PGA970 GUI User's Guide

User's Guide

Literature Number: SLDU023
November 2016

Contents

GUI Software Installation

The PGA970 GUI allows the users to communicate to the PGA970 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 or higher
- Recommended RAM memory – 4GB or higher
- Recommended CPU Operating Speed – 3.3 GHz or higher

1.2 Installation Procedure

The following procedure describes the steps involved in the PGA970 GUI installation.

1. Double click on the Setup_PGA970_EVM.exe from the installer folder as shown in Figure 1-1.

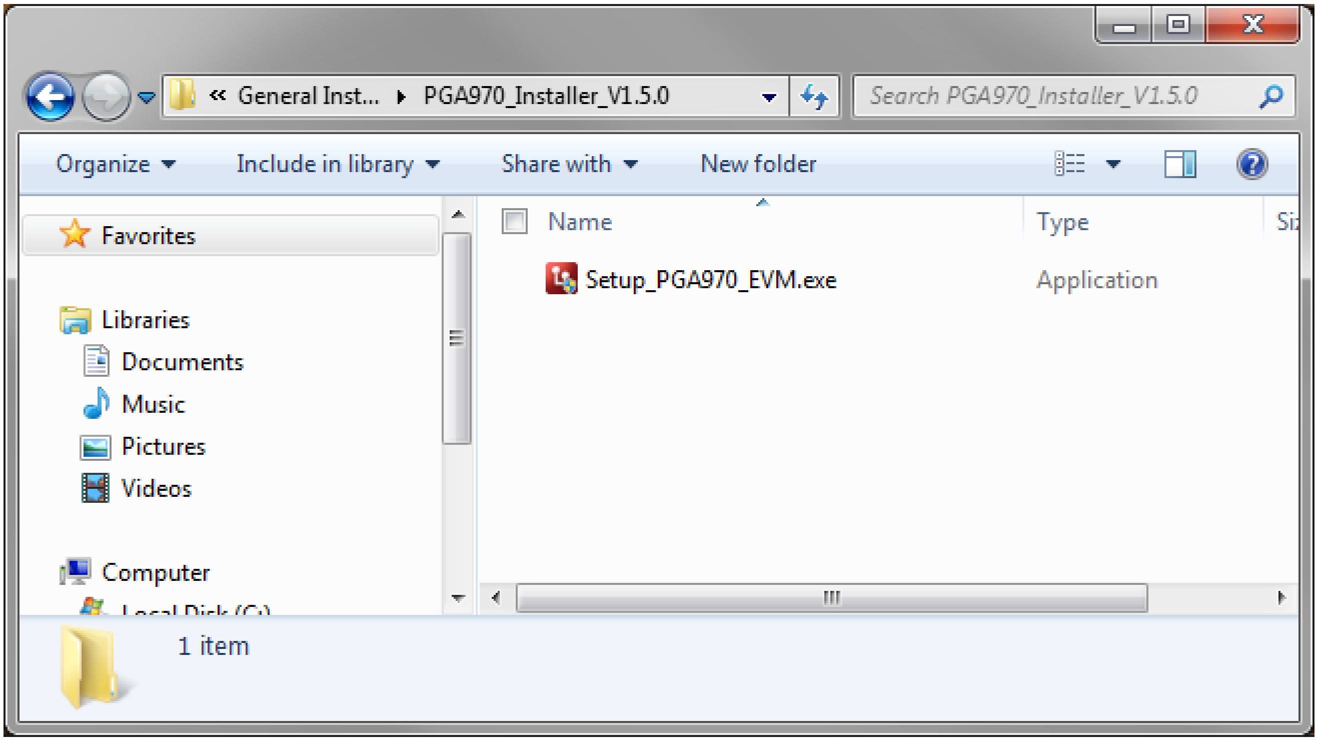
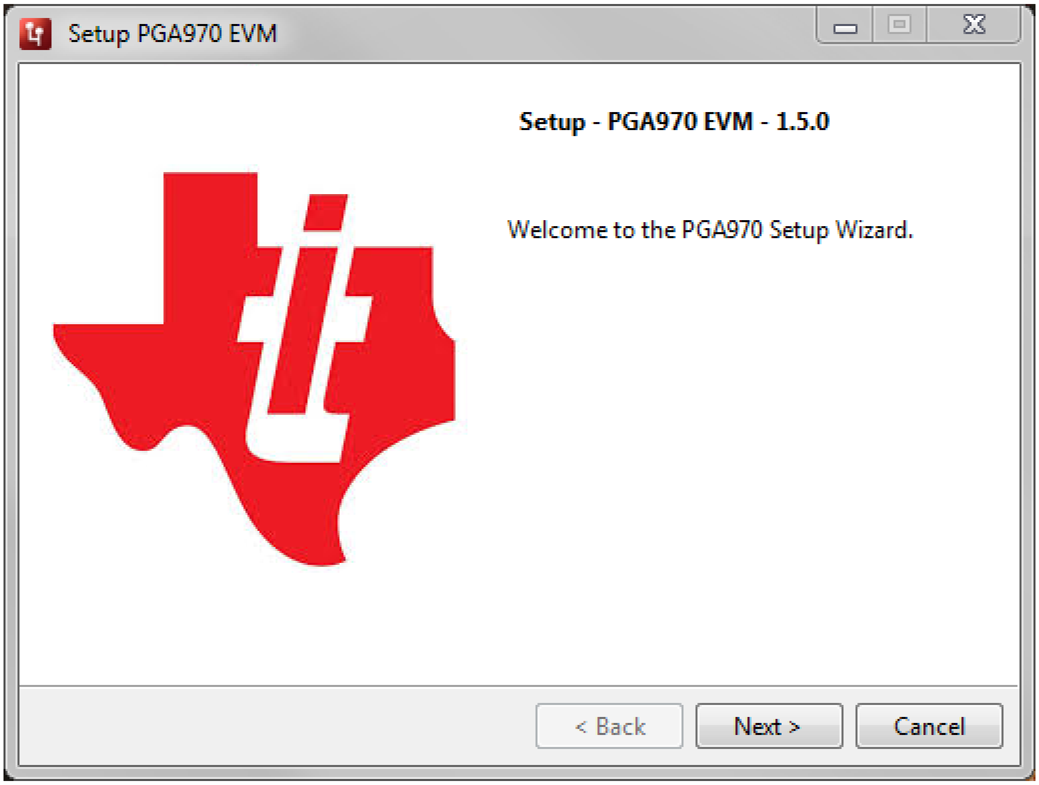


Figure 1-1. Setup_PGA970_EVM.exe

A screen shown as in Figure 1-2 appears.

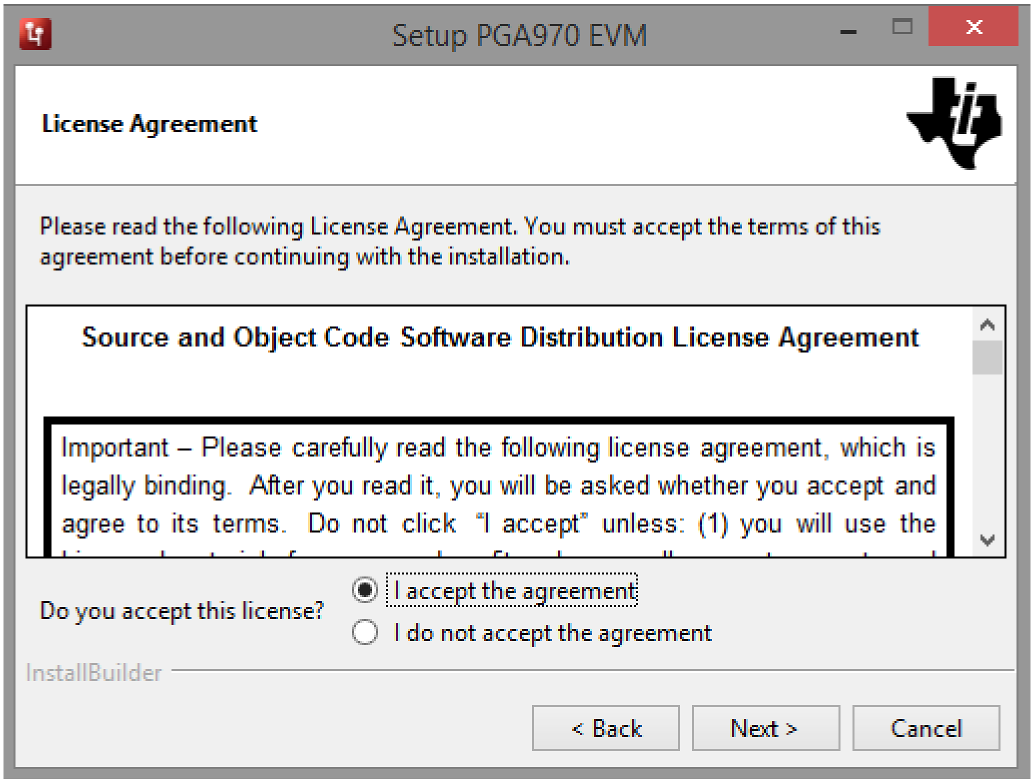


Setup - PGA970 EVM - 1.5.0

Welcome to the PGA970 Setup Wizard.

Figure 1-2. Installation Initialization

2. The License Agreement for PGA970 GUI appears as shown in Figure 1-3. Read through the agreement carefully, enable the “I accept the agreement” option 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.

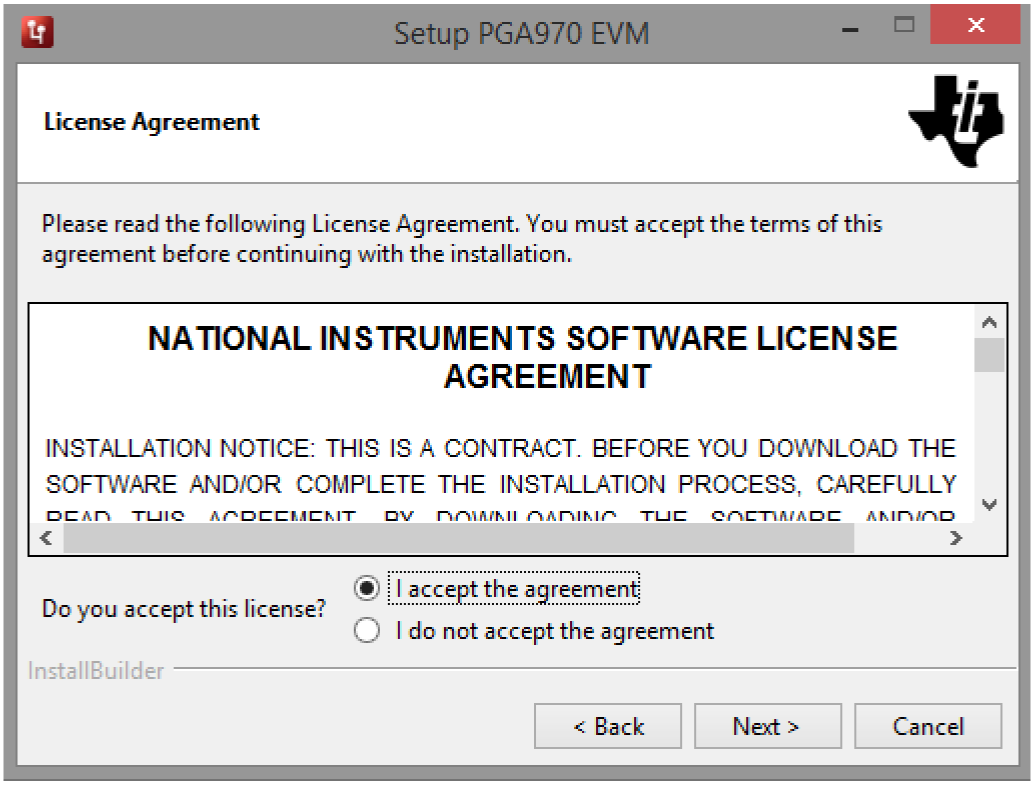
Source and Object Code Software Distribution License Agreement

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. License Agreement – GUI

The License Agreement for National Instruments appears as shown in Figure 1-4. Read through the agreement carefully 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.

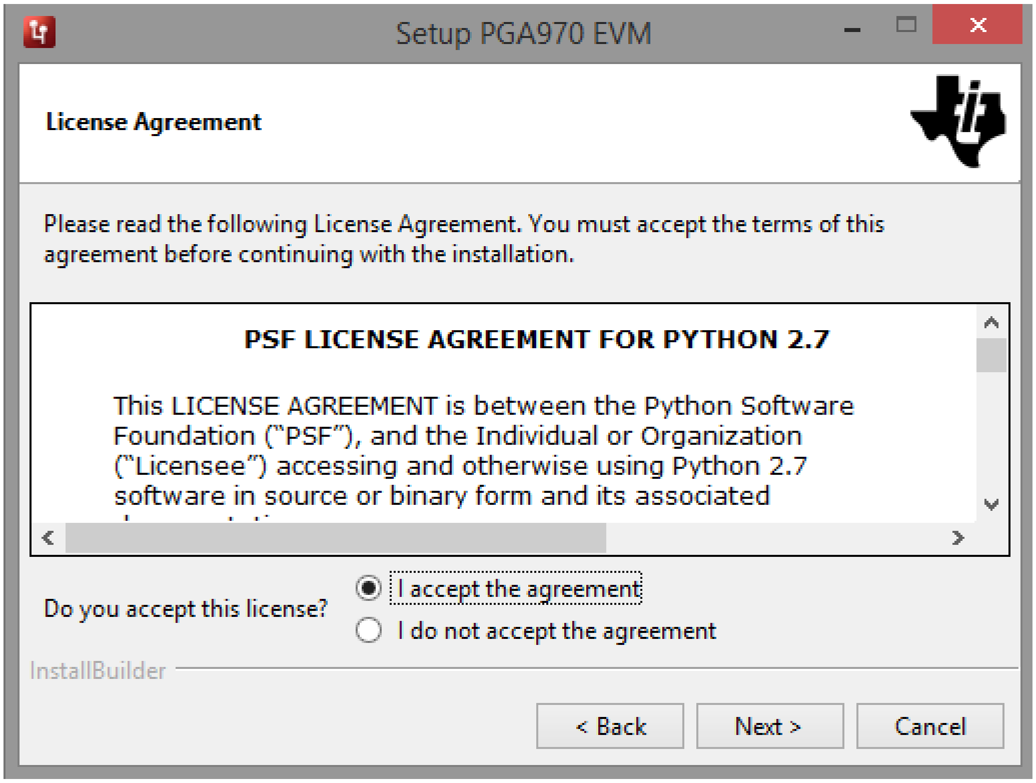
NATIONAL INSTRUMENTS SOFTWARE LICENSE AGREEMENT

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 - NI

The License Agreement for Python 2.7 appears as shown in Figure 1-5. Read through the agreement carefully and enable the “I accept the agreement” option and click the Next» button.



Setup PGA970 EVM

License Agreement

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

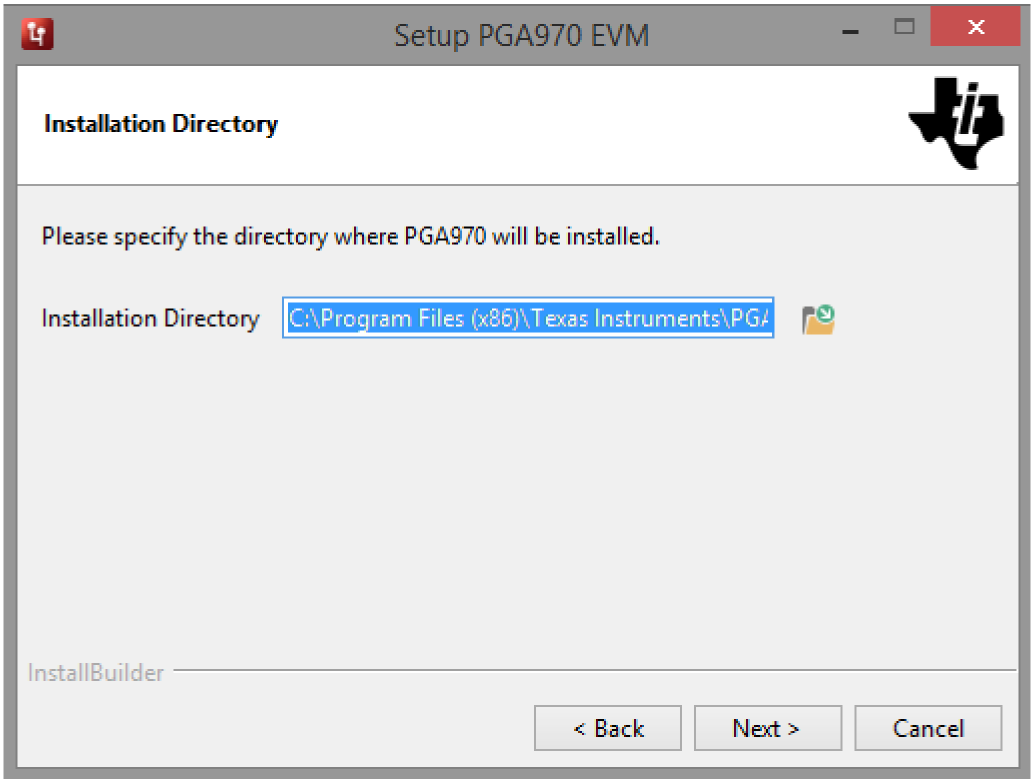
This LICENSE AGREEMENT is between the Python Software Foundation ("PSF"), and the Individual or Organization ("Licensee") accessing and otherwise using Python 2.7 software in source or binary form and its associated

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

Figure 1-5. License Agreement - Python

NOTE: It is highly recommended to keep the default values as provided in the installer.

