PGA900 Pressure and Temperature Sensor Signal Conditioner

User's Guide

Literature Number: SLDU016A
May 2015–Revised November 2015

Contents

List of Figures

Introduction

The PGA900 user’s guide provides a general overview of the PGA900 evaluation module (EVM) GUI which includes:

- Required software installations
- Description of the features of the GUI
- Functions to be considered while using the GUI

Chapter 1

SLDU016A–May 2015–Revised November 2015

GUI Software Installation

The PGA900 GUI allows the user to communicate to the PGA900 EVM. The following section explains the location and the procedure for installing the software properly.

NOTE: Do not make any USB connections to the EVM until the installation is complete.

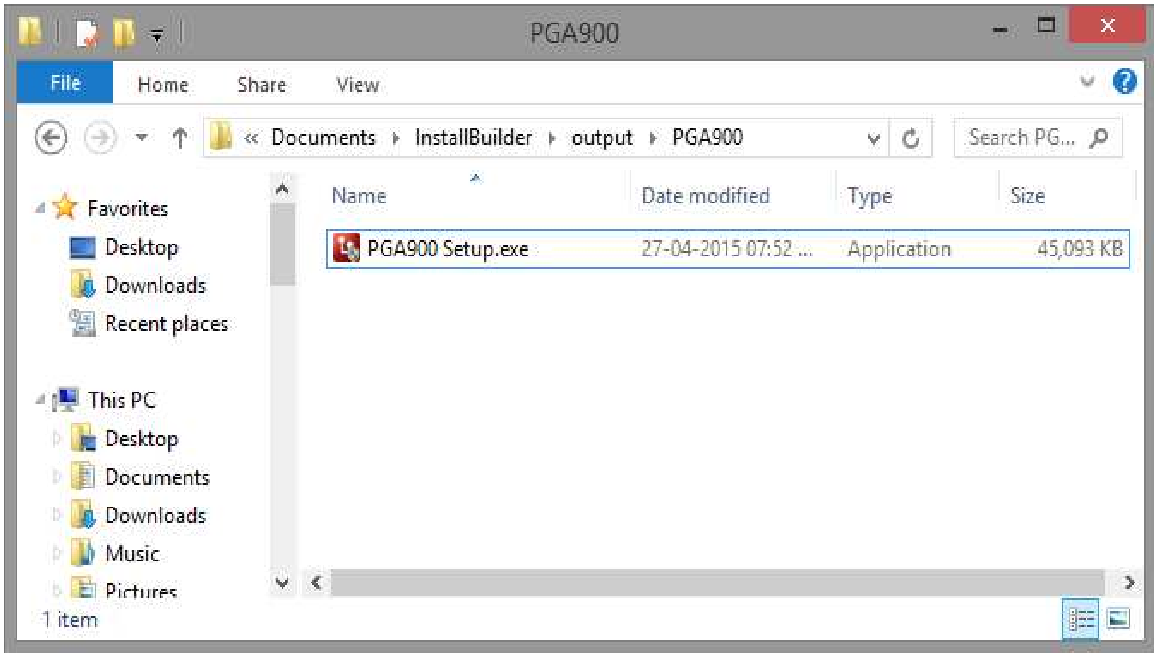
1.1 System Requirements

- Supported OS – Windows® XP, Windows 7 (32 bit, 64 bit)
- Recommended RAM memory – 4GB or higher
- Recommended CPU operating speed – 3.3 GHz or higher

1.2 Installation Procedure

Follow this procedure to install the PGA900 GUI.

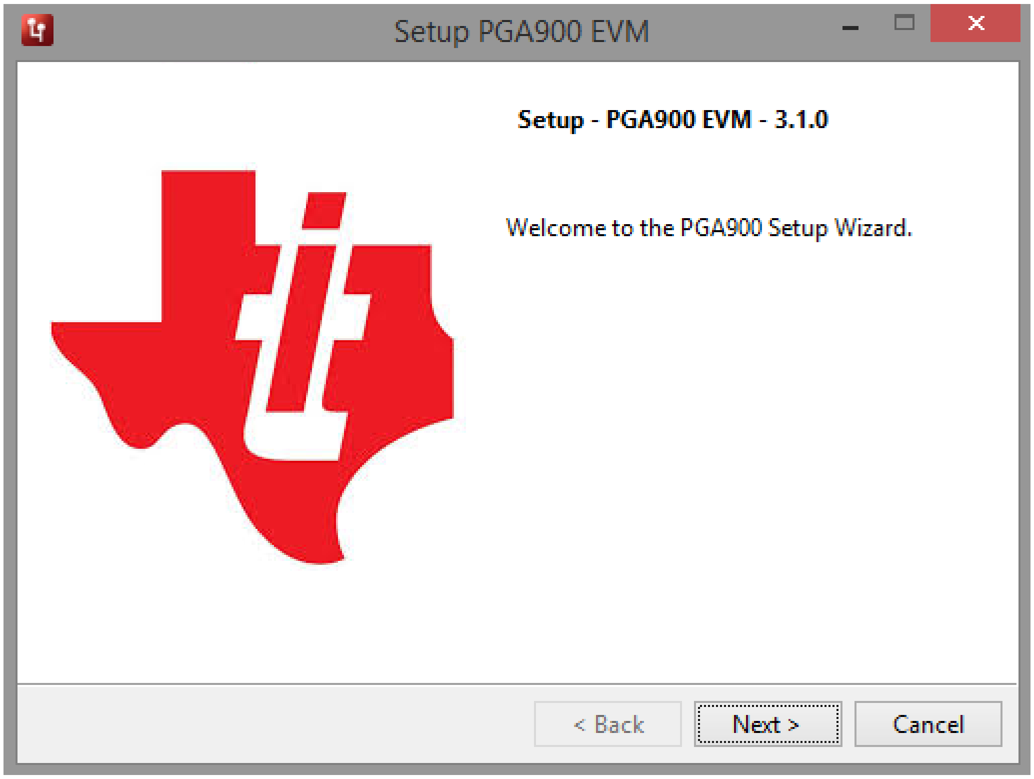
1. Double click on the Setup.exe from the Volume folder as shown in Figure 1-1.



Name	Date modified	Type	Size
PGA900 Setup.exe	27-04-2015 07:52 ...	Application	45,093 KB

Figure 1-1. Setup.exe from Volume Folder

The screen shown in Figure 1-2 appears.

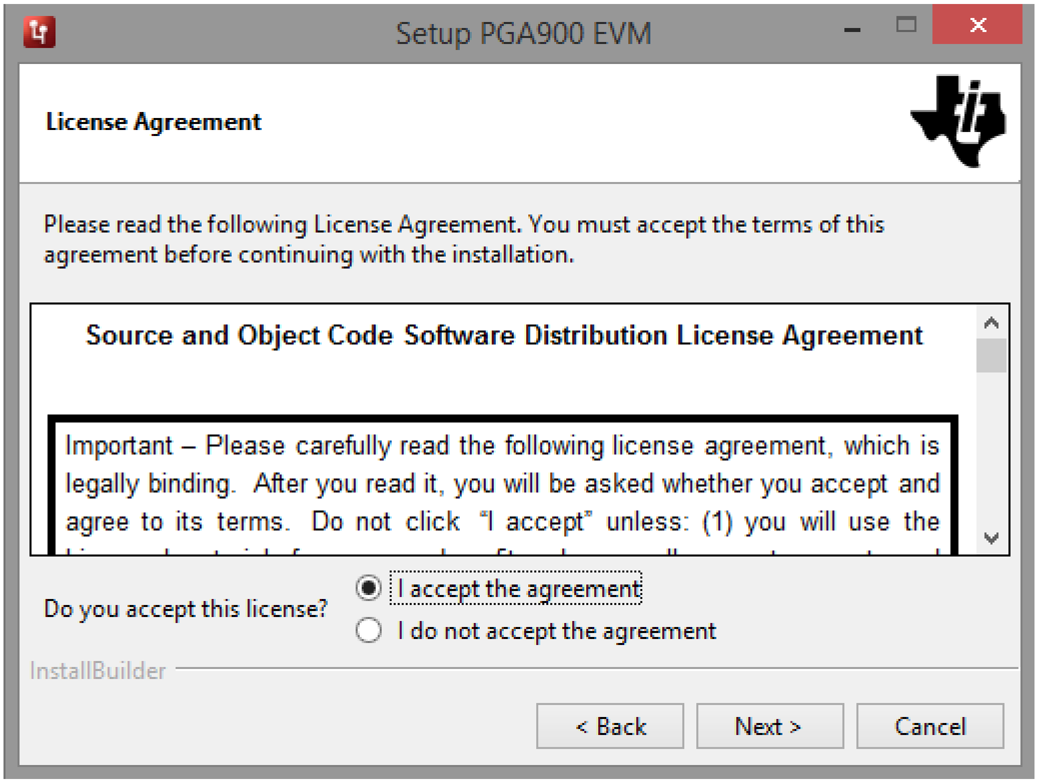


Setup - PGA900 EVM - 3.1.0

Welcome to the PGA900 Setup Wizard.

Figure 1-2. Installation Initialization

2. The license agreement for PGA900 GUI appears as shown in Figure 1-3. Read through the agreement and enable the “I accept the agreement” radio button and click the **Next >** button



Setup PGA900 EVM

License Agreement

Please read the following License Agreement. You must accept the terms of this agreement before continuing with the installation.

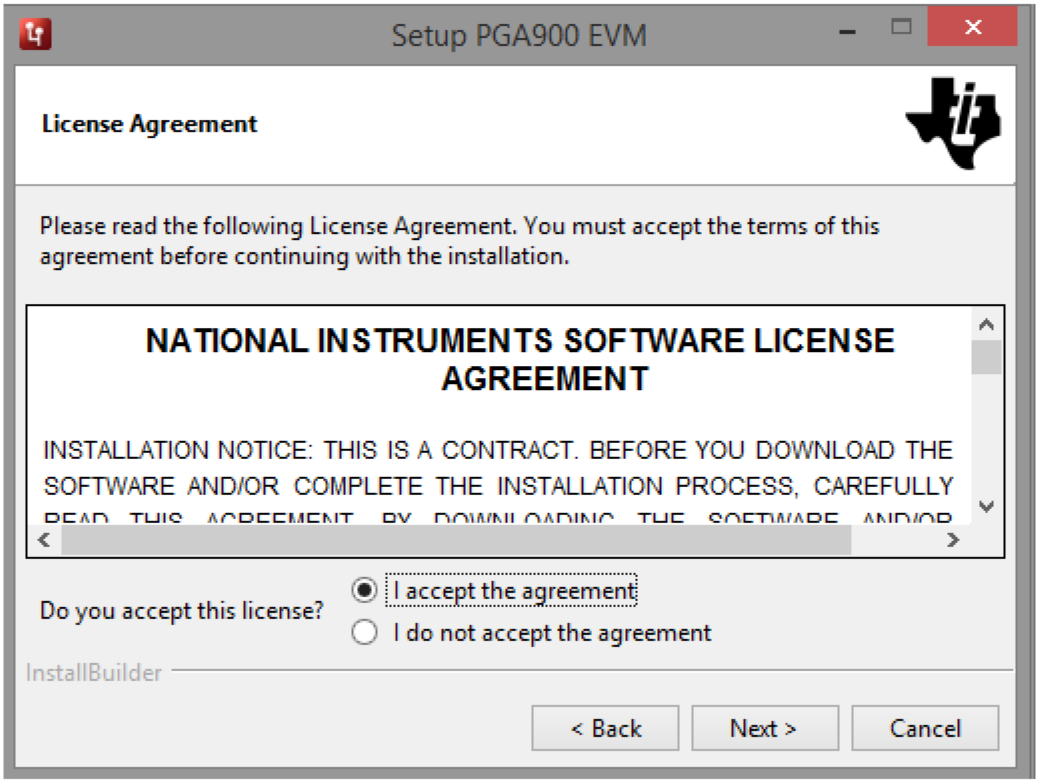
Source and Object Code Software Distribution License Agreement

Important – Please carefully read the following license agreement, which is legally binding. After you read it, you will be asked whether you accept and agree to its terms. Do not click “I accept” unless: (1) you will use the

Do you accept this license?
● I accept the agreement
○ I do not accept the agreement

Figure 1-3. GUI Software Evaluation and Internal Use License Agreement

The license agreement for National Instruments™ will appear as shown in Figure 1-4. Read through the agreement and enable the "I accept the agreement" radio button and click the **Next >** button.



License Agreement

Please read the following License Agreement. You must accept the terms of this agreement before continuing with the installation.

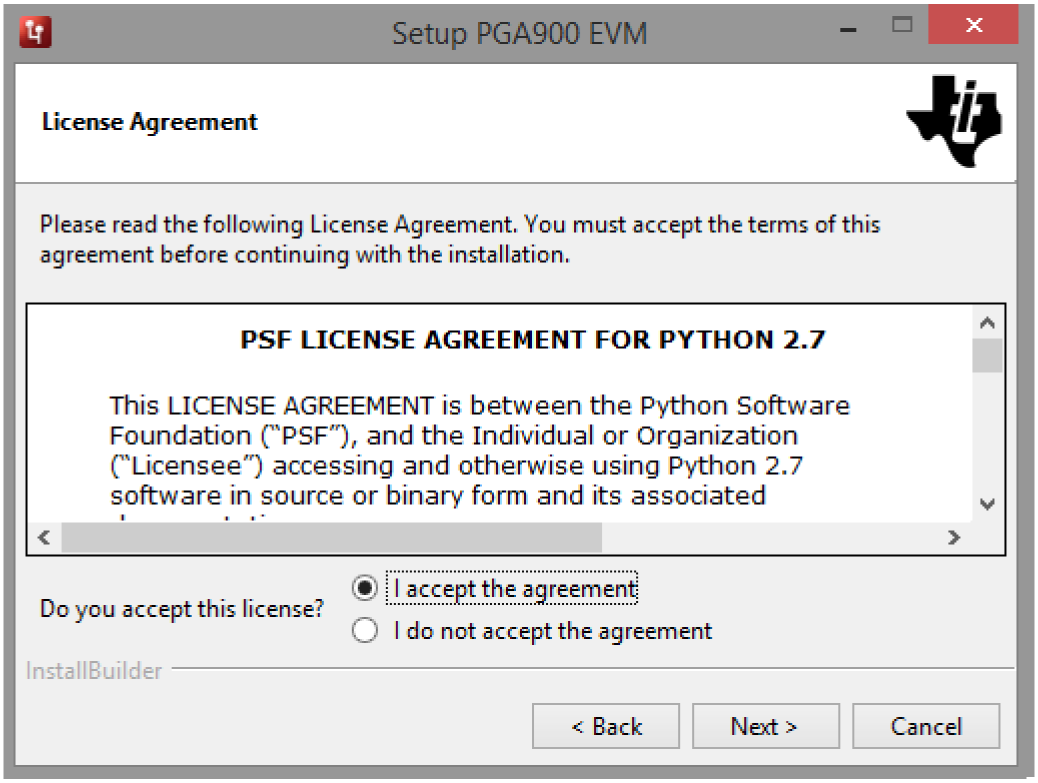
NATIONAL INSTRUMENTS SOFTWARE LICENSE AGREEMENT

INSTALLATION NOTICE: THIS IS A CONTRACT. BEFORE YOU DOWNLOAD THE SOFTWARE AND/OR COMPLETE THE INSTALLATION PROCESS, CAREFULLY READ THIS AGREEMENT. BY DOWNLOADING THE SOFTWARE AND/OR

Do you accept this license?
● I accept the agreement
○ I do not accept the agreement

Figure 1-4. License Agreement for National Instruments

The license agreement for Python® 2.7 appears as shown in Figure 1-5. Read through the agreement and enable the “I accept the agreement” radio button and click the **Next >** button.



Please read the following License Agreement. You must accept the terms of this agreement before continuing with the installation.

PSF LICENSE AGREEMENT FOR PYTHON 2.7

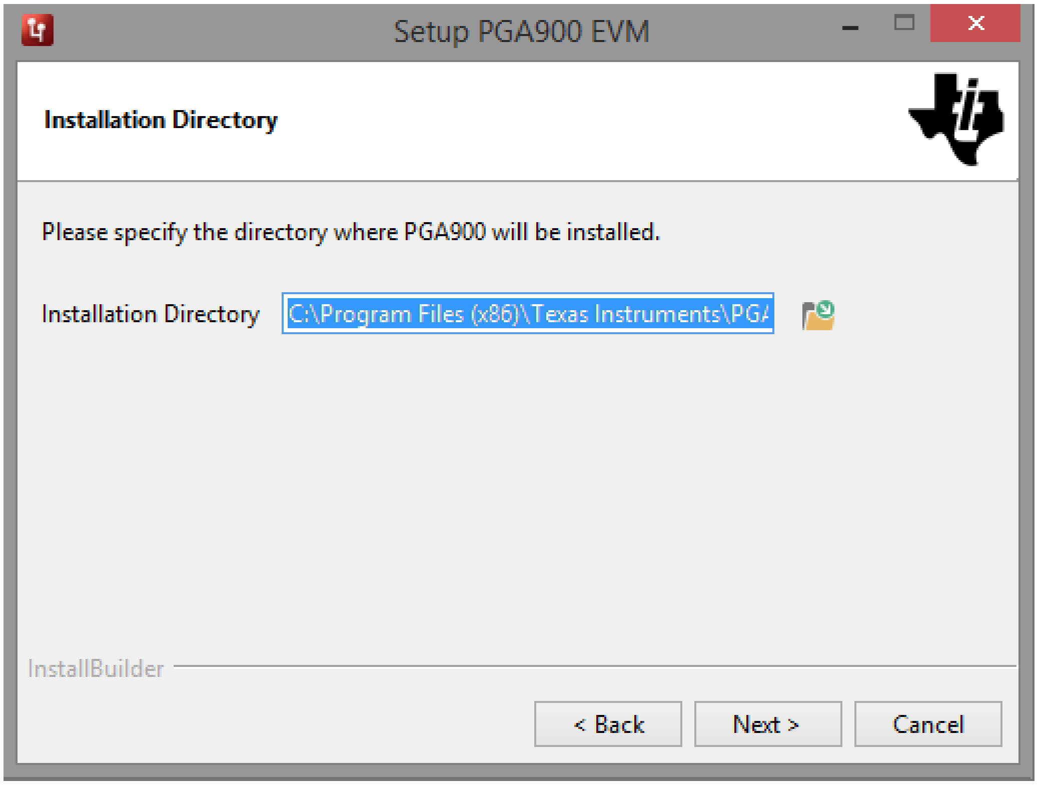
This LICENSE AGREEMENT is between the Python Software Foundation ("PSF"), and the Individual or Organization ("Licensee") accessing and otherwise using Python 2.7 software in source or binary form and its associated

Do you accept this license? ● I accept the agreement ○ I do not accept the agreement

Figure 1-5. PSF License Agreement for Python 2.7

NOTE: TI highly recommends to keep the default values as provided in the installer.

3. Set the destination directories for the PGA900 GUI installation and click the **Next >** button as shown in Figure 1-6.

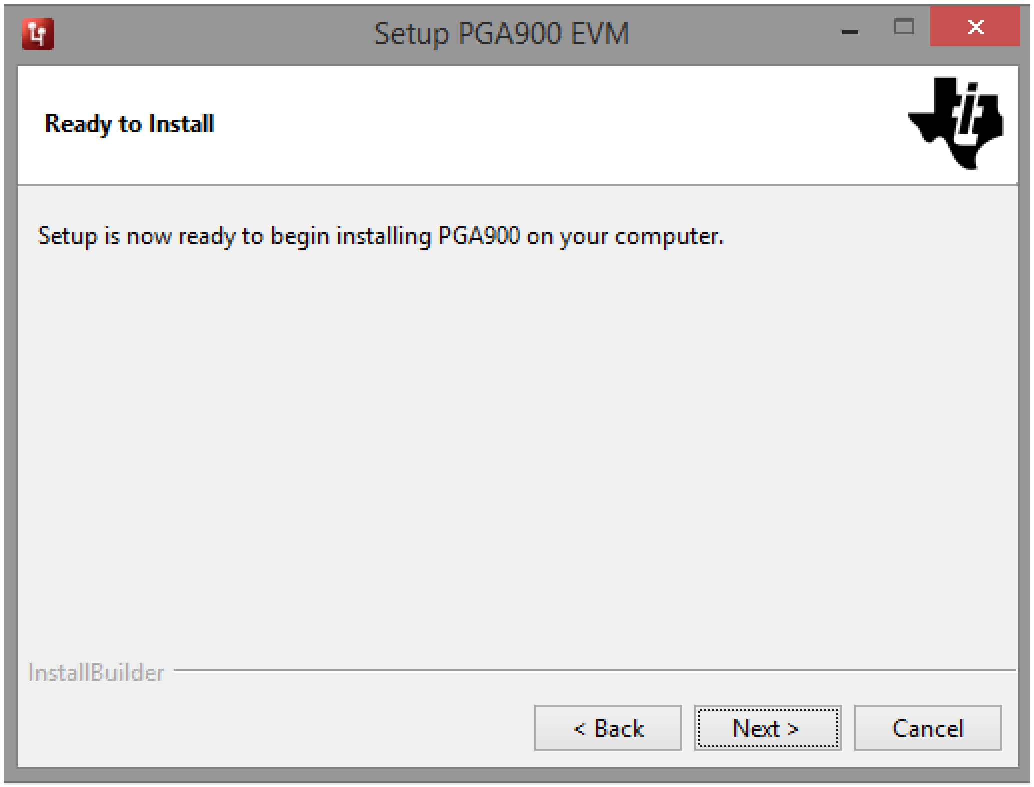


Please specify the directory where PGA900 will be installed.

