	91h	r		Real Number	Returns value of the fan-2 speed in linear format
READ_PIN Example: >r 97	97h	r		Real Number	Returns value of the input power consumed in linear format
MFR_ID Example: >r 99	99h	r		Character	Returns five bytes in ASCII format
MFR_MODEL Example: >r 9A	9Ah	r		Character	Returns fifteen bytes in ASCII format
MFR_REVISION Example: >r 9B	9Bh	r		Character	Returns six bytes in ASCII format
MFR_SERIAL Example: >r 9E	9Eh	r		Character	Returns fifteen bytes in ASCII format
STATUS_SUMMARY Example: >r D0	D0h	r		Array	Returns thirteen bytes in ASCII format
STATUS_UNIT Example: >r D1	D1h	r		Character	Returns two bytes in ASCII format
STATUS_ALARM Example: >r D2	D2h	r		Character	Returns three bytes in ASCII format
READ_FAN_SPEED Example: >r D3	D3h	r		Real number	Returns four bytes in ASCII format
READ_INPUT Example: >r D4	D4h	r		Character	Returns four bytes in ASCII format
READ_FIRMWARE_REV Example: >r D5	D5h	r		Character	Returns six bytes in ASCII format
READ_RUN_TIMER Example: >r D6	D6h	r		Real Number	Returns three bytes in ASCII format

Command	Code	Capability	Name	Value	Function
STATUS_BUS **Example:** `>r D7`	D7h	r		Character	Returns one byte
TAKE_OVER_BUS_CONTROL **Example:** `>r D8`	D8h	r		None	
EEPROM RECORD **Example:** `>r D9` `> w D9 ASCII`	D9h	r/w		Character	Up to 128 characters of the user choosing can be written into the internal EEPROM section of the processor. See data sheet for further information.
READ_TEMP_EXHAUST **Example:** `>r DA`	DAh	r		Real Number	Not supported
READ_TEMP_INLET **Example:** `>r DB`	DBh	r		Real Number	Returns the inlet temperature into the power supply.
PASSWORD **Example:** `>r E0`	E0h	r		Character	Returns four bytes in ASCII format. See Data Sheet. Cannot be changed. Not usable.
TARGET_LIST **Example:** `>r E1`	E1h	r		Character	Returns the list of the processors in ASCII to be upgraded. See Data Sheet.
COMPATIBILITY CODE **Example:** `>r E2`	E2h	r		Character	Returns 32 bytes in ASCII defining the compatibility code. See Data Sheet
SOFTWARE_VERSION **Example:** `>r E3`	E3h	r		Real Number	Returns seven bytes of the software revision. See Data Sheet.
MEMORY_CAPABILITY **Example:** `>r E4`	E4h	r		Real Number	Returns seven bytes describing the memory capability in the processor. See Data Sheet
APPLICATION _STATUS **Example:** `>r E5`	E5h	r		Hex	Returns one byte of the status of the application. See Data Sheet
BOOT_LOADER **Example:** `>r E6` `> w E6 #`	E6h	r/w		Real Number	Not supported
DATA_TRANSFER **Example:** `>r E7` `> w E7 ASCII`	E7h	r/w		Character	Uploads into the power supply the upgraded application in 32 byte increments. See Data Sheet
PRODUCT_COMCODE **Example:** `>r E8`	E8h	r		Character	Reads 11 ASCII characters. See data sheet
UPLOAD_BLACK_BOX **Example:** `>r F0`	F0h	r		Character	Uploads contents of the black box in 32 byte increments. See Data Sheet

7 DPI-CPGUI (Simple GUI for the CP Rectifier Family[1])

The DPI-CPGUI is a graphically-oriented tool for interfacing to up to eight CP family rectifiers or PEMs. Tool capabilities include