3. Set the destination directories for the PGA970 GUI installation and click the Next» button as shown in Figure 1-6.

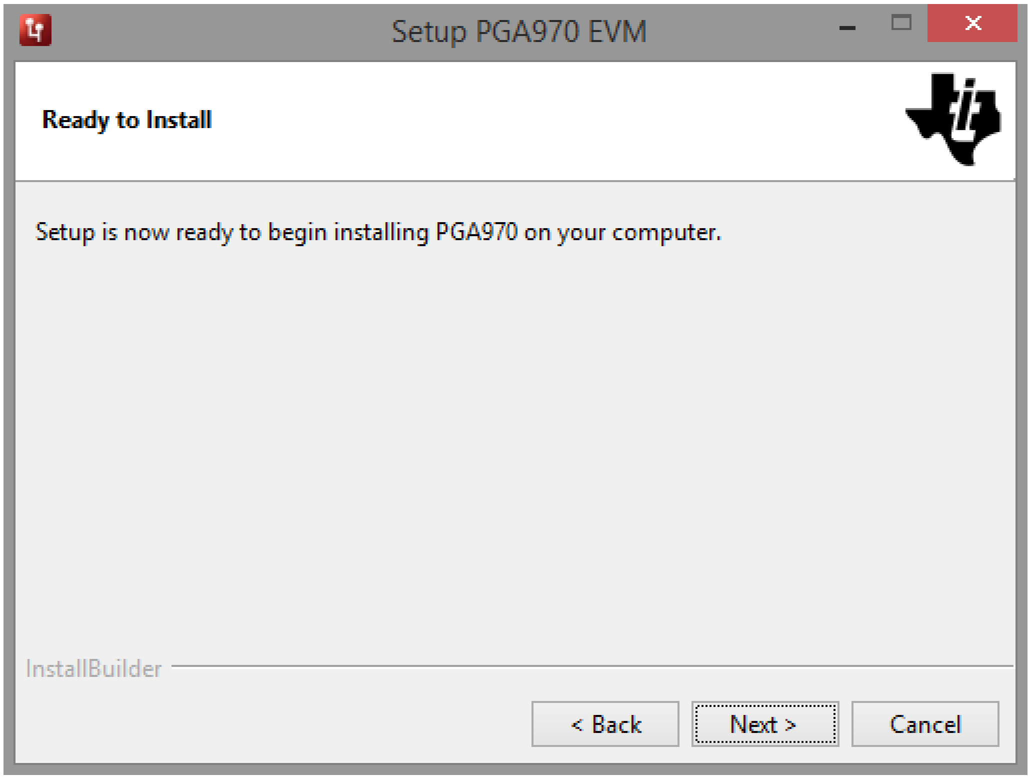


Please specify the directory where PGA970 will be installed.

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

Figure 1-6. Destination Directory

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

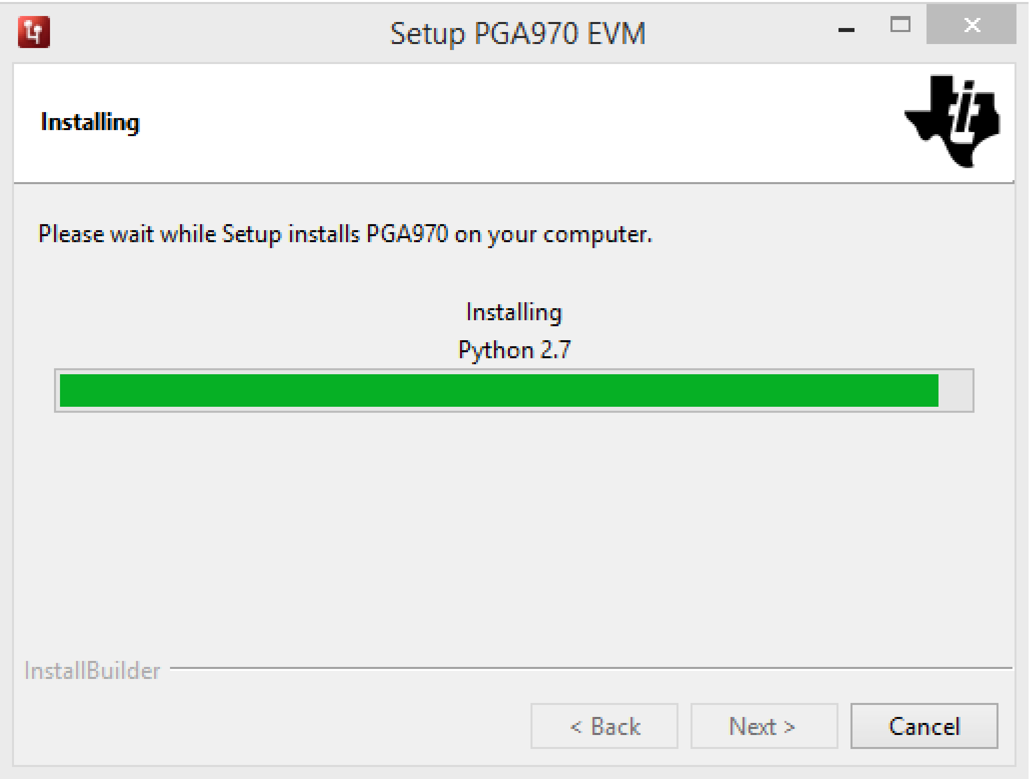


Ready to Install

Setup is now ready to begin installing PGA970 on your computer.

Figure 1-7. Start Installation

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



Setup PGA970 EVM

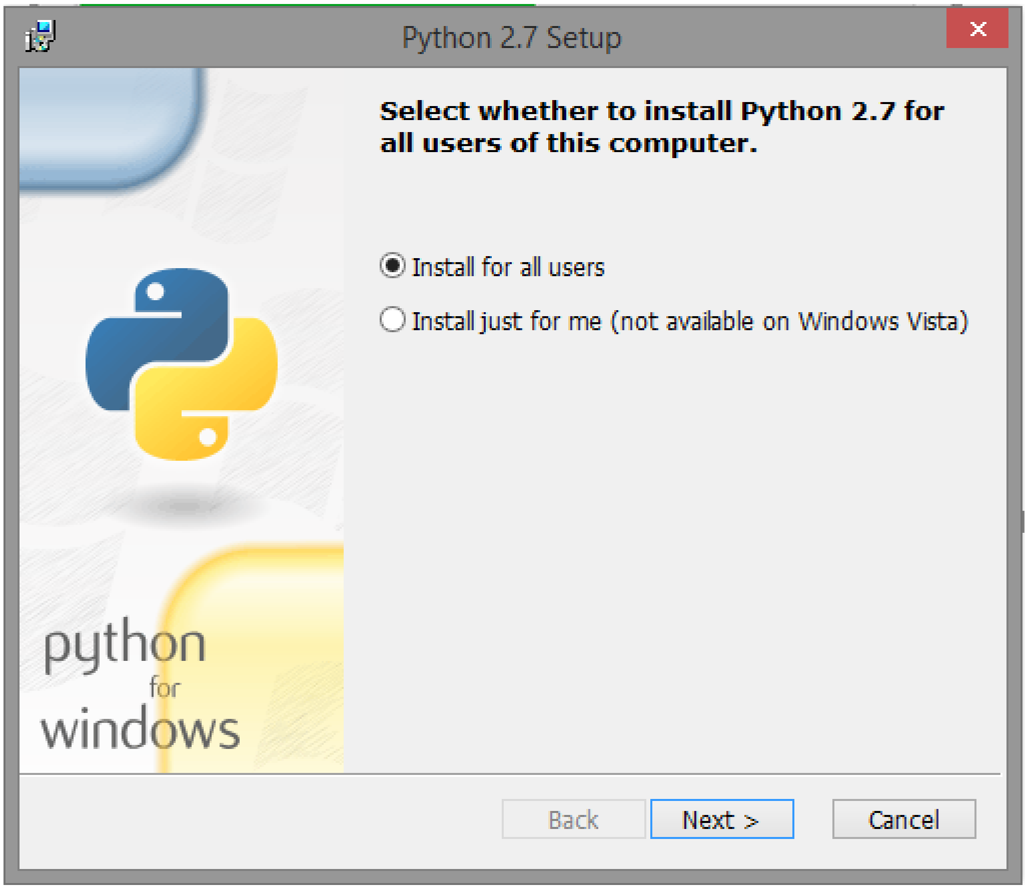
Installing

Please wait while Setup installs PGA970 on your computer.

Installing
Python 2.7

Figure 1-8. Installation in Progress

6. Python installation starts towards the end of PGA970 GUI installation. Select the required option and click the Next» button.



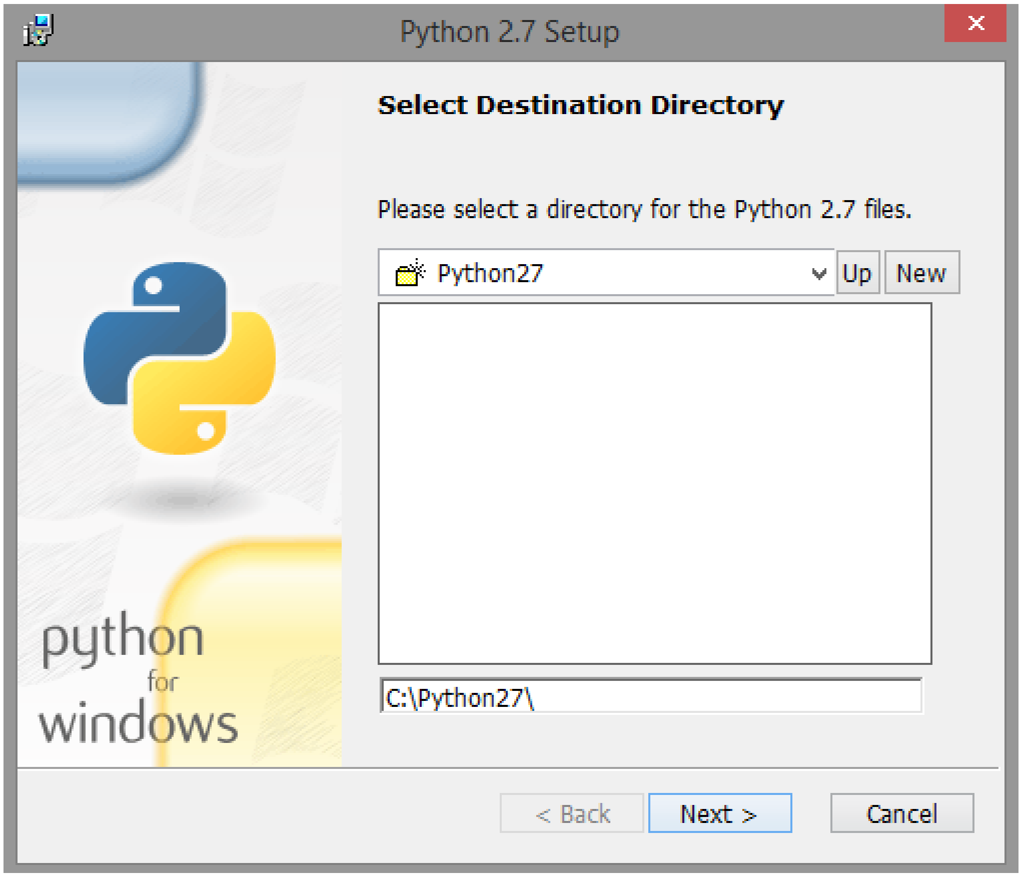
Select whether to install Python 2.7 for all users of this computer.

● 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 the Next» button.



Select Destination Directory

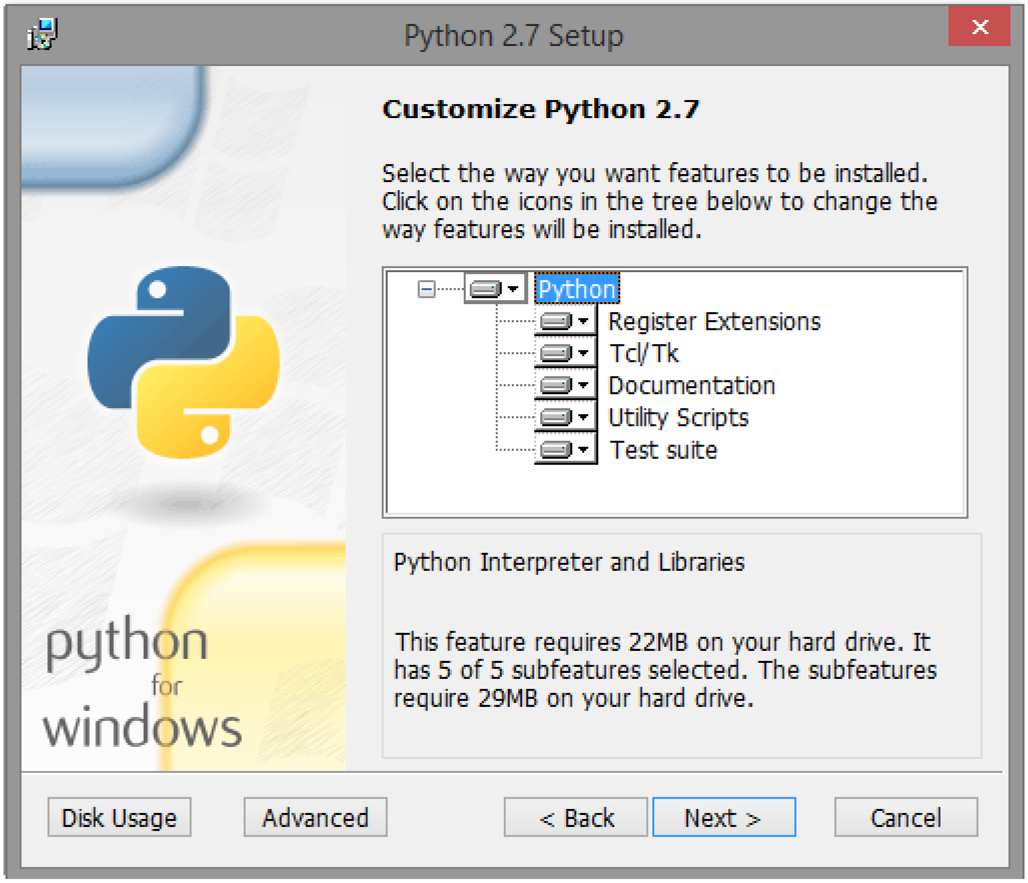
Please select a directory for the Python 2.7 files.

Python27

C:\Python27\

Figure 1-10. Python Installation Directory

8. A dialog as shown in Figure 1-11 appears. Click the Next» button to proceed with the installation.



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.

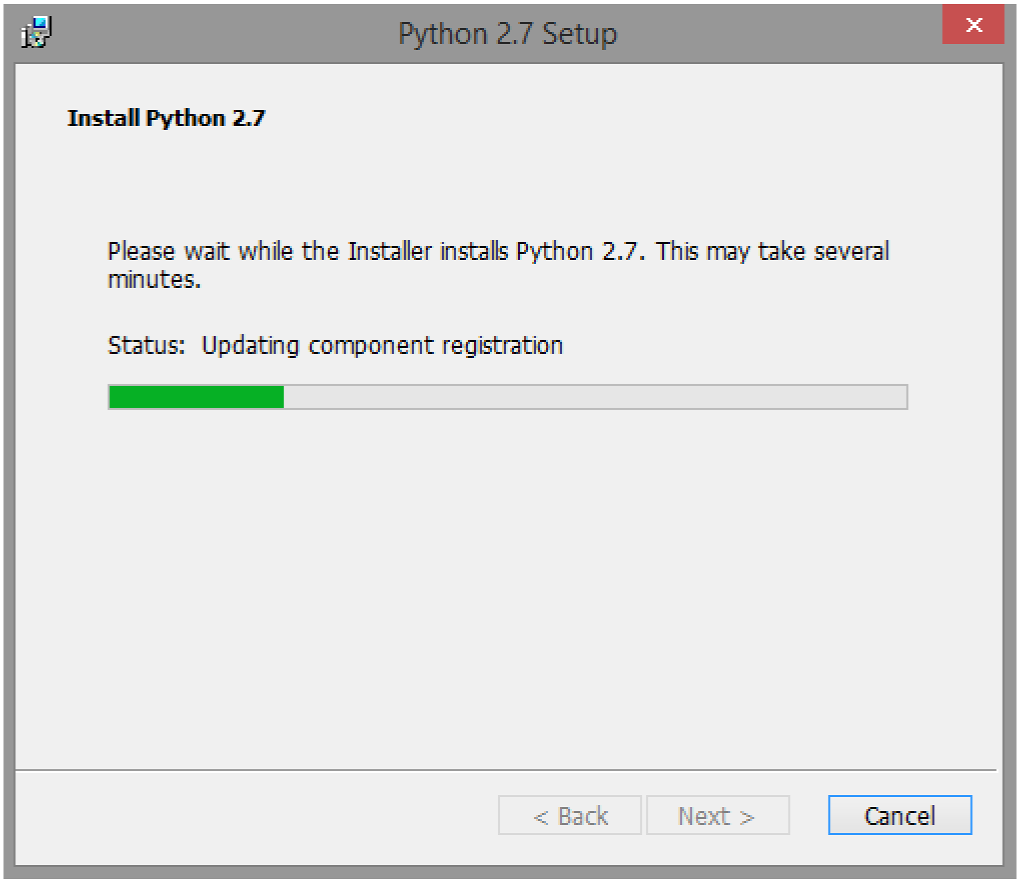
- Python
  - Register Extensions
  - Tcl/Tk
  - Documentation
  - Utility Scripts
  - Test suite

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 updated as shown in Figure 1-12.