Installation Directory C:\Program Files (x86)\Texas Instruments\PG/

Figure 1-6. Destination Directory

4. A screen as shown in Figure 1-7 appears. Click **Next >** to begin installation.



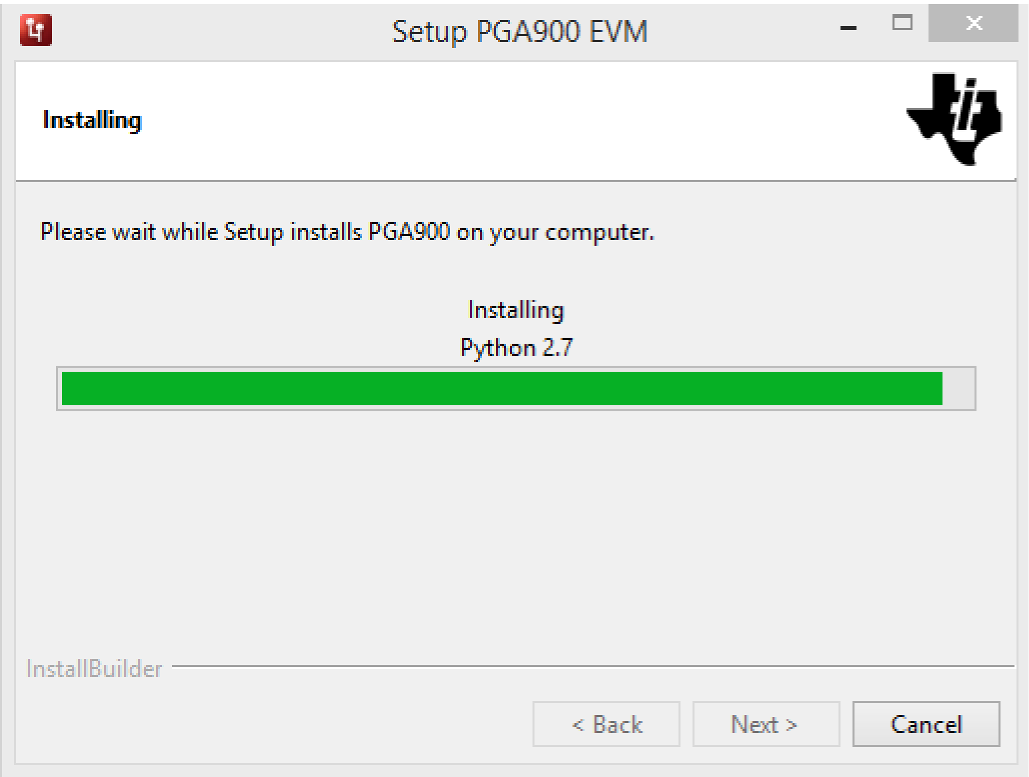
Setup PGA900 EVM

Ready to Install

Setup is now ready to begin installing PGA900 on your computer.

Figure 1-7. Start Installation

5. The installer begins self-extraction and proceed with the installation as shown in Figure 1-8.



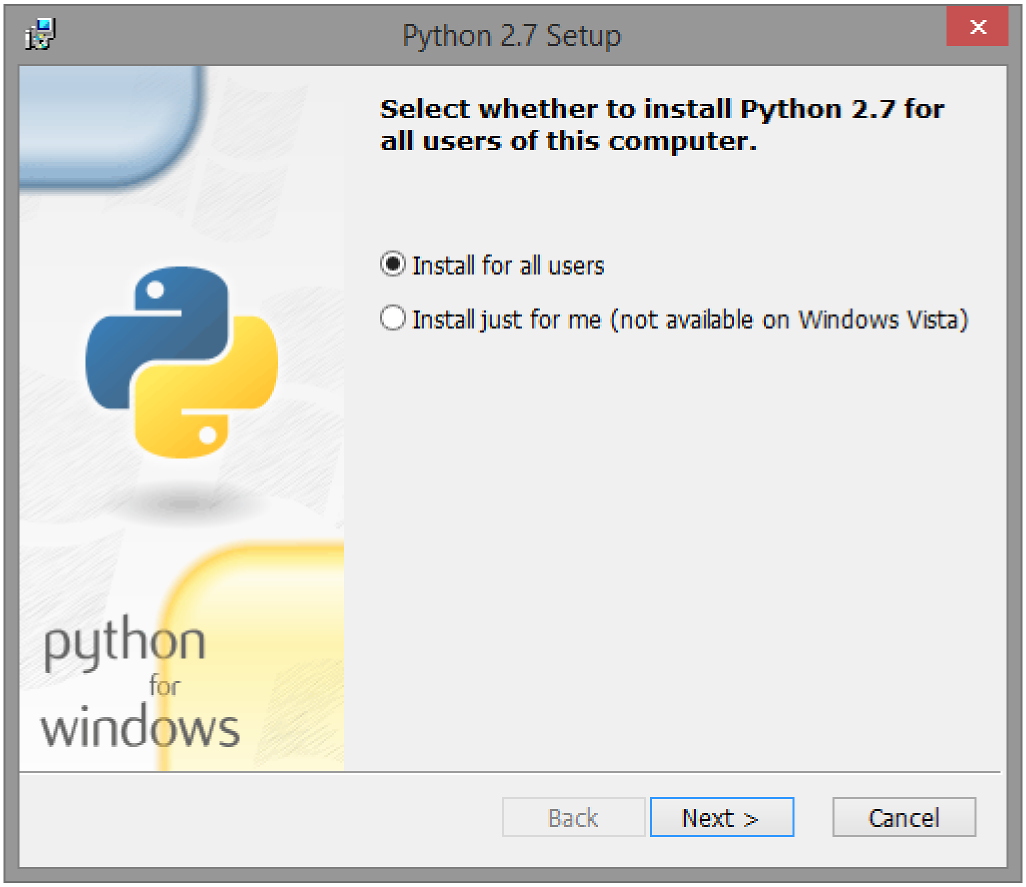
Installing

Please wait while Setup installs PGA900 on your computer.

Installing
Python 2.7

Figure 1-8. Installation in Progress

6. Towards the end of PGA900 GUI installation, Python installation starts. Select the required option and click the **Next >** button.



Python 2.7 Setup

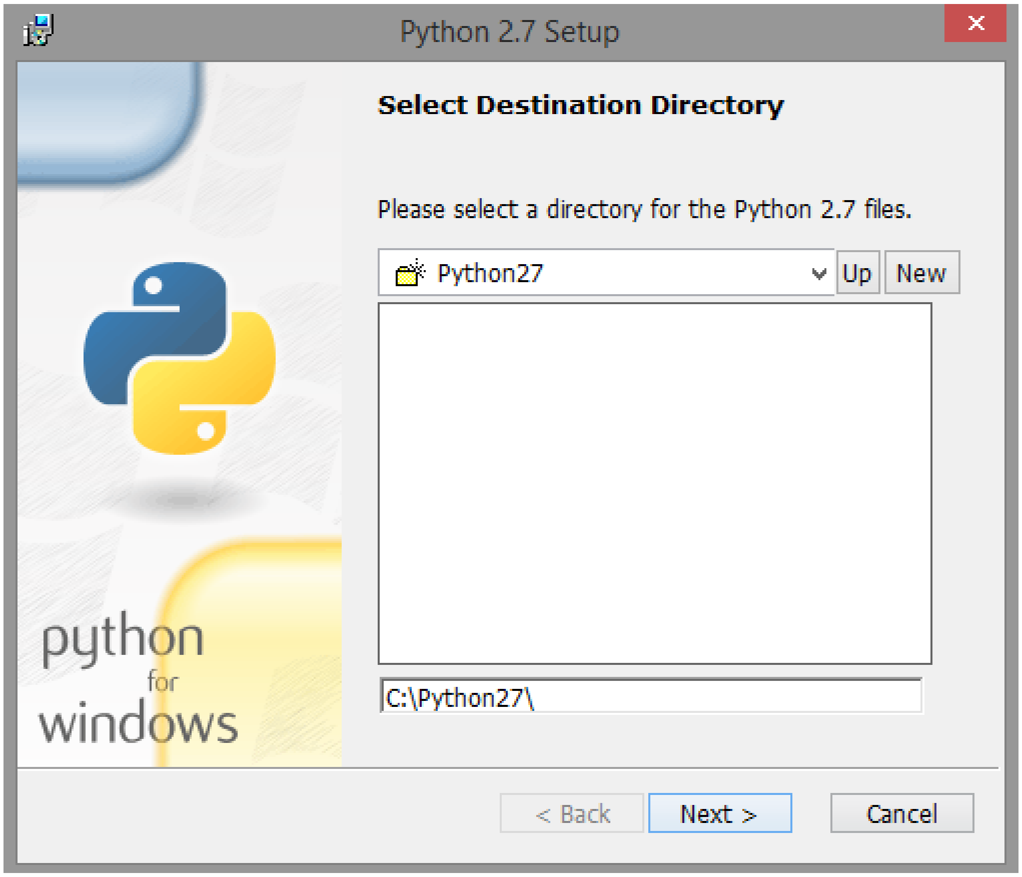
Select whether to install Python 2.7 for all users of this computer.

● Install for all users

○ Install just for me (not available on Windows Vista)

Figure 1-9. Python Installation

7. Select the installation folder for Python and click **Next >** button.



Select Destination Directory

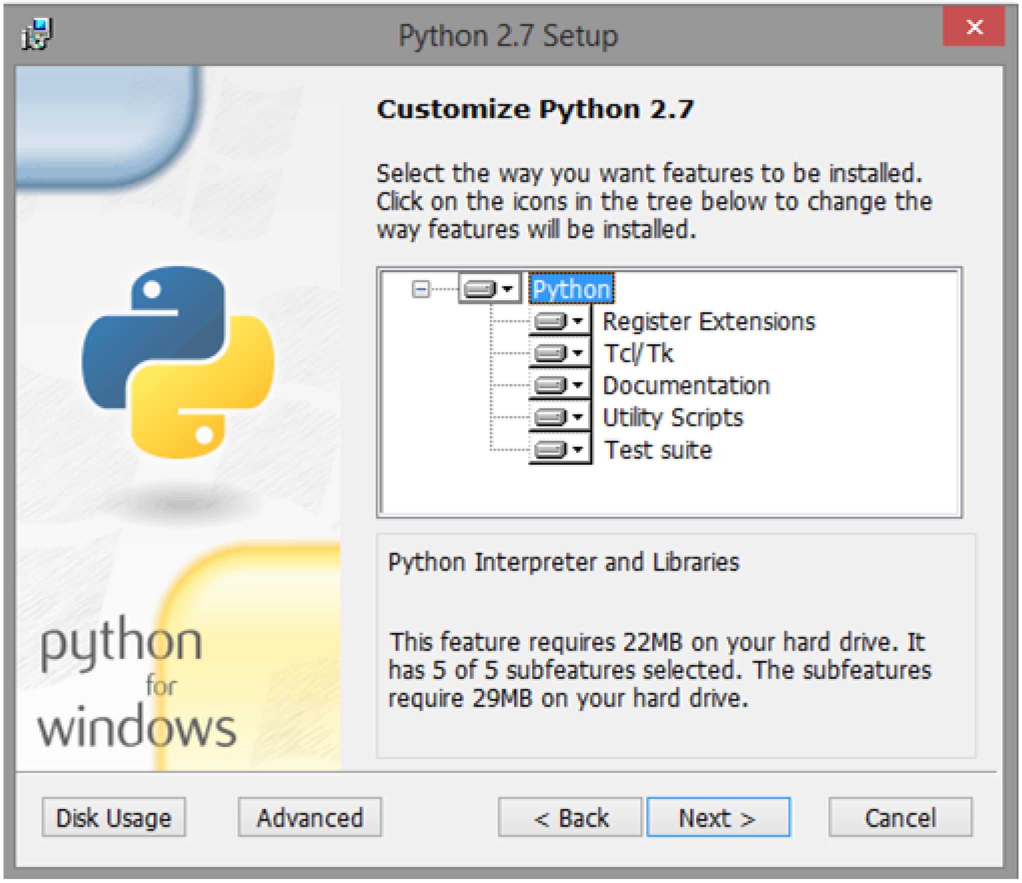
Please select a directory for the Python 2.7 files.

Python27

C:\Python27\

Figure 1-10. Python Installation Directory

8. A dialog as shown in Figure 1-11 appears. Click **Next >** button from the dialog.



Python 2.7 Setup

Customize Python 2.7

Select the way you want features to be installed. Click on the icons in the tree below to change the way features will be installed.

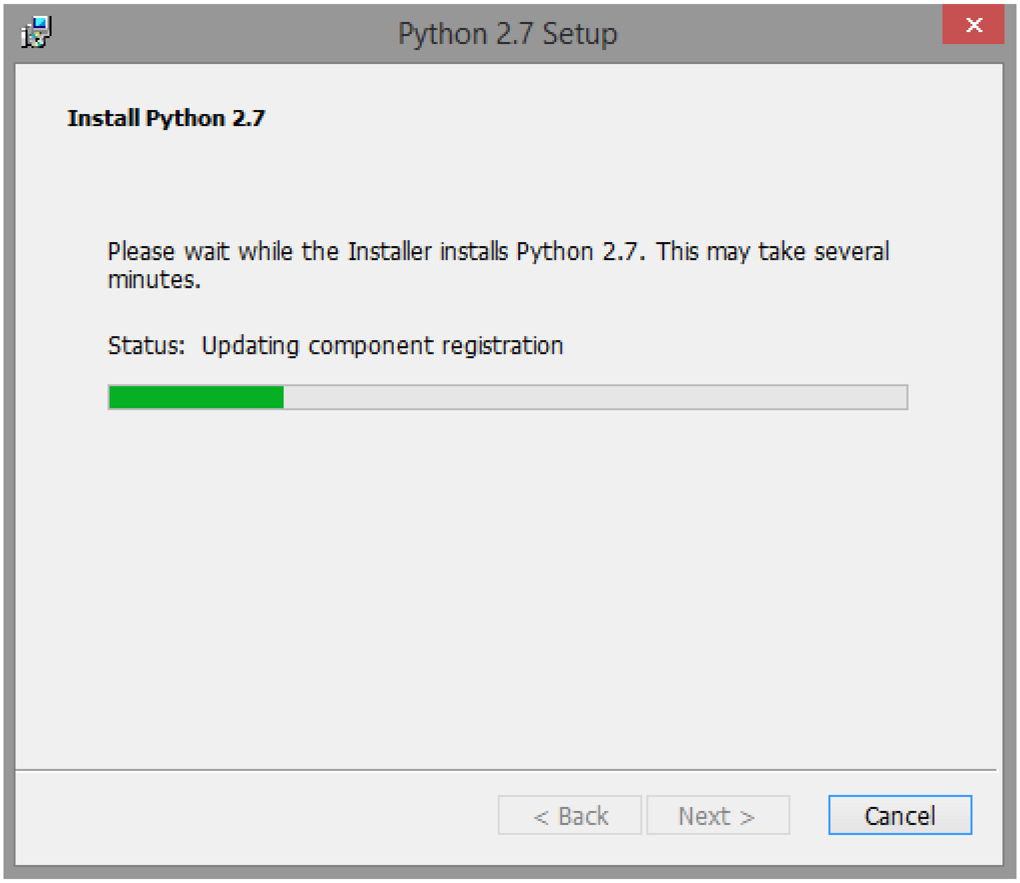
- Python
  - Register Extensions
  - Tcl/Tk
  - Documentation
  - Utility Scripts
  - Test suite

Python Interpreter and Libraries

This feature requires 22MB on your hard drive. It has 5 of 5 subfeatures selected. The subfeatures require 29MB on your hard drive.

Figure 1-11. Python Customization

9. Python installation starts. The progress is shown as in Figure 1-12.



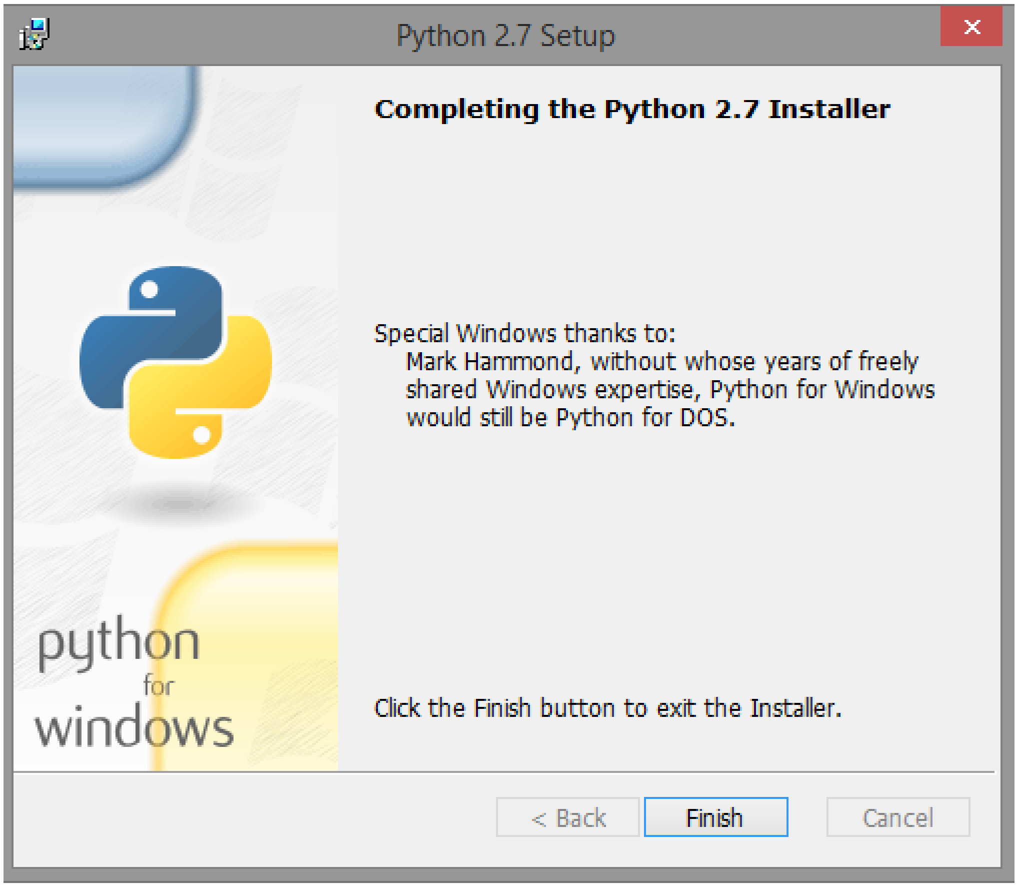
Install Python 2.7

Please wait while the Installer installs Python 2.7. This may take several minutes.

Status: Updating component registration

Figure 1-12. Python Installation Progress

10. After Python installation is finished, click the **Finish** button.



Python 2.7 Setup

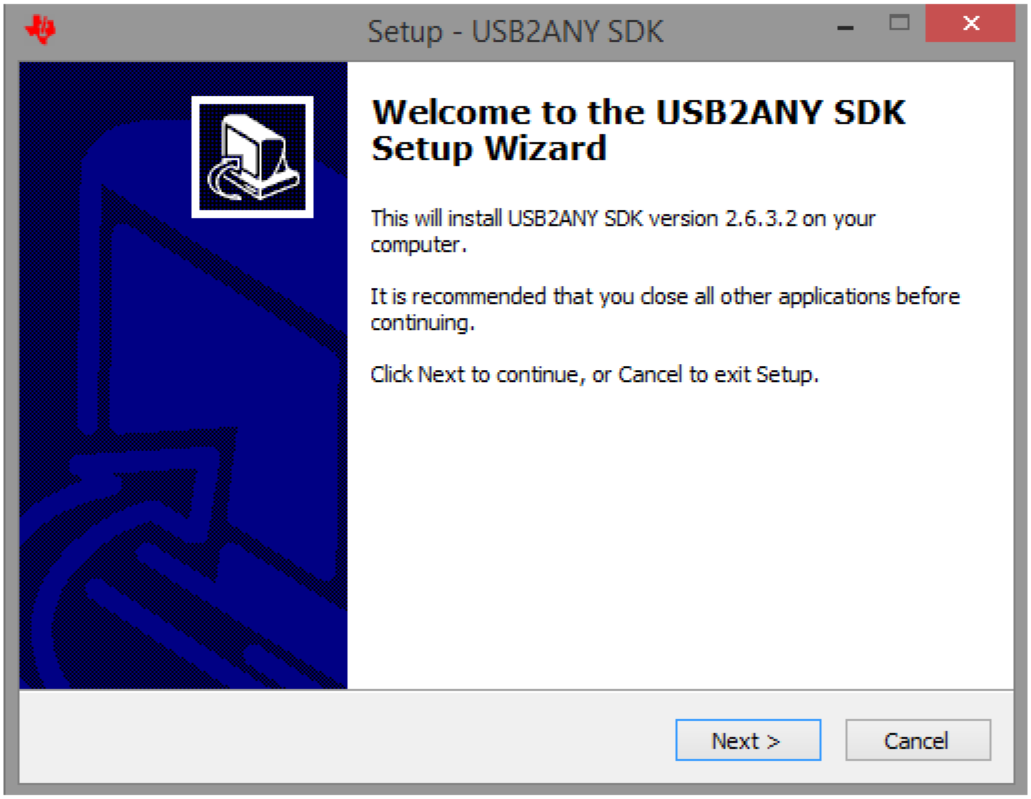
Completing the Python 2.7 Installer

Special Windows thanks to:
Mark Hammond, without whose years of freely shared Windows expertise, Python for Windows would still be Python for DOS.

Click the Finish button to exit the Installer.

Figure 1-13. Python Installation Complete

11. The USB2ANY installation starts after Python installation is finished. From the dialog as shown in Figure 1-14, click the **Next >** button.



Setup - USB2ANY SDK

Welcome to the USB2ANY SDK Setup Wizard

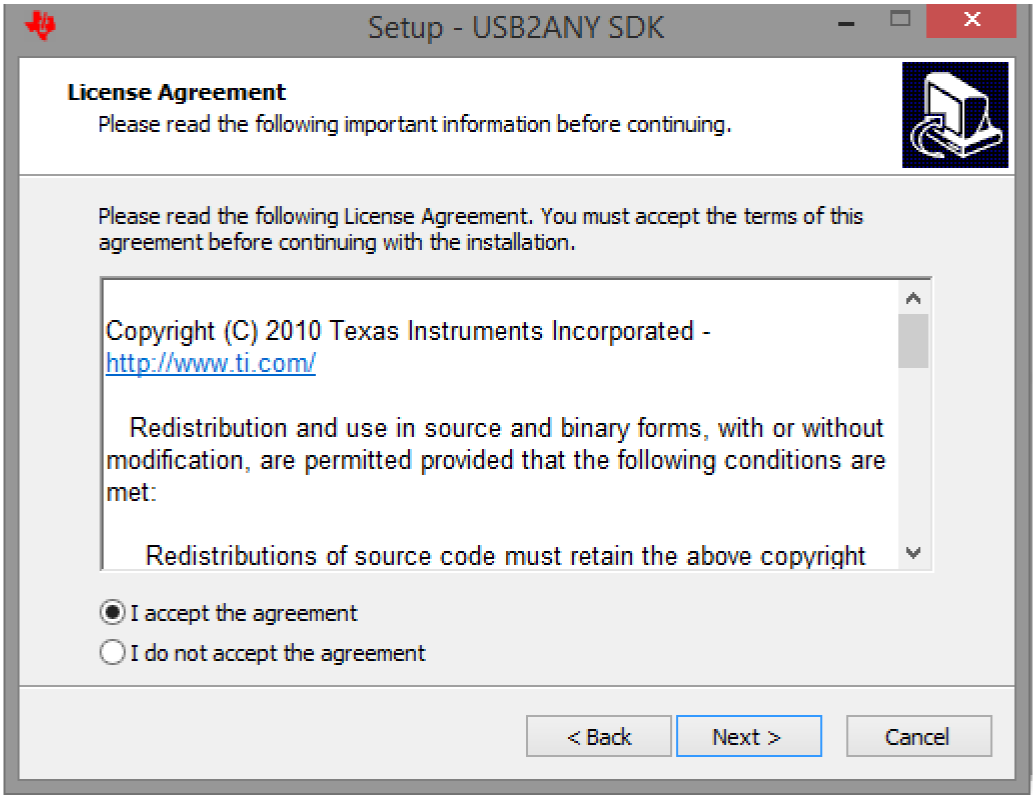
This will install USB2ANY SDK version 2.6.3.2 on your computer.