- Identifying and configuring the addresses of up to eight units
- Retrieving and displaying bus status and alarm register information for the units
- Acquiring and showing data including input voltage, input power, output voltage, output current, unit temperature, fan speed settings and fan speeds, micro controller firmware revisions and run time.
- Setting of key parameters such as output voltage, output voltage shutdown level, fan speed, or issuing a custom command
- Control of key parameters; who should be controlled (one or all units), output ON/OFF, LED test routine, read a single status, fan speed control, EEPROM write control, shutdown control (latch or restart), I^2C bus control, isolation test, inhibit droop output voltage control, issue bad command or a bad PEC byte.
- Setting up periodic polling and display of key information from the units
- Automatically saving a communications log containing all commands issued on the I^2C bus and status changes with associated time stamps
- Saving to and retrieving data from the EEPROM in the rectifier or PEM
- Executing automated bus control commands – taking over bus control

7.1 Starting the CPGUI Tool

When the GE USB Interface Adapter is plugged into the USB port on the Personal Computer, the Green LED of the Adapter should be ON. Apply input power to the rectifier or PEM and start the GUI application. Note that multiple DPI applications cannot run simultaneously as they share the same COM port connection to the USB Interface Adapter on the personal computer.

The DPI-CPGUI tool can be started by double clicking on the *cpgui.exe* icon located on the desktop, or using the standard *Windows Start – Programs – DPI – CP GUI* option. The tool starts up and displays the screen shown on the next page. Note: it may take up a few seconds for this screen to appear. This is normal.

Note that the GE USB Interface adapter needs to be plugged into the computer for the DPI-CPGUI program to work properly. If the GE USB Interface adapter cannot be accessed by the software, it will either display the following window during startup, or it may not start up at all.


In some cases, this can be corrected by removing and re-inserting the USB cable connected to the GE USB Interface adapter, and restarting the DPI-CPGUI program. Correcting this condition may require two reinsertions .To exit the DPI-CPGUI program, just click on the X in the top right corner of the tool display window.

---

[1] The CP family includes the CP1800AC52, CP2000AC54, CP2500AC54, CP2725AC54, CP2000DC54, CP2500DC54 and all variations thereof. (The CP3500AC54 has a different command set that is part of the global platform)

7.2   DPI-CPGUI Description

The screen shot below shows the tool display once it starts up. This is the only screen displayed by the tool.
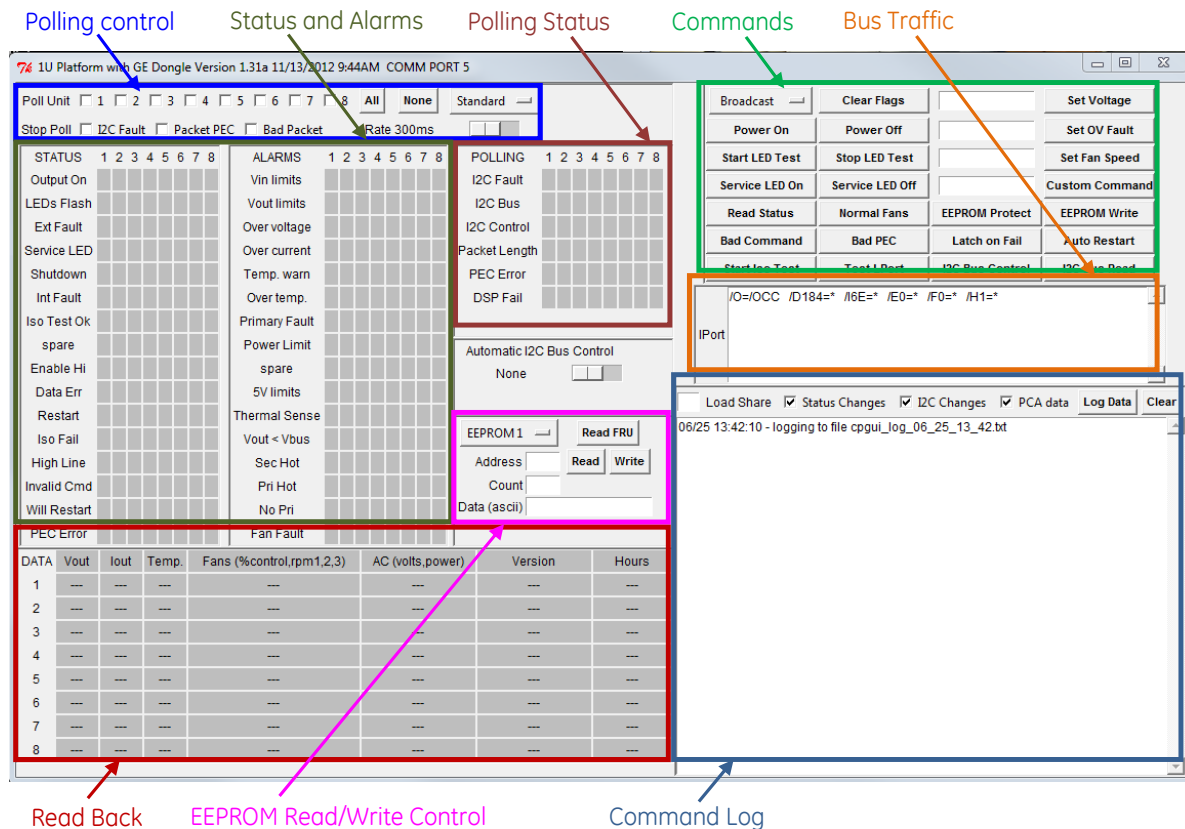