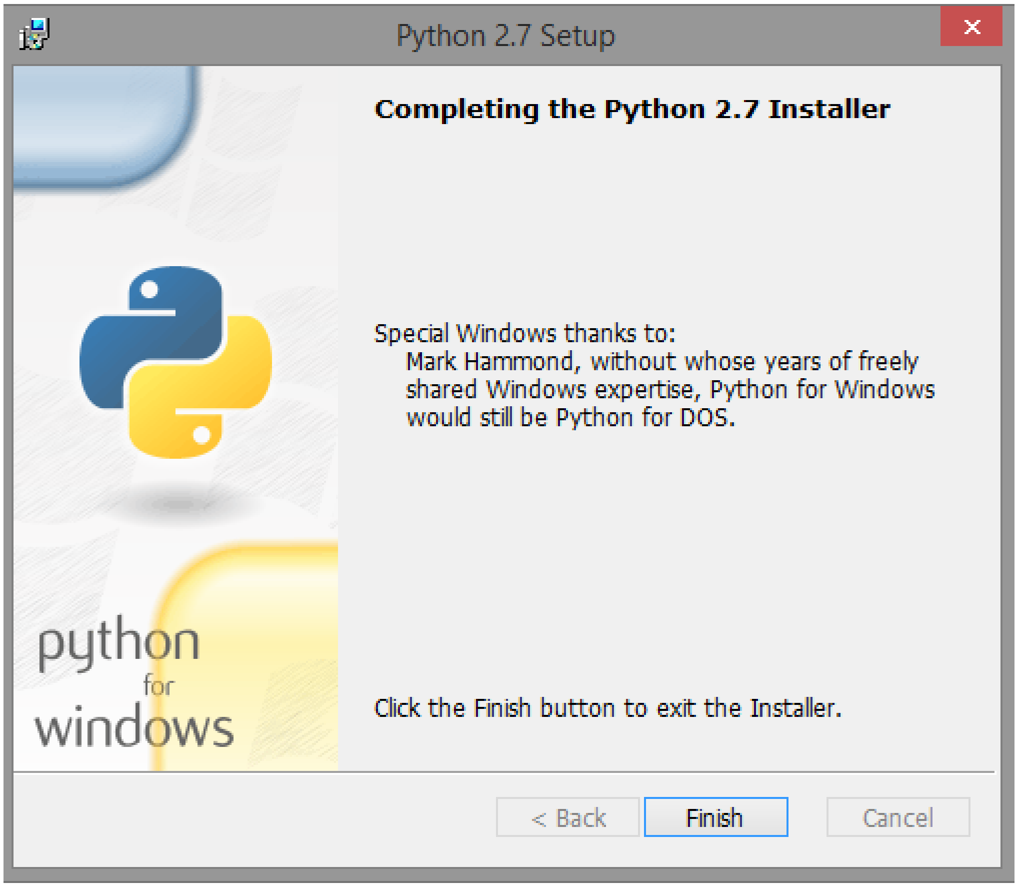
Install Python 2.7

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 the completion of Python installation, click the Finish button.



Python 2.7 Setup

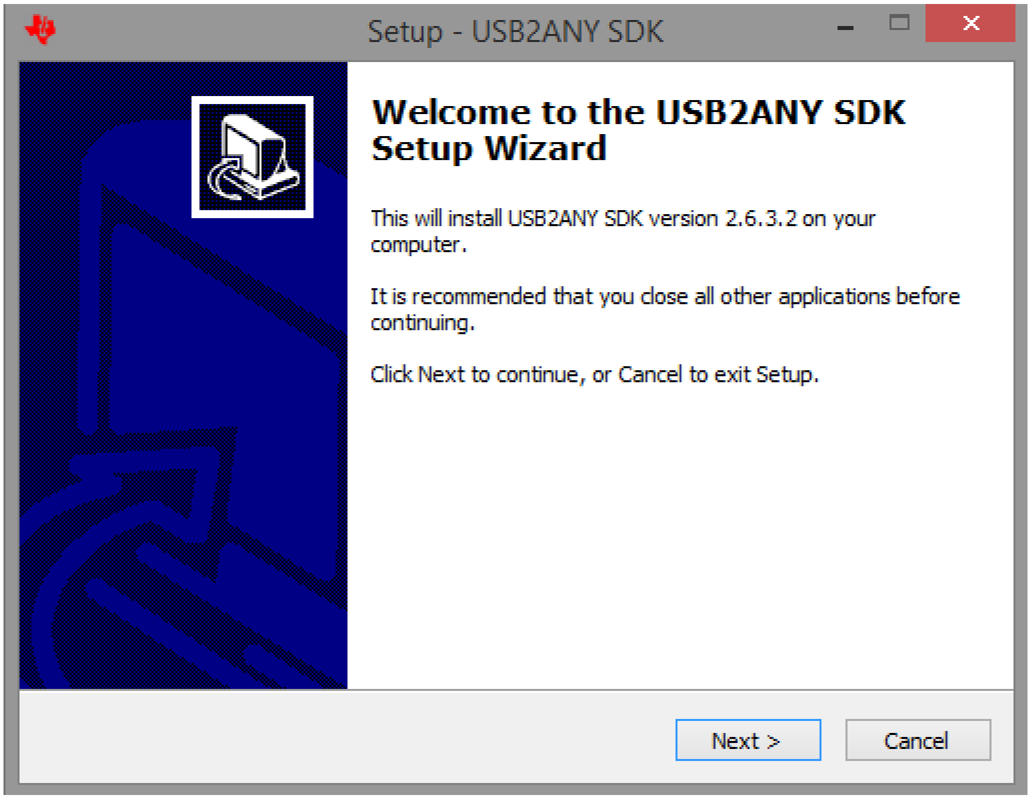
Completing the Python 2.7 Installer

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 the completion of Python installation. From the dialog shown in Figure 1-14, click the Next» button.



Welcome to the USB2ANY SDK Setup Wizard

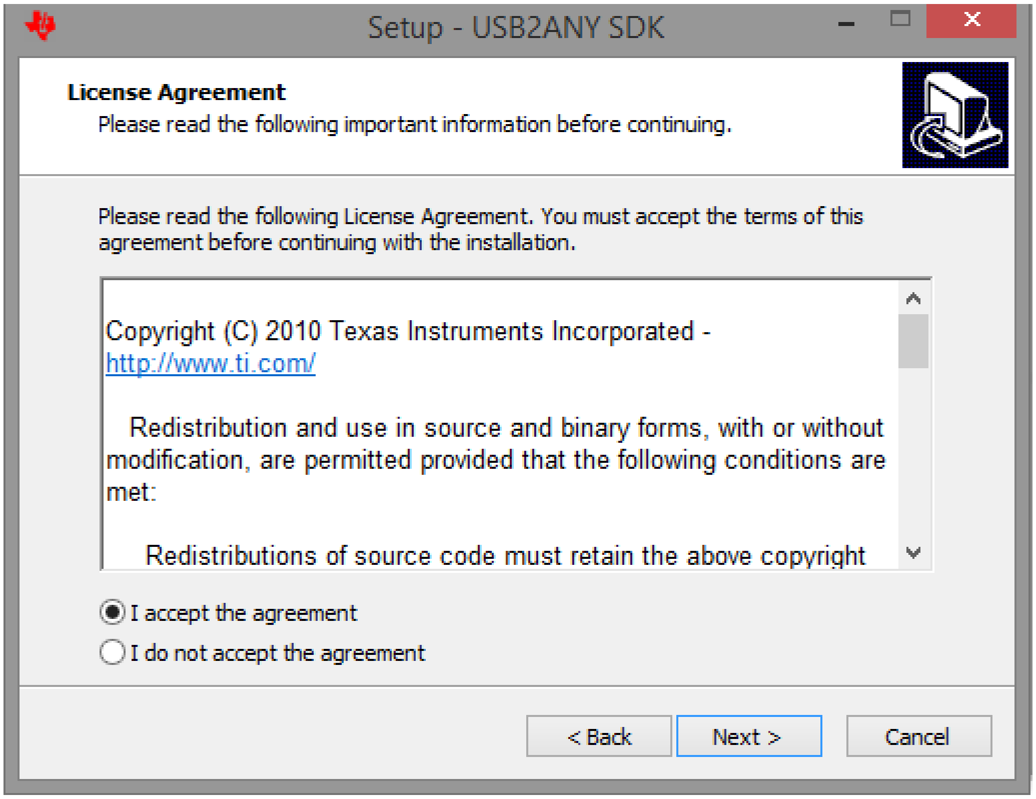
This will install USB2ANY SDK version 2.6.3.2 on your computer.

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 carefully and enable the “I accept the agreement” option 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:

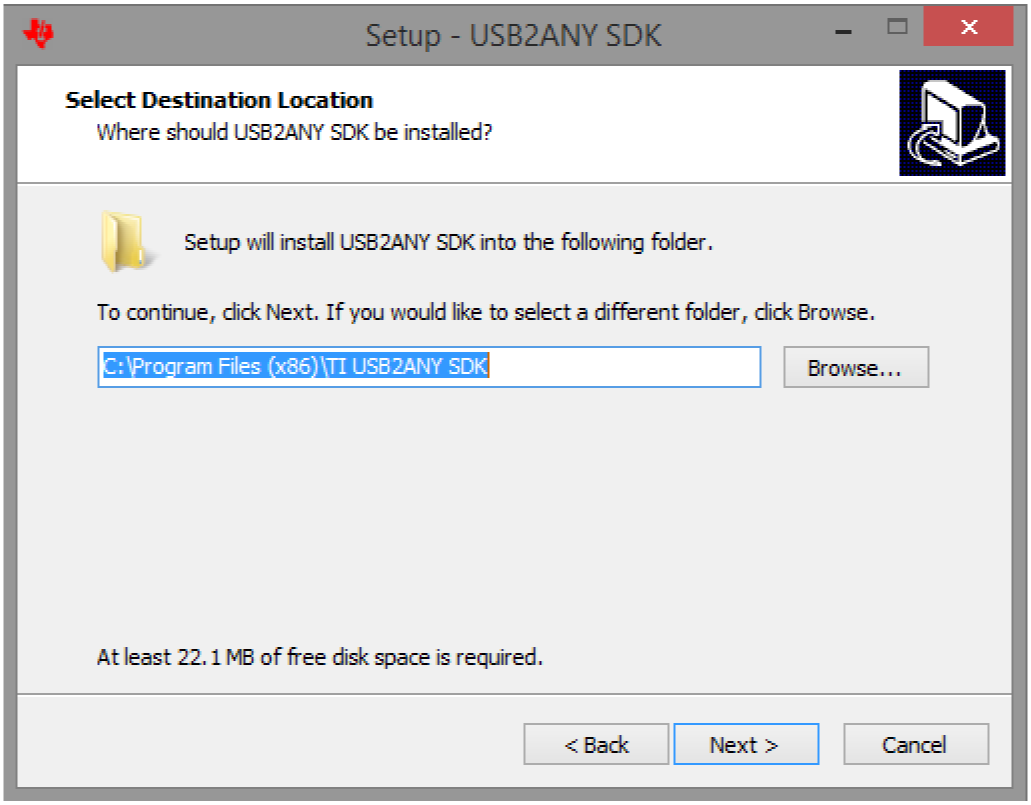
Redistributions of source code must retain the above copyright

● I accept the agreement

○ I do not accept the agreement

Figure 1-15. USB2ANY License Agreement

13. Set the destination directories for the USB2ANY 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.

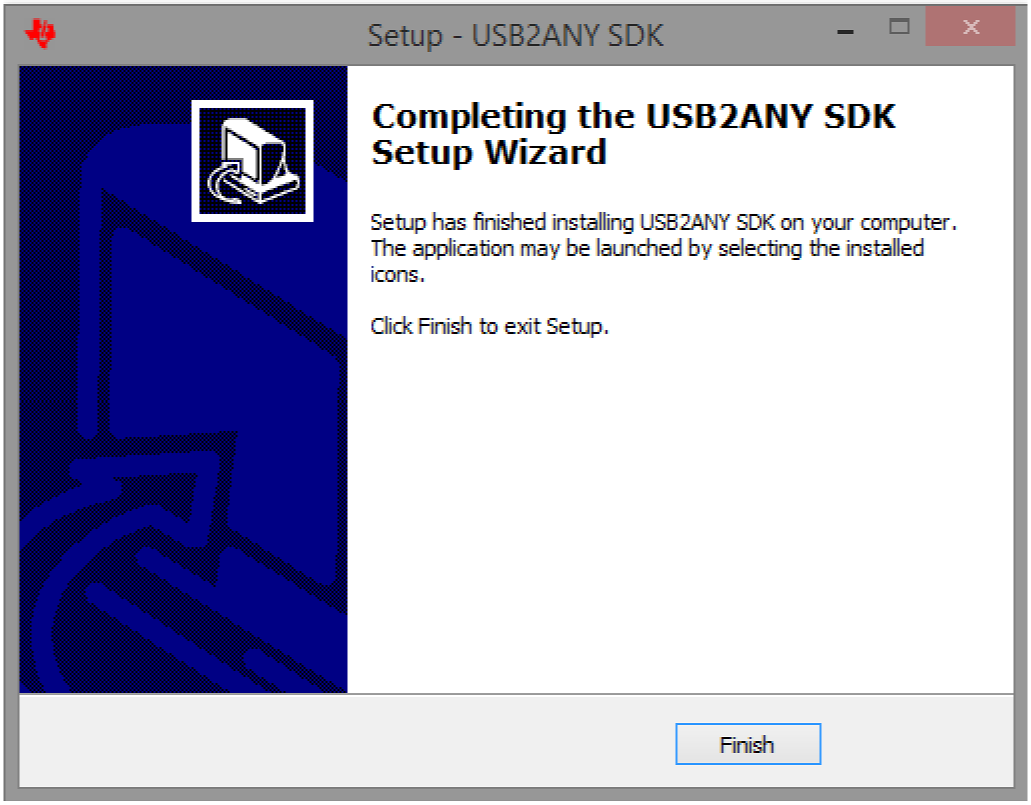
To continue, click Next. If you would like to select a different folder, click Browse.

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 completed click the Finish button.



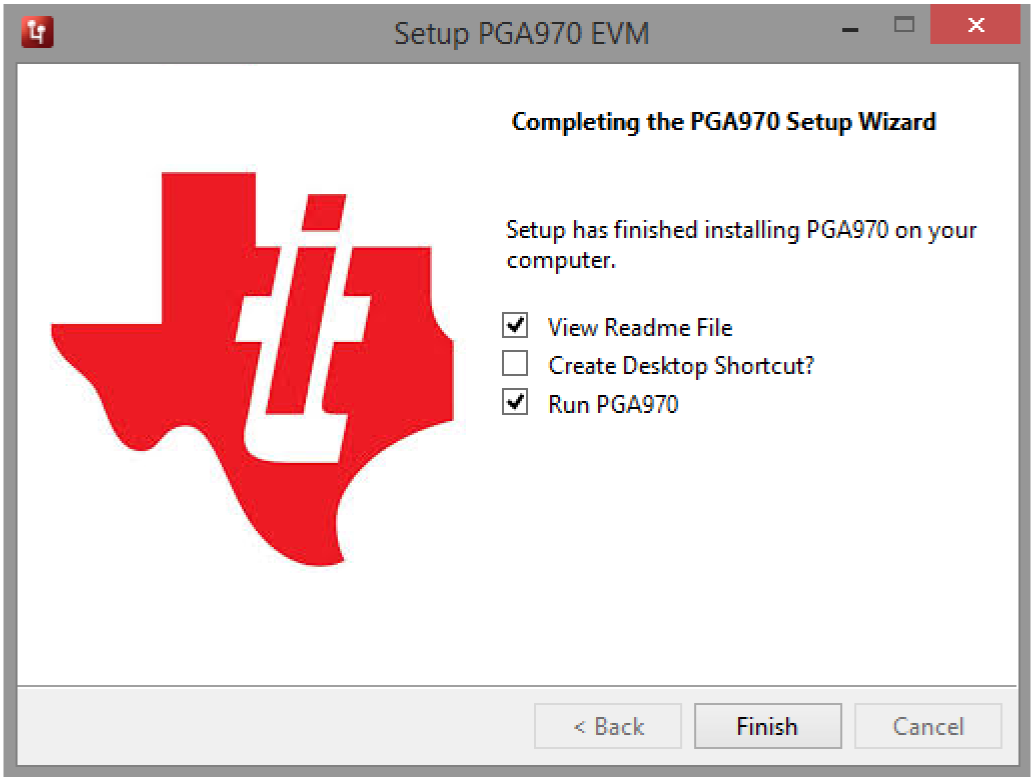
Completing the USB2ANY SDK Setup Wizard

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 as shown in Figure 1-18 appears and denotes the end of the PGA970 GUI installation.



Setup PGA970 EVM

Completing the PGA970 Setup Wizard

Setup has finished installing PGA970 on your computer.

☑ View Readme File
□ Create Desktop Shortcut?
☑ Run PGA970

Figure 1-18. Installation Complete

CAUTION

The PGA970 GUI requires the following software to be installed before the GUI is executed.

- National Instruments LabVIEW Run-Time Engine 2012: www.ni.com/download/labview-run-time-engine-2012/3433/en/

NOTE: The PGA970 GUI executable has been built in LabVIEW 2012 (32-Bit) version and it expects the LabVIEW Run-Time Engine version to be LabVIEW Run-Time Engine 2012 (32-Bit) Version.

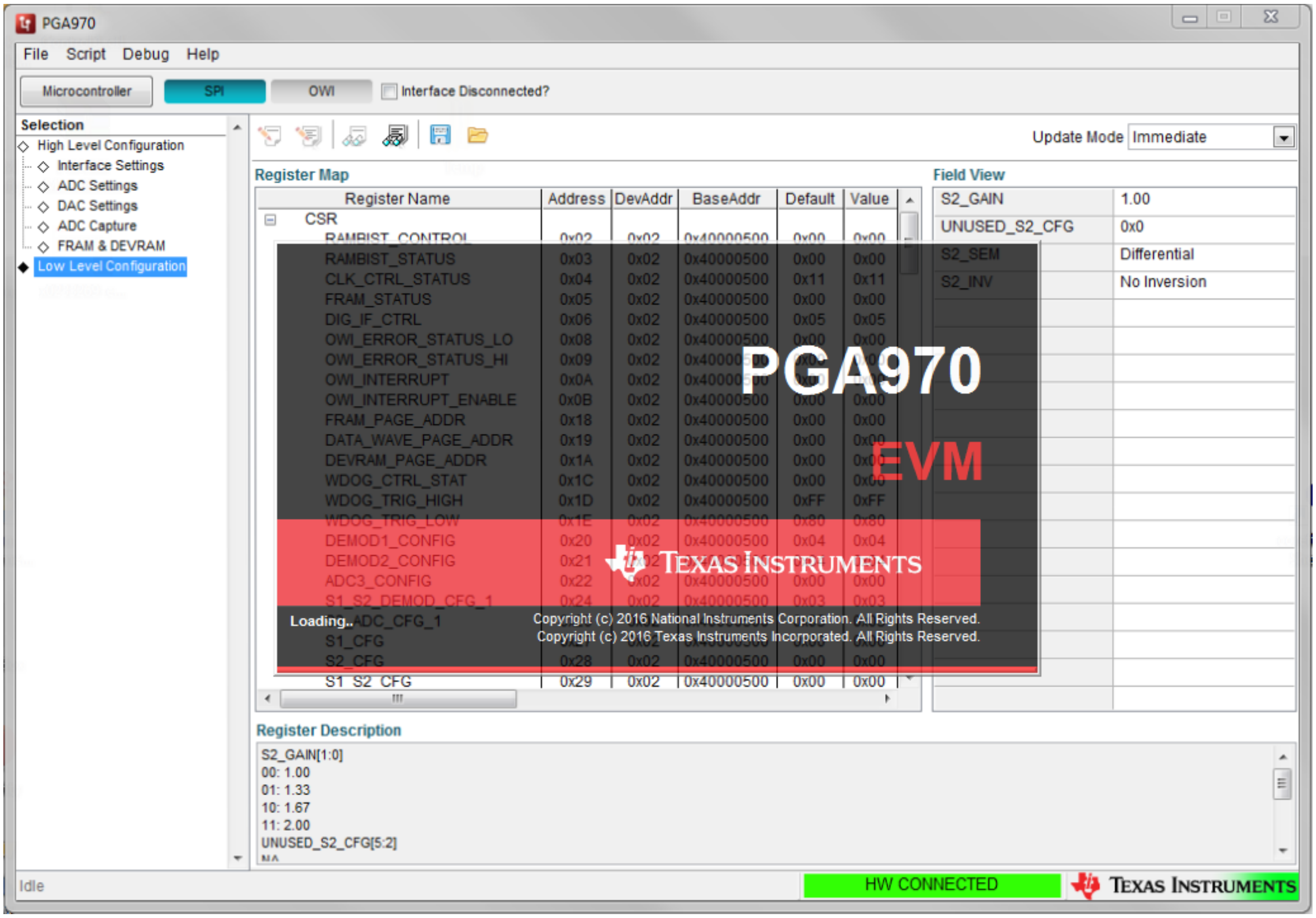
PGA970 – User Interface

This section gives a detailed description of the features of the PGA970 GUI.

