GE Energy

Digital Power Insight™
User Manual

Table of Contents

1. Introduction

GE Energy Digital Point-of-Load (POL) modules and Digital Bus Converters 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^2C 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™ 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 two tools.

1. A graphically-oriented User Interface, referred to here as the Digital Power Insight 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 the 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 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.
2. The second tool is a low level command line interface tool called the Digital Power Insight 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 both the DPI-GUI and DPI-CLI tools.

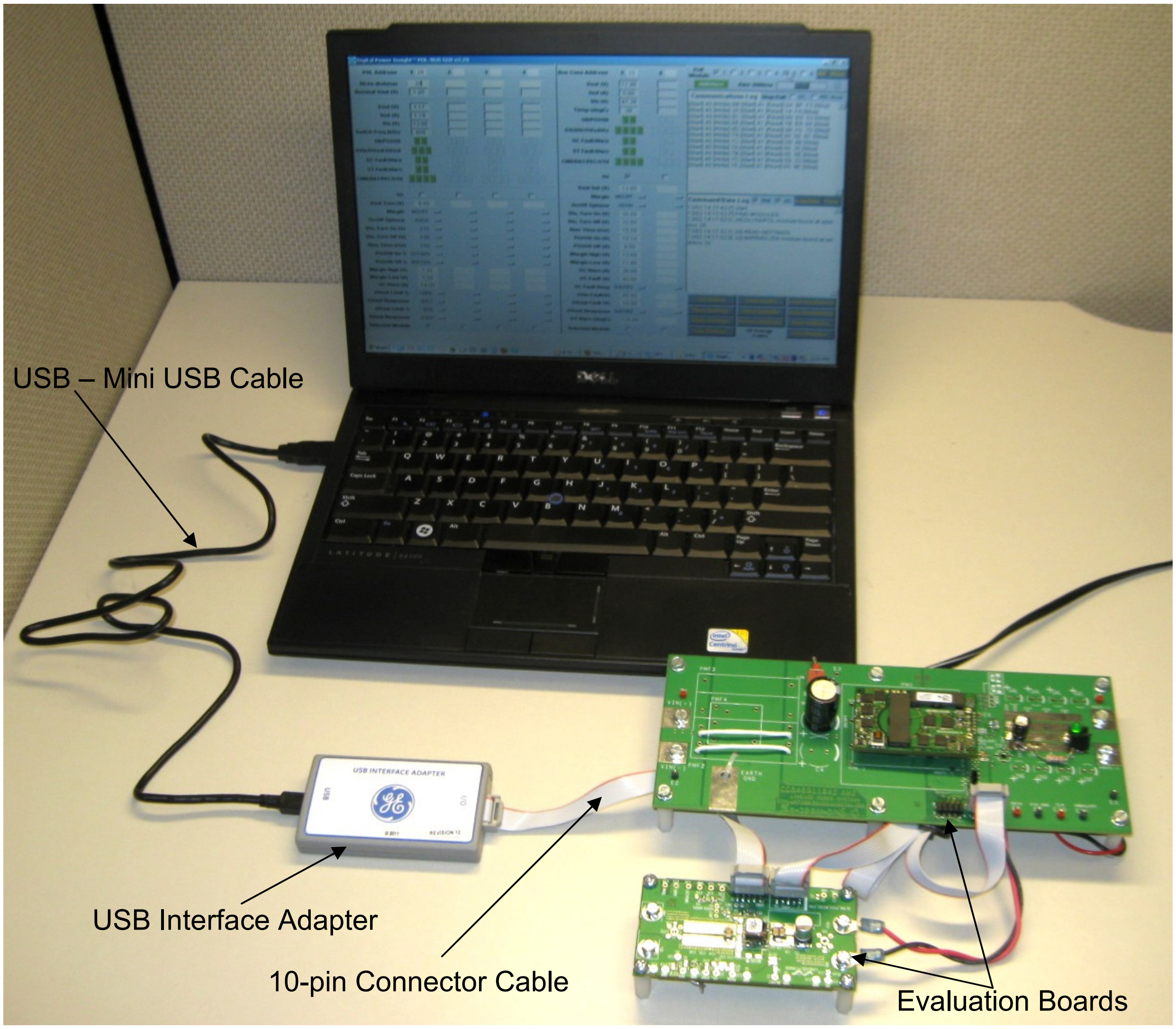
2. Setup

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 Energy
- 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 Energy 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: Digital Power Insight only works with the GE Energy USB Interface Adapter.