7.3   Polling Control

This section is used to select the units to be polled, the rate at they should be polled, and under what conditions polling should be automatically stopped. Unit addresses are set by the unit_ID and shelf_ID signal pins of the units. The internal DSP translates these two signal pins to three bits A2, A1, and A0 components of the address field. …..000 is unit #1 and ……111 is unit #8. This is all done automatically. There are three devices that can be addressed, the micro controller that communicates power supply related information and control, the EEPROM that contains FRU_ID information, and the PCA9541 multiplexer that controls which I2C line is being communicated to. The GUI automatically sets the correct address depending on which feature is selected by the user.

7.3.1   Initiation and Manual Termination of Polling

The tool can be set up to poll any of the 8 units by clicking on the buttons to the left of the unit number. If all units are to be pulled the user can alternatively click the **All** button once. To turn OFF polling simply click on the **None** button. The polling rate can be changed by moving the scrolling bar to the right or left. The chosen polling rate is displayed to the left of the scrolling bar.
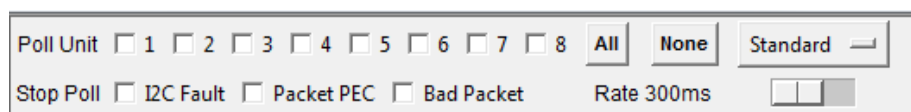
7.3.2   Automatic Poll Termination

Polling can be automatically stopped for any or all of the three failure modes, **I2C fault**, **Packet error**, or **Bad Packet** detected by the tool if the appropriate box is selected by clicking on the box area.

7.3.3 Changing the rate of Polling

The **polling rate** can be changed by dragging the bar to the right of the displayed polling rate. The selected rate is displayed. The rate between polls can vary from 200ms to 5 seconds.

7.3.4 Choosing What to Poll and Which Product is being Polled

The **Standard** button above the scrolling bar determines what is being polled. Clicking on the button displays a drop down menu that selects other possible features beside the 'standard' polling feature. The following features are currently available:

Standard: Polls the digital STATUS and ALARM registers and output voltage, output current, and temperature.
Enhanced: Invokes the polling of all analog read backs in addition to the 'standard' poll. These include input voltage and power, fan speeds, Controller revisions, and run time. Default numbers are set in registers that are not supported by the module being polled.
CDC2100 and **CAC3000:** These are two custom codes that the tool can support.
CP2500DC: Clicking on this icon changes the GUI display to communicate with the latest CP2500DC module. In this mode only the 'enhanced' mode of read back is supported.