It is recommended that you close all other applications before continuing.

Click Next to continue, or Cancel to exit Setup.

Figure 1-14. USB2ANY Installation

12. The license agreement appears as shown in Figure 1-15. Read through the agreement, enable the “I accept the agreement” radio button, and click the **Next >** button.



License Agreement

Please read the following important information before continuing.

Please read the following License Agreement. You must accept the terms of this agreement before continuing with the installation.

Copyright (C) 2010 Texas Instruments Incorporated - http://www.ti.com/

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

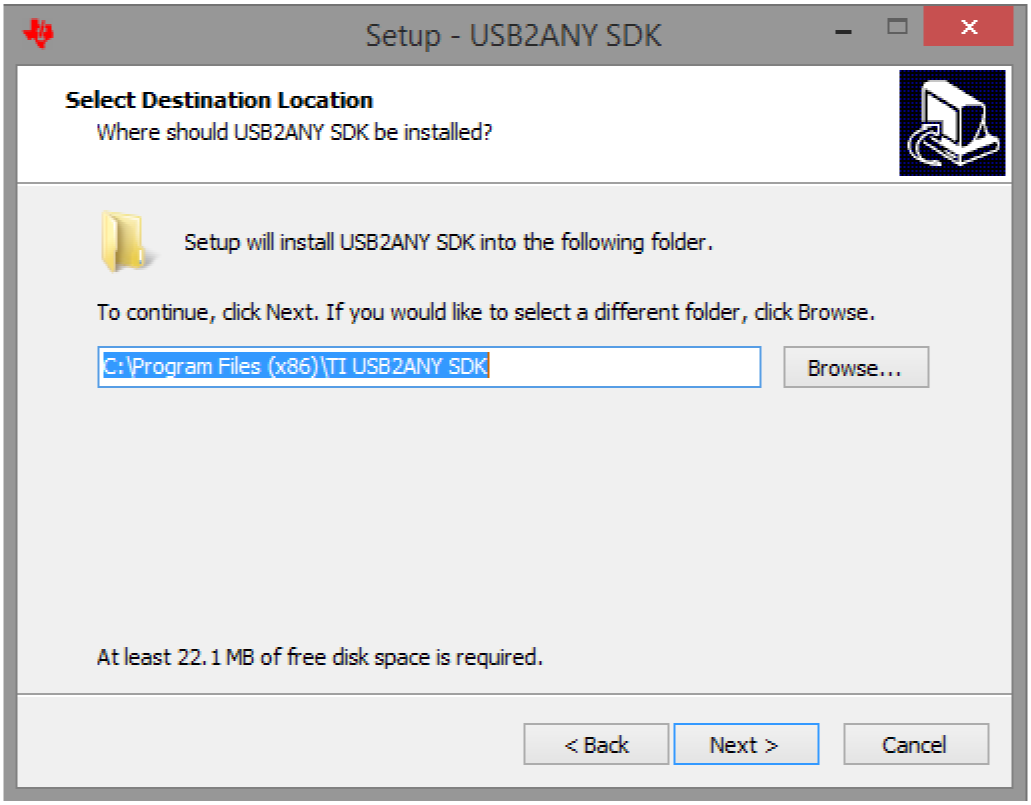
Redistributions of source code must retain the above copyright

● I accept the agreement

○ I do not accept the agreement

Figure 1-15. USB2ANY License Agreement

13. Set the destination directories for the PGA900 GUI installation and click the **Next >** button as shown in Figure 1-16.



Select Destination Location

Where should USB2ANY SDK be installed?

Setup will install USB2ANY SDK into the following folder.

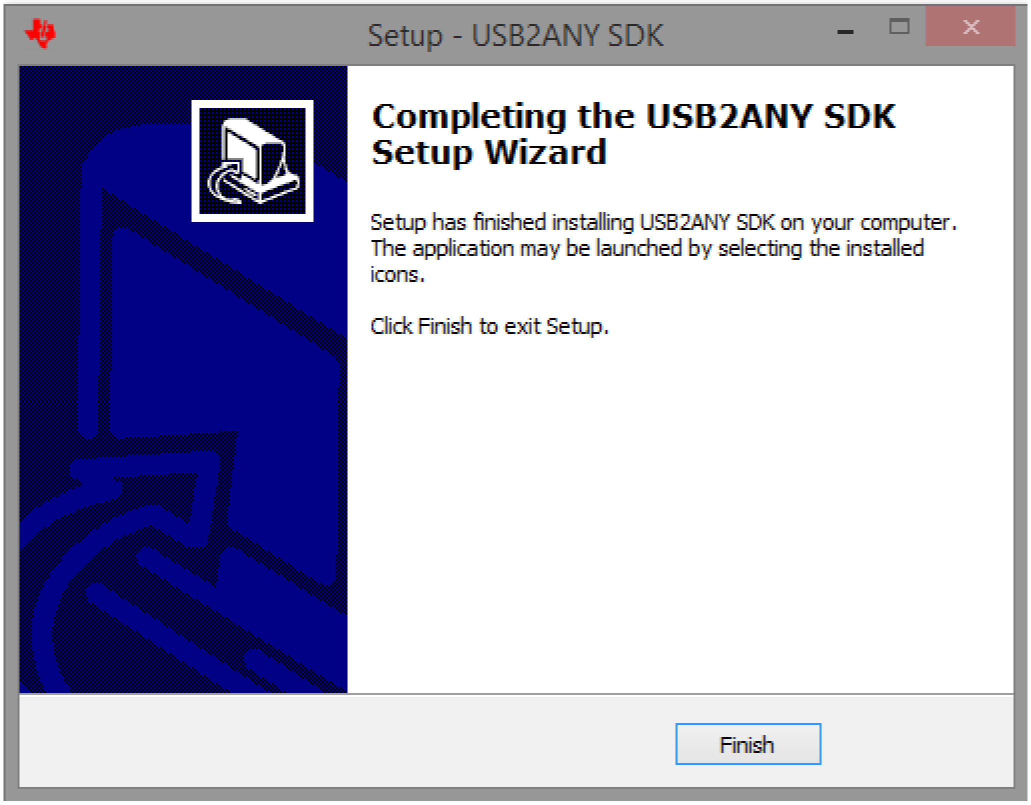
To continue, click Next. If you would like to select a different folder, click Browse.

C:\Program Files (x86)\TI USB2ANY SDK

At least 22.1 MB of free disk space is required.

Figure 1-16. USB2ANY Installation Folder

14. After the installation is complete, click the **Finish** button.



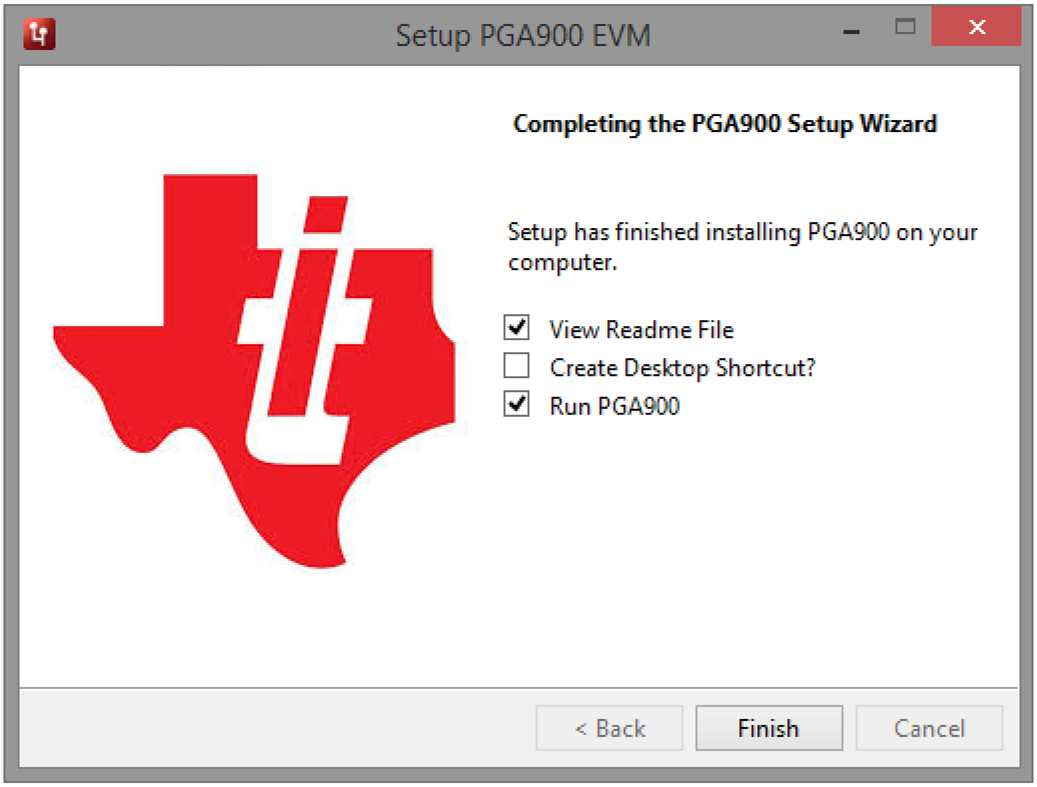
Completing the USB2ANY SDK Setup Wizard

Setup has finished installing USB2ANY SDK on your computer. The application may be launched by selecting the installed icons.

Click Finish to exit Setup.

Figure 1-17. USB2ANY Installation Complete

15. The screen shown in Figure 1-18 appears and denotes the end of the PGA900 GUI installation.



Setup PGA900 EVM

Completing the PGA900 Setup Wizard

Setup has finished installing PGA900 on your computer.

☑ View Readme File

□ Create Desktop Shortcut?

☑ Run PGA900

Figure 1-18. Installation Complete

NOTE: The installer also installs Python 2.7, USB2ANY SDK along with the GUI installation.

NOTE: The PGA900 GUI requires the following software to be installed before the GUI is executed.

National Instruments LabVIEW™ runtime engine 2012 from this link. http://www.ni.com/download/labview-run-time-engine-2012/3433/en/

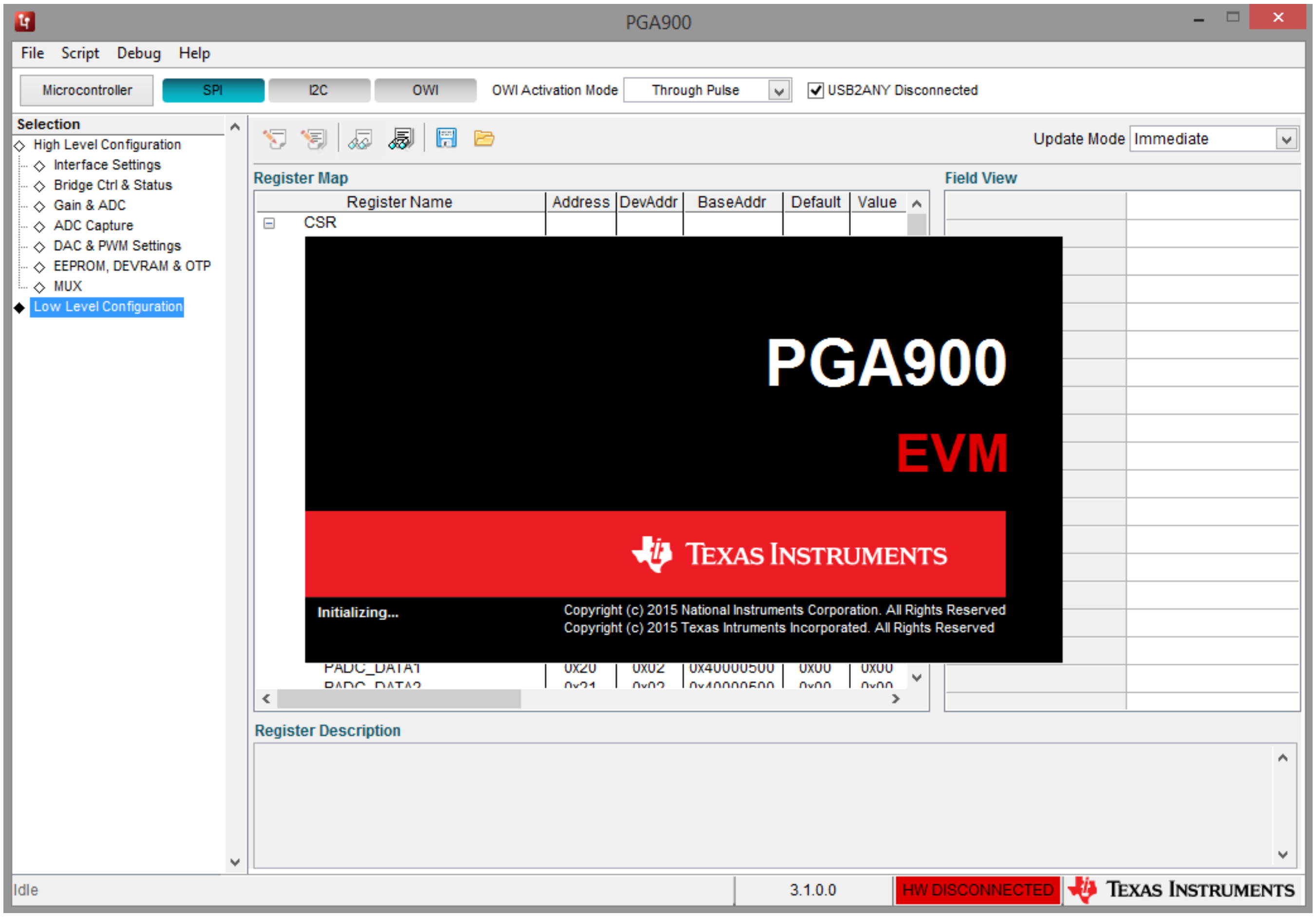
NOTE: The PGA900 GUI executable has been built in LabVIEW 2012 (32-bit) version and it expects the LabVIEW runtime engine version to be LabVIEW runtime engine 2012 (32-bit) version.

PGA900 – User Interface

This section gives a detailed description of the features of the PGA900 GUI. The PGA900 GUI is an intuitive UI for the PGA900 device and EVM that allows the user to read and configure the registers of the PGA900 device and control some EVM components.

NOTE: Launch the PGA900 GUI with administrator privileges.

When the PGA900 GUI is launched, the following screen pops up indicating that the GUI is initializing.



OWI Activation Mode: Through Pulse

☑ USB2ANY Disconnected

Update Mode: Immediate

PADC_DATA1	0x20	0x02	0x40000500	0x00	0x00
PADC_DATA2	0x21	0x02	0x40000500	0x00	0x00

Initializing...

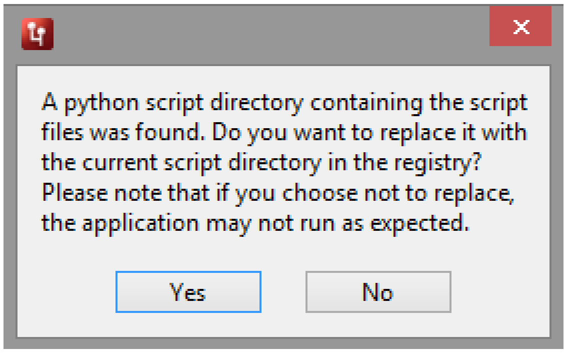
Copyright (c) 2015 National Instruments Corporation. All Rights Reserved
Copyright (c) 2015 Texas Intruments Incorporated. All Rights Reserved

Idle | 3.1.0.0 | HW DISCONNECTED

Figure 2-1. GUI Initializing

When the PGA900 GUI is launched for the first time, a pop-up window (as shown in Figure 2-2) appears on the screen.

1. Click the **Yes** button on the pop-up for the GUI to run as expected. This updates the Python installation path in the Windows registry, so that when the macro window is launched from the GUI, the python IDLE IDE is called.

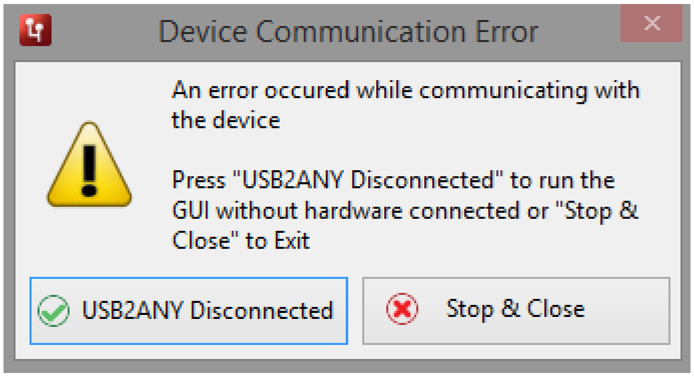


A python script directory containing the script files was found. Do you want to replace it with the current script directory in the registry? Please note that if you choose not to replace, the application may not run as expected.

Figure 2-2. Update Registry

2. After the PGA900 GUI is launched, the user can invoke it in two different ways:
   - USB2ANY disconnected
   - USB2ANY connected

   USB2ANY disconnected mode is invoked when the USB2ANY is not connected. When the PGA900 GUI is loaded with no hardware interface connected, a message pops up as shown in Figure 2-3. This allows the user to either run the PGA900 GUI in USB2ANY disconnected mode or to terminate the usage.



Device Communication Error

An error occured while communicating with the device

Press "USB2ANY Disconnected" to run the GUI without hardware connected or "Stop & Close" to Exit

Figure 2-3. Device Communication Error

When the user continues in USB2ANY disconnected mode, the checkbox is set. This shows that the PGA900 GUI is running in simulation. When the GUI is in USB2ANY disconnected mode, the UI controls may appear to function as if the device is connected and will read simulated data.

2.1 PGA900 GUI Overview

The PGA900 GUI consists of the following pages:

- High Level Configuration Pages
  - Interface Settings
  - Bridge Ctrl and Status
  - Gain and ADC
  - ADC Capture
  - DAC and PWM Settings
  - EERPOM, DEVRAM, and OTP
  - MUX
- Low Level Configuration Page

The PGA900 resources can be accessed by the digital interface or microcontroller. The user can change how the PGA900 resources are accessed by selecting the microcontroller button at the top-left corner of the PGA900 GUI (as shown in Figure 2-4).

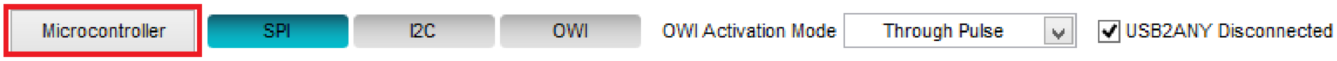


Figure 2-4. Microcontroller Selection

The microprocessor can be put into reset state by writing 1 to the MICRO_RESET bit in the MICRO_INTERFACE_CONTROL register using any of the digital interfaces. Access to the digital interface is enabled by writing 3 to MICRO_INTERFACE_CONTROL register. Disabling any of the bits that correspond to the digital interface results in disabling the digital interface.

NOTE: All pages except the Low Level configuration page, Interface Settings page, and Bridge Control and Status Settings page are disabled when the digital interface is disabled.

2.1.1 Interface Settings

- The user can configure I^{2}C settings like speed and CS active state.
- UART speed settings for OWI
- Rloop resistance configuration
- Additional voltage configuration

2.1.2 Bridge Ctrl and Status

- The bridge control and status page is used to operate the internal bridge circuit present in the PGA900 EVM.
- This page also contains the status section where different status registers could be read and reset based on the flags.

2.1.3 Gain and ADC

- The page contains the temperature and pressure sensor settings for the ADC.
- The table at the bottom of the page is used to read the ADC register values and display the corresponding voltage.
- The data can also be dumped into a file by selecting the corresponding ADC.

2.1.4 ADC Capture

- The ADC Graph allows the user the configure the ADC settings for Capture.
- It has options to perform continuous ADC capture with the configured settings.

2.1.5 DAC and PWM Settings

- The DAC and PWM settings page is used to control the DAC configurations and PWM configurations.
- User can read DAC data and ADC loopback data. DAC Gain can also be configured here.

2.1.6 EERPOM, DEVRAM, and OTP

- The PGA900 has OTP and DevRAM blocks along with the register memory and these blocks can be accessed by using the CSR EERPOM and OTP Programming page of the PGA900 GUI.
- These memories can be programmed by using HEX files.
- EEPROM has an additional functionality to calculate the cyclic redundancy check (CRC) for all the EEPROM registers.

2.1.7 MUX

- The MUX and Sensor settings page is used to control the MUX of the temperature and the pressure ADC path.
- The PGA900 EVM board has two types of MUX connected to it.
  - Analog text MUX
  - Digital test MUX
- The MUX can be used to select a particular channel from the device.

2.1.8 Low Level Configuration Page

- The Low Level Configuration Page lists all the registers that are present in the PGA900 GUI.
- This page can be used to write to and read from the register fields of the PGA900 device.

2.1.9 About Page

- About page can be accessed from Help Menu → About
- The About page contains information such as GUI name, version information, and supported OS.
- It also contains copyright information of the PGA900 GUI.

2.2 PGA900 Communication Interfaces

The communication interfaces are used to communicate (read and write) with the device’s registers. Each interface uses different pins for communication. The device has three separate modes of communication:

- Serial peripheral interface (SPI)
- Inter-integrated circuit (I^2C)
- One-wire interface (OWI)

Each communication mode has its own protocol; however, all three access the same memory elements within the device. For all three communication modes, the PGA900 device operates as a slave device.

NOTE: Whenever the user changes the protocols, a sequence of operations are performed at the backend

2.3 During SPI and I^2C Selection

- The digital interface is enabled by writing 3 to MICRO_INTERFACE_CONTROL.
- GPIO_OWI_TX is set LOW.
- GPIO_OWI_ACT is set HIGH.
- Disables the OWI_XCVR bit in the DIG_IF_CTRL register

2.4 For OWI Activation

- Through I^{2}C:
  - The digital interface is enabled by writing 3 to MICRO_INTERFACE_CONTROL.
  - The I^{2}C communication parameters (speed, address length, and pullup speed) are set.
  - The 3.3-V and 5.0-V power outputs are enabled.
  - GPIO_OWI_ACT is set to LOW.
  - GPIO_OWI_TX is set to HIGH.
  - DIG_IF_CTRL written with F [SPI_EN, I2C_EN, OWI_EN, and OWI XCVR gets enabled]
  - TOPDIG is writen with 1B [OWI RECEIVE DATA gets enabled]
  - ALPWR is writen with 4 [SD = 0 , ADC_EN = 1]
  - DLPWR is written with 1 [OWI_CLK_EN gets enabled]
  - DIG_IF_CTRL written with 4F [SPI_EN, I2C_EN, OWI_EN, OWI_XCVR_EN, and I2C_DEGLITCH_EN gets enabled]
- Through pulse:
  - The digital interface is disabled by writing 0 to MICRO_INTERFACE_CONTROL.
  - GPIO_OWI_TX is set to LOW.
  - GPIO_OWI_ACT is set to LOW.
  - Wait for 10 ms. (This configuration is hardcoded in the GUI software.)
  - GPIO_OWI_ACT is set HIGH.
  - Wait for 10 ms. (This configuration is hardcoded in the GUI software.)
  - GPIO_OWI_TX is set HIGH.