2.1 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 I^2C/SMBus commands and also translates data received over I^2C/SMBus back to the PC. This adapter is designed specifically to work with GE Energy products. Check with the GE Energy Technical Support team for other uses.

2.1.1 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.

2.1.2 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

2.1.3 I^2C Clock Speed Settings

The default value of the I^2C 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.

2.2 Installing the DPI Software

The DPI software is installed using a self-executable install file. 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.

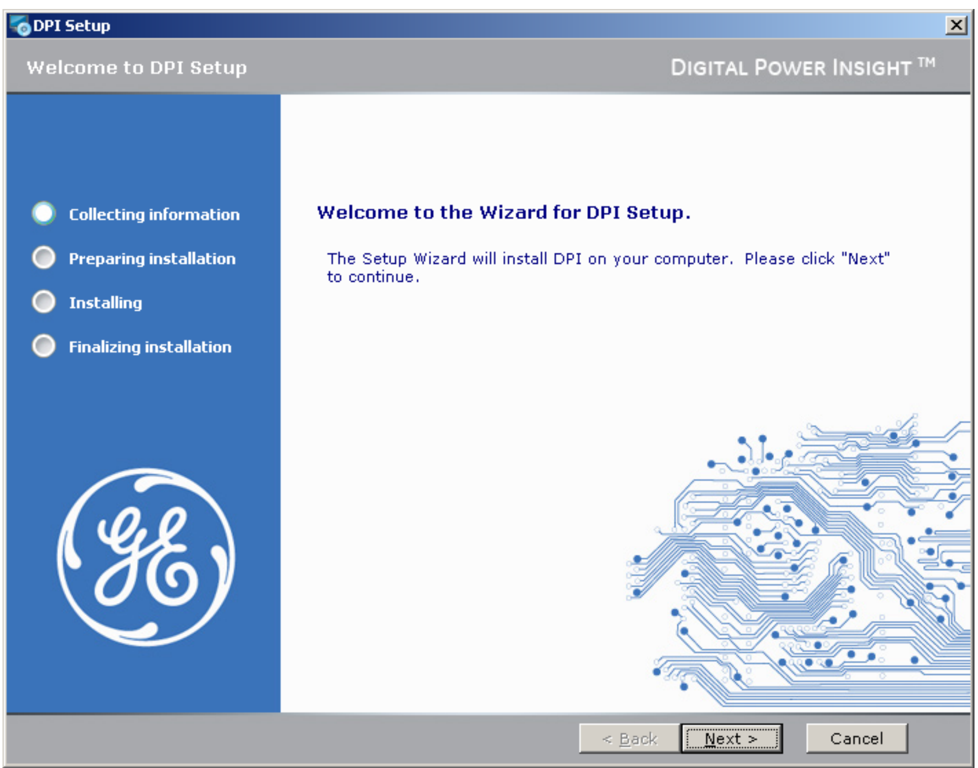
Note:

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

Since the DPI-GUI program also creates a log file 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-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 Energy USB Interface Adapter. See 2.2.1 for instructions on installing the driver file.



2.2.1 USB Interface Adapter driver setup

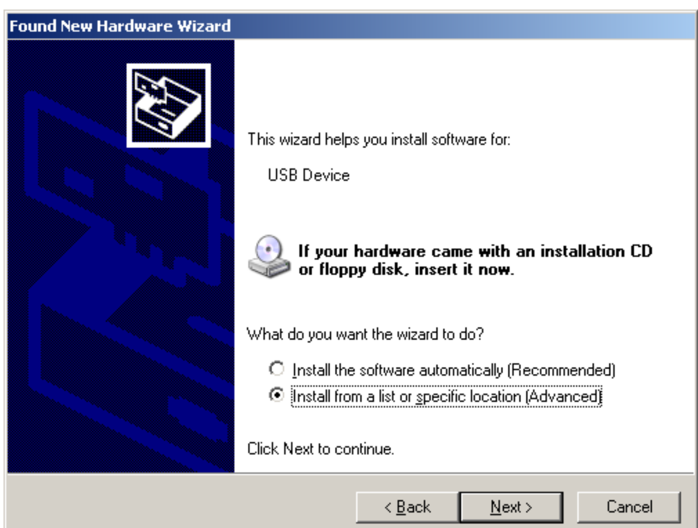
The files

GE_Power_Electronics_USB_Interface_Adapter.inf and ***GE_Power_Electronics_USB_Interface_Adapter_64bit.inf***

contain the Windows driver information. During the DPI software installation, these files are automatically placed in the C:\WINDOWS\System32 directory.