7.4 Returned data Display Area

There are three areas in the tool that display the status and key module parameters as a combination of numerical values and LED-like status displays. The **Status and Alarms** and **Polling Status** use LED-like status displays, while **Read Back** displays analog data as numbers. These registers display updated states every time the unit is polled.

7.4.1 Status and Alarms

The **Status and Alarms** area displays status of up to eight units. The user selects in the Polling and Stop Control area which units are to be monitored and the time rate at which they should be polled. The displayed color indicates event severity. Green – normal, Yellow – warning, Red – fault which requires immediate attention. Since these registers are continually being updated, it is not unusual to see changes in the displayed states as the unit goes through a number of event changes prior to settling down to the final state.

In the example display shown on the left, unit #1 has been shut down because of an over-temperature event on the secondary side. **Internal_fault** or **external_fault** are predictions of the likely cause of the event.

Units #2 and #3 are working correctly and their outputs are ON. The 'will restart' indicator shows that the three units are in the restart mode.

7.4.2 Polling Status

The **Polling Status area** displays the status of communications to the units being read or polled.

I2C Fault – RED Communications failed with the I²C μController.
I2C Bus – YELLOW Communications are not established on the bus.
 GREEN Module at address #5 Established communications
I2C Control – YELLOW Does not have control of this I²C line.
 GREEN Module at address #5 Has control of the connected bus
Packet Length – RED Length of the data packet is incorrect.
PEC error – RED The calculated PEC does not agree with the transmitted PEC.
DSP Fail – RED Internal communications between the DSP and the I²C μC failed

7.4.3   Read Back

In the lower left corner analog information is read from the selected units. The data read back from all 8 possible units (the maximum capacity of a single bus line) can be displayed at once. There are two modes of read back, **standard** and **extended**.

The **standard** read back records the output voltage, output current, and temperature of the units.

The **extended** read back also records the commanded fan control percentage and the speed of up to three fans in RPM, input voltage and power, firmware revision of the three µCs, and operational running time. The extended read back is supported by the latest revisions of the units.

DATA	Vout	Iout	Temp.	Fans (%control,rpm1,2,3)	AC (volts,power)	Version	Hours
1	-----	-----	-----	---	---	---	---
2	-----	-----	-----	---	---	---	---
3	-----	-----	-----	---	---	---	---
4	-----	-----	-----	---	---	---	---
5	53.99	0.0	29	0% (0.0k,0.0k,0.0k)	117V 17W	FE, 73, 25	105
6	-----	-----	-----	---	---	---	---
7	-----	-----	-----	---	---	---	---
8	-----	-----	-----	---	---	---	---

Vout, Iout, Temp. – standard & extended mode reads
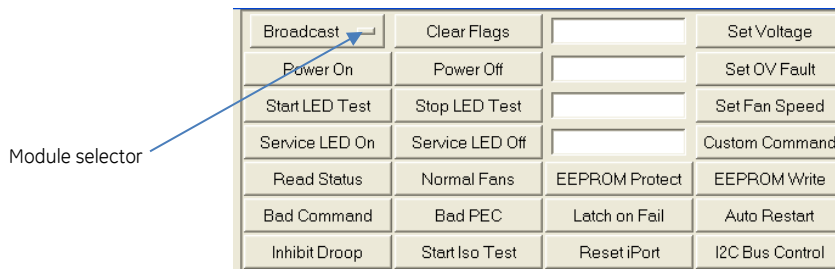Fans – commanded duty cycle(RPM#1,RPM#2,RPM#3)
AC – input voltage, input power
Version –primary µC, secondary DSP,I²C µC in succession
Hours – Cumulated run time hours

The **digital STATUS and ALARM register information** area shows the last read state of all polled units. The registers are cleared by either clicking on the **Clear Flags** button or by clicking on the **Read Status** button. If the fault or alarm is still persistent after clearing the register flags are reset again. The communication fault bits are an exception. These bits must be cleared using the **Clear Flags command**.