Whenever the OWI is activated through PULSE, and if the user wants to activate the OWI again using I^{2}C, the user must activate SPI or I^{2}C first, then reactivate OWI through I^{2}C.

2.4.1 Overview of SPI

- SPI is a synchronous, serial, master-slave, communication standard that requires the following four pins:
  - MOSI: SPI master out slave in, input pin
  - MISO: SPI master in slave out, serial output pin (tri-state output)
  - SCK: SPI clock which controls the communication
  - CSN: Chip select (active low)
- SPI communicates in a master/slave style where only one device, the master, can initiate the data transmissions.
- The PGA900 always acts as the slave in SPI communication where the external device that is communicating with it becomes the master. Both devices begin data transmission with the most significant bit (MSB) first.
- Because multiple slave devices can exist on one bus, the master node is able to notify the specific slave node that it is ready to begin communicating with, by driving the CSN line to a low logic level.
- In the absence of active transmission, the master SPI device places the device in reset by driving the CSN pin to a high logic level.
- During the reset state, the MISO pin operates in tri-state mode.

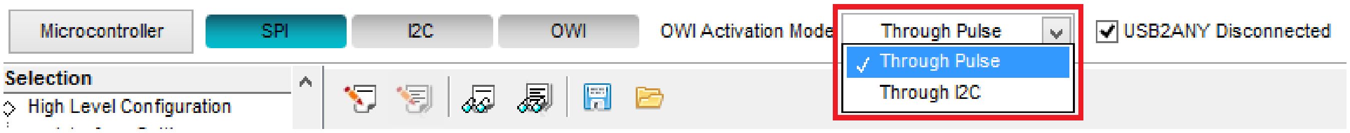
2.4.2 Overview of I^{2}C Interface

- I^{2}C is a synchronous serial communication standard that requires the following two pins for communication:
  - SDA: I^{2}C serial data line (SDA)
  - SCL: I^{2}C serial clock line (SCL)
- In addition, the CSN pin is used to select the I^{2}C device address of PGA900, specifically,

  - CSN – Logic 1 device address – 0x20-0x27
  - CSN – Logic 0 device address – 0x40-0x47
- It is noted that for valid I^{2}C communication to occur:
  - CSN should not change value during an I^{2}C transaction
  - SPI clock (SCLK) should not be active when CSN is logic 0, when selecting the I^{2}C device address
- I^{2}C communicates in a master/slave style communication bus where one device, the master, can initiate data transmission.
- The device always acts as the slave device in I^{2}C communication, where the external device that is communicating to it acts as the master.
- The master device is responsible for initiating communication over the SDA line and supplying the clock signal on the SCL line.
- When the I^{2}C SDA line is pulled low, it is considered a logical 0, and when the I^{2}C SDA line is floating high, it is considered a logical 1.
- For the I^{2}C interface to have access to memory locations other than test register space, the IF_SEL bit in the Micro/Interface Control Test register (MICRO_IF_SEL_T) must be set to logic 1.

2.4.3 Overview of OWI

- The OWI digital communication is a master-slave communication link in which the PGA900 operates as a slave device only.
- The master device controls when the data transmission begins and ends.
- The slave device does not transmit data back to the master until it is commanded to do so by the master. The VDD pin of PGA900 is used as an OWI, so that when PGA900 is embedded inside a system module, only two pins are needed (VDD and GND) for communication.
- The OWI master communicates with PGA900 by modulating the voltage on the VDD pin while PGA900 communicates with the master by modulating current on VDD pin.
- When using the OWI, the user can select the activation mode from one of the following options (also shown in Figure 2-5).
  - Through pulse
  - Through I^{2}C



OWI Activation Mode: Through Pulse

✓ Through Pulse
Through I2C

☑ USB2ANY Disconnected

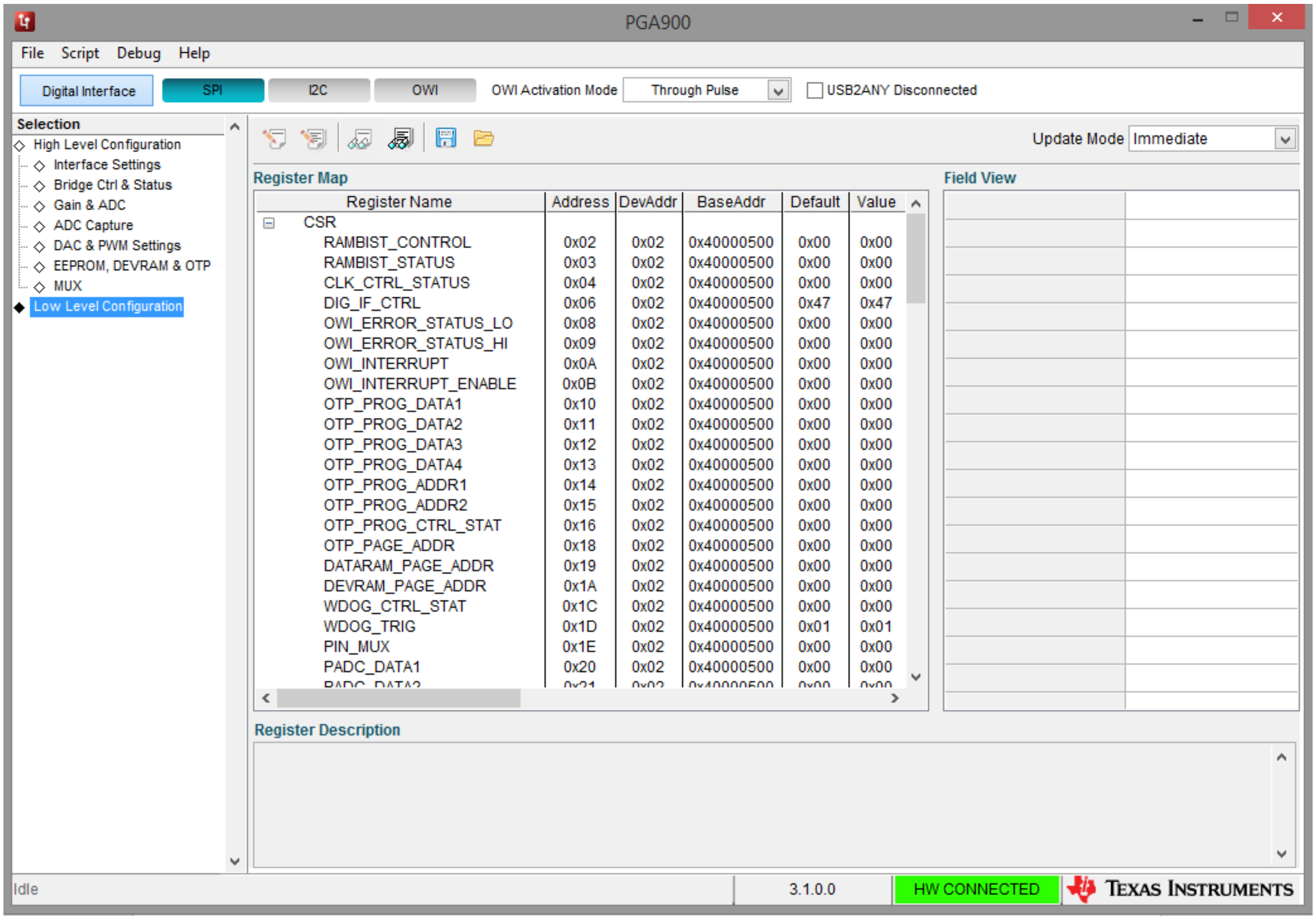
Figure 2-5. OWI Activation Mode

NOTE: The OWI activation mode is applicable only to the OWI and not to the SPI and I^{2}C interface.

- OWI Write:
  - No specific configuration is needed to write using the OWI.
  - Write the data in the format defined by the data sheet (SLDS209).
- OWI Read:
  - The following sequence is executed for the read operation.
  - Check if the USB2ANY buffer has residual data and empty the buffer.
  - Send the read command with the sync byte, read initialization and read response command as defined in the data sheet (SLDS209).
  - Check for data in the buffer until the data is received.
  - Read the data from the buffer and display it. This reads all data available in the buffer. Ideally, there should be only one byte of data.

2.5 Page Selection

2.5.1 Low Level Configuration Page



OWI Activation Mode: Through Pulse

□ USB2ANY Disconnected

Update Mode: Immediate

Register Name	Address	DevAddr	BaseAddr	Default	Value
CSR					
RAMBIST_CONTROL	0x02	0x02	0x40000500	0x00	0x00
RAMBIST_STATUS	0x03	0x02	0x40000500	0x00	0x00
CLK_CTRL_STATUS	0x04	0x02	0x40000500	0x00	0x00
DIG_IF_CTRL	0x06	0x02	0x40000500	0x47	0x47
OWI_ERROR_STATUS_LO	0x08	0x02	0x40000500	0x00	0x00
OWI_ERROR_STATUS_HI	0x09	0x02	0x40000500	0x00	0x00
OWI_INTERRUPT	0x0A	0x02	0x40000500	0x00	0x00
OWI_INTERRUPT_ENABLE	0x0B	0x02	0x40000500	0x00	0x00
OTP_PROG_DATA1	0x10	0x02	0x40000500	0x00	0x00
OTP_PROG_DATA2	0x11	0x02	0x40000500	0x00	0x00
OTP_PROG_DATA3	0x12	0x02	0x40000500	0x00	0x00
OTP_PROG_DATA4	0x13	0x02	0x40000500	0x00	0x00
OTP_PROG_ADDR1	0x14	0x02	0x40000500	0x00	0x00
OTP_PROG_ADDR2	0x15	0x02	0x40000500	0x00	0x00
OTP_PROG_CTRL_STAT	0x16	0x02	0x40000500	0x00	0x00
OTP_PAGE_ADDR	0x18	0x02	0x40000500	0x00	0x00
DATARAM_PAGE_ADDR	0x19	0x02	0x40000500	0x00	0x00
DEVRAM_PAGE_ADDR	0x1A	0x02	0x40000500	0x00	0x00
WDOG_CTRL_STAT	0x1C	0x02	0x40000500	0x00	0x00
WDOG_TRIG	0x1D	0x02	0x40000500	0x01	0x01
PIN_MUX	0x1E	0x02	0x40000500	0x00	0x00
PADC_DATA1	0x20	0x02	0x40000500	0x00	0x00
PADC_DATA2	0x21	0x02	0x40000500	0x00	0x00

Idle | 3.1.0.0 | HW CONNECTED

Figure 2-6. Low Level Configuration Page

- Low Level Configuration page provides a detailed view of all the registers that the device possesses.
- This page allows the users to read from and write to the registers.
- When a particular register is selected, the corresponding register description is displayed at the bottom of the page.
- Register write modes:

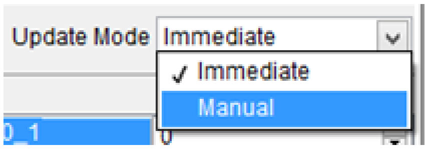
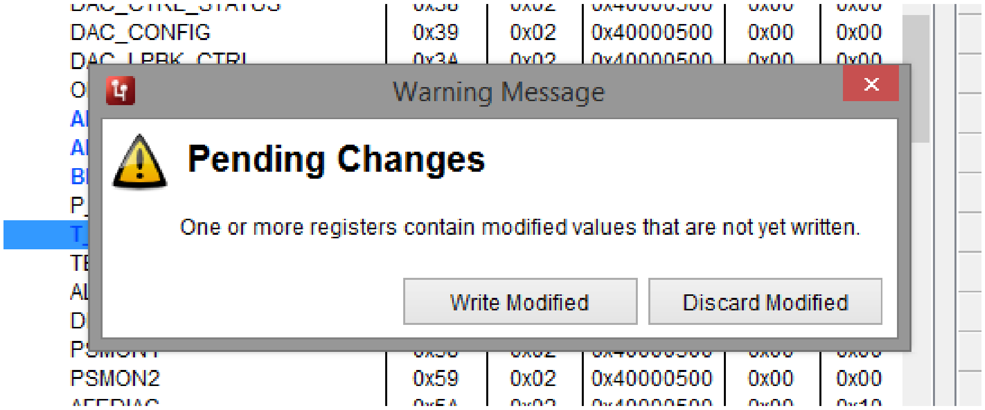


Figure 2-7. Update Mode

  – Immediate mode - The register values are written to device immediately.
  – Manual mode - The register values will be written to the device when the **Write Register** or **Write Modified** button is clicked. The changed register values will be highlighted in blue which means that these changes are not yet written to the device. When a highlighted register is written to the device, the register text changes into normal state (black text). If there are some pending changes and the update mode is changed from manual to immediate, a dialog box will appear. Choose the

required operation to be carried out from the dialog box.



DAC_CONFIG	0x39	0x02	0x40000500	0x00	0x00
PSMON2	0x59	0x02	0x40000500	0x00	0x00

Warning Message

Pending Changes

One or more registers contain modified values that are not yet written.

Figure 2-8. Pending Changes Dialog Box

2.5.1.1 Load and Save Register Configuration Feature

- **Save Config** - When you click this button, the current register configuration is saved into a file which can be later loaded into the GUI using the load option.

Figure 2-9. Save Config

- **Load Config** – Click this button to load the configuration file which was saved previously to bring the device to a known state. Ensure the microcontroller is reset before you load the configuration.

Figure 2-10. Load Config

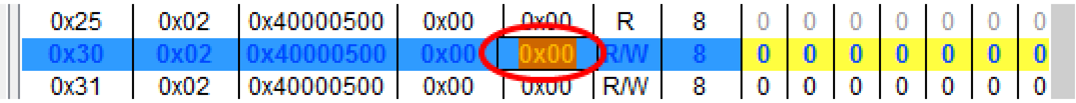
- When you select any of the above options, a message pops up on the screen. Select the file path (to load or save the configuration file) and click OK.

NOTE: Load Config overwrites the existing data in registers with the value specified in the .cfg file loaded.

2.5.1.2 Register Read and Write

The user can change the value of a register by:

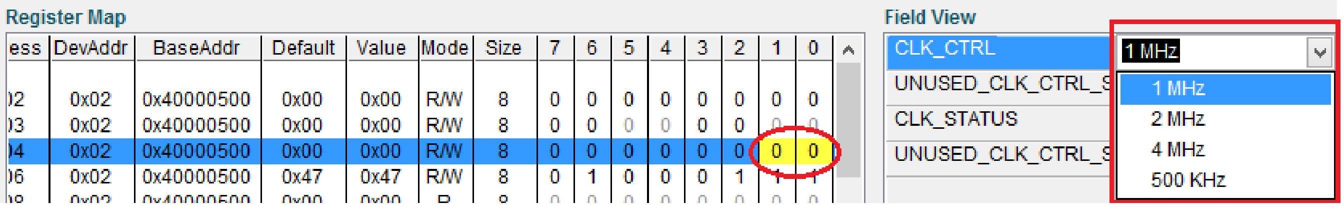
- Register level operation
  - Select the register to edit
  - Double click on the value column corresponding to the register
  - Enter the register value (HEX) in the edit box



0x25	0x02	0x40000500	0x00	0x00	R	8	0	0	0	0	0	0	0
0x30	0x02	0x40000500	0x00	0x00	R/W	8	0	0	0	0	0	0	0
0x31	0x02	0x40000500	0x00	0x00	R/W	8	0	0	0	0	0	0	0

Figure 2-11. Register Level Value Change

- Field level operation:
  - Select the register to edit
  - The fields corresponding to the register will be listed in the ‘Field View’ section
  - When you hover mouse over the value of the field, dropdown list or numeric control will appear and the corresponding bits will be highlighted. The user can change the appropriate value in the dropdown list or numeric control.



Register Map

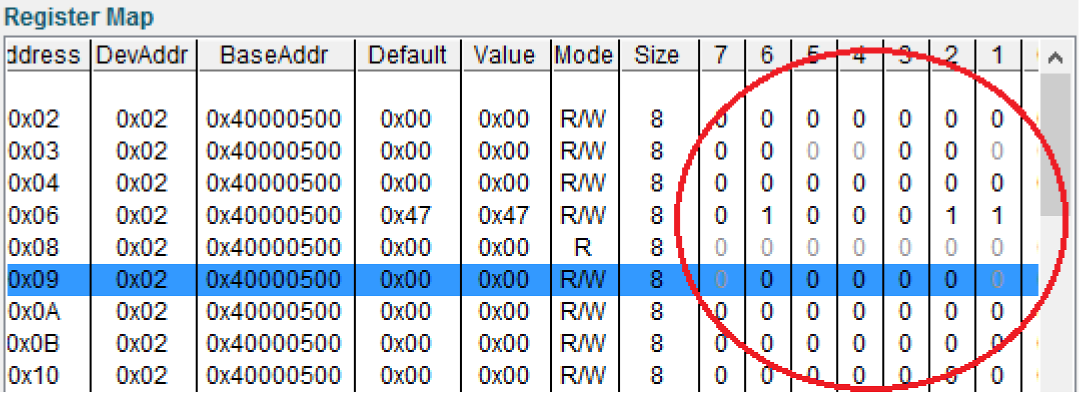
ess	DevAddr	BaseAddr	Default	Value	Mode	Size	7	6	5	4	3	2	1	0
)2	0x02	0x40000500	0x00	0x00	R/W	8	0	0	0	0	0	0	0	0
)3	0x02	0x40000500	0x00	0x00	R/W	8	0	0	0	0	0	0	0	0
)4	0x02	0x40000500	0x00	0x00	R/W	8	0	0	0	0	0	0	0	0
)6	0x02	0x40000500	0x47	0x47	R/W	8	0	1	0	0	0	1	1	1
)8	0x02	0x40000500	0x00	0x00	R	8	0	0	0	0	0	0	0	0

Field View

CLK_CTRL	1 MHz
UNUSED_CLK_CTRL_S	
CLK_STATUS	
UNUSED_CLK_CTRL_S	

Figure 2-12. Field Level Value Change

- Bit level operation:
  - Select the register to edit.
  - Value of each bit in the register can be changed by clicking the 0 or 1 in the corresponding bit column. The bits that are greyed out are read only bits. These bits cannot be written.



Register Map

ddress	DevAddr	BaseAddr	Default	Value	Mode	Size	7	6	5	4	3	2	1
0x02	0x02	0x40000500	0x00	0x00	R/W	8	0	0	0	0	0	0	0
0x03	0x02	0x40000500	0x00	0x00	R/W	8	0	0	0	0	0	0	0
0x04	0x02	0x40000500	0x00	0x00	R/W	8	0	0	0	0	0	0	0
0x06	0x02	0x40000500	0x47	0x47	R/W	8	0	1	0	0	0	1	1
0x08	0x02	0x40000500	0x00	0x00	R	8	0	0	0	0	0	0	0
0x09	0x02	0x40000500	0x00	0x00	R/W	8	0	0	0	0	0	0	0
0x0A	0x02	0x40000500	0x00	0x00	R/W	8	0	0	0	0	0	0	0
0x0B	0x02	0x40000500	0x00	0x00	R/W	8	0	0	0	0	0	0	0
0x10	0x02	0x40000500	0x00	0x00	R/W	8	0	0	0	0	0	0	0

Figure 2-13. Bit Level Value Change

- The Field View displays the fields of the selected register and value of each field.
- **Read Register** – When the Read Register button is pressed, the value will be read from the device and displayed in the register map tree. The filed view will also be updated with the new values.
- **Write Register** – The field view will display the values to be written to register (field-wise view). The hex equivalent data that will be written to the register is displayed in the value column corresponding to the selected register in Register Map tree.

- **Read All** – When you press this button, the data is read from all the registers based on the mode (Read and Read/Write mode).
- **Write Modified** – When you press this button, any value entered in the display box is written to all the registers that are in Write and Read/ Write mode. This option is enabled only when update mode is manual.

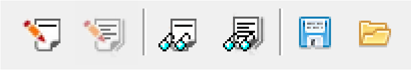


Figure 2-14. Low Level Page Operations

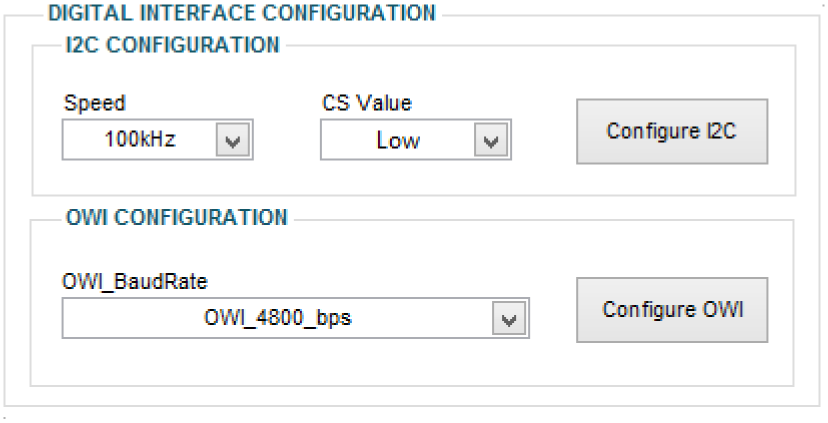
2.5.2 High Level Configuration Page

- The High Level Configuration page provides an abstract view of the device.
- Different functions of the device are represented structurally.
- Each control that is placed in the High Level Configuration page is linked to a register or a field of the device.

The High Level Configuration pages consist of the following functions. Each of the following sections explains a different function of the PGA900 EVM.

2.5.2.1 Interface Settings

- The Interface Settings page contains the configuration parameters necessary for the communication protocols.
- The page includes options for configuring the I^{2}C and OWI communications.
- The default values are displayed when the GUI is loaded.