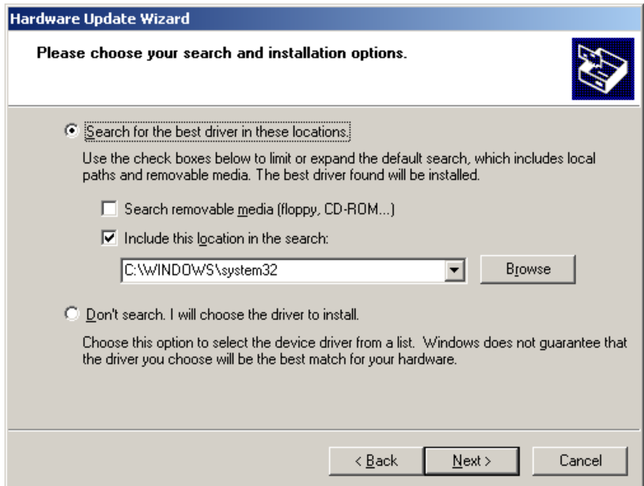
Plug the USB Interface Adapter into an open USB port on your computer. The computer should recognize that new hardware has been connected and it will ask you for a location for the driver for this new hardware.

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.

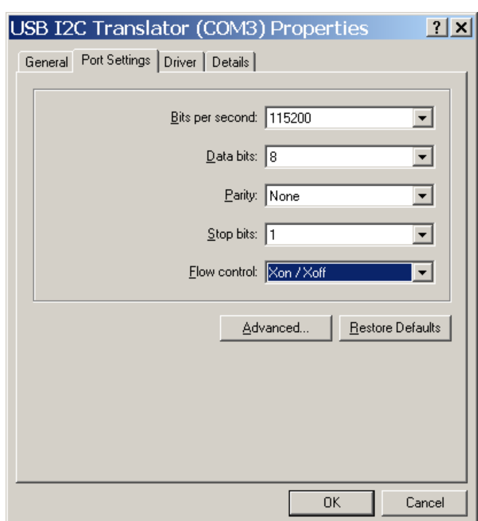


Select **Search for the best driver in these locations,** followed by **Include this location in the search** and enter C:\WINDOWS\System32

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.

2.2.2 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.

2.3 Files Created by DPI

The 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 4.2.18. 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. DPI-GUI

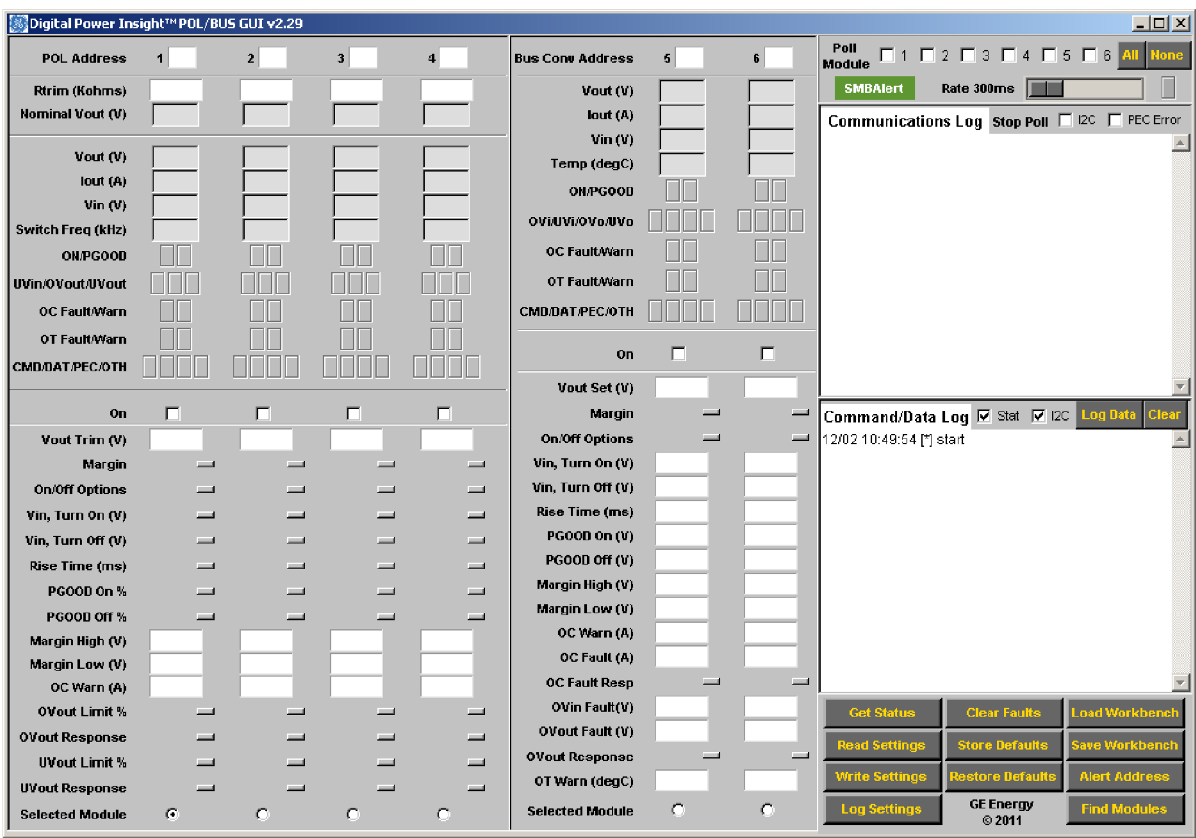
The DPI-GUI is a graphically-oriented tool for interfacing to up to six modules. The tool capabilities include