The PGA970 GUI is an intuitive UI, for the PGA970 device and EVM that allows the user to read and configure the registers of the PGA970 device and control some EVM components.

NOTE: It is advised to launch the PGA970 GUI with administrator privileges.

When the PGA970 GUI is launched, the following screen pops up indicating that the GUI is initializing.



Register Map

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	0x11	0x11
FRAM_STATUS	0x05	0x02	0x40000500	0x00	0x00
DIG_IF_CTRL	0x06	0x02	0x40000500	0x05	0x05
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
FRAM_PAGE_ADDR	0x18	0x02	0x40000500	0x00	0x00
DATA_WAVE_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_HIGH	0x1D	0x02	0x40000500	0xFF	0xFF
WDOG_TRIG_LOW	0x1E	0x02	0x40000500	0x80	0x80
DEMOD1_CONFIG	0x20	0x02	0x40000500	0x04	0x04
DEMOD2_CONFIG	0x21	[illegible]	[illegible]	[illegible]	[illegible]
ADC3_CONFIG	0x22	0x02	0x40000500	0x00	0x00
S1_S2_DEMOD_CFG_1	0x24	0x02	0x40000500	0x03	0x03
ADC_CFG_1	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
S1_CFG	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
S2_CFG	0x28	0x02	0x40000500	0x00	0x00
S1_S2_CFG	0x29	0x02	0x40000500	0x00	0x00

Field View

S2_GAIN	1.00
UNUSED_S2_CFG	0x0
S2_SEM	Differential
S2_INV	No Inversion

Register Description

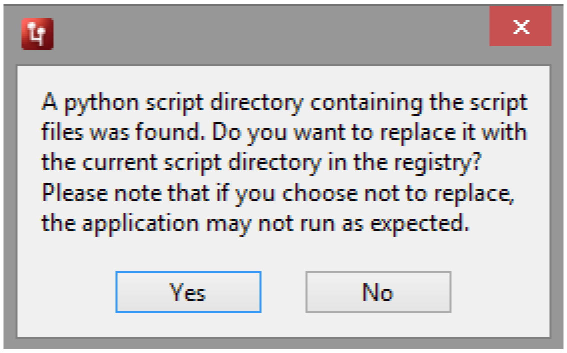
S2_GAIN[1:0]
00: 1.00
01: 1.33
10: 1.67
11: 2.00
UNUSED_S2_CFG[5:2]
NA

Copyright (c) 2016 National Instruments Corporation. All Rights Reserved.
Copyright (c) 2016 Texas Instruments Incorporated. All Rights Reserved.

Figure 2-1. GUI Initializing

When the PGA970 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 IDLE IDE is called.

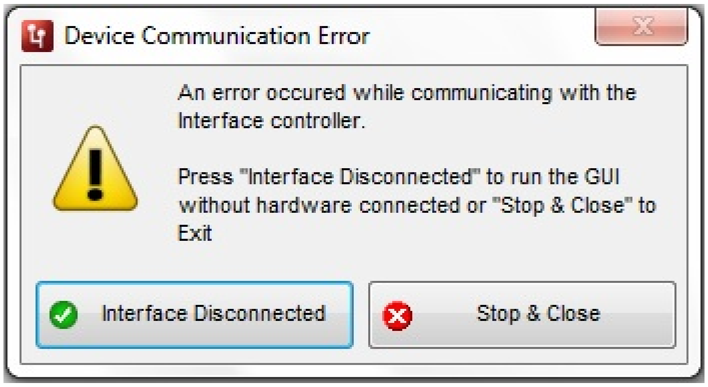


A python script directory containing the script files was found. Do you want to replace it with the current script directory in the registry? Please note that if you choose not to replace, the application may not run as expected.

Figure 2-2. Update Registry

2. After the PGA970 GUI is launched, the user can invoke it in two different ways:
   - Interface Disconnected
   - Interface Connected

   Interface Disconnected mode is invoked when the USB2ANY is not connected. When the PGA970 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 PGA970 GUI in Interface Disconnected mode or to terminate the usage.



Device Communication Error

An error occured while communicating with the Interface controller.

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

Figure 2-3. Device Communication Error

When the user continues in Interface Disconnected mode, the checkbox at the top of the GUI is set. This shows that the PGA970 GUI is running in USB2ANY Disconnected mode. When the GUI is in Interface Disconnected mode the UI controls may appear to function as if the device is connected and will read simulated data.

2.1 PGA970 GUI Overview

The PGA970 GUI consists of the following pages:

- High Level Configuration Pages
  - Interface Settings
  - ADC Settings
  - ADC Capture
  - DAC Settings
  - FRAM & DEVRAM
- Low Level Configuration Page

Interface Settings

- UART speed settings for OWI
- Rloop resistance configuration
- Additional voltage configuration

ADC Settings

- The page contains the settings for the ADCs available in PGA970 device.
- The table at the bottom of the page is used to read the ADC register values and display the corresponding amplitudes in volts.
- The data could also be dumped into a file by selecting the corresponding ADC.

ADC Capture

- The ADC Graph allows the user to configure the ADC settings for Capture.
- It has options to perform continuous ADC capture with the configured settings
- This Page also allows the user to export the graph data to a text file or save the graph plot as an image. Right click the graph for the export options.

DAC Settings

- User can read DAC data and ADC or waveform loopback data. DAC Gain can also be configured here.
- This page has Loopback option that will read back the waveform generated or the DAC values through the ADC available on the device.

FRAM & DEVRAM

- The PGA970 has FRAM and DEVRAM blocks along with the register memory and these blocks can be accessed by using the FRAM & DEVRAM page of the PGA970 GUI.
- These memories can be programmed by using HEX files.
- This page also enables the user to read the RAM memory and save it into a file or verify the RAM contents against a hex file to find if the programmed data and the file data are same.

Low Level Configuration Page

- The Low Level Configuration Page lists down all the registers that are present in the PGA970 GUI.
- This page can be used to write to or read from, the register fields of the PGA970 device.

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 copy-write information of the PGA970 GUI.

The PGA970 resources could be accessed by the Digital interface or by the Microcontroller. The user can change how the PGA970 resources are accessed by selecting the microcontroller button at the top left corner of the PGA970 GUI as shown in Figure 2-4.

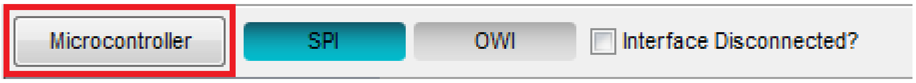


Figure 2-4. Microcontroller Selection

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 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 the pages except “Low Level Configuration” page and “Interface Settings” page are disabled when the Digital Interface is disabled.)

2.2 PGA970 Communication Interfaces

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

- Serial Peripheral Interface (SPI)
- One-Wire Interface (OWI)

Each communication mode has its own protocol; however, both access the same memory elements within the device. For all communication modes, the PGA970 device operates as a slave device.

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

During SPI Selection

- The Digital Interface is enabled by writing 3 to MICRO_INTERFACE_CONTROL
- GPIO_OWI_TX is set LOW
- GPIO_OWI_TX is set LOW
- Disables the OWI_XCVR bit in DIG_IF_CTRL register

For OWI Activation

- Digital Interface is disabled by writing 0 to MICRO_INTERFACE_CONTROL
- GPIO_OWI_TX is set to LOW
- GPIO_OWI_ACT is set to HIGH
- Wait for 10 ms *(This configuration is hardcoded in the GUI software)*
- GPIO_OWI_ACT is set LOW
- GPIO_OWI_TX is set HIGH

2.2.1 Overview of SPI Interface

- 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 PGA970 always acts as the slave in SPI communication, where, whatever external device that is

communicating with it becomes the master. Both devices begin data transmission with the most significant bit (MSB) first.

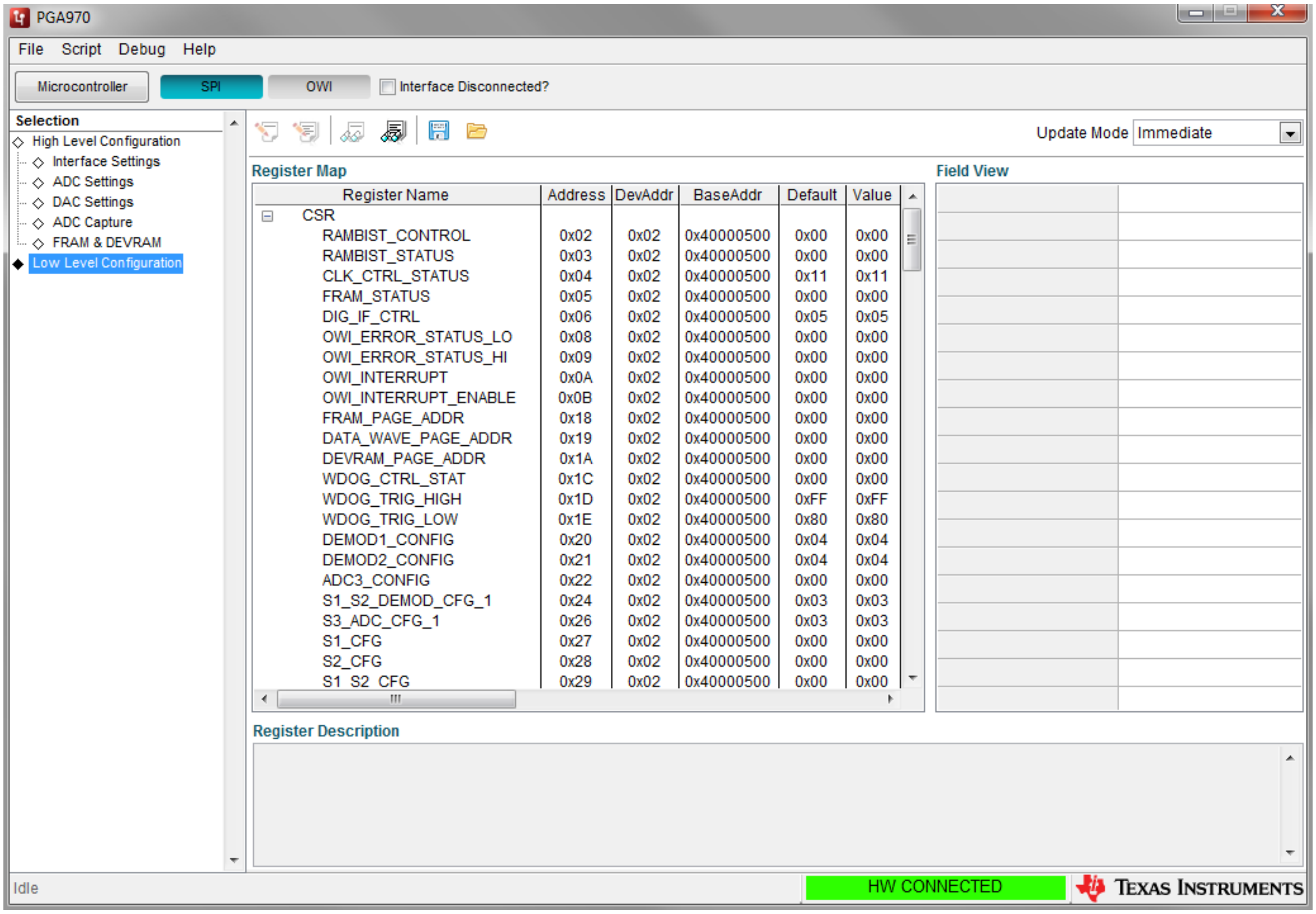
- Because multiple slave devices can exist on one bus, the master node is able to notify the specific slave node that it is ready to begin communicating with, by driving the CSN line to a low logic level.
- In the absence of active transmission, the master SPI device places the device in reset by driving the CSN pin to a high logic level.
- During the reset state the MISO pin operates in the tri-state mode.

2.2.2 Overview of OWI Interface

- The OWI digital communication is a master-slave communication link in which the PGA970 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 PGA970 is used as OWI interface, so that when PGA970 is embedded inside a system module, only two pins are needed (VDD and GND) for communication.
- The OWI master communicates with PGA970 by modulating the voltage on the VDD pin while PGA970 communicates with the master by modulating current on VDD pin.
- **OWI Write**
  - No specific configuration is needed to write using the OWI.
  - Write the data in the format defined by the data sheet.
- **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.
  - Check for data in the buffer until it receives it.
  - Read the data from the buffer and display it. This will read all data available in the buffer. Ideally there should be only one byte of data.

2.3 Page Selection

2.3.1 Low Level Configuration Page



☐ Interface Disconnected?

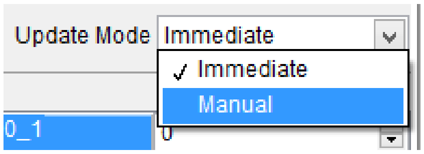
Update Mode: Immediate

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	0x11	0x11
FRAM_STATUS	0x05	0x02	0x40000500	0x00	0x00
DIG_IF_CTRL	0x06	0x02	0x40000500	0x05	0x05
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
FRAM_PAGE_ADDR	0x18	0x02	0x40000500	0x00	0x00
DATA_WAVE_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_HIGH	0x1D	0x02	0x40000500	0xFF	0xFF
WDOG_TRIG_LOW	0x1E	0x02	0x40000500	0x80	0x80
DEMOD1_CONFIG	0x20	0x02	0x40000500	0x04	0x04
DEMOD2_CONFIG	0x21	0x02	0x40000500	0x04	0x04
ADC3_CONFIG	0x22	0x02	0x40000500	0x00	0x00
S1_S2_DEMOD_CFG_1	0x24	0x02	0x40000500	0x03	0x03
S3_ADC_CFG_1	0x26	0x02	0x40000500	0x03	0x03
S1_CFG	0x27	0x02	0x40000500	0x00	0x00
S2_CFG	0x28	0x02	0x40000500	0x00	0x00
S1_S2_CFG	0x29	0x02	0x40000500	0x00	0x00

Figure 2-5. 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 or write to the registers.
- When a particular register is selected, the corresponding register description is displayed at the bottom of the page.

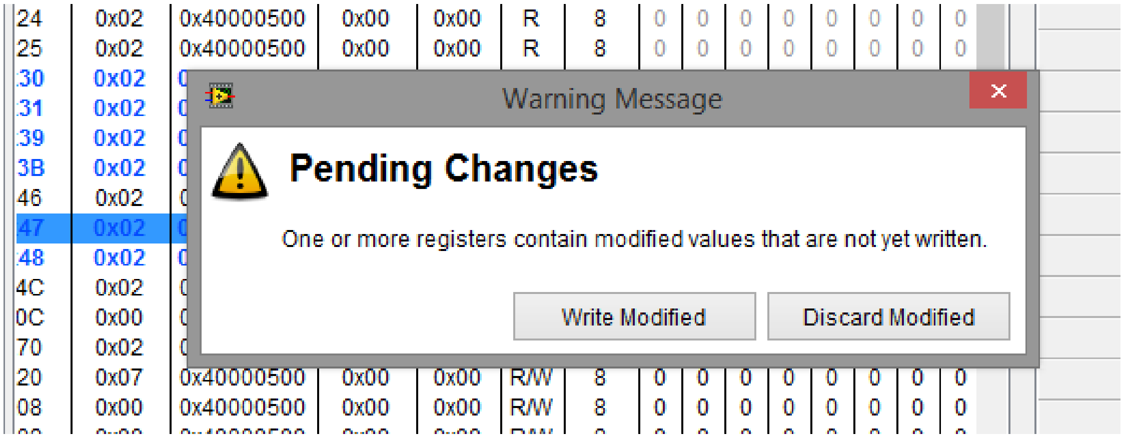
- Register write modes



Update Mode Immediate

Figure 2-6. Update Mode

- **Immediate mode -** The register values will be written to device immediately when a register value or a field value is changed.
- **Manual mode -** The register values will be written to the device only when Write Register or Write Modified button is pressed. The changed register values will be highlighted in blue color which means that these changes are not yet written into the device. When a highlighted register is written to the device, the register will again turn into normal state (black). If there are some pending changes and 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.



Warning Message

Pending Changes

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

24	0x02	0x40000500	0x00	0x00	R	8	0	0	0	0	0	0	0	0
25	0x02	0x40000500	0x00	0x00	R	8	0	0	0	0	0	0	0	0
30	0x02													
31	0x02													
39	0x02													
3B	0x02													
46	0x02													
47	0x02													
48	0x02													
4C	0x02													
0C	0x00													
70	0x02													
20	0x07	0x40000500	0x00	0x00	R/W	8	0	0	0	0	0	0	0	0
08	0x00	0x40000500	0x00	0x00	R/W	8	0	0	0	0	0	0	0	0

Figure 2-7. Pending Changes Dialog

2.3.1.1 Load and Save Register Configuration Feature

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

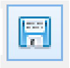


Figure 2-8. Save Config

- **Load Config** - Click this button to load the configuration file which was saved previously to bring the device into a known state. Ensure that the Microcontroller is reset before the configuration is loaded.