DIGITAL INTERFACE CONFIGURATION

I2C CONFIGURATION

Speed: 100kHz

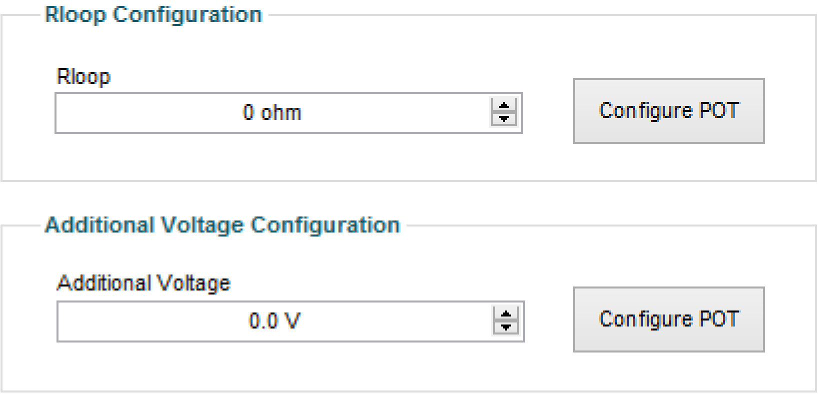
CS Value: Low

OWI CONFIGURATION

OWI_BaudRate: OWI_4800_bps

Figure 2-15. Interface Configurations

- The user has the provisions to change the speed, chip select value of I^{2}C protocol and baud rate for the UART protocol.
- The default settings for I^{2}C is 100 kHz and CSN low. For UART, the default is 4800 bps.
- After the change is made, click the **Configure** button for the changes to take effect.
- The controls Rloop and Additional Voltage are used to configure the potentiometers on the PGA900EVM when OWI is used as a digital interface. These potentiometers are configured using I2C protocol. The Rloop refers to the resistance in the loop when the PGA900 is operating in 4-20 mA loop mode. The PGA900EVM has a 10 Ω loop resistor and therefore, when in 4-20 mA mode, Rloop should be set to 10 Ω. When in voltage mode, Rloop should be set to 0 Ω. The additional voltage configuration is needed when constant voltage drops (such as diodes) are present in the loop. For the PGA900EVM, the additional voltage should be set to 0.0 V.



Rloop Configuration

Rloop: 0 ohm

Additional Voltage Configuration

Additional Voltage: 0.0 V

Figure 2-16. Potentiometers configuration

OWI configurations – UART configuration involves three settings:

1. Setting the baud rate to the selected configuration.
2. Setting the receiver mode is configured as 2.
   (a) Configures the communication to be half duplex
   (b) USB2ANY transmits data with 2 stop bits, but accepts data with a single stop bit.

2.5.2.2 Bridge Ctrl and Status

- The Bridge Control and Status page shows the status bits of the PGA900 device.
- The GUI provides the ability to read and reset the flags.

  A flag is represented by a red highlighted button. The status can be noted by clicking the read button on top of the indicators. To reset the flag, check the box corresponding to the flag on the left side of the indicator and press the Write button on top of the checkbox. Multiple checkboxes can be selected simultaneously.

NOTE: The status controls and indicators on this page remain disabled until the digital interface is enabled.

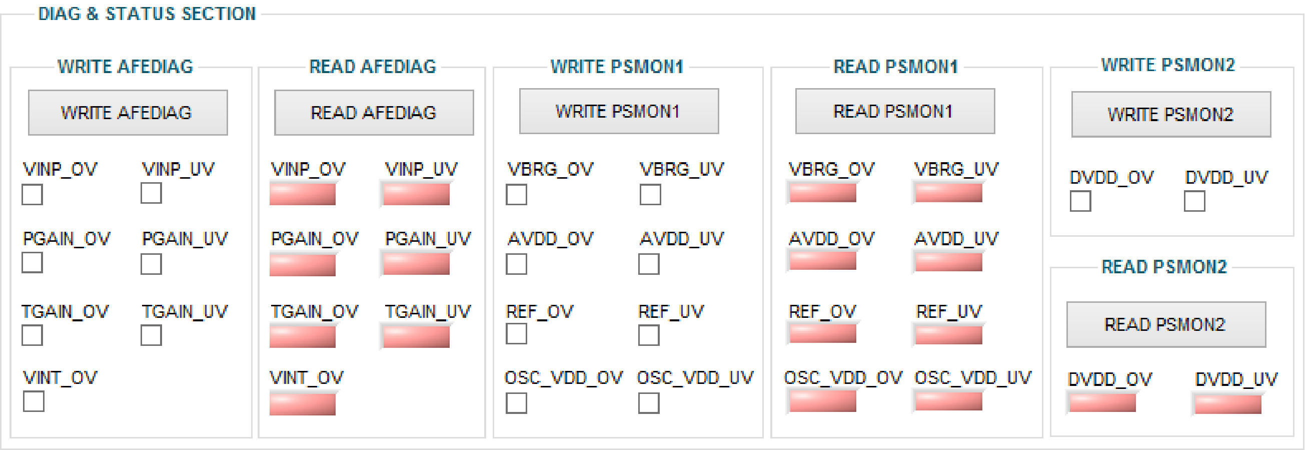


Figure 2-17. Diag and Status Section

- The PGA900 device also has an external bridge circuit. The external bridge has four legs and the arrow near the leg corresponds to the variable resistor. The balance button at the middle is used to balance the leg resistances. The top-left leg is a configurable resistance. The variable resistance value ranges to three decimal places, but this is a theoretical calculation based on the device specification. Hence, there could be minor variations in the equivalent resistance in the top-left leg.

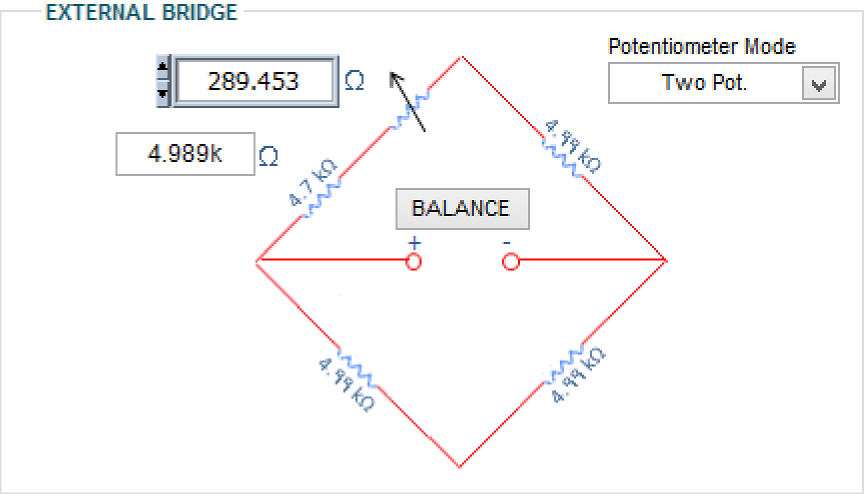


Figure 2-18. External Bridge Circuit

The Bridge Control voltage can be enabled and disabled by using the BRDG_EN checkbox and the voltage can be specified using the VBRDG_CTRL as shown in Figure 2-19.

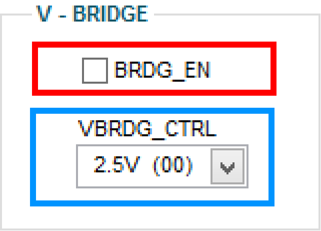


Figure 2-19. Bridge Control Circuit

2.5.2.3 Gain and ADC

- The page has options to enable VREF, ADC_CFG_1, and set the analog power.

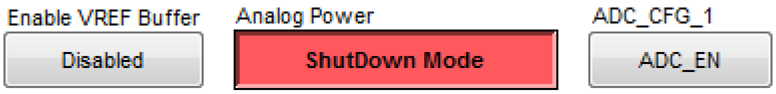
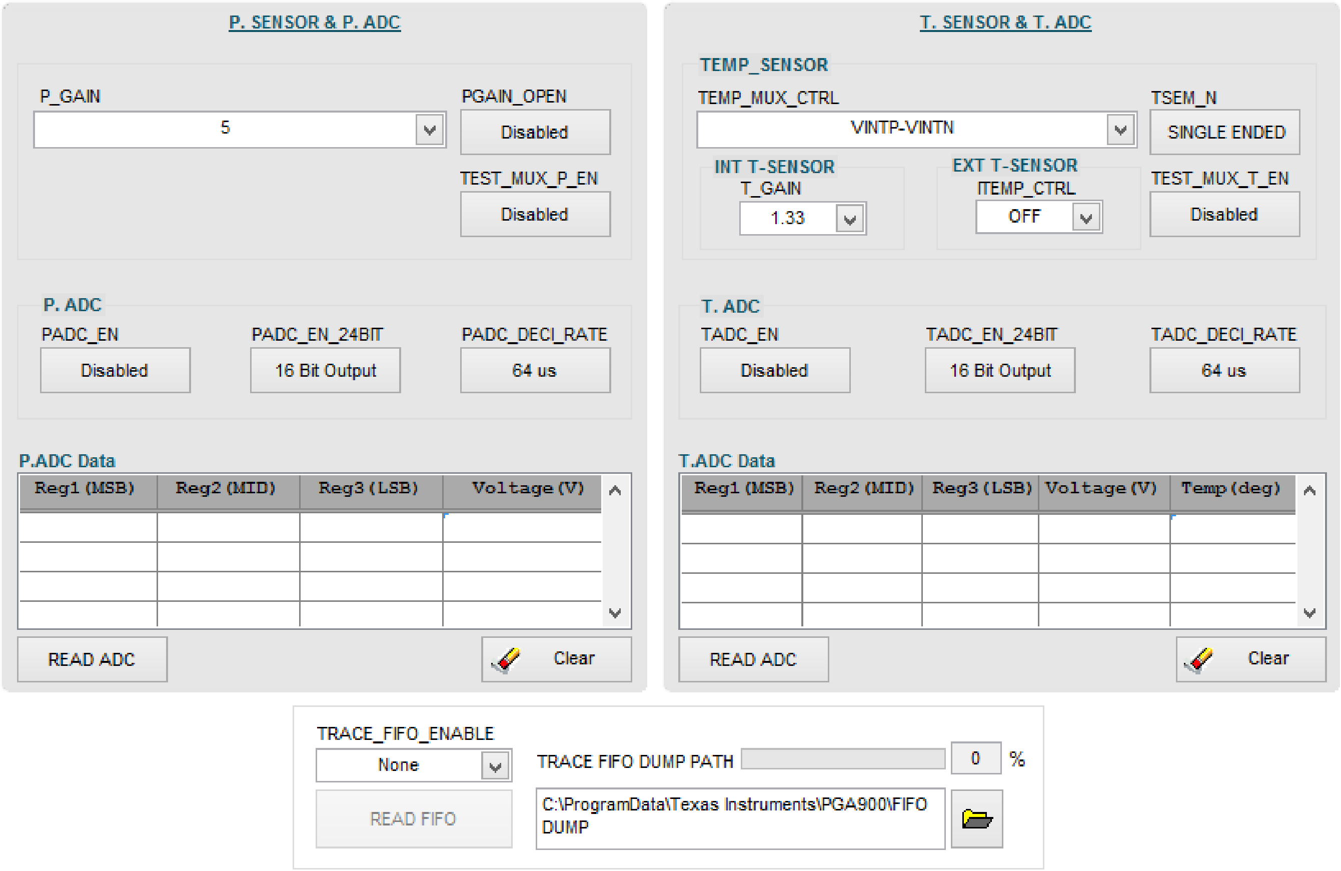


Figure 2-20. Enable Analog Config

- The enable VREF button enables or disables the ADC_EN_VREF bit.
- The **Analog Power** button enables the SD bit when set to ShutDown Mode and disables it when set to Out of ShutDown Mode.
- The **ADC_CFG_1** button writes 0x10 to the ADC_CFG_1 register when enabled and writes 0x00 to the register when disabled.
- The page also contains the temperature and pressure sensor settings for the ADC.
- The respective ADC is enabled by selecting the TADC_EN or the PADC_EN button.

- The table at the bottom of the tab control is used to read the ADC register values and display the corresponding voltage.
- The data could also be dumped into a file by selecting the corresponding ADC in the TRACE_FIFO_ENABLE dropdown menu. This enables the **READ FIFO** button and configures the Trace source.
- The path of the dump can be provided by clicking the **READ FIFO** button. The values would be dumped into the user-defined location.



P. SENSOR & P. ADC

P_GAIN: 5
PGAIN_OPEN: Disabled
TEST_MUX_P_EN: Disabled

P. ADC

PADC_EN: Disabled
PADC_EN_24BIT: 16 Bit Output
PADC_DECI_RATE: 64 us

P.ADC Data

Reg1 (MSB)	Reg2 (MID)	Reg3 (LSB)	Voltage (V)

T. SENSOR & T. ADC

TEMP_SENSOR

TEMP_MUX_CTRL: VINTP-VINTN
TSEM_N: SINGLE ENDED
INT T-SENSOR: T_GAIN: 1.33
EXT T-SENSOR: ITEMP_CTRL: OFF
TEST_MUX_T_EN: Disabled

T. ADC

TADC_EN: Disabled
TADC_EN_24BIT: 16 Bit Output
TADC_DECI_RATE: 64 us

T.ADC Data

Reg1 (MSB)	Reg2 (MID)	Reg3 (LSB)	Voltage (V)	Temp (deg)

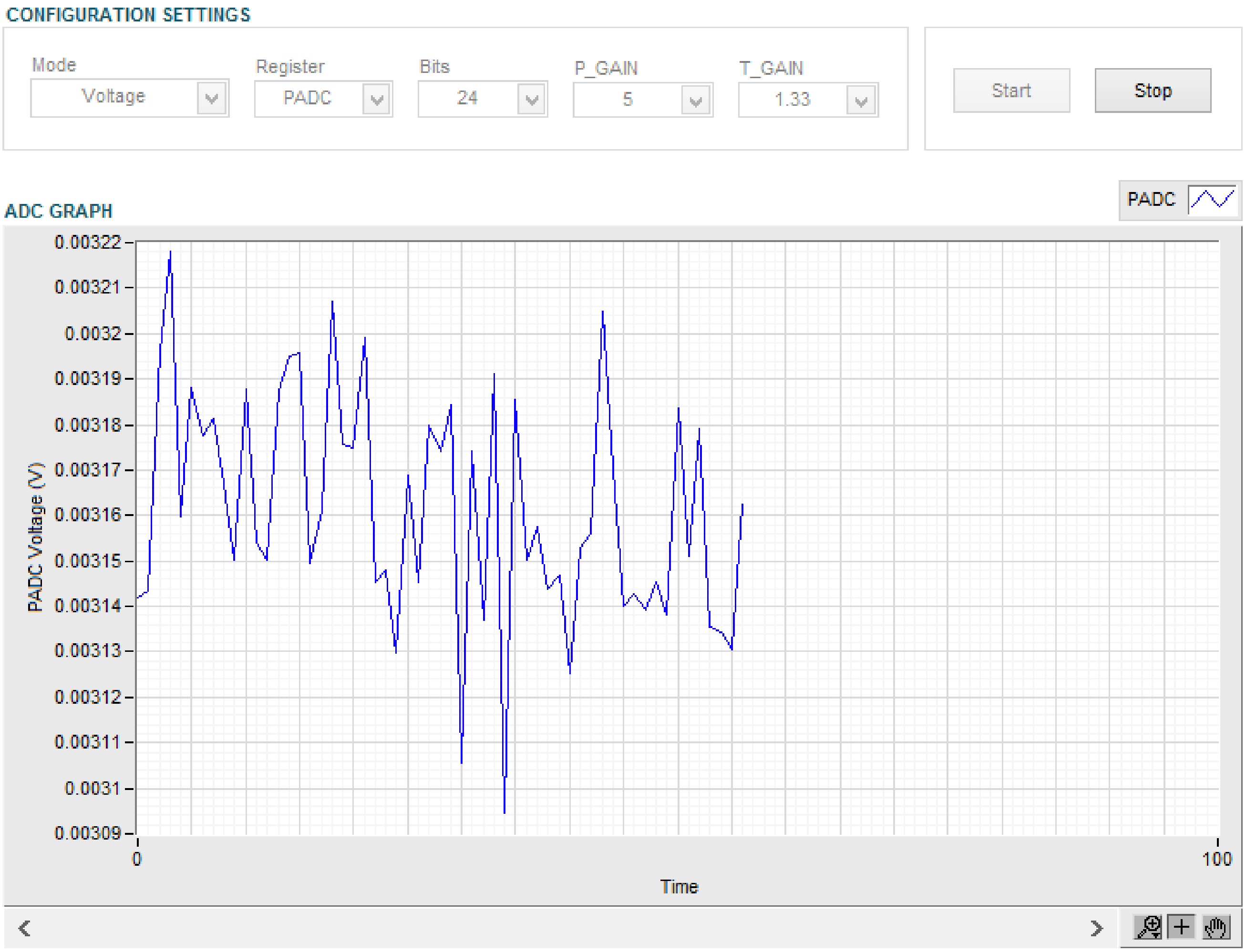
TRACE_FIFO_ENABLE: None
TRACE FIFO DUMP PATH: 0 %
C:\ProgramData\Texas Instruments\PGA900\FIFO DUMP

Figure 2-21. Temperature ADC With FIFO Enabled

- When the user presses the **Read FIFO** button, the following set of operations are performed.
  - The REMAP bit in the register is set to 0 to come out of the OTP.
  - The FIFO is flushed.
  - The FIFO is enabled again to have fresh data on the DEVRAM.
  - The GUI starts reading from the DEVRAM.

2.5.2.4 ADC Capture Page

- The ADC Capture Page allows the user to configure the required ADC configuration and perform a continuous ADC capture.
- This page allows the user to switch between ADC codes and voltages.
- After the required configurations are done on the UI, the capture can be started by clicking the **Start** button.
- The continuous capture data is displayed on the waveform graph on the UI.
- Click the **Stop** button to stop the capture at any point.



CONFIGURATION SETTINGS

Mode: Voltage
Register: PADC
Bits: 24
P_GAIN: 5
T_GAIN: 1.33

Figure 2-22. ADC Capture

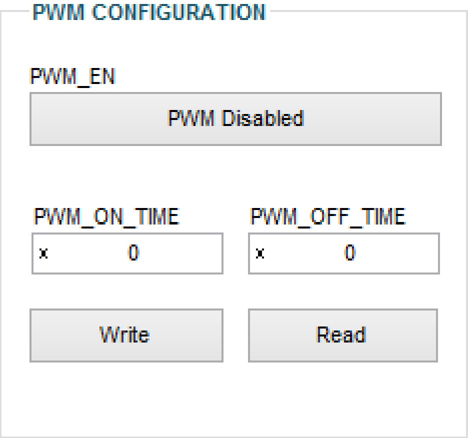
2.5.2.5 DAC and PWM Settings

The DAC and PWM Settings Page contain the DAC and the PWM features of the device, which are explained in the following sections.

- The device has PWM logic that can drive the DAC gain buffer directly.
- The PWM functionality uses a 16-bit 4-MHz free running timer. The PWM functionality can be enabled by writing a 1 to the PWM_EN bit in PWM_EN register. This enables the free running timer used for PWM functionality.

2.5.2.6 PWM ON and OFF Times

- The PWM ON and OFF times can be configured by writing the ON and OFF time values to PWM_ON_TIME and PWM_OFF_TIME registers, respectively.
- If the PWM_ON_TIME and PWM_OFF_TIME registers are updated, the updated ON and OFF times take effect at the start of the next ON time.
- The individual **Read** and **Write** buttons configure both the ON time and OFF time.


PWM CONFIGURATION

PWM_EN: PWM Disabled

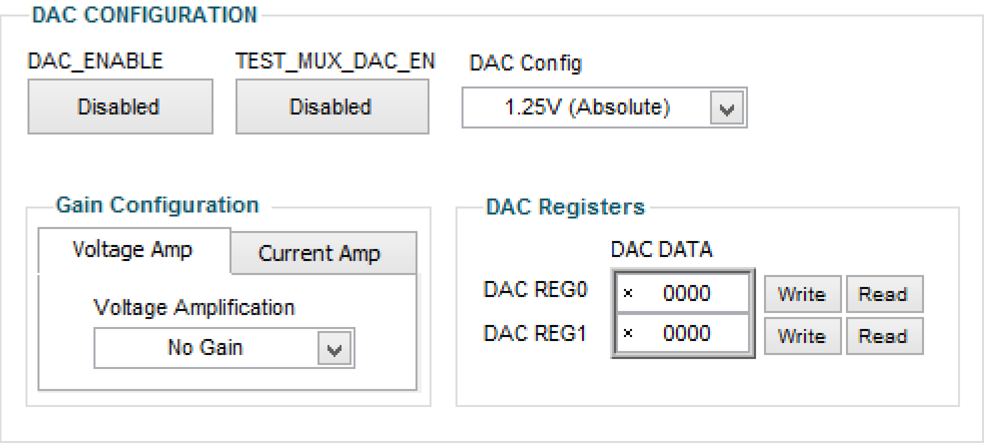
PWM_ON_TIME: x 0

PWM_OFF_TIME: x 0

Figure 2-23. PWM Configurations

2.5.2.7 DAC Configuration

- The device includes a 14-bit digital-to-analog converter that produces an absolute output voltage with respect to the accurate reference voltage or ratiometric output voltage with respect to the VDD supply.
- The DAC can be disabled by writing 0 to DAC_ENABLE bit in the DAC_CTRL_STATUS register. When the microprocessor undergoes a reset, the DAC registers are driven to 0x000 codes.
- The page also has options to Write and Read the DAC registers data.


DAC CONFIGURATION

DAC_ENABLE: Disabled

TEST_MUX_DAC_EN: Disabled

DAC Config: 1.25V (Absolute)

Gain Configuration

Voltage Amp | Current Amp

Voltage Amplification: No Gain

DAC Registers

DAC DATA

DAC REG0: x 0000

DAC REG1: x 0000

Figure 2-24. Disable DAC

2.5.2.8 Ratiometric vs Absolute

- The DAC output can be configured to be either in ratiometric-to-VDD mode or independent-of-VDD (or absolute) mode using the DAC_RATIOMETRIC bit in DAC_CONFIG.
- In ratiometric mode, changes in the VDD voltage result in a proportional change in the output voltage because the current reference for the DAC is derived from VDD.

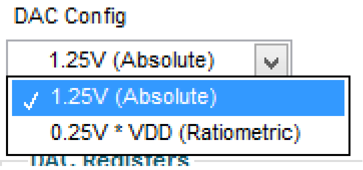


Figure 2-25. Configure DAC

2.5.2.9 DAC Gain

- The DAC gain buffer is a configurable buffer stage for the DAC output.
- The DAC gain amplifier can be configured to operate in voltage amplification mode for voltage output or current amplification mode for 4-20-mA applications.
- In voltage output mode, DAC gain can be configured for a specific gain value by setting the DAC_GAIN bits in the DAC_CONFIG register to a specific value.
- The DAC gain can be configured to one of five possible gain configurations using the 2-bit DAC_GAIN field.
- The final stage of DAC gain is connected to Vddp and ground. This gives the ability to drive VOUT voltage close to VDD voltage.
- The DAC gain buffer also implements a COMP pin to allow the implementation of compensation when driving large capacitive loads.