- Identifying and configuring the addresses of up to six modules connected to the I^2C bus interfaced by the GE Energy 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 I^2C bus
- Automatically storing all commands and status changes with associated time stamps in a log file

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

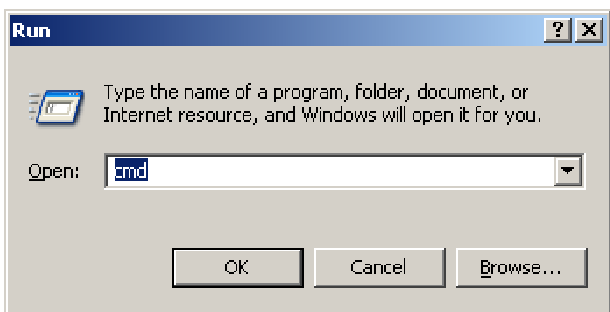
In some cases, this can be corrected by removing and re-inserting the USB cable connected to the GE Energy USB Interface adapter, and restarting the DPI-GUI program. Correcting this condition may require at least two reinsertions .To exit the DPI-GUI program, just click on the X in the top right corner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

3.1.1 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.

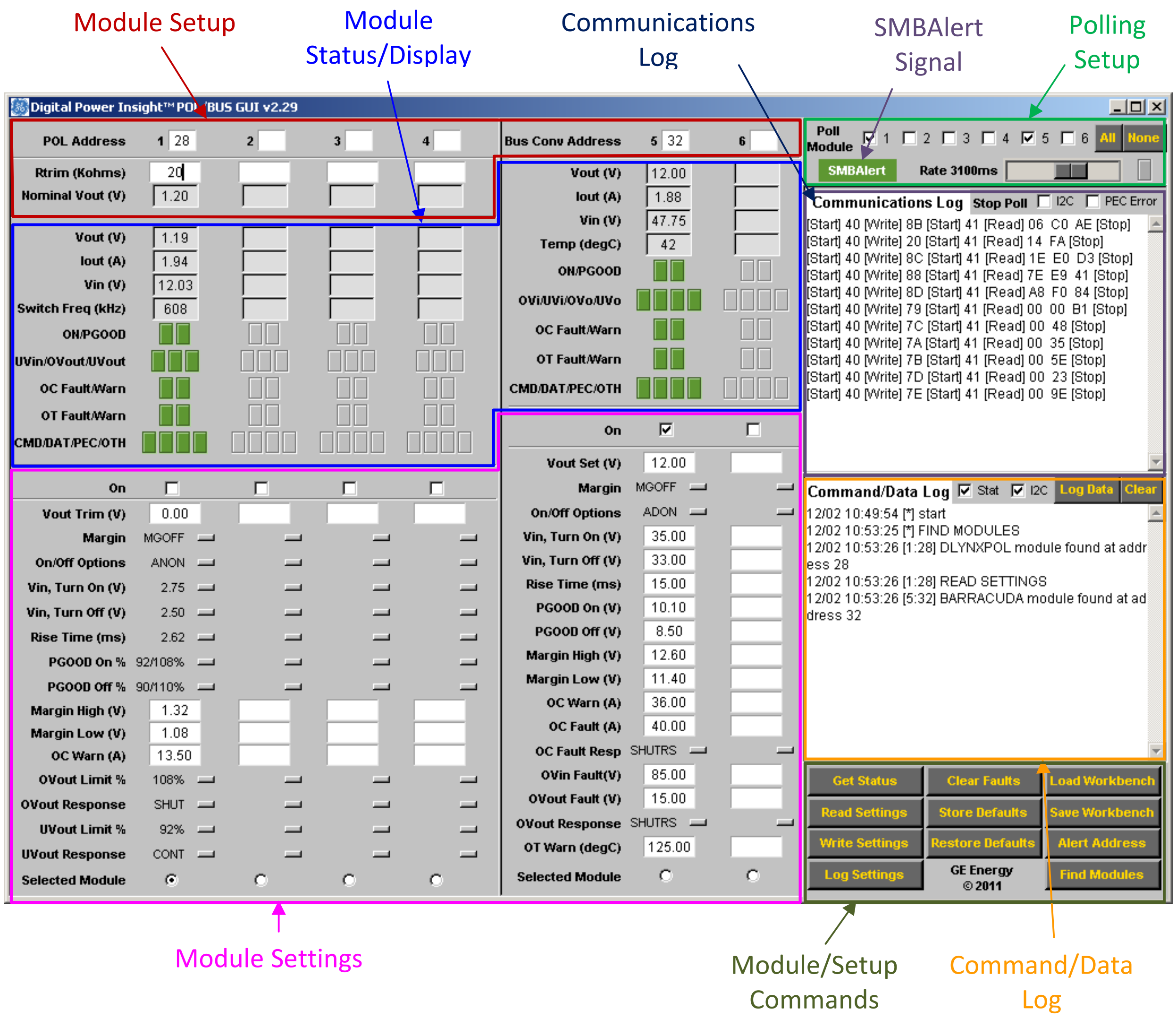
For example, typing the command

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

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