7.5   Commands Area

This area of the tool provides access to a number of commands. Each of the commands is summarized below

Broadcast ▾	Clear Flags		Set Voltage
Power On	Power Off		Set OV Fault
Start LED Test	Stop LED Test		Set Fan Speed
Service LED On	Service LED Off		Custom Command
Read Status	Normal Fans	EEPROM Protect	EEPROM Write
Bad Command	Bad PEC	Latch on Fail	Auto Restart
Inhibit Droop	Start Iso Test	Reset iPort	I2C Bus Control

Module selector

Broadcast/module: The user may select between addressing an individual module or all modules simultaneously as indicated by the 'broadcast' label. In the view above, the GUI indicates that 'broadcast' is the present addressing choice. To change to a specific address click on the **module selector** tab that opens a selection menu showing all possible combinations. Select the module to be addressed. The GUI responds by showing the new address in the box, i.e. unit 1 or unit 2 etc. Only a single module, or all modules, can be selected.

Clear_Flags: The alarm states of the four status and alarm registers can be cleared by clicking on this button. Note that the cleared states of the units will not be observed on the screen until **Read Status** is clicked to refresh the rectifier status in the tool. An alternate approach is to poll the status of all units periodically to maintain a continuously refreshed status display. If the fault/warning condition in the unit persists, the refreshed display will get reinstated into its alarm state to indicate the continuing fault/warning condition.

Power On and Power Off: The main output of the unit is commanded to turn ON or OFF via these commands.

Start LED Test and Stop LED Test: These buttons command the unit to execute the LED test which starts/stops the blinking of all LEDs simultaneously. The ON time is longer than the OFF time. The ON time is chosen to enable the user to scan through a large number of units operating in parallel.

Service LED ON and Service LED OFF: These buttons exercise the Service LED commands.

Read Status: This button executes a single **read** to the selected unit.

Normal Fans: This button relinquishes control of the unit fan speed to the unit controller.

Bad command and Bad PEC: These commands purposely issue either a bad command or an incorrect PEC so that the user could verify that the power supply properly sets the correct STATUS flags identifying the incorrect instruction. An executed read back should verify the correct behaviour .

Inhibit droop: This command is only applicable to units such as the PEM that incorporate droop regulation where the output voltage decreases proportionately to output current. The command tells the unit to ignore drooping. There isn't a command to revert back to droop. The only way to reinstate droop is to unplug and reinsert the unit in a multiple unit arrangement. (This is required in order to remove the bias voltage from the control DSP and thus reset it into its default mode).

Start Iso Test: This command instructs the selected unit to commence an isolation test. The isolation test can only be performed when multiple units are being tested. Prior to initiating this command the user needs to verify that the other units on-line have sufficient capacity to provide the output power without the unit being tested. If sufficient power is not available the units on-line will go into a power /current limit mode that may negate the validity of the test.

Note: Since the verification of sufficient power capacity is a contingency for validity of this test, a test failure is provided as 'information only' and does not trigger an automatic power fault. The using system must determine whether the unit is in fact faulty.

Set Voltage and Set OV fault: The output voltage of the unit and the OV shutdown set point can be changed with these two buttons. The changes are only temporary. Removal of the bias voltage to the DSP would reset these register values into their defaults.

Set Fan Speed: This button can be used to increase the speed of the fans beyond what the unit requires for its internal operation. The entry is in % duty cycle.

Custom Command: This button enables the exercising of a command that is not listed in this command area. The command needs to be entered in Hex and it should be supported by the rectifier or PEM.

EEPROM Protect and EEPROM Write: These commands instructs the unit to either 'enable' or 'disable' the ability to write into the upper ¼ of memory space of the EEPROM. The EEPROM in the CP platform contains 64kbits. Memory locations 0 x 1800 and above are in the protected section of the EEPROM.