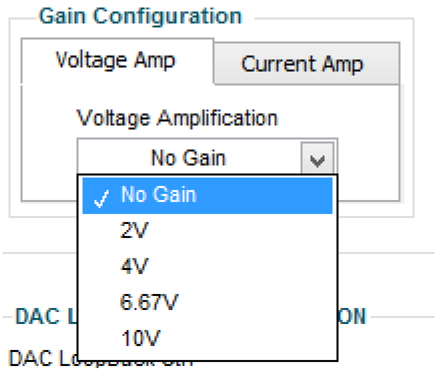
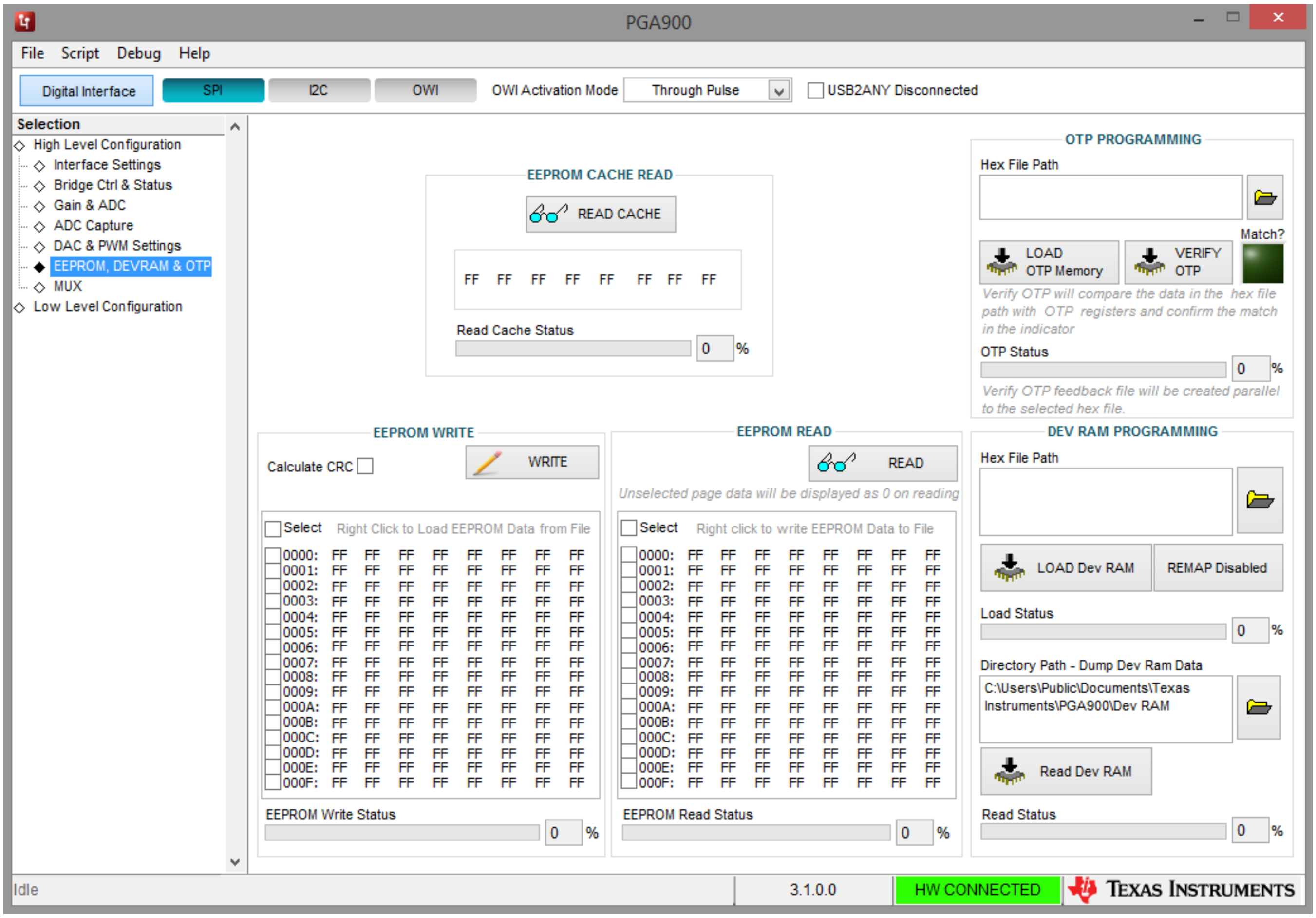


Figure 2-26. DAC Gain Configuration

2.5.2.10 EEPROM, DEVRAM, and OTP

This page describes the EEPROM, OTP, and the DEVRAM programming functionalities of the PGA900 EVM as shown in Figure 2-27. Each of the device functions are discussed in the following sections.



OWI Activation Mode: Through Pulse

□ USB2ANY Disconnected

EEPROM CACHE READ: FF FF FF FF FF FF FF FF

Read Cache Status: 0 %

Verify OTP will compare the data in the hex file path with OTP registers and confirm the match in the indicator

OTP Status: 0 %

Verify OTP feedback file will be created parallel to the selected hex file.

Calculate CRC □

Unselected page data will be displayed as 0 on reading

EEPROM WRITE: □ Select Right Click to Load EEPROM Data from File

```
□0000: FF FF FF FF FF FF FF FF
□0001: FF FF FF FF FF FF FF FF
□0002: FF FF FF FF FF FF FF FF
□0003: FF FF FF FF FF FF FF FF
□0004: FF FF FF FF FF FF FF FF
□0005: FF FF FF FF FF FF FF FF
□0006: FF FF FF FF FF FF FF FF
□0007: FF FF FF FF FF FF FF FF
□0008: FF FF FF FF FF FF FF FF
□0009: FF FF FF FF FF FF FF FF
□000A: FF FF FF FF FF FF FF FF
□000B: FF FF FF FF FF FF FF FF
□000C: FF FF FF FF FF FF FF FF
□000D: FF FF FF FF FF FF FF FF
□000E: FF FF FF FF FF FF FF FF
□000F: FF FF FF FF FF FF FF FF
```

EEPROM Write Status: 0 %

EEPROM READ: □ Select Right click to write EEPROM Data to File

```
□0000: FF FF FF FF FF FF FF FF
□0001: FF FF FF FF FF FF FF FF
□0002: FF FF FF FF FF FF FF FF
□0003: FF FF FF FF FF FF FF FF
□0004: FF FF FF FF FF FF FF FF
□0005: FF FF FF FF FF FF FF FF
□0006: FF FF FF FF FF FF FF FF
□0007: FF FF FF FF FF FF FF FF
□0008: FF FF FF FF FF FF FF FF
□0009: FF FF FF FF FF FF FF FF
□000A: FF FF FF FF FF FF FF FF
□000B: FF FF FF FF FF FF FF FF
□000C: FF FF FF FF FF FF FF FF
□000D: FF FF FF FF FF FF FF FF
□000E: FF FF FF FF FF FF FF FF
□000F: FF FF FF FF FF FF FF FF
```

EEPROM Read Status: 0 %

Load Status: 0 %

Directory Path - Dump Dev Ram Data: C:\Users\Public\Documents\Texas Instruments\PGA900\Dev RAM

Read Status: 0 %

Idle | 3.1.0.0 | HW CONNECTED

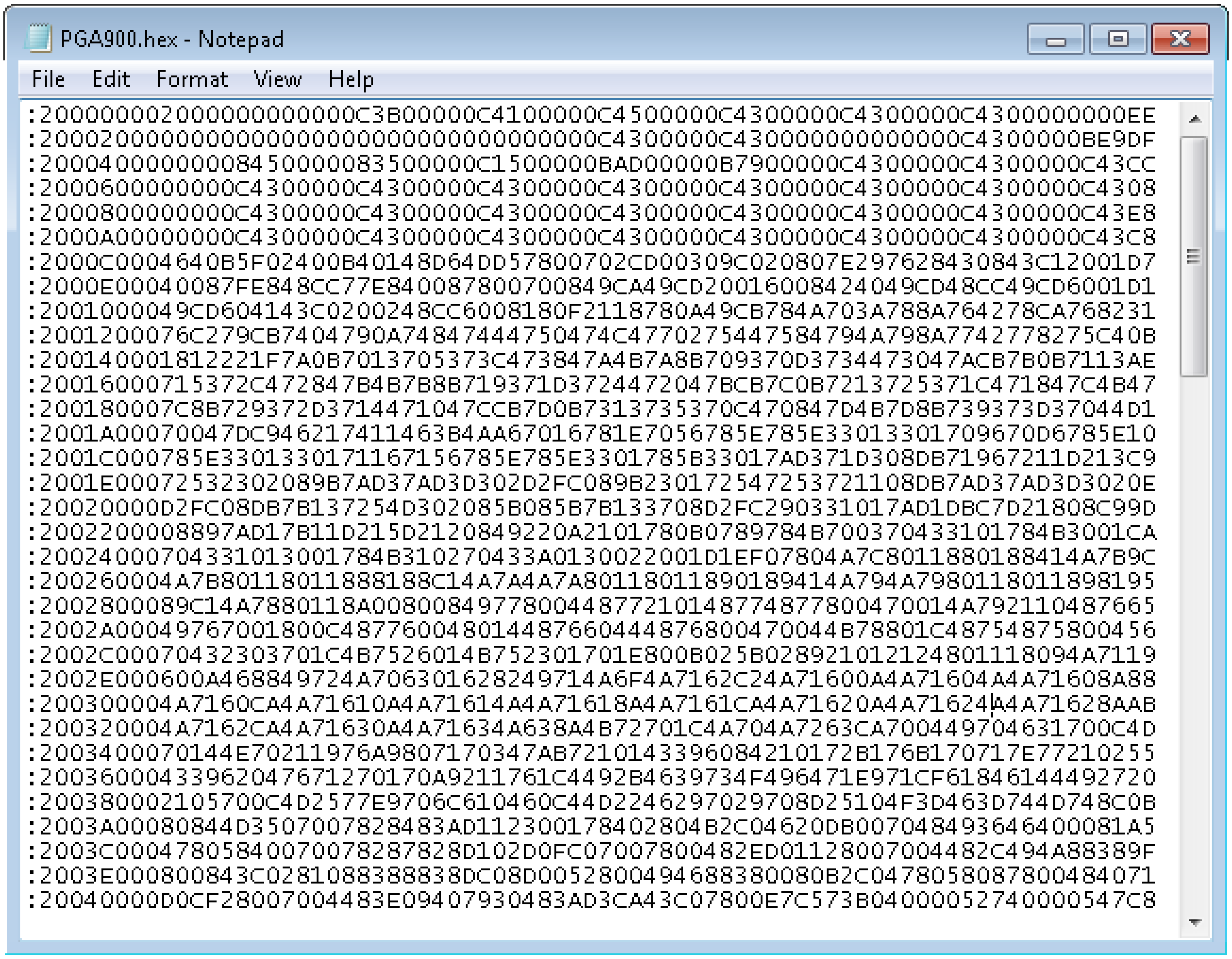
Figure 2-27. EEPROM, DEVRAM, and OTP

2.5.2.10.1 OTP Programming

- OTP memory is 8-bit addressable.
- To allow the communication interface to access these memories using 8-bit address, the memory space is organized as 256 bytes/page.
- There are a total of 32 pages.

2.5.2.10.1.1 Load OTP Memory

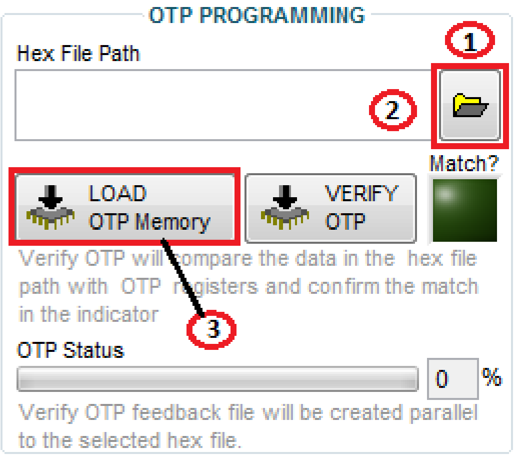
- The data to be written to the OTP is provided to the device as a HEX file. Figure 2-28 shows a typical HEX file.



```
:200000002000000000000C3B00000C4100000C4500000C4300000C4300000C4300000000EE
:200020000000000000000000000000000000000C4300000C430000000000000C4300000BE9DF
:2000400000000845000008350000 0C1500000BAD00000B7900000C4300000C4300000C43CC
:2000600000000C4300000C4300000C4300000C4300000C4300000C4300000C4300000C4308
:2000800000000C4300000C4300000C4300000C4300000C4300000C4300000C4300000C43E8
:2000A00000000C4300000C4300000C4300000C4300000C4300000C4300000C4300000C43C8
:2000C0004640B5F02400B40148D64DD57800702CD00309C020807E297628430843C12001D7
:2000E00040087FE848CC77E840087800700849CA49CD20016008424049CD48CC49CD6001D1
:2001000049CD604143C0200248CC6008180F2118780A49CB784A703A788A764278CA768231
:2001200076C279CB7404790A74847444750474C4770275447584794A798A7742778275C40B
:200140001812221F7A0B7013705373C473847A4B7A8B709370D3734473047ACB7B0B7113AE
:20016000715372C472847B4B7B8B719371D3724472047BCB7C0B7213725371C471847C4B47
:200180007C8B729372D3714471047CCB7D0B7313735370C470847D4B7D8B739373D37044D1
:2001A00070047DC946217411463B4AA67016781E7056785E785E33013301709670D6785E10
:2001C000785E3301330171167156785E785E3301785B33017AD371D308DB71967211D213C9
:2001E00072532302089B7AD37AD3D302D2FC089B230172547253721108DB7AD37AD3D3020E
:20020000D2FC08DB7B137254D302085B085B7B133708D2FC290331017AD1DBC7D21808C99D
:2002200008897AD17B11D215D2120849220A2101780B0789784B700370433101784B3001CA
:2002400070433101300178 4B310270433A0130022001D1EF07804A7C8011880188414A7B9C
:200260004A7B80118011888188C14A7A4A7A80118011890189414A794A798011801189 8195
:2002800089C14A7880118A008008497780044877210148774877800470014A792110487665
:2002A00049767001800C4877600480144876604448768004700 44B78801C487548758004 56
:2002C0007043230370 1C4B7526014B752301701E800B025B028921012124801118094A7119
:2002E000600A468849724A706301628249714A6F4A7162C24A71600A4A71604A4A71608A88
:200300004A7160CA4A71610A4A71614A4A71618A4A7161CA4A71620A4A71624A4A71628AAB
:200320004A7162CA4A71630A4A71634A638A4B72701C4A704A7263CA700449704631700C4D
:2003400070144E70211976A9807170347AB721014339608421 0172B176B170717E77210255
:2003600043396204767127 0170A9211761C4492B4639734F496471E971CF61846144492720
:200380002105700C4D2577E9706C610460C44D2246297029708D25104F3D463D744D748C0B
:2003A00080844D3507007828483AD112300178402804B2C04620DB007048493646400081A5
:2003C00047805840070078287828D102D0FC07007800482ED01128007004482C494A88389F
:2003E000800843C0281088388838DC08D005280049468838 0080B2C0478058087800484071
:20040000D0CF28007004483E09407930483AD3CA43C07800E7C573B040000527400005 47C8
```

Figure 2-28. Sample HEX File

- Select the HEX file path by clicking the browse button as shown in Figure 2-29(1). The OTP memory is programmed with the contents of the HEX file after clicking the **Load OTP Memory** button.



Verify OTP will compare the data in the hex file path with OTP registers and confirm the match in the indicator

Verify OTP feedback file will be created parallel to the selected hex file.

(1) Click the folder button to load HEX file into GUI.

(2) Select required file from browser window.

(3) Click **Load OTP Memory** button.

Figure 2-29. OTP Programming

- If the **Verify OTP** button is pressed, the GUI compares the contents of the OTP memory with the selected HEX file. If the OTP memory matches the contents of the .HEX file, the GUI sets the *Match?* LED.

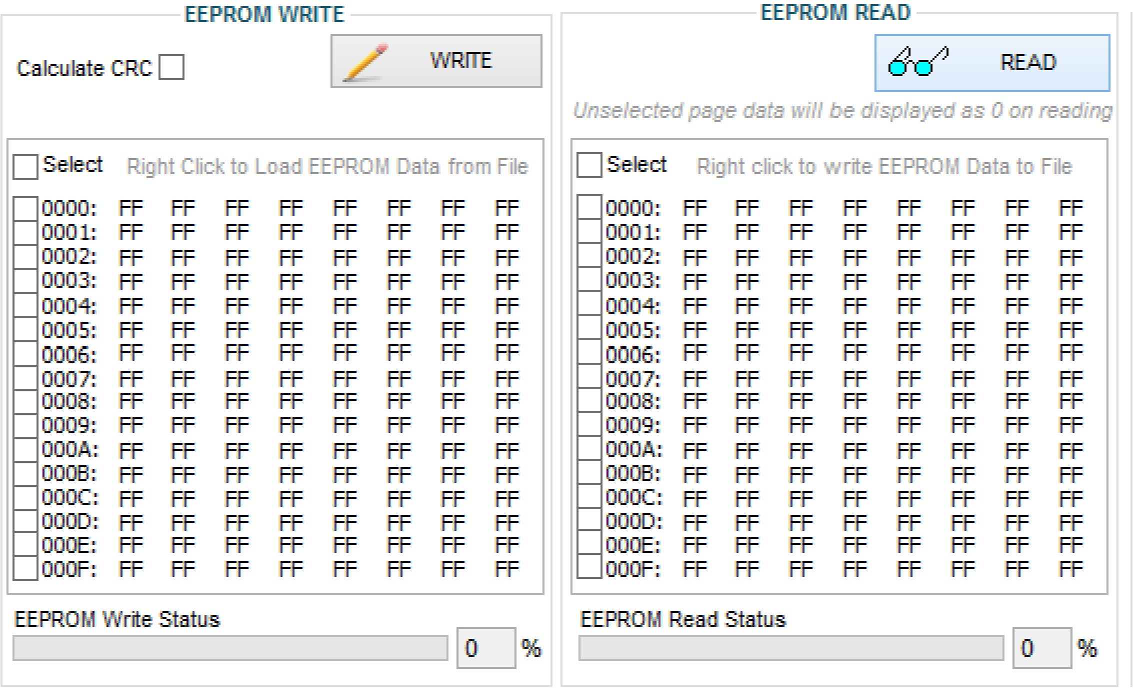
Verify OTP function does these activities:

- It reads all 32 pages of the OTP registers.
- Compares the defined register data in the HEX file for matching data from the read array and checks if the rest of the registers (which are not defined in the HEX file) are 0s (because the OTP function cannot compare values for non-existing registers in the hex file).
- If both these match, the GUI enables the *Match?* Boolean in the front panel.
- A verification log file is generated in the directory parallel to the selected hex file with the same file name as the selected hex file except with the text “_OTPLog” appended. This file contains the details (address, hex file data, and device data, both data in decimal representation) about all the location where a data mismatch was encountered.

NOTE: If the user has a few registers already configured and wrote only the remaining registers using the HEX file, then the VERIFY OTP function will give a FALSE for that HEX file because the existing OTP register values are not known to the GUI.

2.5.2.10.2 EEPROM WRITE and READ

- This section allows you to read-from or write-to the HEX files.
- In the EEPROM Write area in the GUI, right-click to select an EEPROM file.



□	0000:	FF	FF	FF	FF	FF	FF	FF	FF
□	0001:	FF	FF	FF	FF	FF	FF	FF	FF
□	0002:	FF	FF	FF	FF	FF	FF	FF	FF
□	0003:	FF	FF	FF	FF	FF	FF	FF	FF
□	0004:	FF	FF	FF	FF	FF	FF	FF	FF
□	0005:	FF	FF	FF	FF	FF	FF	FF	FF
□	0006:	FF	FF	FF	FF	FF	FF	FF	FF
□	0007:	FF	FF	FF	FF	FF	FF	FF	FF
□	0008:	FF	FF	FF	FF	FF	FF	FF	FF
□	0009:	FF	FF	FF	FF	FF	FF	FF	FF
□	000A:	FF	FF	FF	FF	FF	FF	FF	FF
□	000B:	FF	FF	FF	FF	FF	FF	FF	FF
□	000C:	FF	FF	FF	FF	FF	FF	FF	FF
□	000D:	FF	FF	FF	FF	FF	FF	FF	FF
□	000E:	FF	FF	FF	FF	FF	FF	FF	FF
□	000F:	FF	FF	FF	FF	FF	FF	FF	FF

EEPROM Write Status: 0 %

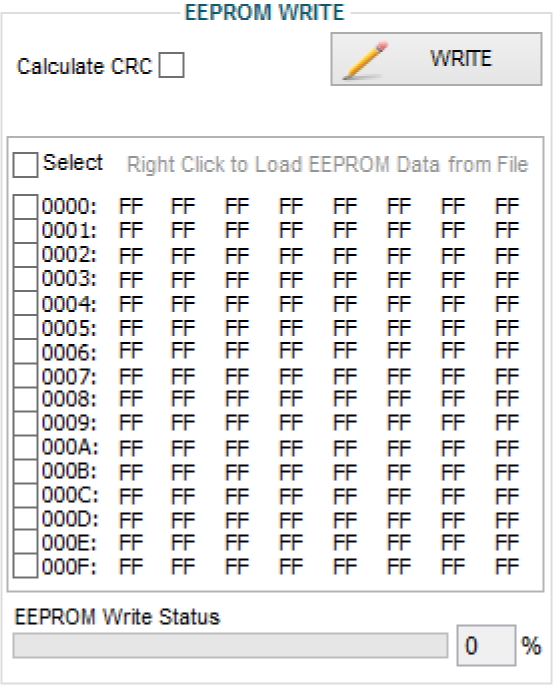
Unselected page data will be displayed as 0 on reading

□	0000:	FF	FF	FF	FF	FF	FF	FF	FF
□	0001:	FF	FF	FF	FF	FF	FF	FF	FF
□	0002:	FF	FF	FF	FF	FF	FF	FF	FF
□	0003:	FF	FF	FF	FF	FF	FF	FF	FF
□	0004:	FF	FF	FF	FF	FF	FF	FF	FF
□	0005:	FF	FF	FF	FF	FF	FF	FF	FF
□	0006:	FF	FF	FF	FF	FF	FF	FF	FF
□	0007:	FF	FF	FF	FF	FF	FF	FF	FF
□	0008:	FF	FF	FF	FF	FF	FF	FF	FF
□	0009:	FF	FF	FF	FF	FF	FF	FF	FF
□	000A:	FF	FF	FF	FF	FF	FF	FF	FF
□	000B:	FF	FF	FF	FF	FF	FF	FF	FF
□	000C:	FF	FF	FF	FF	FF	FF	FF	FF
□	000D:	FF	FF	FF	FF	FF	FF	FF	FF
□	000E:	FF	FF	FF	FF	FF	FF	FF	FF
□	000F:	FF	FF	FF	FF	FF	FF	FF	FF