3.2 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.



3.2.1 Module Setup

The DPI-GUI can simultaneously display up to six modules. In the default configuration, four POL modules 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.

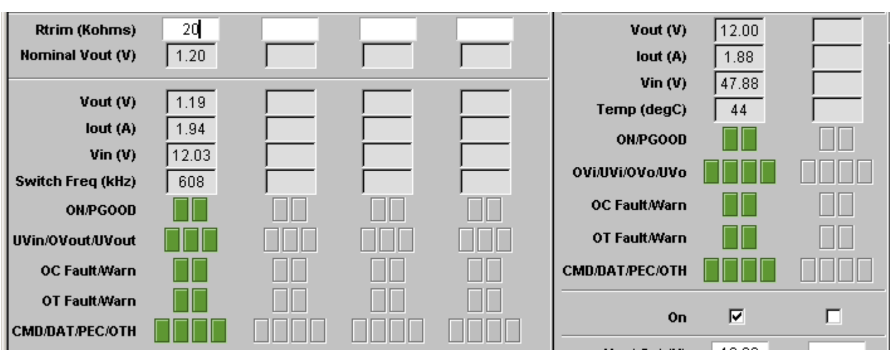
Please refer to the module data sheet for additional information on how the module address and RTrim are specified for the particular module.

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.

3.2.2 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.

3.2.3 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.

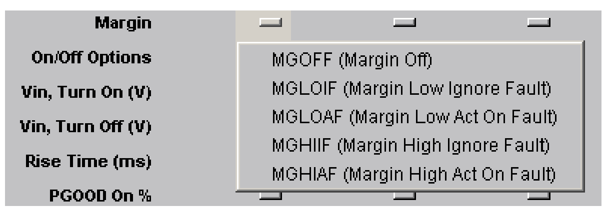
3.2.4 Module Settings Area (POL)

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.

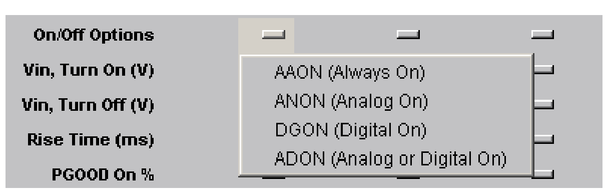
Vout Trim (V) : The offset to the nominal output voltage value can be entered here. The module output voltage will then be changed the new value (Vout,nom + Vout Trim) when the parameters are "written" to the module. This parameter applies only to POL modules.