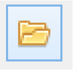


Figure 2-9. Load Config

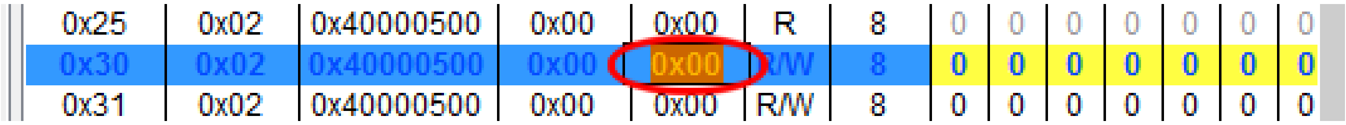
- When the user selects any of the above options, a message pops up on the screen. Select the file path (to load /save the configuration file) and click OK

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

2.3.1.2 Register Read and Write

The user can change the value of a register by:

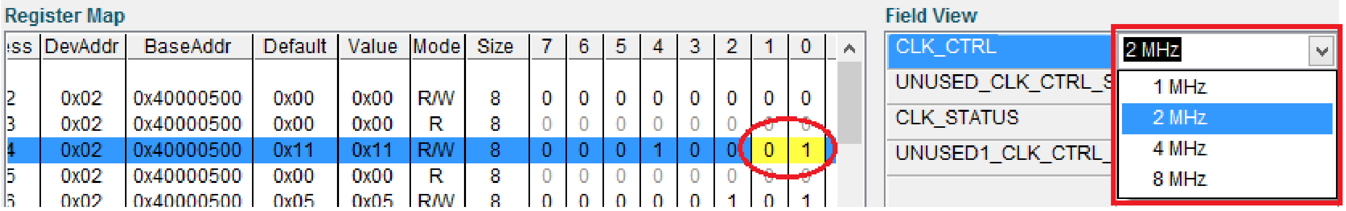
- **Register level operation**
  - Select the register to edit
  - Double click on the value column corresponding to the register
  - Enter the register value (Hex) in the edit box



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 2-10. 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 the user hovers the mouse over the value of a field, a dropdown list or a numeric control appears and the corresponding bits get highlighted. The user can change the appropriate value in the dropdown list or numeric control.



Register Map

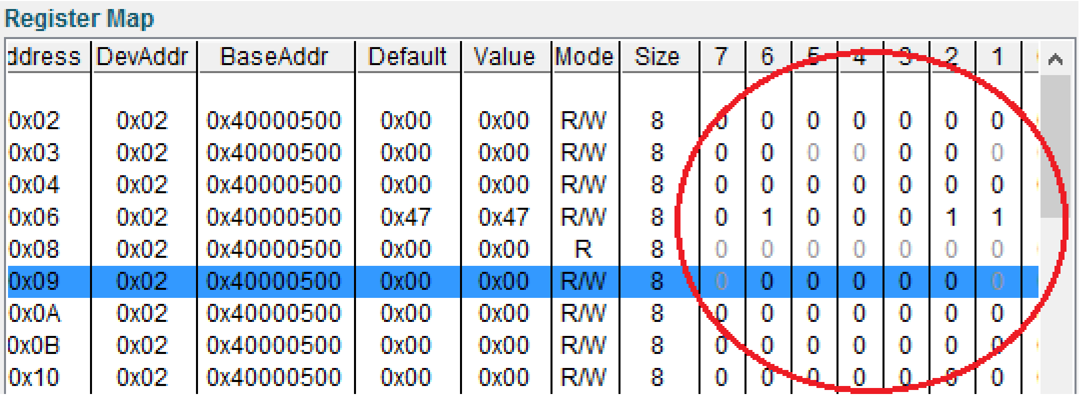
ss	DevAddr	BaseAddr	Default	Value	Mode	Size	7	6	5	4	3	2	1	0
2	0x02	0x40000500	0x00	0x00	R/W	8	0	0	0	0	0	0	0	0
3	0x02	0x40000500	0x00	0x00	R	8	0	0	0	0	0	0	0	0
4	0x02	0x40000500	0x11	0x11	R/W	8	0	0	0	1	0	0	0	1
5	0x02	0x40000500	0x00	0x00	R	8	0	0	0	0	0	0	0	0
6	0x02	0x40000500	0x05	0x05	R/W	8	0	0	0	0	0	1	0	1

Field View

CLK_CTRL	2 MHz
UNUSED_CLK_CTRL_S	
CLK_STATUS	
UNUSED1_CLK_CTRL_	

Figure 2-11. Field Level Value Change

- **Bit level operation**
  - Select the register that has to be edited.
  - 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 2-12. 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 field 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 the user presses this button, the data is read from all the registers based on the mode (Read and Read/Write mode).
- **Write Modified** - When the user presses this button all the pending registers that are not written to the device gets written into the device. This option gets enabled only when update mode is Manual.

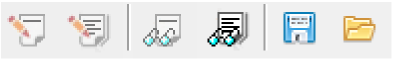


Figure 2-13. Low Level Page Operations

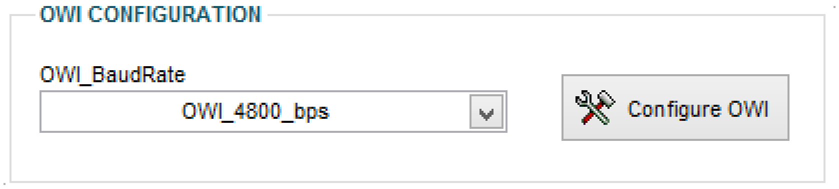
2.3.2 High Level Configuration Page

- 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 sections below explain a different function of the PGA970 EVM.

2.3.2.1 Interface Settings

- The Interface Settings page contains the configuration parameters necessary for the communication protocols.
- The page includes options for configuring OWI communication.
- The default values are displayed when the GUI is loaded.

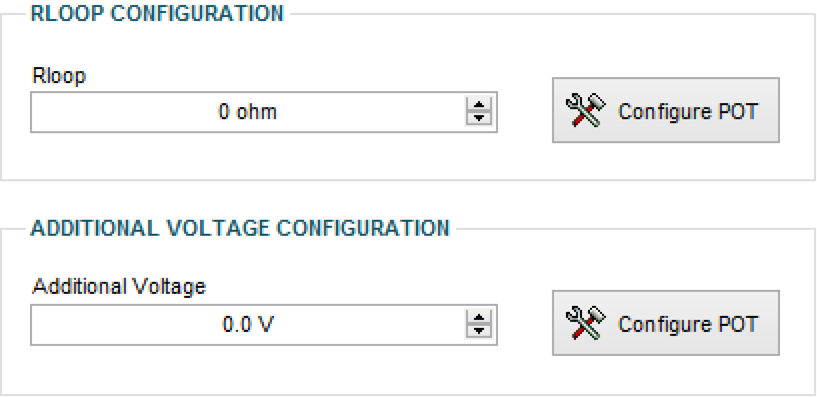


OWI CONFIGURATION

OWI_BaudRate: OWI_4800_bps

Figure 2-14. Interface Configurations

- The user has the provisions to change the Baud rate for the UART protocol.
- The default baud rate for UART it is 4800 bps.
- After the change is made, the Configure button has to be clicked for the changes to take effect.
- The controls Rloop and Additional Voltage are used to configure the Potentiometers on the PGA970 EVM board. These Potentiometers are configured using I2C protocol. The additional voltage configuration is needed when constant voltage drops (such as diodes) are present in the loop. For the PGA970EVM, the additional voltage should be set to 0.0 V.



Rloop: 0 ohm

Additional Voltage: 0.0 V

Figure 2-15. POT Configuration

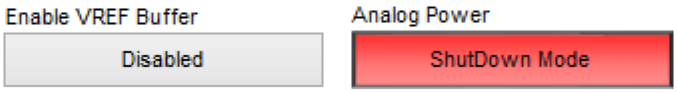
2.3.2.1.1 OWI Configuration

UART configuration involves 3 settings,

1. Setting the baud rate to the selected configuration
2. Setting the receiver mode to ‘2’
   (a) Configures the communication to be half duplex.
   (b) USB2ANY transmits data with 2 stop bits but accepts data with single stop bit.
3. Setting the USB2ANY internal timeout to make sure USB2ANY waits for enough time to receive the data while at lower baud rates.

2.3.2.2 ADC Settings

- The page has options to enable VREF and set the Analog Power.

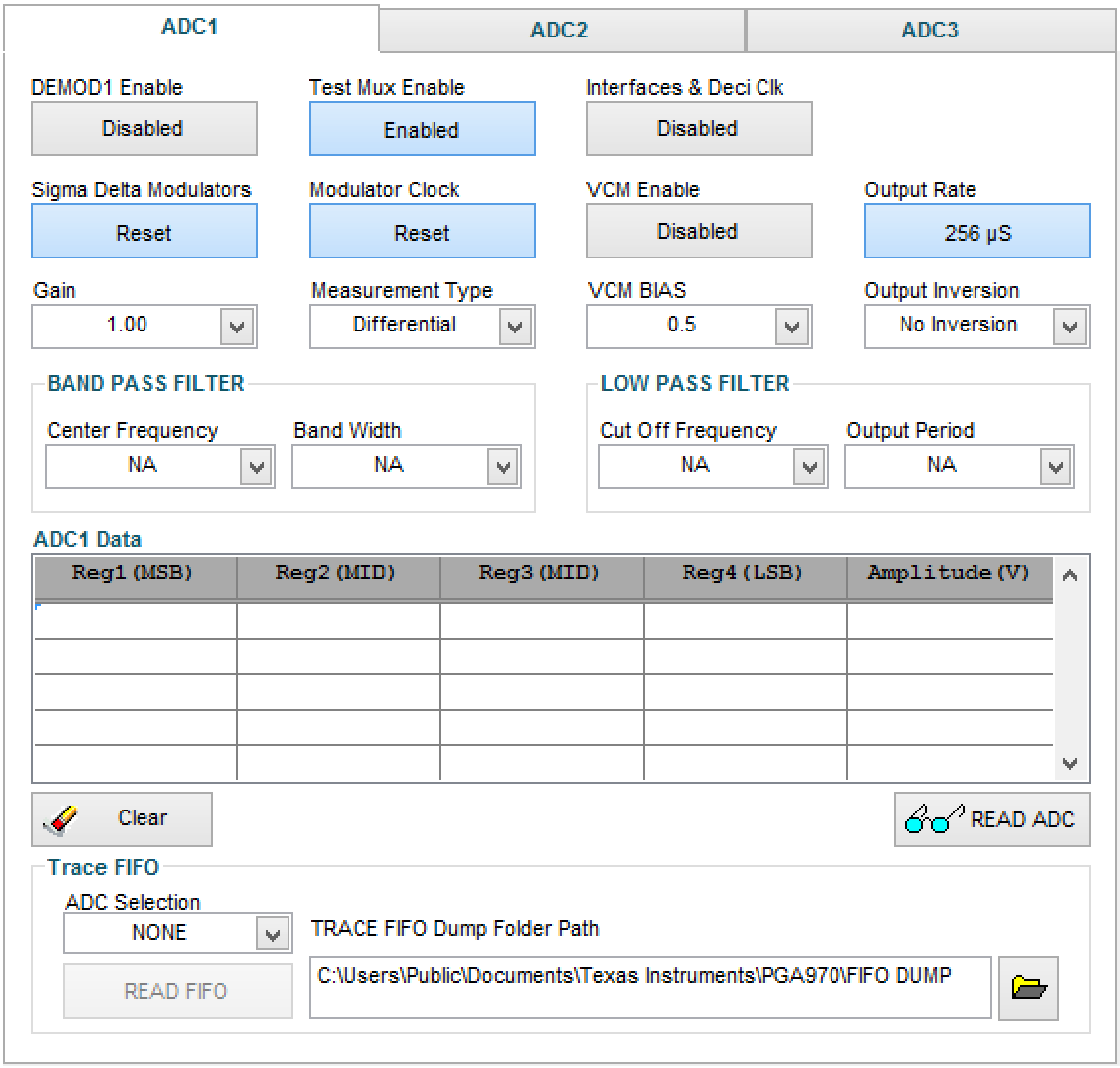


Enable VREF Buffer: Disabled

Analog Power: ShutDown Mode

Figure 2-16. Enable Analog Config

- The enable VREF button enables/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 page contains the settings for ADC1, ADC2 and ADC3 grouped under corresponding tabs.
- The table at the bottom of the page is used to read the ADC register values and display the corresponding amplitudes in volts. For ADC1 and ADC2 click on the Read ADC button to read the values.



DEMOD1 Enable: Disabled

Test Mux Enable: Enabled

Interfaces & Deci Clk: Disabled

Sigma Delta Modulators: Reset

Modulator Clock: Reset

VCM Enable: Disabled

Output Rate: 256 µS

Gain: 1.00

Measurement Type: Differential

VCM BIAS: 0.5

Output Inversion: No Inversion

BAND PASS FILTER

Center Frequency: NA

Band Width: NA

LOW PASS FILTER

Cut Off Frequency: NA

Output Period: NA

ADC1 Data

Reg1 (MSB)	Reg2 (MID)	Reg3 (MID)	Reg4 (LSB)	Amplitude (V)

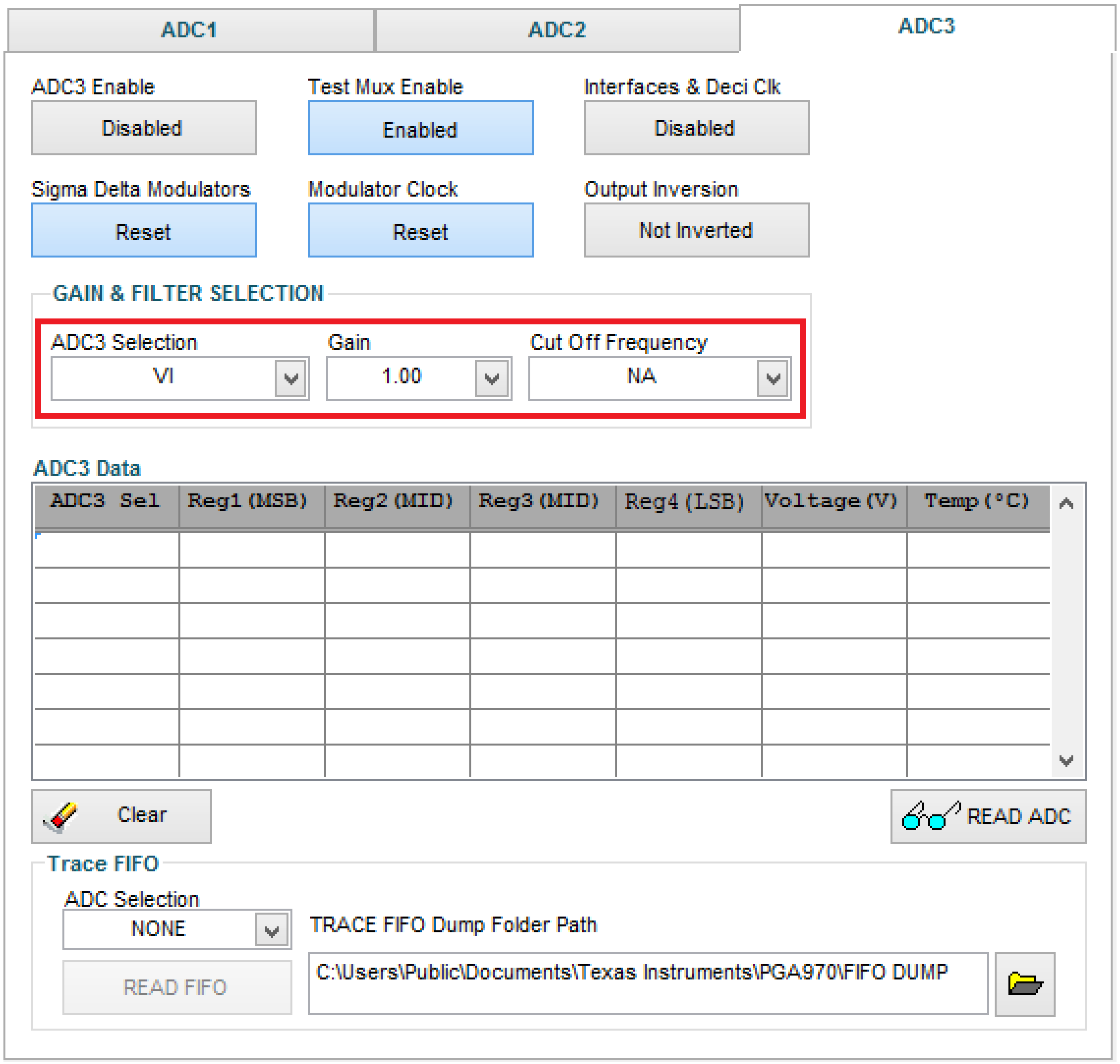
Trace FIFO

ADC Selection: NONE

TRACE FIFO Dump Folder Path: C:\Users\Public\Documents\Texas Instruments\PGA970\FIFO DUMP

Figure 2-17. ADC1 Settings

- For ADC3 select any one of the option from the ADC3 Selection dropdown and then click on Read ADC button. Note that the gain value is different for each selection and make sure that proper gain is selected before reading the ADC data.