Latch on Fail and Auto Restart: These buttons instruct the unit to either 'latch' after a OV, OC, or OT fault or 'restart' after an OV, OC, OT fault. The instruction cannot be commanded to individual functions. These commands do not change the default state of the unit.

Reset Iport: This function is no longer supported. It was used prior to the introduction of the **GE Interface Adapter.**

I2C Bus Control: The rectifier or PEM has two independent and isolated I²C ports. This button instructs the PEM to take over control to the I²C port this application is connected to. This feature is further discussed in the next section.

7.6   Communicating with the External EEPROM

This section of the tool enables writing to and reading data from the external EEPROM. Besides storing and reading data the 'write protect' feature protection the upper ¼ of memory can be demonstrated.
The tool can select which of the 8 possible units is being communicated to. Clicking on the little square to the right of the EEPROM1 labeled button brings up a dropdown menu that can select any of the 8 units.
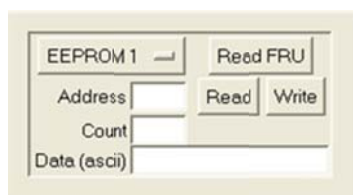
Read FRU: This button extracts the FRU_ID information that is stored in the EEPROM. The information is displayed in the Command Log area of the tool. It is also stored in the test record file that has been created automatically by the tool.

Address: The starting hex memory location for either the **Read** or **Write** functions needs to be entered next to the address box prior to clicking either of these two functions. Note that a **Read** also needs the count area to be populated.

Count: Tells the tool how many bytes should be read back. The entry is a number indicating how many bytes are to be read back. The count number needs to be populated prior to clicking on the **Read** function.

Data (ASCII): The read back information is displayed in this box. The Command Log displays both the hex and ascii equivalent of the read back information. This information gets logged into the test record file.

7.7   Command/Data Log, Log File and Bus Traffic Areas

7.7.1   Command/Data Log

Another feature of the CP GUI is its automated time stamp driven capture of all changes in the STATUS and ALARM registers, including the execution of all instructions. Four different types of events can be time stamp recorded and these are:
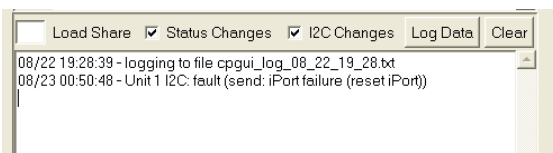
Load share – an entry in this table compares the difference in current readings between the units. If the difference is greater than the value entered, this data will be recorded.
Status changes – records changes in STATUS and ALARM registers
I2C changes – records initiated commands and communications errors

Log Data: Records the analog data captured on the screen. This is a one- time stamped event. The time stamp captures the date, time of the screen capture, and the unit # transmitting the information. If the Standard read is selected then the three parameters recorded are tabulated. The unit # is also identified. If the Enhanced read is selected, then all other parameters are recorded.

Standard read capture

Enhanced read capture

Clear: Clicking this button clears the Log screen, however, it does not erase the data that has already been captured in the Log file.

7.7.2   Log File

The Log File is recorded automatically in the same directory where the executable .exe file resides. If the downloaded executable is located on the desktop than the Log file is also automatically placed on the desktop. It is named automatically by date i.e. 6_27 and time(hours_minutes) recorded 13_25 in military time (1:25PM),
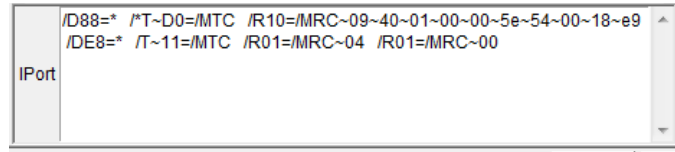

The log file can be renamed and located into any directory just like any other file. Or, if not needed, simply delete it.