EEPROM Read Status: 0 %

Figure 2-30. Selecting EEPROM Data File

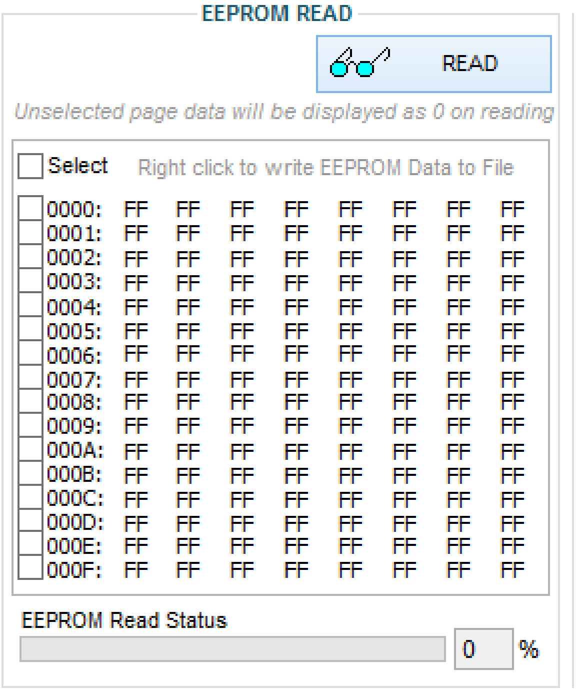
- When a particular file is selected, the data is loaded as shown in Figure 2-31. An individual byte can also be edited using the GUI before writing to the device. The Calculate CRC calculates the CRC as per the logic defined in the data sheet (SLDS209) for the write operation.
- CRC conditions:
  - When checked without all the pages selected, the GUI reads all 128 bytes of existing EEPROM data and replaces the data of the selected page with the newly configured data from the EEPROM write box and calculates the resulting CRC. After the calculation, the GUI writes the data of the selected pages and the last page with updated CRC value to the EEPROM.
  - When all the pages are selected, the GUI does not read all the data, but directly calculates CRC from the EEPROM WRITE BOX and writes to all the pages with the updated CRC value to the last byte.



```
□ 0000: FF FF FF FF FF FF FF FF
□ 0001: FF FF FF FF FF FF FF FF
□ 0002: FF FF FF FF FF FF FF FF
□ 0003: FF FF FF FF FF FF FF FF
□ 0004: FF FF FF FF FF FF FF FF
□ 0005: FF FF FF FF FF FF FF FF
□ 0006: FF FF FF FF FF FF FF FF
□ 0007: FF FF FF FF FF FF FF FF
□ 0008: FF FF FF FF FF FF FF FF
□ 0009: FF FF FF FF FF FF FF FF
□ 000A: FF FF FF FF FF FF FF FF
□ 000B: FF FF FF FF FF FF FF FF
□ 000C: FF FF FF FF FF FF FF FF
□ 000D: FF FF FF FF FF FF FF FF
□ 000E: FF FF FF FF FF FF FF FF
□ 000F: FF FF FF FF FF FF FF FF
```

Figure 2-31. EEPROM Data Loaded

Read EEPROM function provides the user the feature to read back the EEPROM DATA


EEPROM READ
READ
Unselected page data will be displayed as 0 on reading
□ Select
Right click to write EEPROM Data to File
EEPROM Read Status
0 %

```
□ 0000: FF FF FF FF FF FF FF FF
□ 0001: FF FF FF FF FF FF FF FF
□ 0002: FF FF FF FF FF FF FF FF
□ 0003: FF FF FF FF FF FF FF FF
□ 0004: FF FF FF FF FF FF FF FF
□ 0005: FF FF FF FF FF FF FF FF
□ 0006: FF FF FF FF FF FF FF FF
□ 0007: FF FF FF FF FF FF FF FF
□ 0008: FF FF FF FF FF FF FF FF
□ 0009: FF FF FF FF FF FF FF FF
□ 000A: FF FF FF FF FF FF FF FF
□ 000B: FF FF FF FF FF FF FF FF
□ 000C: FF FF FF FF FF FF FF FF
□ 000D: FF FF FF FF FF FF FF FF
□ 000E: FF FF FF FF FF FF FF FF
□ 000F: FF FF FF FF FF FF FF FF
```

Figure 2-32. EEPROM Read Data

EEPROM pages can be individually selected and written or read using the checkbox on the left side of the data map.

EEPROM cache read: This function helps the user to read back the 8 bytes of data from the EEPROM cache.

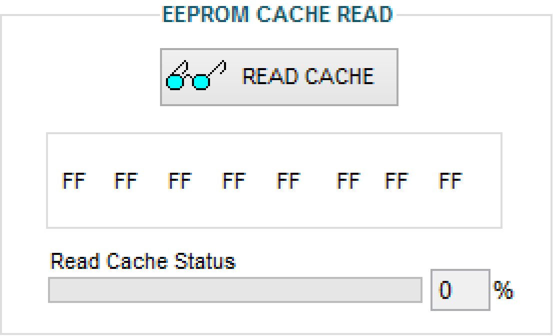
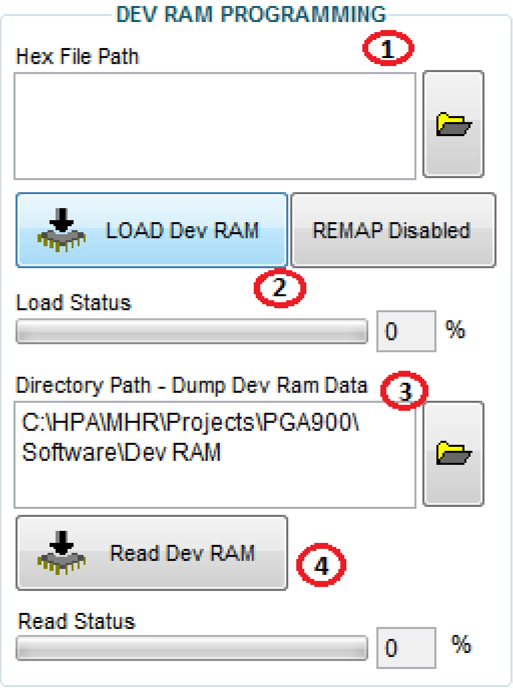


Figure 2-33. EEPROM Cache Read

2.5.2.10.3 DEVRAM Programming

- PGA900 has 8KB of development RAM (DEVRAM) that can overlay the OTP memory address. This allows convenient programming of the device.
- If the REMAP checkbox is selected, the GUI sets the REMAP bit to 0, loads the DEVRAM, and then sets back the REMAP to 1. TI always recommends this method so the data gets written to the correct memory and not to OTP in case the REMAP was set to 1 before this operation.



C:\HPA\MHR\Projects\PGA900\Software\Dev RAM

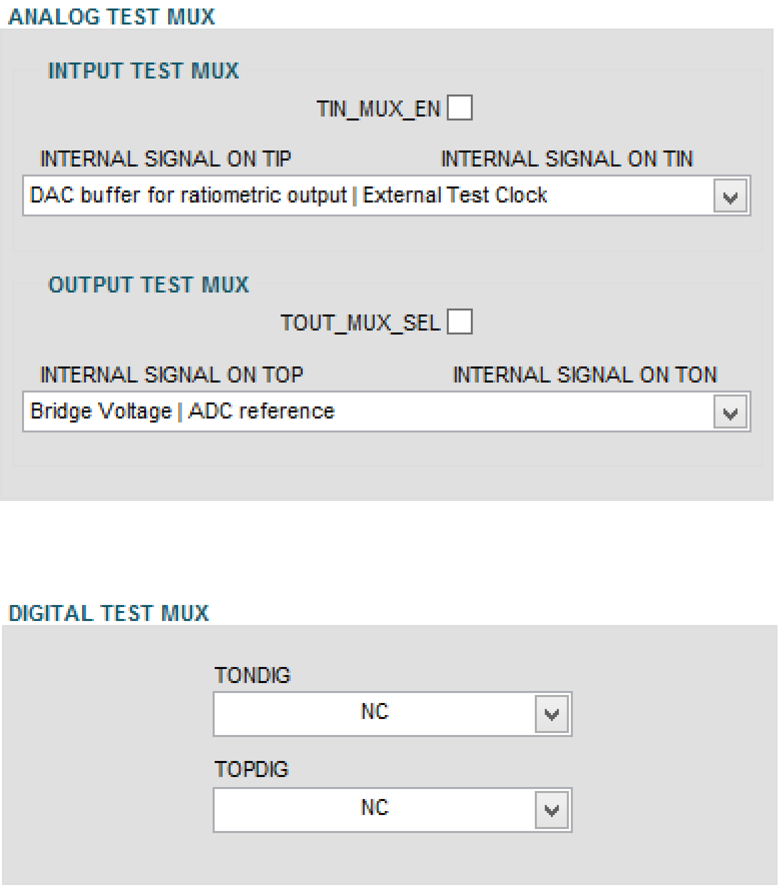
Figure 2-34. DEVRAM Programming

1. Browse the .HEX file path to load the HEX file to the GUI.
2. Click the **LOAD Dev RAM** button to start programming the DEVRAM.
3. Browse the path to save the DEVRAM.
4. Click **Read Dev RAM** to read the data from development RAM.

Note that the status of read is shown by the progress bar.

2.5.2.11 MUX

- The ADC, MUX, and Sensor settings page displays the various digital and analog MUX used by the PGA900 EVM board.
- The page provides various options to control these MUX as shown in Figure 2-35.
- The TIN_MUX_EN checkbox enables or disables the input test MUX.
- Similarly, the TOUT_MUX_SEL checkbox enables or disables the output test MUX.
- The required signal can be MUXed out using the dropdown selection control.



ANALOG TEST MUX

INTPUT TEST MUX

TIN_MUX_EN □

INTERNAL SIGNAL ON TIP INTERNAL SIGNAL ON TIN

DAC buffer for ratiometric output | External Test Clock

OUTPUT TEST MUX

TOUT_MUX_SEL □

INTERNAL SIGNAL ON TOP INTERNAL SIGNAL ON TON

Bridge Voltage | ADC reference

DIGITAL TEST MUX

TONDIG

NC

TOPDIG

NC

Figure 2-35. Analog and Digital MUX Options

2.6 Menu Options

2.6.1 File

The File menu contains the Exit option as shown in Figure 2-36. The Exit option is used to stop the execution of the PGA900 GUI.

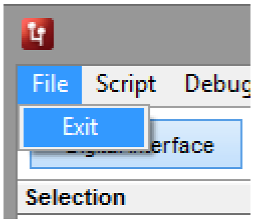


Figure 2-36. File Menu

2.6.2 Script

- Scripting is used to automate the device operations and reduces the time consumption while repeating similar operations.
- This is helpful in situations where performing a particular device function requires setting 10 to 15 registers to a particular value. In these circumstances, scripts can be recorded and run whenever needed.
- In the PGA900 GUI, the scripting is done using Python because:
  - It is easier to implement
  - More widely used
  - More user friendly

2.6.2.1 Performing Macro Recording

- To create a custom macro, click Script → Launch Window.

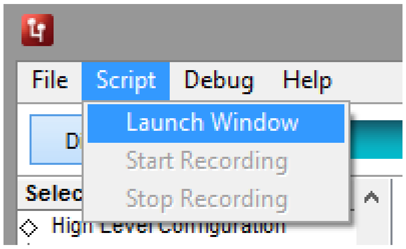


Figure 2-37. Launch Window

- The Python IDLE window appears as shown in Figure 2-38.

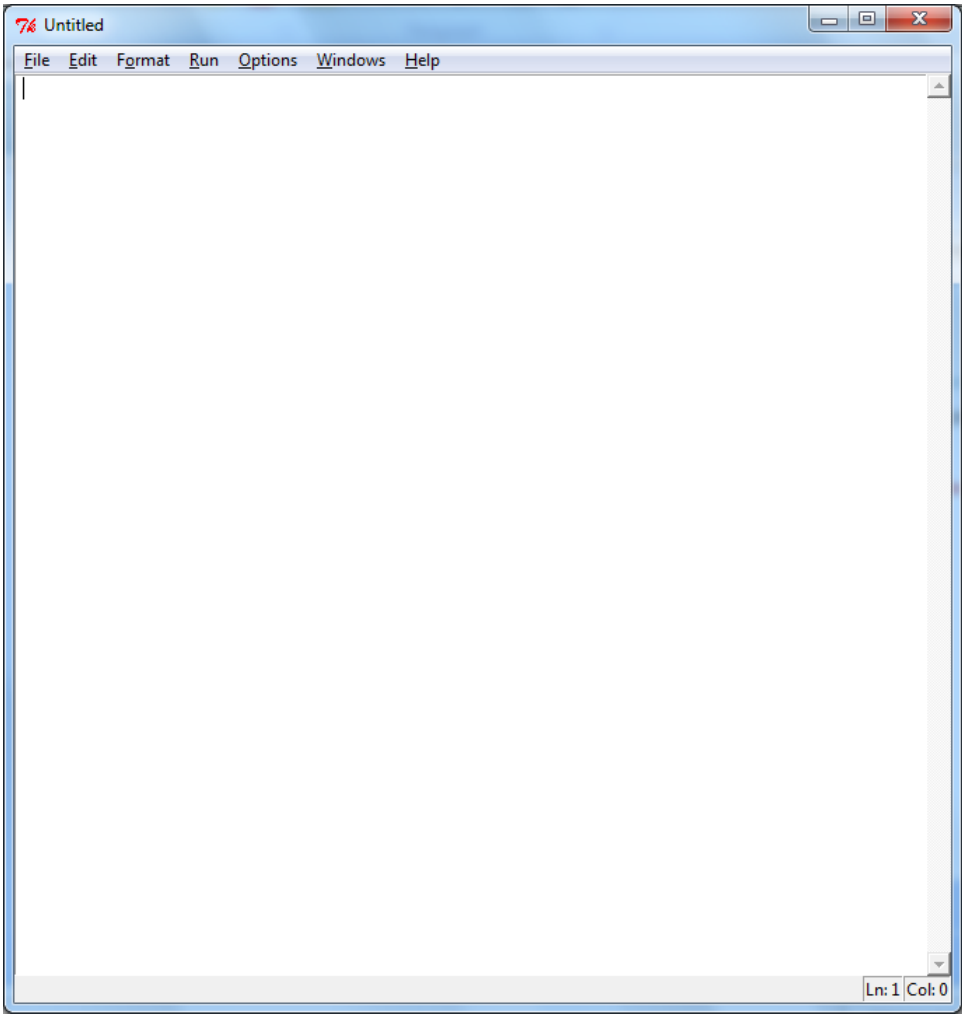


Figure 2-38. Python IDLE Window

NOTE: Selecting the launch window again opens another untitled window, and the one opened last will be active.

- Click Script → Start Recording. The IDLE window becomes green to show that recording has started.

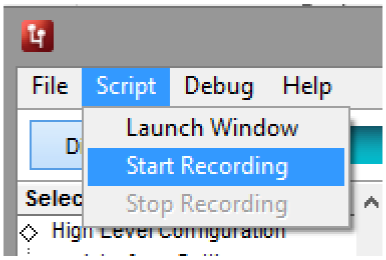
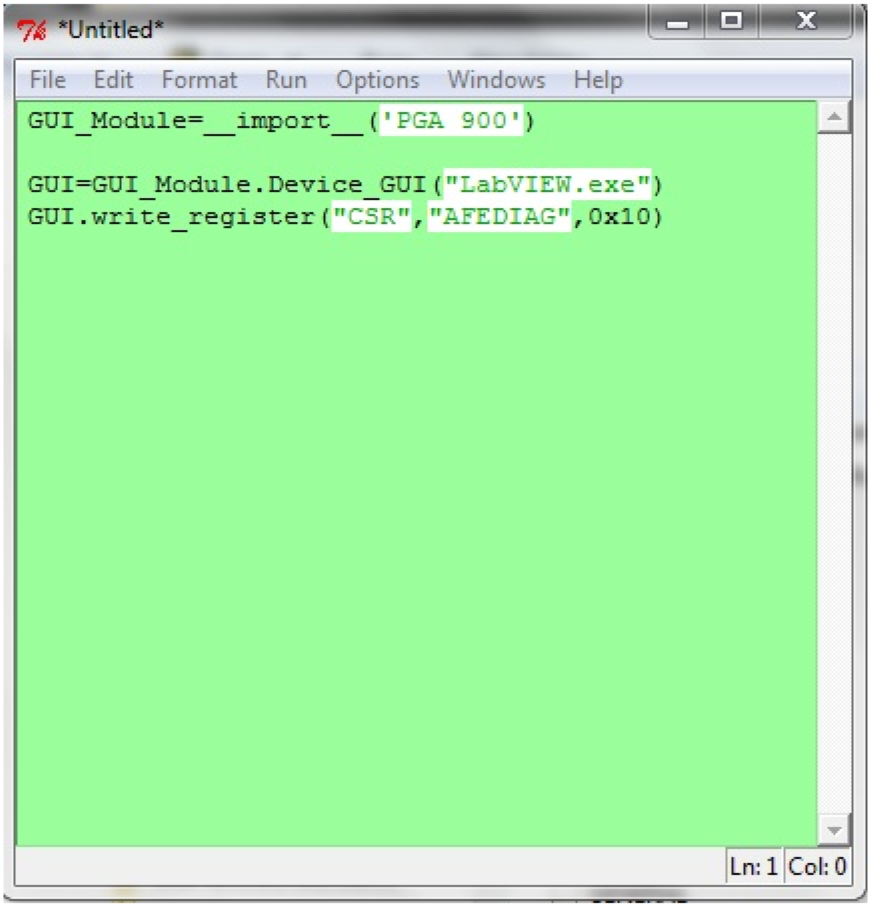


Figure 2-39. Start Recording

- Go to Low Level Page of PGA900, select AFEDIAG register and write 10 to “Data” control and click **Write Register**. The action is recorded as shown in Figure 2-40.



```
GUI_Module=__import__('PGA 900')

GUI=GUI_Module.Device_GUI("LabVIEW.exe")
GUI.write_register("CSR","AFEDIAG",0x10)
```

Figure 2-40. Macro Recording

- Stop the recording by clicking on Script → Stop Recording.

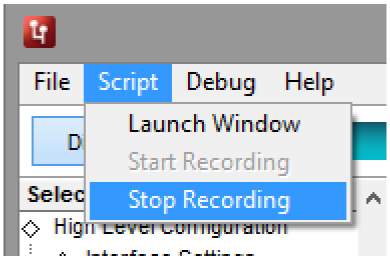
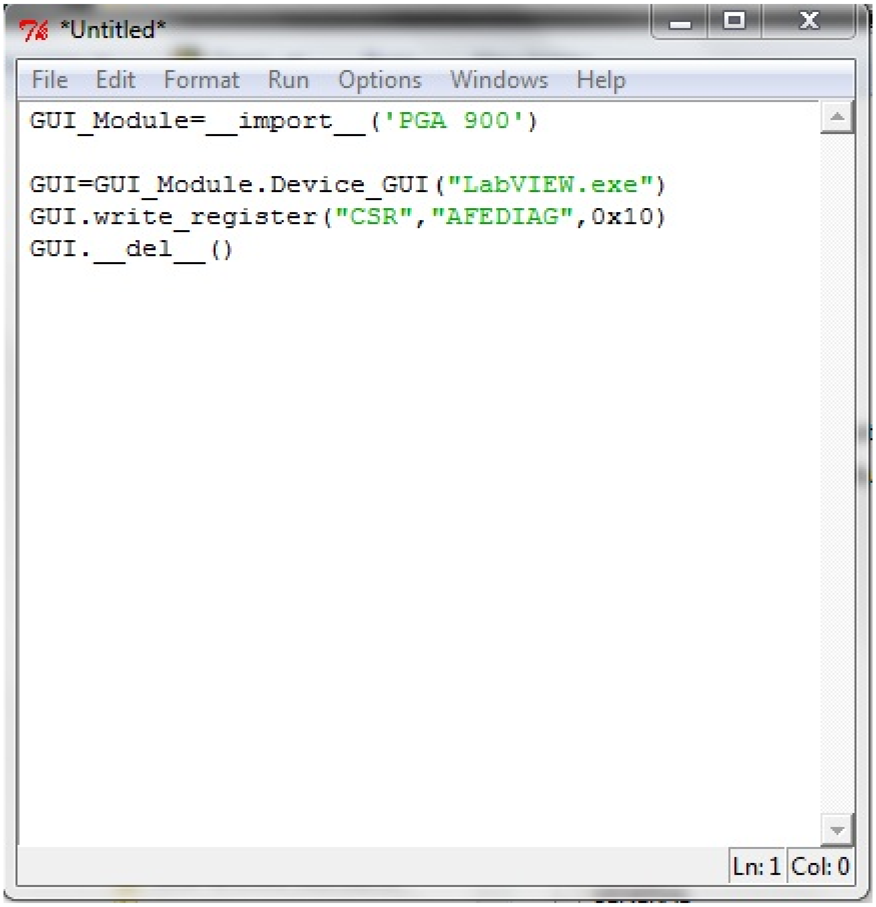


Figure 2-41. Stop Recording

- The Python IDLE window will no longer be green, which indicates that the recording stopped.

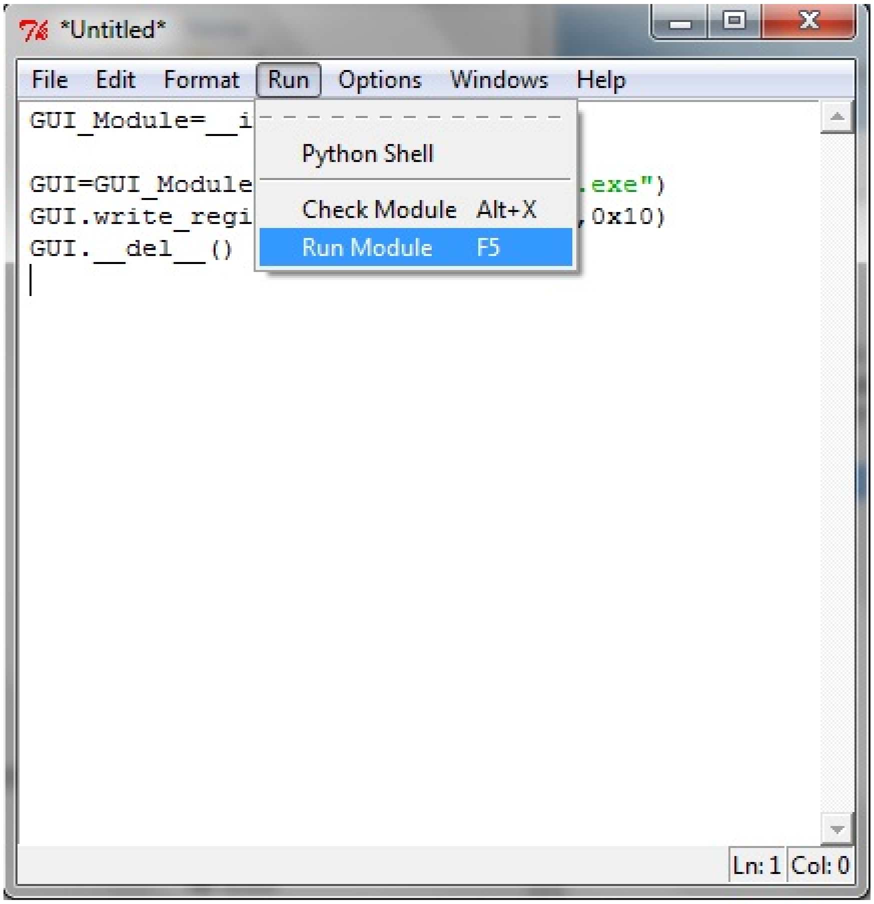


```
GUI_Module=__import__('PGA 900')

GUI=GUI_Module.Device_GUI("LabVIEW.exe")
GUI.write_register("CSR","AFEDIAG",0x10)
GUI.__del__()
```

Figure 2-42. Finished Macro Recording

- Run the script by either clicking Run → Run Module in the IDLE window or pressing F5.