Margin : Clicking the button in line with this parameter opens a drop down menu as shown below

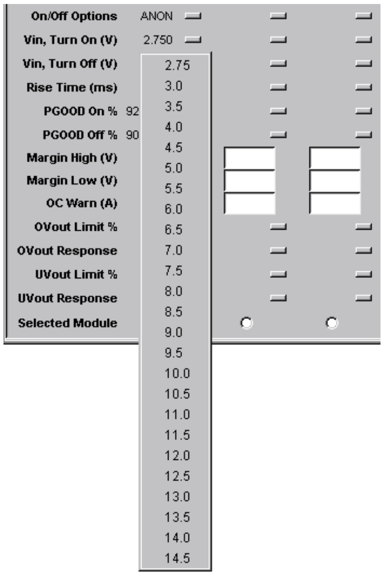


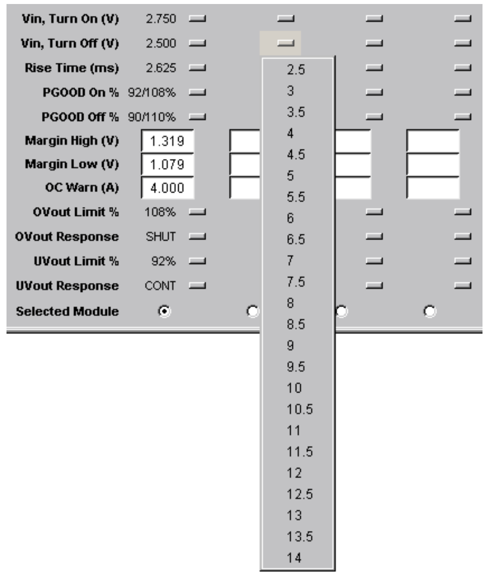
The drop down menu has five choices. For all dropdown menus, the pointer location is highlighted by the background. To select a particular choice, click on the highlighted area within the drop down menu. Both the mnemonic label and its meaning are shown (e.g. MGOFF for Margin Off).

On/Off Options : When this button is clicked, the parameter choices shown below are displayed.

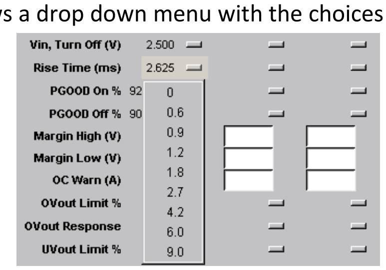


Vin, Turn On (V) and Vin,Turn Off (V): Clicking these buttons show the drop down menus with the choices shown below for POL Modules.

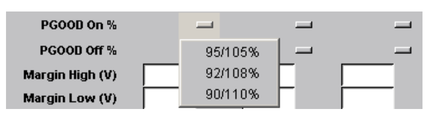




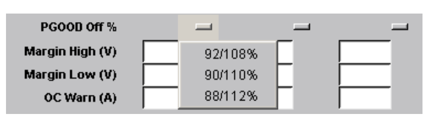
Rise Time (ms) : Clicking this button shows a drop down menu with the choices shown below for POL Modules.



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.

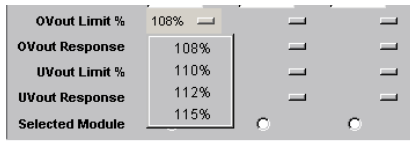


Margin High (V) : The margin high voltage of the module can be entered here. The module goes to the commanded margin high voltage when Margin High is enabled and the new settings are written to the module.

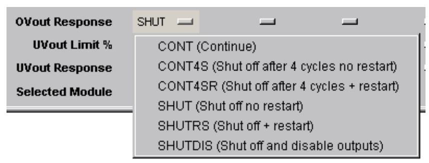
Margin Low (V) : The margin low voltage of the module can be entered here. The module goes to the commanded margin low voltage when Margin Low is enabled and the new settings are written to the module.

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

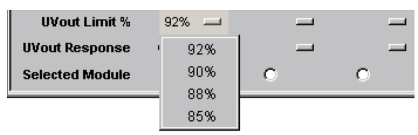
OVout Limit % : Clicking this button shows a drop down menu with the choices shown below for POL Modules. The user can select the Output Overvoltage protection threshold from the available choices expressed as a percentage of the set output voltage.



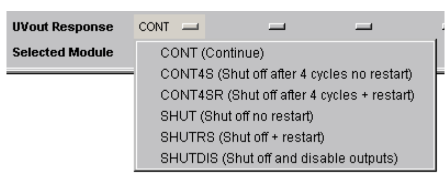
OVout Response : Clicking this button shows a drop down menu with the choices shown below for POL Modules. These choices control the module response in the event of an output overvoltage condition.



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.

3.2.5 Module Settings Area (Bus Converter)

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

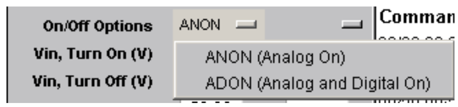
On : Clicking the white box will insert a **√** indicating that the module is to be turned ON. For this feature to actively control the ON/OFF of the module, the On/Off Options should be set to ADON (Analog and Digital ON). Note that clicking the module ON or OFF will immediately turn the module ON or 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.