ADC3 Enable: Disabled

Test Mux Enable: Enabled

Interfaces & Deci Clk: Disabled

Sigma Delta Modulators: Reset

Modulator Clock: Reset

Output Inversion: Not Inverted

ADC3 Selection: VI

Gain: 1.00

Cut Off Frequency: NA

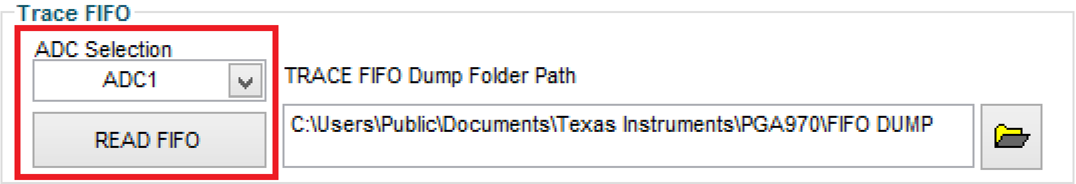
ADC3 Sel	Reg1 (MSB)	Reg2 (MID)	Reg3 (MID)	Reg4 (LSB)	Voltage (V)	Temp (°C)

ADC Selection: NONE

TRACE FIFO Dump Folder Path: C:\Users\Public\Documents\Texas Instruments\PGA970\FIFO DUMP

Figure 2-18. ADC3 Settings

- The data could also be dumped into a file by selecting the corresponding ADC in the TRACE_FIFO_ENABLE drop down. This will enable the READ FIFO button and configure the Trace source.
- Provide a folder path where the ADC values that are read from the TRACE FIFO has to be saved. Press READ FIFO button to read data and dump it into a file which gets saved with the present timestamp as the file name. The read progress is shown when the GUI reads the data from the development RAM. Note that the file name cannot be specified for the TRACE FIFO dump. It gets generated by the GUI based on the time at which TRACE FIFO read gets completed.

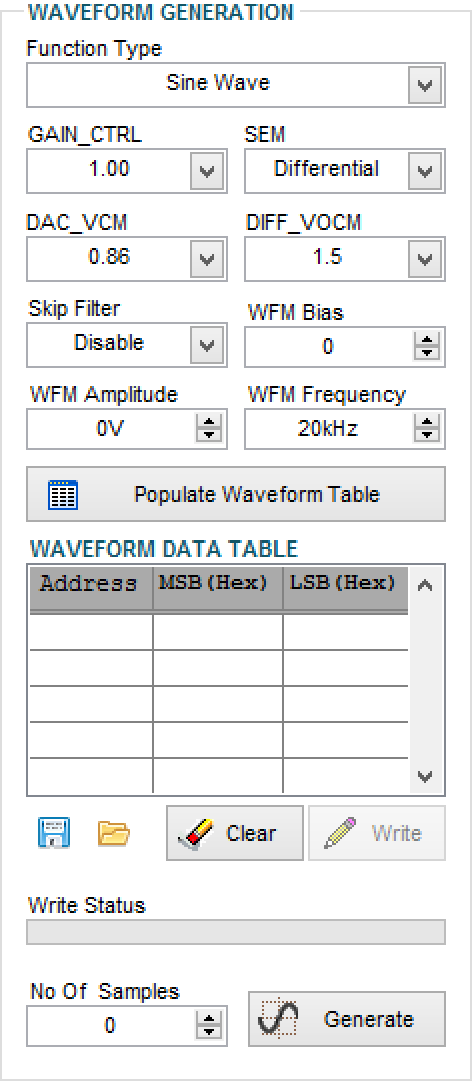


C:\Users\Public\Documents\Texas Instruments\PGA970\FIFO DUMP

Figure 2-19. Trace FIFO

- When the user presses the Read FIFO Button, the following set of operations are performed,
  - The REMAP bit in the Register is set to zero to come out of the OTP
  - The FIFO is flushed
  - The FIFO is enabled again to have a fresh data on the DEVRAM
  - GUI waits till TRACE FIFO is full. Then the GUI starts reading from the DEVRAM
- The waveform generation section of the ADC page is used to generate continuous Sine wave. Steps to generate sine wave:
  1. Specify required values in GAIN_CTRL, SEM, DAC_VCM, DIFF_VOCM, Skip Filter.
  2. Specify required offset for the wave to be generated in WFM Bias. The range for this value is 0.82 to 0.88
  3. Specify the amplitude for the sine wave in WFM Amplitude. The range of WFM Amplitude value depends on the GAIN_CTRL, SEM and WFM_Bias Value in the following manner.
     - In Single-Ended Mode
       WFM_Amplitude < (WFM_Bias - 0.15)/GAIN_CTRL
     - In Differential mode
       WFM_Amplitude < 1.235 - WFM Bias

     When GUI is initializing WFM Amplitude and WFM_BIAS controls will be loaded with typical values. User can able to change the values during runtime and when entered value lies outside limit or if it is not satisfying the above condition, Popup will be displayed and entered value will be coerced to proper value.
  4. Specify the frequency with which the wave has to be generated in WFM Frequency. The range for frequency is 1 kHz to 20 kHz.
  5. After specifying the waveform parameters, press Populate Waveform Table button. The Waveform table gets populated with the values that has to be written to the Waveform RAM. No. of Samples shows the number of samples required to generate quarter of the specified sine wave. The table values can be edited and number of samples can also be specified manually through No. of Samples edit box. There can be between 1 and 256.
  6. To write the table values to the device, click the Write button at the bottom of waveform table.
  7. Clear will erase the waveform table values. Save and Open is used to save the waveform data to a text file or load the waveform data from a text file.
  8. Click Generate button to generate the waveform. When the button is clicked, a set of registers related to the waveform generation gets written to the device.
  9. After the waveform generation starts, click the Stop button to stop the waveform generation.



WAVEFORM GENERATION

Function Type: Sine Wave

GAIN_CTRL: 1.00

SEM: Differential

DAC_VCM: 0.86

DIFF_VOCM: 1.5

Skip Filter: Disable

WFM Bias: 0

WFM Amplitude: 0V

WFM Frequency: 20kHz

WAVEFORM DATA TABLE

Address	MSB (Hex)	LSB (Hex)

Write Status

No Of Samples: 0

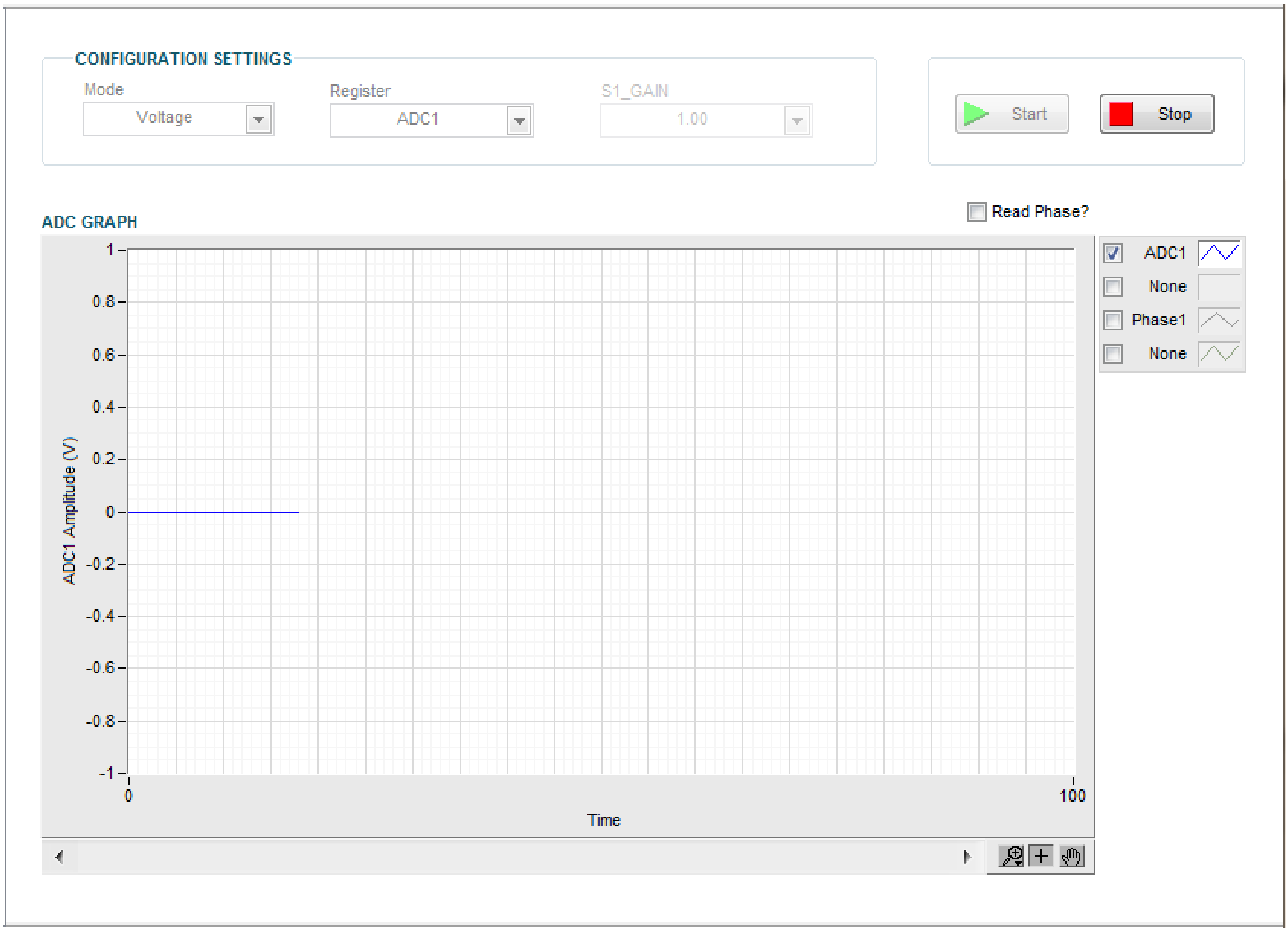
Figure 2-20. Waveform Generation

2.3.2.3 ADC Capture

- 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.
- The Register dropdown is used to switch between the various ADCs and the Gain dropdown sets corresponding gain to the selected ADC.
- If Mode is Voltage, the graph displays the voltage values and if the Mode is ADC Code, then the graph displays the ADC register values. Mode can also be selected to Degree or Fahrenheit when temperature ADC Register is selected to view the data in degree or Fahrenheit scale.
- Check or uncheck the Read Phase? Selection to read the Phase value of ADC. This option will be available only for ADC1 and ADC2 registers.
- After the required configurations are done on the UI, the capture can be started by clicking the Start button.
- The continuous capture data will be displayed on the waveform graph on the UI
- Click the Stop button to stop the capture at any point of time.

- To export the ADC data to a text file, Right click the graph area and select Export data from the shortcut menu. From the dialog box that appears, select a text file to which the values has to be stored. Maximum number of samples that can be stored to a file is 1024 samples.
- To export the ADC data to a text file, Right click the graph area and select Export data from the shortcut menu. From the dialog box that appears, select a text file to which the values has to be stored. Maximum number of samples that can be stored to a file is 1024 samples.

NOTE: When Read Phase? Option is selected, ADC data read will perform using USB2ANY scripting.



CONFIGURATION SETTINGS

Mode: Voltage

Register: ADC1

S1_GAIN: 1.00

□ Read Phase?

☑ ADC1
□ None
□ Phase1
□ None

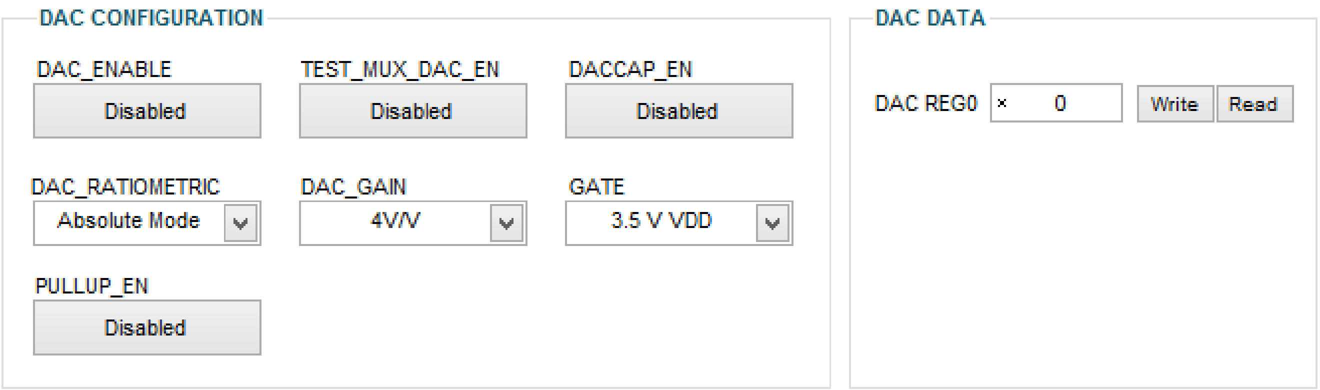
Figure 2-21. ADC Capture

2.3.2.4 DAC Settings

The DAC Settings Page contain the DAC features of the device that's has been explained in the below sections.

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 ratio metric output voltage with respect to the VDD supply.
- The DAC can be disabled by writing 0 to DAC_ENABLE bit in 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

DACCAP_EN: Disabled

DAC_RATIOMETRIC: Absolute Mode

DAC_GAIN: 4V/V

GATE: 3.5 V VDD

PULLUP_EN: Disabled

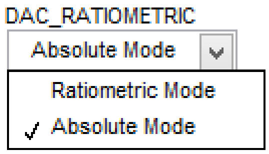
DAC DATA

DAC REG0: x 0

Figure 2-22. DAC Settings

Ratio Metric 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 Ratio metric 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.



DAC_RATIOMETRIC: Absolute Mode

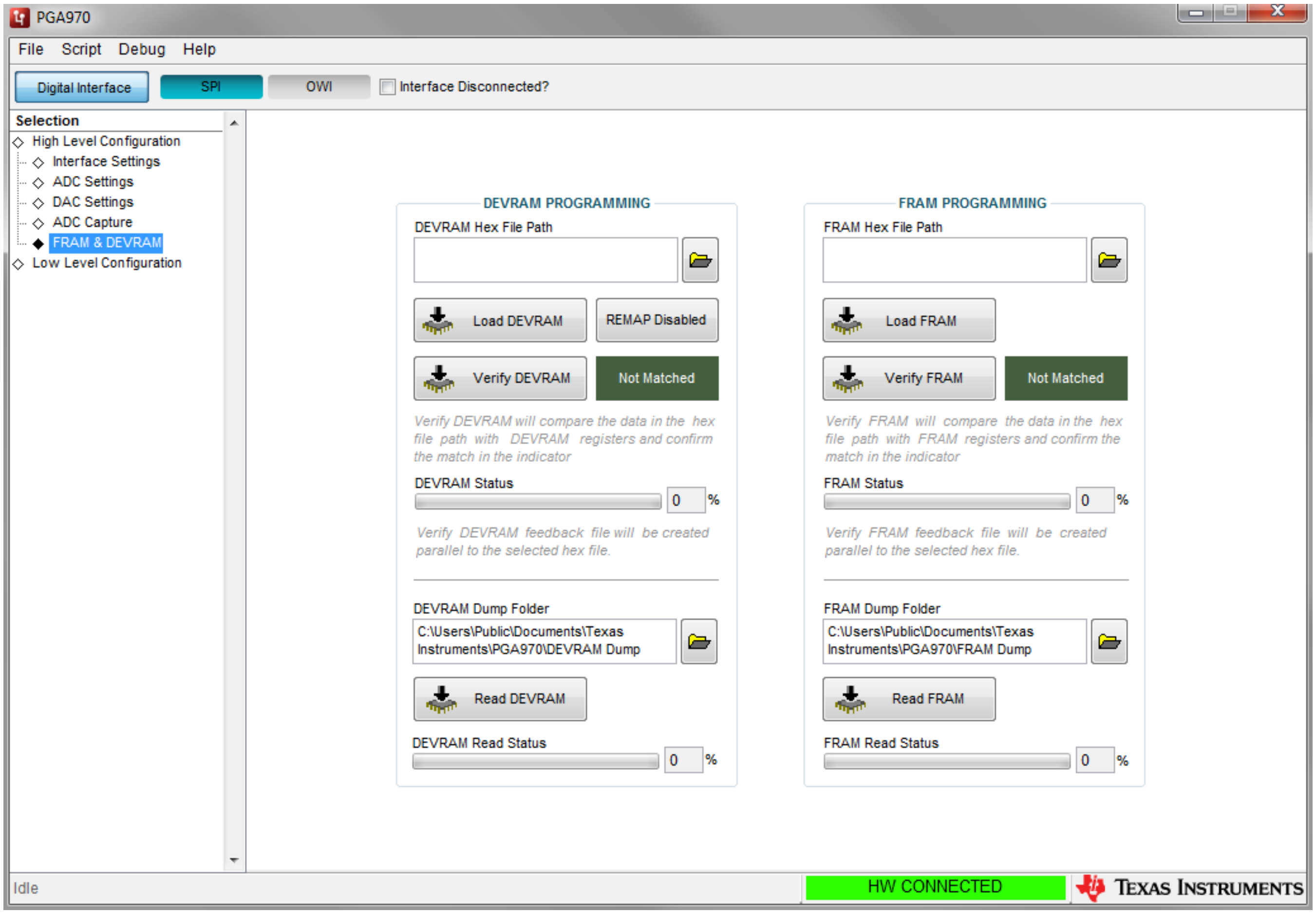
Figure 2-23. Configure DAC

DAC Gain

- The DAC Gain buffer is a configurable buffer stage for the DAC Output.
- In voltage output mode, DAC Gain can be configured for a specific gain value by setting the DAC_GAIN bits in 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.

2.3.2.5 FRAM and DEVRAM

This page describes the FRAM and DEVRAM programming functionalities of the PGA970 EVM as shown in Figure 2-24. Each of the device functions are discussed in the following sections.



□ Interface Disconnected?