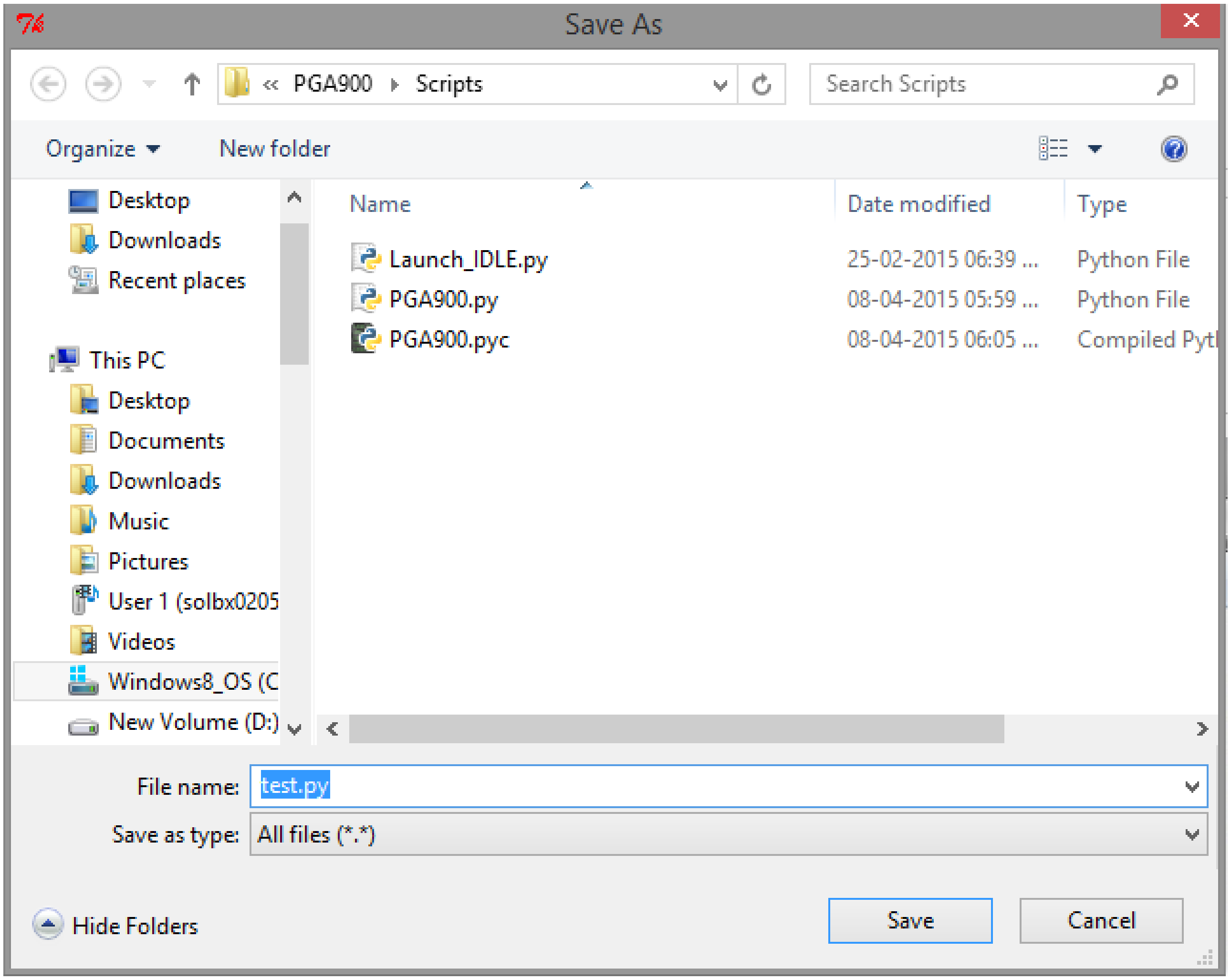
```
GUI_Module=__i

GUI=GUI_Module                          .exe")
GUI.write_regi                          ,0x10)
GUI.__del__()
```

Figure 2-43. Run Module

- The *Save As* dialog will appear asking for the file name for the script.
- Give any name, for example, Test.py and click **Save**.

NOTE: The name of the script should have the extension **.py** as shown in Figure 2-44.



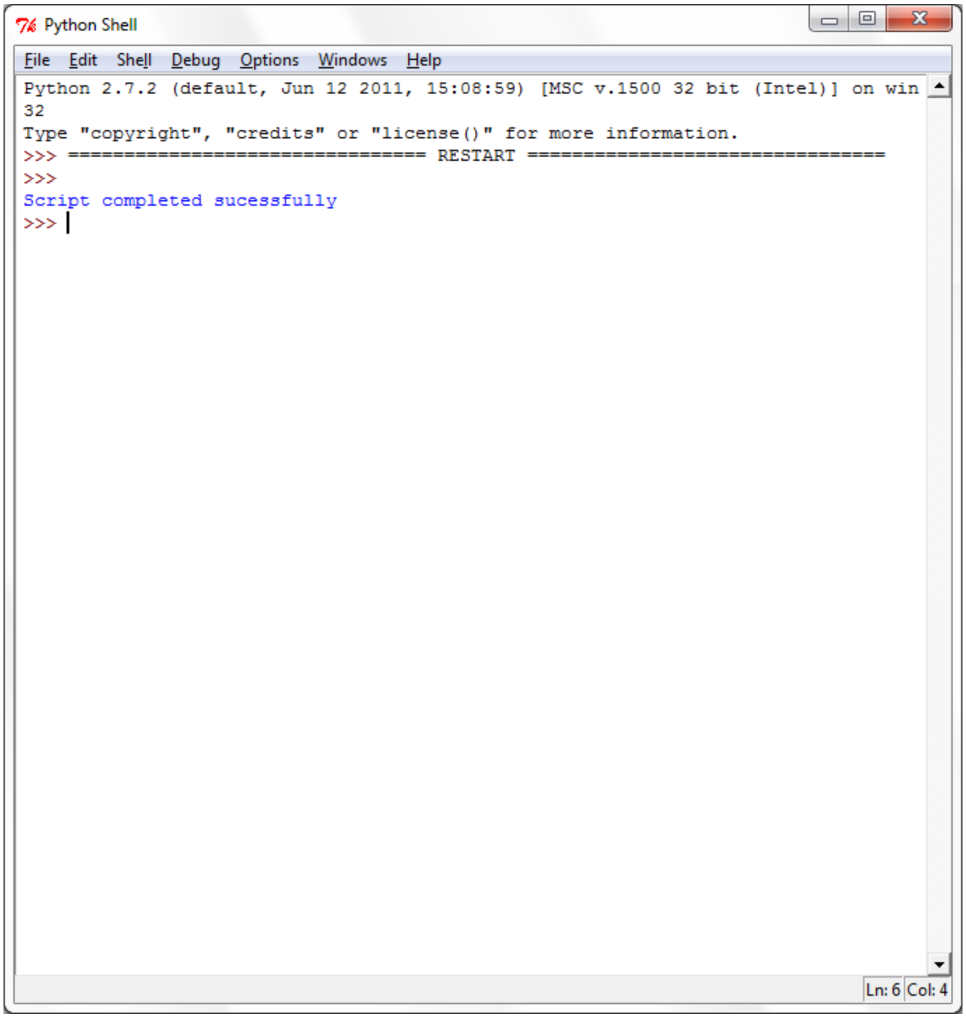
Name	Date modified	Type
Launch_IDLE.py	25-02-2015 06:39 ...	Python File
PGA900.py	08-04-2015 05:59 ...	Python File
PGA900.pyc	08-04-2015 06:05 ...	Compiled Pyt

File name: test.py

Save as type: All files (*.*)

Figure 2-44. Save Browser Window

- Now, the script will run and the status will be displayed in the Python shell window as shown in Figure 2-45.



```
Python 2.7.2 (default, Jun 12 2011, 15:08:59) [MSC v.1500 32 bit (Intel)] on win
32
Type "copyright", "credits" or "license()" for more information.
>>> ================================ RESTART ================================
>>>
Script completed sucessfully
>>> 
```

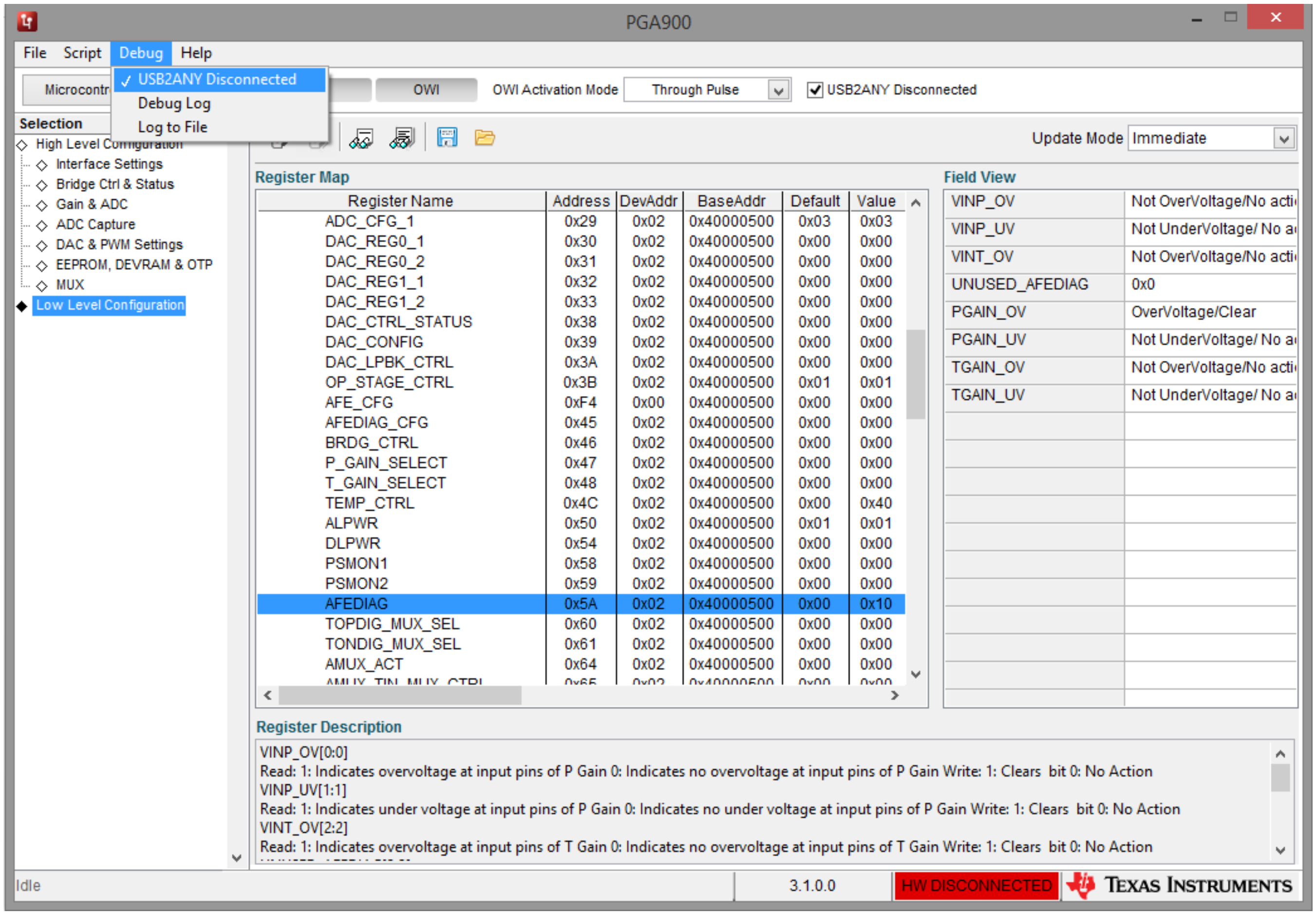
Figure 2-45. Run Saved Macro

- To see the result, refer to the register values in the application. They will be the same as configured by the script. The Read Register operation can also be recorded similarly.

2.6.3 Debug

The Debug option can be used for the following operations.

- USB2ANY Disconnected – By selecting the simulation submenu, the PGA900 GUI is run in simulation mode; by unselecting it, the PGA900 GUI is run in connected mode.
- File Logging – The log to file submenu is used to log the GUI activities to a specified log file.
- Debugging – The Debug log option will enable to log all of the user activities. If debugging is not selected, only the high-level operations are logged.



OWI Activation Mode: Through Pulse

☑ USB2ANY Disconnected

Update Mode: Immediate

Register Map

Register Name	Address	DevAddr	BaseAddr	Default	Value
ADC_CFG_1	0x29	0x02	0x40000500	0x03	0x03
DAC_REG0_1	0x30	0x02	0x40000500	0x00	0x00
DAC_REG0_2	0x31	0x02	0x40000500	0x00	0x00
DAC_REG1_1	0x32	0x02	0x40000500	0x00	0x00
DAC_REG1_2	0x33	0x02	0x40000500	0x00	0x00
DAC_CTRL_STATUS	0x38	0x02	0x40000500	0x00	0x00
DAC_CONFIG	0x39	0x02	0x40000500	0x00	0x00
DAC_LPBK_CTRL	0x3A	0x02	0x40000500	0x00	0x00
OP_STAGE_CTRL	0x3B	0x02	0x40000500	0x01	0x01
AFE_CFG	0xF4	0x00	0x40000500	0x00	0x00
AFEDIAG_CFG	0x45	0x02	0x40000500	0x00	0x00
BRDG_CTRL	0x46	0x02	0x40000500	0x00	0x00
P_GAIN_SELECT	0x47	0x02	0x40000500	0x00	0x00
T_GAIN_SELECT	0x48	0x02	0x40000500	0x00	0x00
TEMP_CTRL	0x4C	0x02	0x40000500	0x00	0x40
ALPWR	0x50	0x02	0x40000500	0x01	0x01
DLPWR	0x54	0x02	0x40000500	0x00	0x00
PSMON1	0x58	0x02	0x40000500	0x00	0x00
PSMON2	0x59	0x02	0x40000500	0x00	0x00
AFEDIAG	0x5A	0x02	0x40000500	0x00	0x10
TOPDIG_MUX_SEL	0x60	0x02	0x40000500	0x00	0x00
TONDIG_MUX_SEL	0x61	0x02	0x40000500	0x00	0x00
AMUX_ACT	0x64	0x02	0x40000500	0x00	0x00
AMUX_TIN_MUX_CTRL	0x65	0x02	0x40000500	0x00	0x00

Field View

VINP_OV	Not OverVoltage/No acti
VINP_UV	Not UnderVoltage/ No a
VINT_OV	Not OverVoltage/No acti
UNUSED_AFEDIAG	0x0
PGAIN_OV	OverVoltage/Clear
PGAIN_UV	Not UnderVoltage/ No a
TGAIN_OV	Not OverVoltage/No acti
TGAIN_UV	Not UnderVoltage/ No a

Register Description

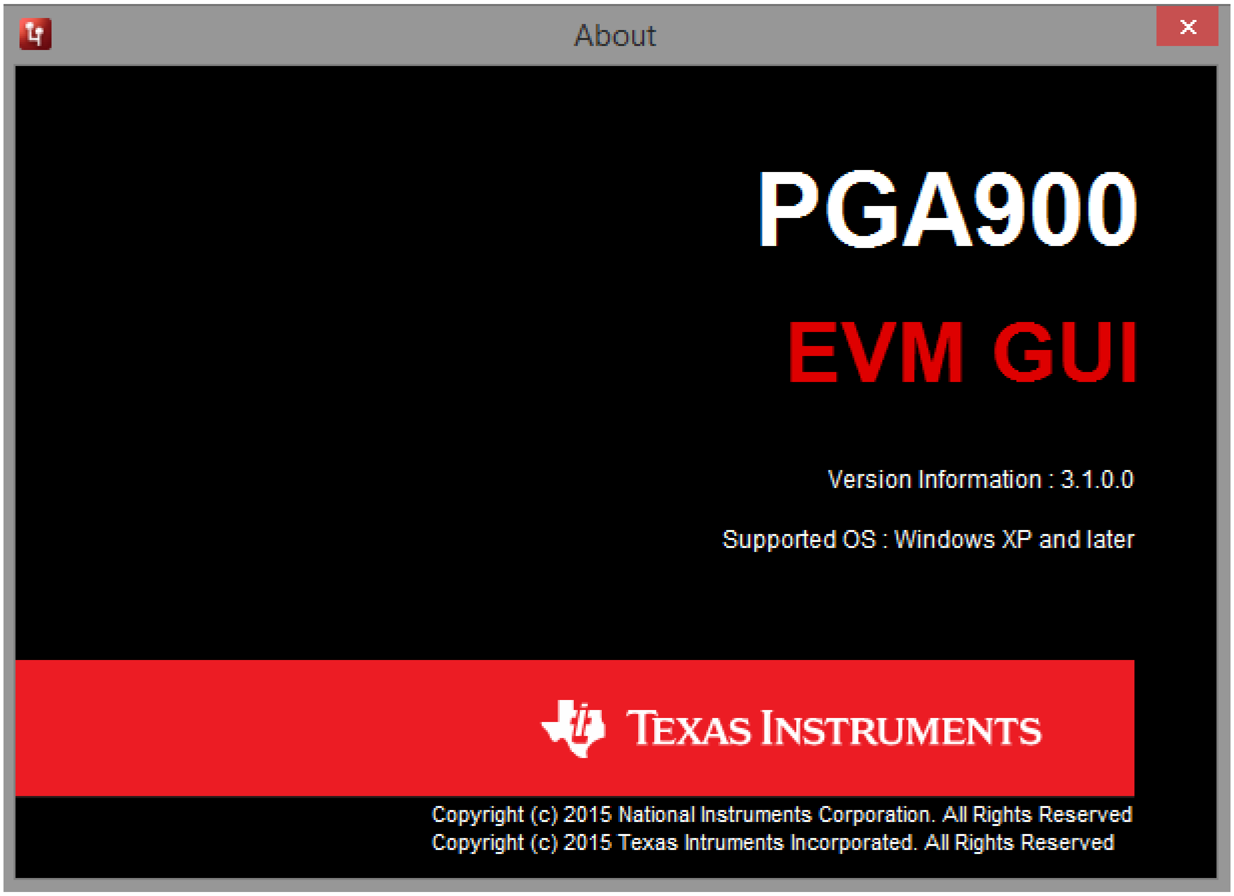
VINP_OV[0:0]
Read: 1: Indicates overvoltage at input pins of P Gain 0: Indicates no overvoltage at input pins of P Gain Write: 1: Clears bit 0: No Action
VINP_UV[1:1]
Read: 1: Indicates under voltage at input pins of P Gain 0: Indicates no under voltage at input pins of P Gain Write: 1: Clears bit 0: No Action
VINT_OV[2:2]
Read: 1: Indicates overvoltage at input pins of T Gain 0: Indicates no overvoltage at input pins of T Gain Write: 1: Clears bit 0: No Action

Figure 2-46. Debug Menu

2.6.4 Help

2.6.4.1 About

- The About dialog can be accessed from Help Menu → About.
- The about dialog contains information such as GUI name, version information, and supported OS.
- It also contains copyright information.



About

PGA900

EVM GUI

Version Information : 3.1.0.0

Supported OS : Windows XP and later

Copyright (c) 2015 National Instruments Corporation. All Rights Reserved
Copyright (c) 2015 Texas Intruments Incorporated. All Rights Reserved

Figure 2-47. About

Revision History

Changes from Original (May 2015) to A Revision	Page
• Updated screenshot for Figure 2-44	52

NOTE: Page numbers for previous revisions may differ from page numbers in the current version.

IMPORTANT NOTICE

Texas Instruments Incorporated and its subsidiaries (TI) reserve the right to make corrections, enhancements, improvements and other changes to its semiconductor products and services per JESD46, latest issue, and to discontinue any product or service per JESD48, latest issue. Buyers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. All semiconductor products (also referred to herein as “components”) are sold subject to TI’s terms and conditions of sale supplied at the time of order acknowledgment.

TI warrants performance of its components to the specifications applicable at the time of sale, in accordance with the warranty in TI’s terms and conditions of sale of semiconductor products. Testing and other quality control techniques are used to the extent TI deems necessary to support this warranty. Except where mandated by applicable law, testing of all parameters of each component is not necessarily performed.

TI assumes no liability for applications assistance or the design of Buyers’ products. Buyers are responsible for their products and applications using TI components. To minimize the risks associated with Buyers’ products and applications, Buyers should provide adequate design and operating safeguards.

TI does not warrant or represent that any license, either express or implied, is granted under any patent right, copyright, mask work right, or other intellectual property right relating to any combination, machine, or process in which TI components or services are used. Information published by TI regarding third-party products or services does not constitute a license to use such products or services or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

Reproduction of significant portions of TI information in TI data books or data sheets is permissible only if reproduction is without alteration and is accompanied by all associated warranties, conditions, limitations, and notices. TI is not responsible or liable for such altered documentation. Information of third parties may be subject to additional restrictions.

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

Buyer acknowledges and agrees that it is solely responsible for compliance with all legal, regulatory and safety-related requirements concerning its products, and any use of TI components in its applications, notwithstanding any applications-related information or support that may be provided by TI. Buyer represents and agrees that it has all the necessary expertise to create and implement safeguards which anticipate dangerous consequences of failures, monitor failures and their consequences, lessen the likelihood of failures that might cause harm and take appropriate remedial actions. Buyer will fully indemnify TI and its representatives against any damages arising out of the use of any TI components in safety-critical applications.

In some cases, TI components may be promoted specifically to facilitate safety-related applications. With such components, TI’s goal is to help enable customers to design and create their own end-product solutions that meet applicable functional safety standards and requirements. Nonetheless, such components are subject to these terms.

No TI components are authorized for use in FDA Class III (or similar life-critical medical equipment) unless authorized officers of the parties have executed a special agreement specifically governing such use.

Only those TI components which TI has specifically designated as military grade or “enhanced plastic” are designed and intended for use in military/aerospace applications or environments. Buyer acknowledges and agrees that any military or aerospace use of TI components which have ***not*** been so designated is solely at the Buyer's risk, and that Buyer is solely responsible for compliance with all legal and regulatory requirements in connection with such use.

TI has specifically designated certain components as meeting ISO/TS16949 requirements, mainly for automotive use. In any case of use of non-designated products, TI will not be responsible for any failure to meet ISO/TS16949.

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

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