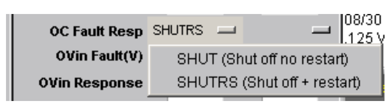
Margin High (V) : The margin high voltage of the module can be entered here. The module goes to the commanded margin high voltage when Margin High is enabled and the new settings are written to the module.

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.

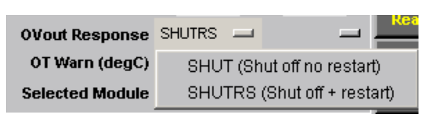
OC Fault Resp : Clicking this button displays the drop down menu with the choices shown below for Bus Converters.



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.

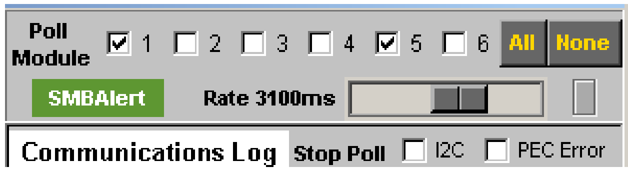


OT Warn (degC) : Allows the Module Temperature Warning threshold in °C to be specified for Bus Converter modules.

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.

3.2.6 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.



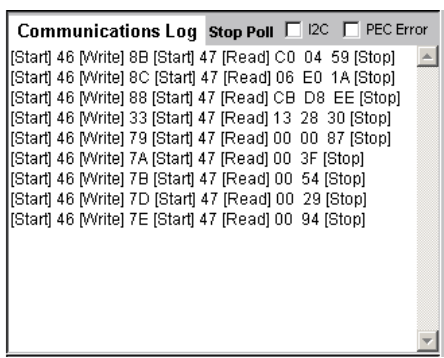
The modules to be polled can be selected along with the desired polling rate (from roughly once every 200ms up to once every 5000ms or 5s). The polling can also be set to stop on I^2C or PEC (Packet Error Checking) errors. The LED button to the right of the polling rate is an indicator that blinks for 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).

3.2.7 Communications Log Area

This area of the DPI-GUI tool shows the I^2C communications traffic in a simplified hex format. Note that what is displayed in this area is only a representation of the I^2C 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.



3.2.8 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

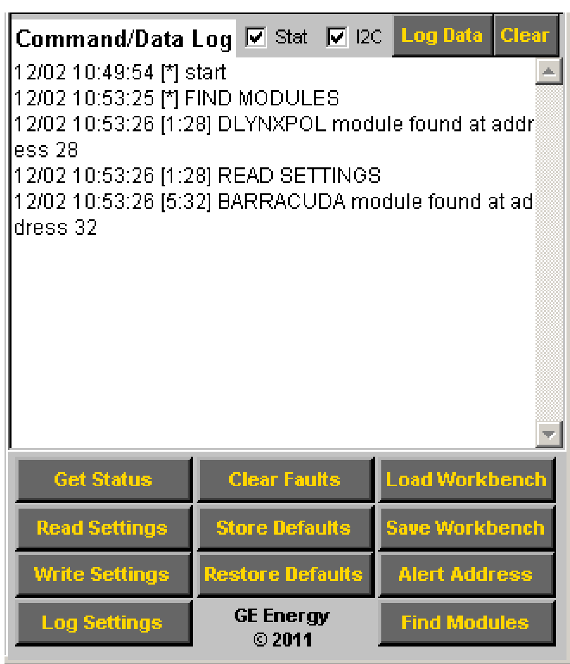
Clear: Erases the content in the Log window but does not erase the contents of the text file.

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.



4. DPI-CLI (Command Line Interface)

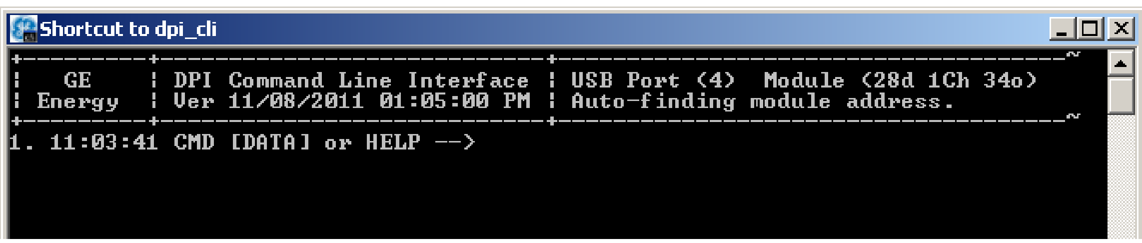
The DPI-CLI is a low-level command-line based tool that is useful for decoding and analyzing basic communications with the module. The low level capability facilitates debugging when specific command interactions need to be examined in detail. The DPI-CLI also has a powerful scripting and data capture/logging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

4.1 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



The tool automatically finds all modules with valid addresses connected to the I^2C 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.