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

DEVRAM Status: 0 %

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

DEVRAM Dump Folder: C:\Users\Public\Documents\Texas Instruments\PGA970\DEVRAM Dump

DEVRAM Read Status: 0 %

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

FRAM Status: 0 %

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

FRAM Dump Folder: C:\Users\Public\Documents\Texas Instruments\PGA970\FRAM Dump

FRAM Read Status: 0 %

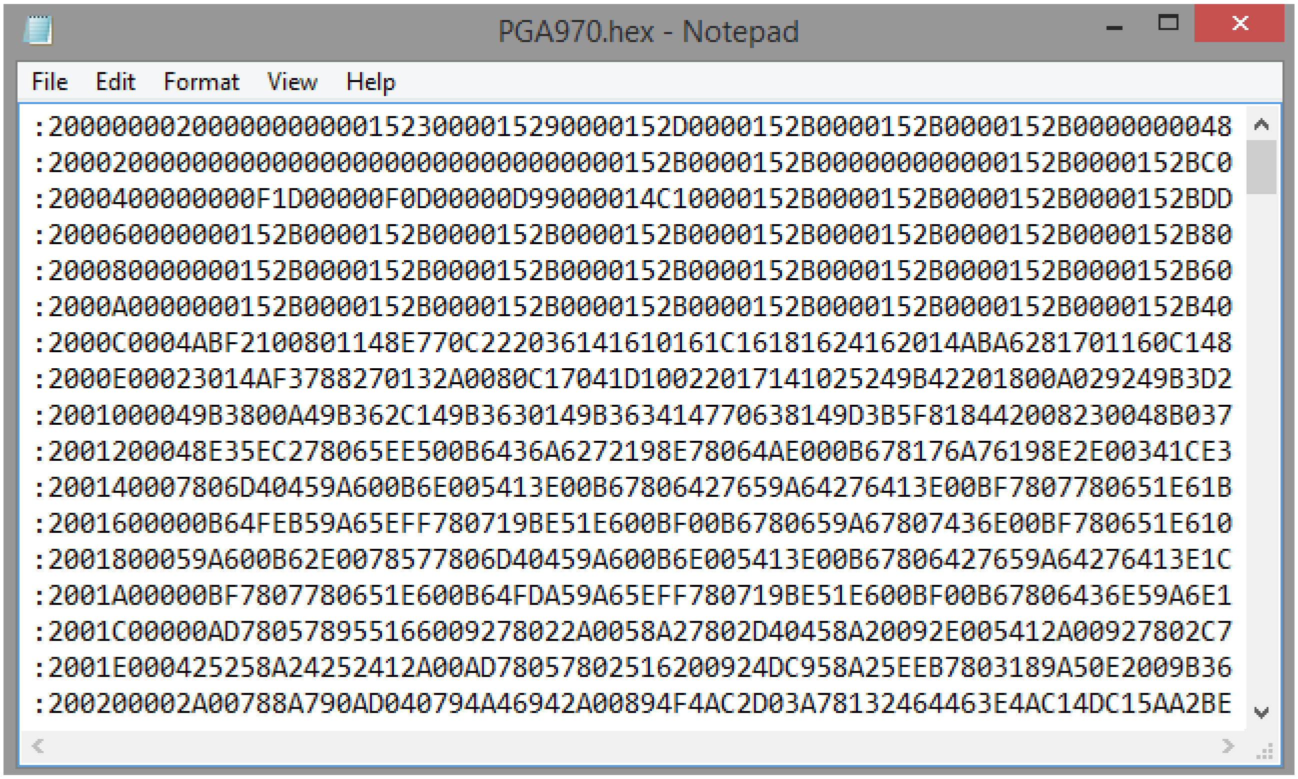
Figure 2-24. FRAM and DEVRAM

FRAM Programming

- FRAM memory is 8-bit addressable.
- In order to allow the Communication interface to access these memories using 8-bit address, the memory space is organized as 256-byte/page.
- There are a total of 32 pages.

Load FRAM Memory

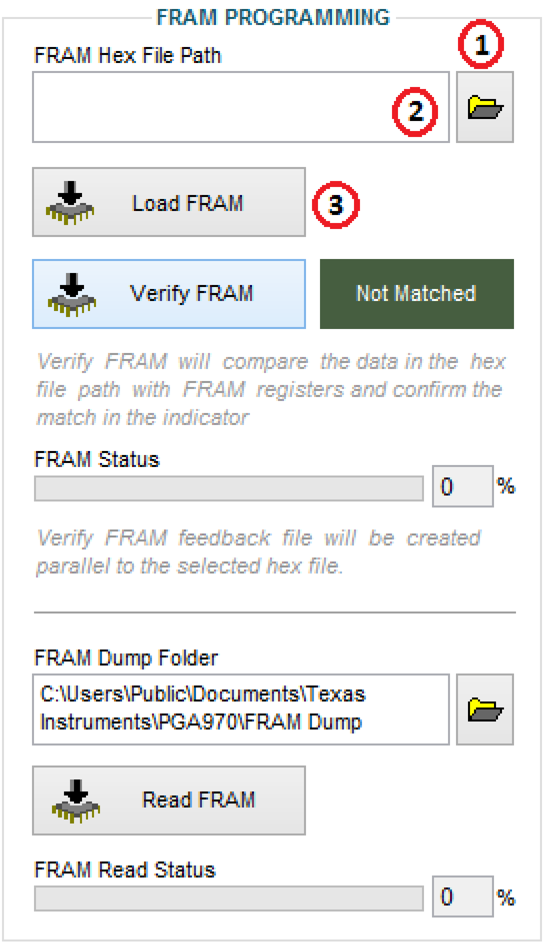
- The data to be written to the OTP is provided to the GUI as a HEX file. A typical HEX file is shown in Figure 2-25.



```
:20000000200000000000152300001529 0000152D0000152B0000152B0000152B0000000048
:200020000000000000000000000000000000152B0000152B000000000000152B0000152BC0
:2000400000000F1D00000F0D00000D99000014C10000152B0000152B0000152B0000152BDD
:200060000000152B0000152B0000152B0000152B0000152B0000152B0000152B0000152B80
:200080000000152B0000152B0000152B0000152B0000152B0000152B0000152B0000152B60
:2000A0000000152B0000152B0000152B0000152B0000152B0000152B0000152B0000152B40
:2000C0004ABF2100801148E770C2220361416101 61C16181624162014ABA6281701160C148
:2000E00023014AF3788270132A0080C17041D10022017141025249B42201800A029249B3D2
:2001000049B3800A49B362C149B3630149B363414770638149D3B5F818442008230048B037
:2001200048E35EC278065EE500B6436A6272198E78064AE000B678176A76198E2E00341CE3
:200140007806D40459A600B6E005413E00B67806427659A64276413E00BF7807780651E61B
:2001600000B64FEB59A65EFF780719BE51E600BF00B6780659A67807436E00BF780651E610
:2001800059A600B62E0078577806D40459A600B6E005413E00B67806427659A64276413E1C
:2001A00000BF7807780651E600B64FDA59A65EFF780719BE51E600BF00B67806436E59A6E1
:2001C00000AD78057895516600927802 2A0058A27802D40458A20092E005412A00927802C7
:2001E000425258A24252412A00AD78057802516200924DC958A25EEB7803189A50E2009B36
:200200002A00788A790AD040794A46942A00894F4AC2D03A78132464463E4AC14DC15AA2BE
```

Figure 2-25. Sample HEX File

- Select the HEX file path by selecting the browse button as shown in Figure 2-26. The OTP memory is programmed with the contents of the HEX file, after the Load FRAM button is clicked.



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

FRAM Status: 0 %

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

FRAM Dump Folder: C:\Users\Public\Documents\Texas Instruments\PGA970\FRAM Dump

FRAM Read Status: 0 %

1. Press button to load HEX file into GUI
2. Select required file from browser window
3. Press **Load FRAM** button

Figure 2-26. FRAM Programming

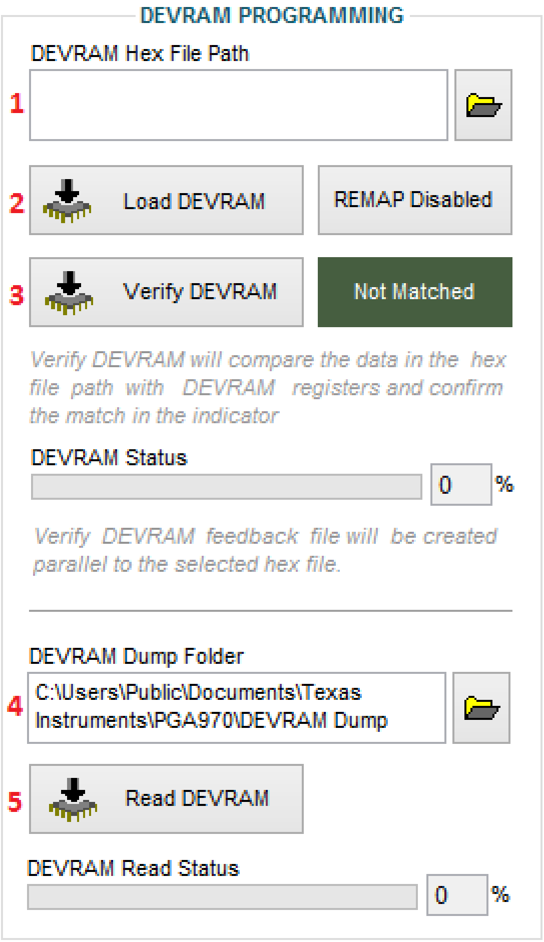
- If the Verify FRAM button is pressed the GUI compares the contents of the FRAM memory with the selected HEX file. If the FRAM memory matches the contents of the .HEX file, the status will be updated from Not Matched to Matched and the LED glows.

Verify FRAM function does the below activities:

- It reads all the 32 pages of the FRAM registers
- Verifies if the data that is read from the device is same as the data present in the HEX file. The verification module ignores the registers that are not present in the HEX file.
- If all the register data matches, the status is updated as Matched and the LED glows.
- A verification log file is generated in the directory, parallel to the selected HEX file, with the file name same as the selected hex file but succeeded with a text _Verification_Log. This file contains the details (Address, Hex File data and Device Data, both in decimal representation) about all the locations where the verification modules encountered a data mismatch.

DEVRAM Programming

- PGA970 has 32 pages of Development RAM with 256 registers each that can overlay the FRAM memory address. This is to allow convenient programming of the device.
- If the REMAP button is clicked, the GUI sets the REMAP bit to 1
- Always ensure that the REMAP is disabled before loading DEVRAM so that the data gets written to the correct memory and not to the FRAM
- DEVRAM also implements verify feature similar to FRAM verify which will compare the DEVRAM data against the selected HEX file data. If both the data matches, the status is updated to Matched and the LED glows.



DEVRAM PROGRAMMING

DEVRAM Hex File Path

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

DEVRAM Status: 0 %

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

DEVRAM Dump Folder: C:\Users\Public\Documents\Texas Instruments\PGA970\DEVRAM Dump

DEVRAM Read Status: 0 %

Figure 2-27. DEVRAM Programming

1. Browse the .HEX file path to load the Hex file to the GUI
2. Click on Load DEVRAM button to start programming the DEVRAM. The status bar shows the progress of DEVRAM programing.
3. Click on Verify DEVRAM button to compare DEVRAM data against the hex file selected in step 1. The status bar shows the progress of DEVRAM verification. A log file is created parallel to the selected hex file. This log file contains details of all the registers that mismatched with the HEX file data.
4. Browse the path to save the DEVRAM data
5. Click on Read Dev RAM to read the data from Development RAM and save to a file. The status of read is shown by the progress bar.

2.4 Menu Options

2.4.1 File

The File menu contains the Exit option as shown in Figure 2-28. The File Menu → Exit option is used to stop the execution of the PGA970 GUI.

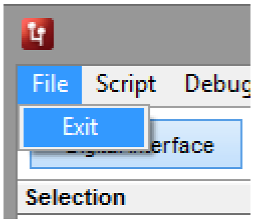


Figure 2-28. File Menu

2.4.2 Script

- Scripting is used to automate the device operations and reduce the time consumption while repeating similar operations.
- This is helpful in situations where, performing a particular device function required setting 10 to 15 registers to a particular value. In these circumstances, scripts could be recorded and run whenever needed.
- In PGA970 GUI, the scripting is done using Python because,
  - It's easier to implement
  - More widely used
  - More user friendly

2.4.2.1 Performing Macro Recording

- To create a custom macro, click Script->Launch Window,

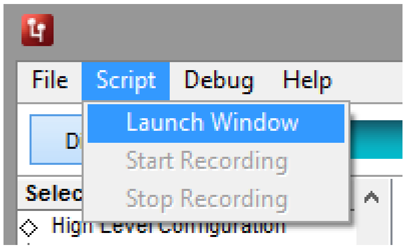


Figure 2-29. Launch Window

- Python IDLE window appears as shown below.

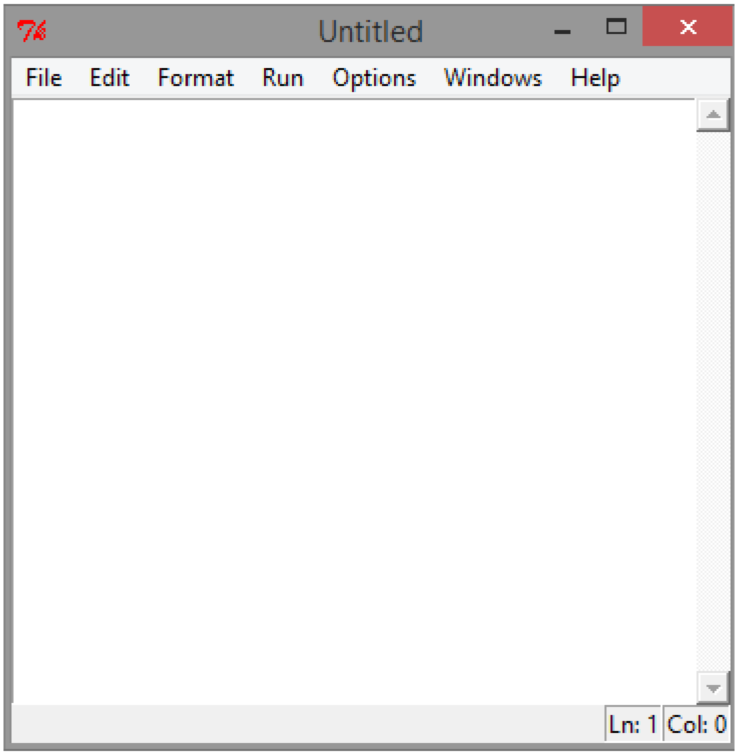


Figure 2-30. Python Idle Window

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

- Click Script → Start Recording, IDLE window becomes green showing that recording has started

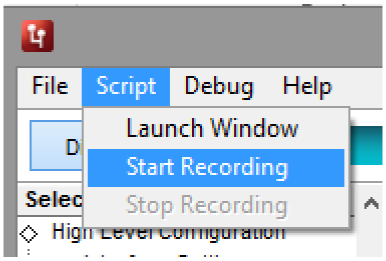
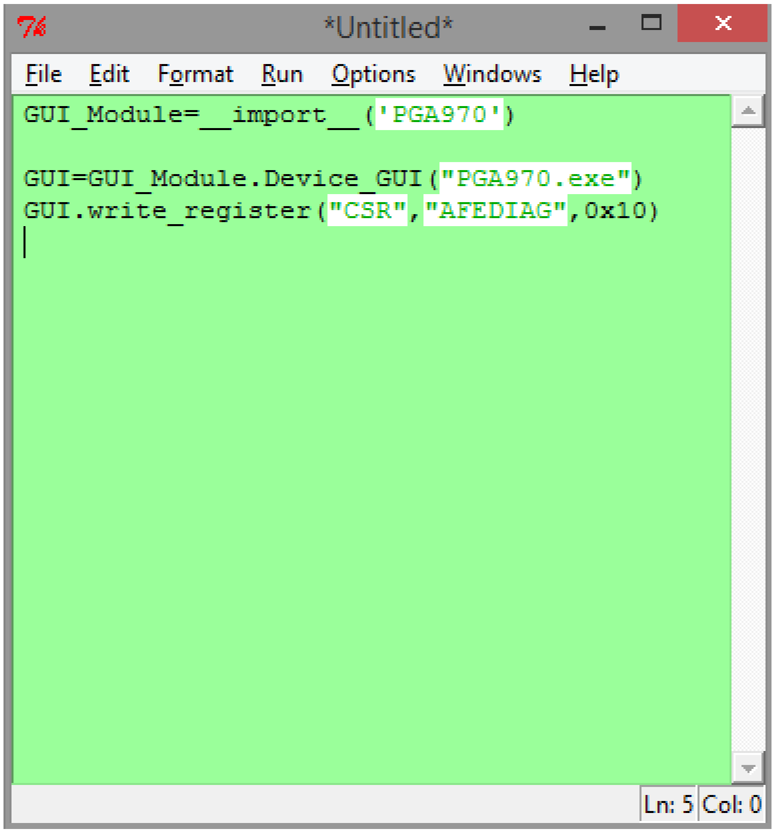


Figure 2-31. Start Recording

- Go to Low Level Configuration Page of PGA970, select AFEDIAG register and write 10 to Data control and press Write Register. Action is recorded as shown in Figure 2-32.



```
GUI_Module=__import__('PGA970')

GUI=GUI_Module.Device_GUI("PGA970.exe")
GUI.write_register("CSR","AFEDIAG",0x10)
```

Figure 2-32. Macro Recording

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

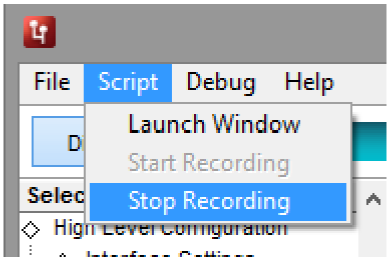
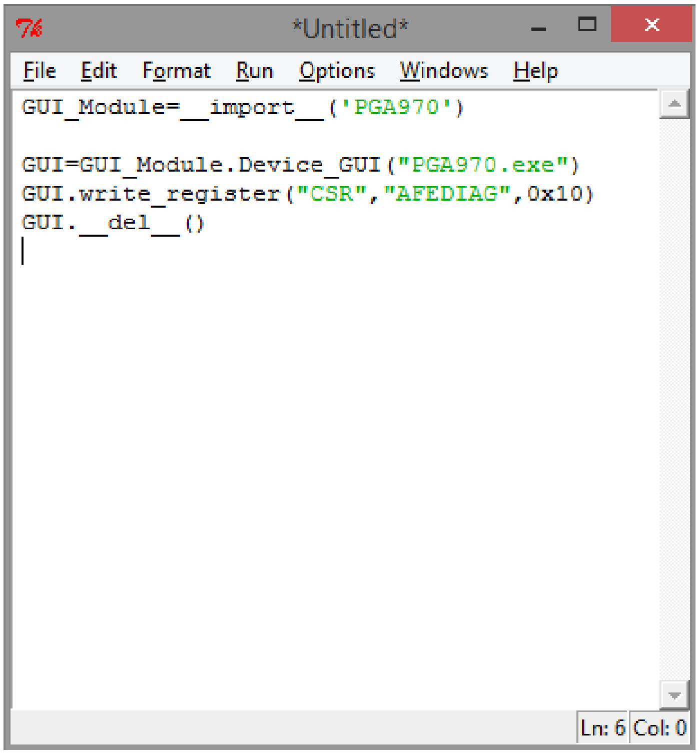