7.7.3   Bus Traffic

The latest bus traffic transmitted over the I2C bus is shown above the Command Log. This detailed information of bus activity is not trivial to decode. The commands are a conglomerate of the Interface Adapter command set, the I2C communications protocol of the CPL platform and the command set of the PCA9541 multiplexer. This data is

normally for information only. The contents have already been translated and displayed by the GUI into the various sections.

Never the less, the information presented may be useful for programmers who are attempting to communicate to the rectifier or PEM but have trouble doing so. The data here shows what is being transmitted over the bus, albeit, commands between the adapter and the computer overshadow some of the information.

```
       /D88=*  /*T~D0=/MTC  /R10=/MRC~09~40~01~00~00~5e~54~00~18~e9    ▲
       /DE8=*  /T~11=/MTC   /R01=/MRC~04  /R01=/MRC~00
IPort
                                                                      ▼
```

Translation of the recorded single poll shown below is as follows: μC of the module at address 44[2] is requested to read status [D0], 10 bytes of data are received, the first byte tells the host that 9 data bytes follow, and these are: STATUS – 2 bytes, ALARM – 2 bytes, output voltage – 2 bytes, output current – 1 byte, temperature – 1 byte, followed by the PEC – 1 byte. (The last data byte of each data packet is always the computed PEC). When this instruction is completed the GUI addresses the multiplexer of the module, 0xE8, at device address 74 and command 11[3] is executed. Two successive reads follow obtaining control and interrupt status.

## 7.8 Demonstration of Dual I^{2}C Communications

The tool can be used to demonstrate the dual I^{2}C communications capability of the units. The Interface board connected to the GE shelf brings out both I^{2}C lines of the power supply. There are two ways to demonstrate the capability.

7.8.1 Using a Single Adapter and Laptop

The first way is to connect the GE Interface Adapter into I^{2}C-0 and set up the tool to communicate to the unit. After the establishment of communications (note the communications information displayed by the LEDs in the polling section of the display area) , disconnect the ribbon cable from I^{2}C-0 and insert it into I^{2}C-1. Observe that the polling section of the display area will indicate that the I2C-1 side does not have control if the power supplies are polled. Issue any operational commands and observe that the power supplies do not respond to the commands. Next, click on the **I2C Bus Control** button in the commands area to take over control. Note that LED indicator changes showing that the I^{2}C-1 side now has control. Issue any operational commands and observe that the power supplies respond to the instructions. This method is not the best alternative because of the switching spikes that are observed when the Interface Adapter is removed and inserted into the I^{2}C ports.

7.8.2 Using Two Adapters and Two Laptops

The preferred method of demonstrating dual communications capability is to connect two GE Interface Adapters, one to I^{2}C-0 and the other to I^{2}C-1, each adapter connected to a separate computer. Start the CPGUI in both computers and observe that the computer connected into I^{2}C-0 has control. The computer connected to I^{2}C-1 should indicate that I^{2}C-1 has no control. Control can be taken over manually by either computer by clicking on the **I2C Bus Control** button. Alternatively, drag the cursor below the **Automatic I2C Bus Control** labeled area in the middle of the tool to set the time delay between automated bus takeover on both computers. The time delay can be as long as 30 minutes. Set different times so that you can recognize which side is being used. (a good suggestion is 1 minute and 2 minutes). The set time delay is displayed instead of the **None** that is initially configured. Set both computers to the poll mode, and observe that control is automatically being taken over by the two applications.

```
Automatic I2C Bus Control
        None    [  ▮  ]
```

---
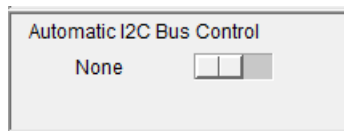
[2] Unit #1 in the 2nd shelf; device 0x88 has the bit pattern 10001000, however, since the LSB is the write/read bit, the actual address is 1000100 which is 0x44. The last three bits are A2, A1, and A0. In the next section **Starting the CPGUI** we explain how we created this address
[3] Command hex byte 0x11 reads the control register, increments by one and reads the interrupt status register of the multiplexer