4.2 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 I^2C 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.

4.2.1 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.

4.2.2 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
```

4.2.3 C or COMMAND Function

Some PMBus™ commands do not have a follow-on data byte and these use the COMMAND function instead of the WRITE function. The following PMBus™ commands use the COMMAND function.

- Clear_Faults 03h
- Store_Default_All 11h
- Restore_Default_All 12h
- Store_Default_Code 13h
- Restore_Default_Code 14h

Examples:

```
>COMMAND CLEAR_ALL
>COMMAND 03
>C 14
```

4.2.4 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.

4.2.5 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 *)

4.2.6 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.

4.2.7 K - Clock Setting Function

This function is used to read or change the I^2C 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 I^2C 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)

4.2.8 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).

4.2.9 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).

4.2.10 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.

4.2.11 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.

4.2.12 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
```

4.2.13 Q or QUIT Function

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

4.2.14 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
```

4.2.15 REGINFO Function

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

Example:

```
>REGINFO
```

4.2.16 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
```

4.2.17 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
```

4.2.18 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.

Example:

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.

4.2.19 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.

4.2.20 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.

4.3 Summary of Supported PMBus™ Commands for 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:** `>c 03`	03h	c			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:** `> c 11`	11h	c	.		Store all the configuration settings into non-volatile memory
RESTORE_DEFAULT_ALL **Example:** `> c 12`	12h	c			Get all configuration settings from non-volatile memory
STORE_DEFAULT_CODE **Example:** `> c 13 29`	13h	c			Store parameters associated with specified command into non-volatile memory
RESTORE_DEFAULT_CODE **Example:** `> c 14 29`	14h	c	.		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
FREQUENCY_SWITCH **Example:** `> r 33`	33h	r		Real Number	Read value of the switching frequency

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
VIN_OFF **Example:** > r 36 > w 36 2.8	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
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
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 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
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 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
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
				Cutoutput	Set module to cut off output on output voltage OV fault and remain OFF until fault is cleared

Command	Code	Capability	Name	Value	Function
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_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_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
VIN_CAL_GAIN **Example:** `>r D7` `>w D7 -0.03`	D7h	r/w		Real Number	Read/write gain factor correction for input voltage measurement

Command	Code	Capability	Name	Value	Function

ALERT_ADDRESS
Example:
`>r alert_address`
`> r a`
	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.

4.4 Summary of Supported PMBus™ Commands for Bus 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
				ActLow	Margin Low but act on 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
			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:** >c 03	03h	c			Clears the Status registers and resets the SMBAlert# signal.
STORE_DEFAULT_ALL **Example:** > c 11	11h	c			Store all the configuration settings from working memory into non-volatile memory
RESTORE_DEFAULT_ALL **Example:** > c 12	12h	c			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 **Example:** > r 41 > w 41 RSP:Shutdown,RS:TryRestart Note: entry required only for the changing function	41h	r/w	RSP	Shutdown	Set module to shutdown on output voltage OV fault
			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
 	
Example: `> r 47`
`> w 47 RSP:Shutdown,RS:TryRestart`
 	
`Note: entry required only for the changing function`
	47h	r/w	RSP	Shutdown	Set module to shutdown on output current OC fault

			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
Example: `> r 50`
`> w 50 RSP:Shutdown,RS:TryRestart`
 	
`Note: entry required only for the changing function`
	50h	r/w	RSP	Shutdown	Set module to shutdown on Overtemperature fault

			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 SEC:ON,POL:POS`	E2h	r/w	**POL**	NEG	Negative signalling polarity of the PGOOD pin
				POS	Positive signalling polarity of the PGOOD pin
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.

GE Energy

World Wide Headquarters
GE Power Electronics Inc.
601 Shiloh Road, Plano, TX 75074, USA
+1-888-546-3243
(Outside U.S.A.: **+1-972-244-WATT(9288)**)
www.ge.com/powerelectronics
e-mail: pe.techsupport1@ge.com

Asia-Pacific Headquarters
Tel: +86.021.54279977*808

Europe, Middle-East and Africa Headquarters
Tel: +49.89.878067-280

India Headquarters
Tel: +91.80.28411633