Figure 2-33. Stop Recording

- Python IDLE window will no longer be green indicating that the recording has been stopped.



```
GUI_Module=__import__('PGA970')

GUI=GUI_Module.Device_GUI("PGA970.exe")
GUI.write_register("CSR","AFEDIAG",0x10)
GUI.__del__()
```

Figure 2-34. Finished Macro Recording

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

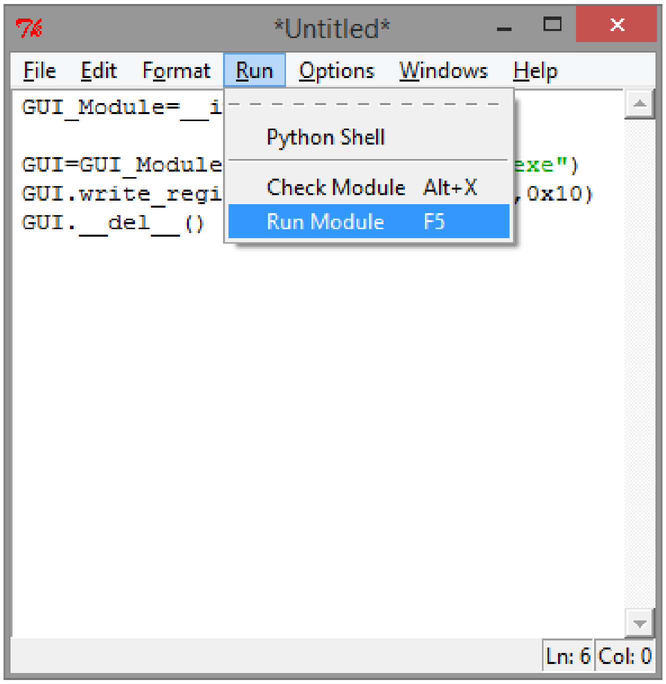
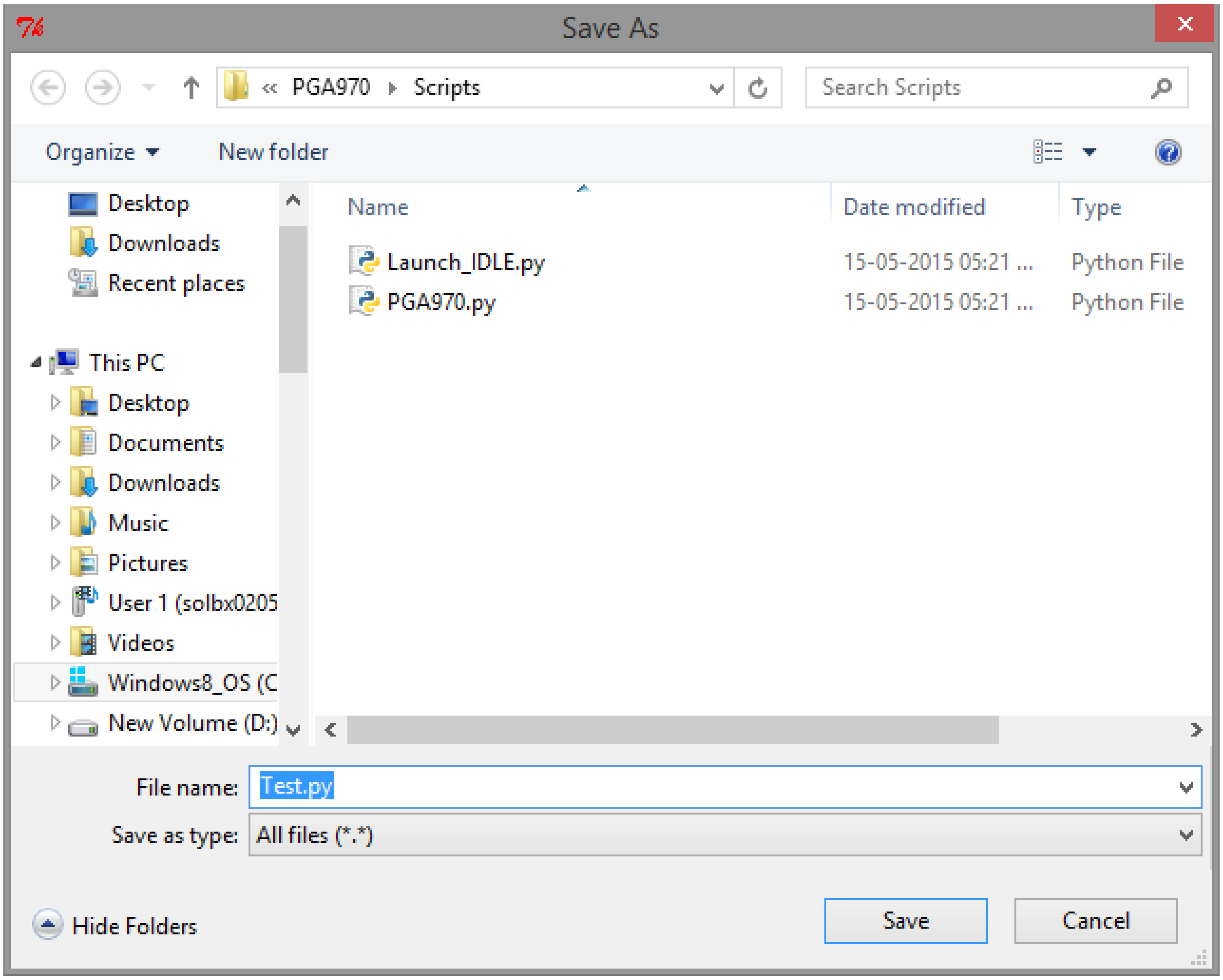


Figure 2-35. Run Module

- Save As dialog appears asking for the file name for the script.
- Specify a relevant name (Test.py) and click Save.

NOTE: Name of the script should have the extension .py



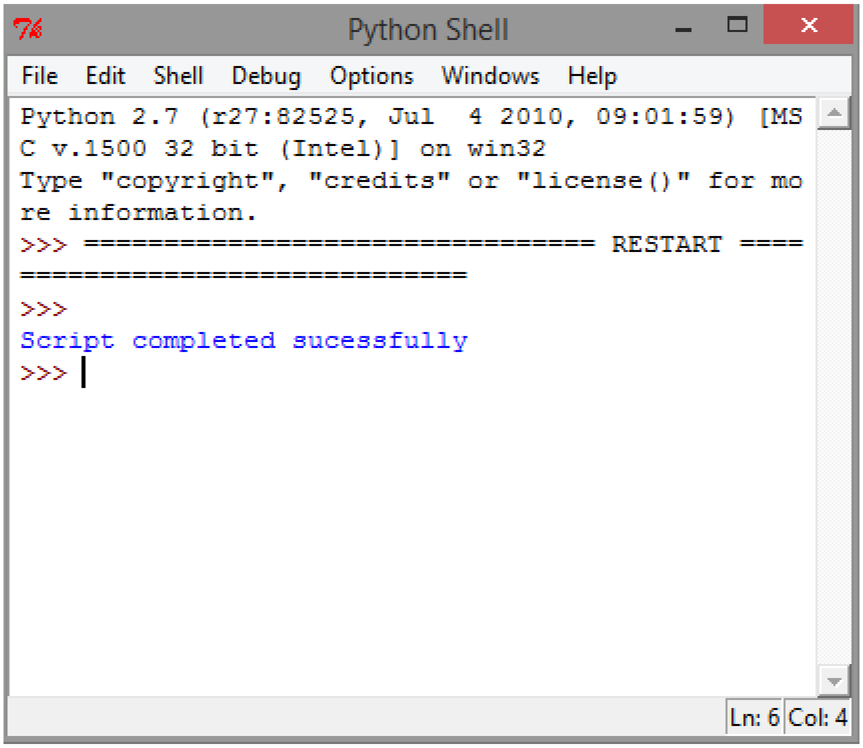
Name	Date modified	Type
Launch_IDLE.py	15-05-2015 05:21 ...	Python File
PGA970.py	15-05-2015 05:21 ...	Python File

File name: Test.py

Save as type: All files (*.*)

Figure 2-36. Save Browser Window

- The script runs and the status is displayed in the shell window as shown below,



```
Python 2.7 (r27:82525, Jul  4 2010, 09:01:59) [MS
C v.1500 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for mo
re information.
>>> ================================ RESTART ====
============================
>>>
Script completed sucessfully
>>> 
```

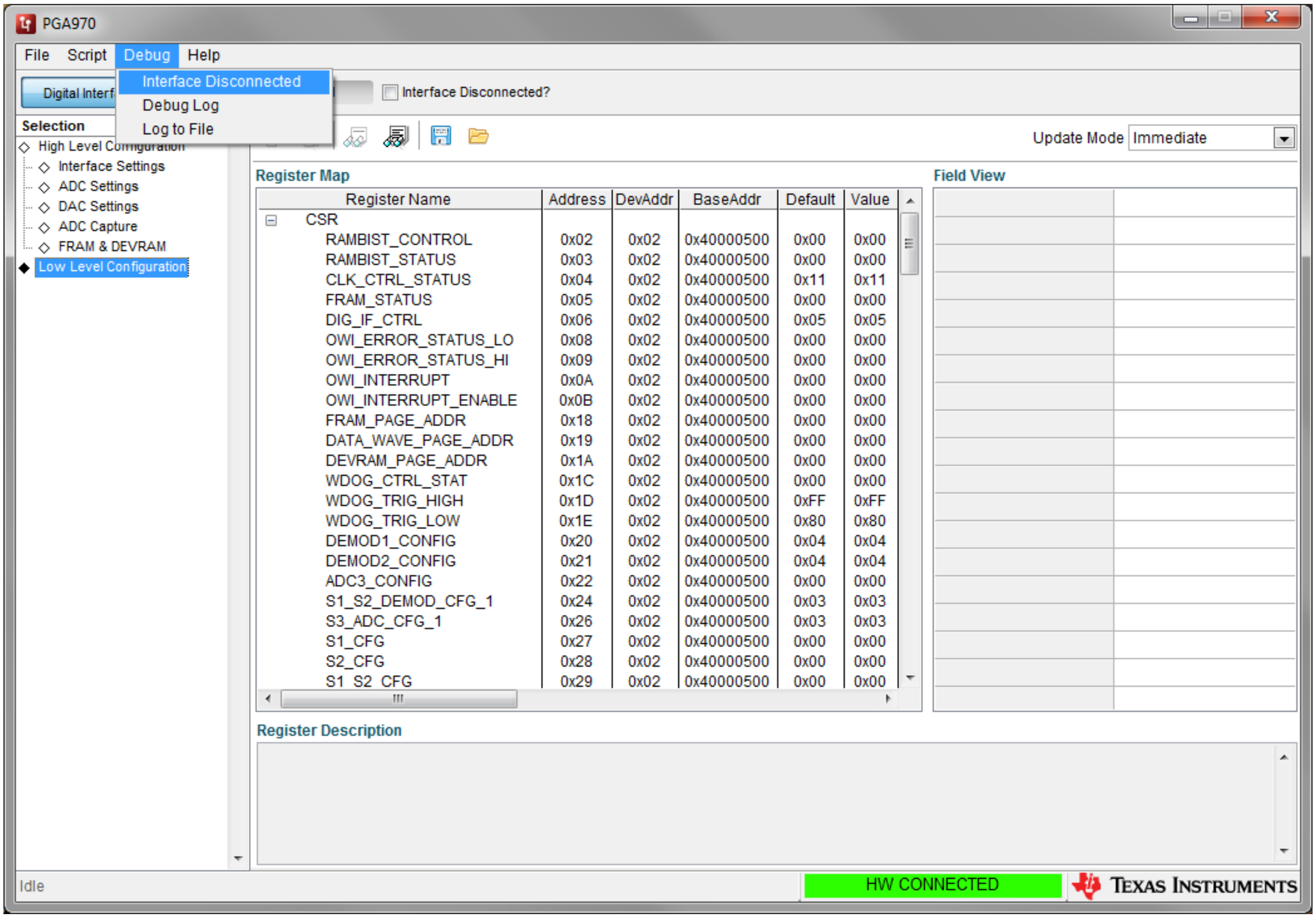
Figure 2-37. Run Saved Macro

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

2.4.3 Debug

he Debug option can be used for the following operations:

- **Interface Disconnected** - By selecting the Interface Disconnected submenu, the PGA970 GUI is run in USB2ANY Disconnected mode and by deselecting it, the PGA970 GUI is run in connected mode.
- **File Logging** - The log to file submenu is used to log the GUI activities to a log file that will be created at [Public Documents]\Texas Instruments\PGA970\Trace Log. The file name will be time stamp.
- **Debugging** - The Debug log option will enable the user to log all the activities of the user. If that is not selected, only the high level operations will be logged.



☐ Interface Disconnected?

Update Mode Immediate

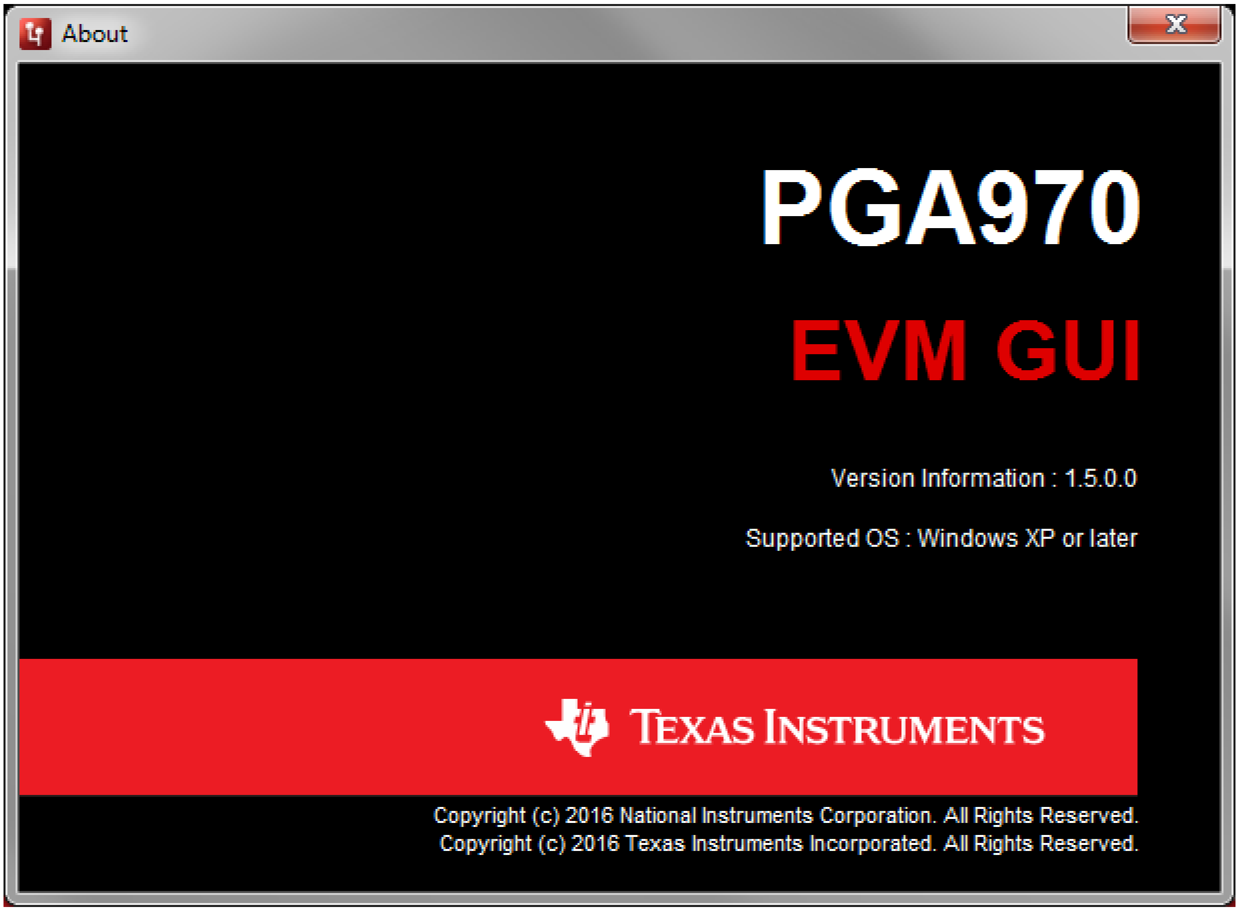
Register Name	Address	DevAddr	BaseAddr	Default	Value
⊟ CSR					
RAMBIST_CONTROL	0x02	0x02	0x40000500	0x00	0x00
RAMBIST_STATUS	0x03	0x02	0x40000500	0x00	0x00
CLK_CTRL_STATUS	0x04	0x02	0x40000500	0x11	0x11
FRAM_STATUS	0x05	0x02	0x40000500	0x00	0x00
DIG_IF_CTRL	0x06	0x02	0x40000500	0x05	0x05
OWI_ERROR_STATUS_LO	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
FRAM_PAGE_ADDR	0x18	0x02	0x40000500	0x00	0x00
DATA_WAVE_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_HIGH	0x1D	0x02	0x40000500	0xFF	0xFF
WDOG_TRIG_LOW	0x1E	0x02	0x40000500	0x80	0x80
DEMOD1_CONFIG	0x20	0x02	0x40000500	0x04	0x04
DEMOD2_CONFIG	0x21	0x02	0x40000500	0x04	0x04
ADC3_CONFIG	0x22	0x02	0x40000500	0x00	0x00
S1_S2_DEMOD_CFG_1	0x24	0x02	0x40000500	0x03	0x03
S3_ADC_CFG_1	0x26	0x02	0x40000500	0x03	0x03
S1_CFG	0x27	0x02	0x40000500	0x00	0x00
S2_CFG	0x28	0x02	0x40000500	0x00	0x00
S1_S2_CFG	0x29	0x02	0x40000500	0x00	0x00

Figure 2-38. Debug Menu

2.4.4 Help

2.4.4.1 About

- 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 of the PGA970 GUI.



PGA970

EVM GUI

Version Information : 1.5.0.0

Supported OS : Windows XP or later

Copyright (c) 2016 National Instruments Corporation. All Rights Reserved.
Copyright (c) 2016 Texas Instruments Incorporated. All Rights Reserved.

Figure 2-39. About

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