Automate your Keysight Test Instruments Using C#

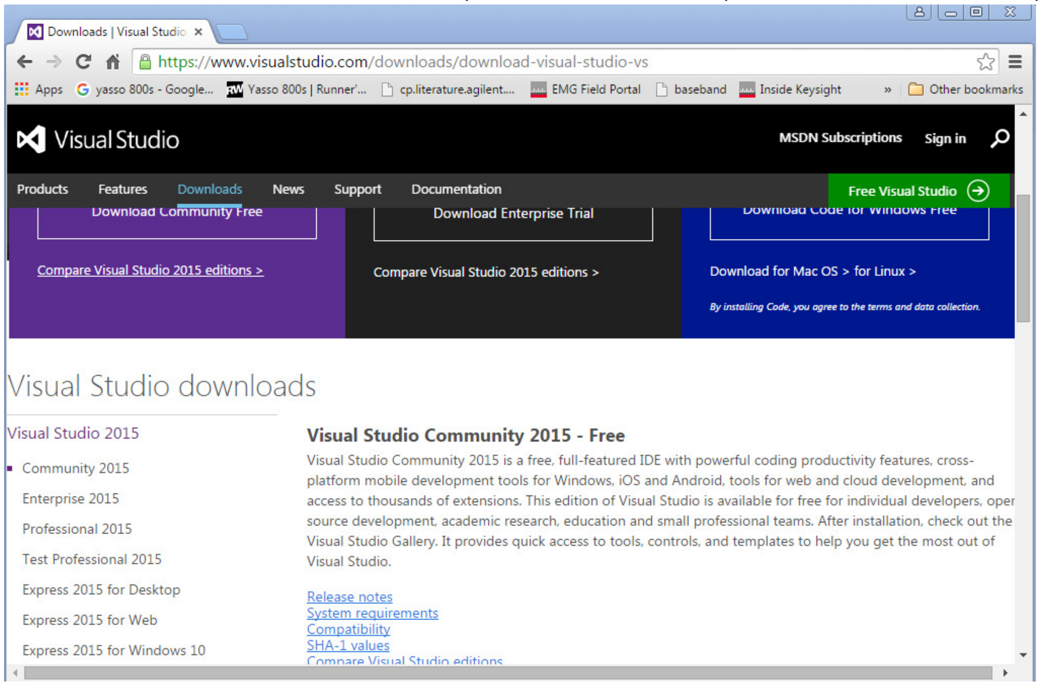
This document provides a step by step guide showing how to communicate with test instruments using C#. The following steps will be reviewed.

1) Install required software applications
    a. Install Keysight IO Libraries
    b. Install Visual Studio Express
2) Verify Instrument Connectivity using Keysight IO Libraries
3) Communicate w/ Test Instruments using SCPI Commands
4) Additional Test and Measurement Resources

Step1) Install Software

1) **Install Keysight IO Libraries**
    a. Download Keysight IO Libraries
    b. **www.keysight.com/find/iolib**
    c. Install Keysight IO Libraries

2) **Install Visual Studio Express**
    a. Download Visual Studio Express from MSDN Website
    b. https://msdn.microsoft.com/en-us/default.aspx
    c. Go to Downloads – Visual Studio. Then under Visual Studio Downloads section, go to the latest version and select the Express version for Desktop and Install Visual Studio Express



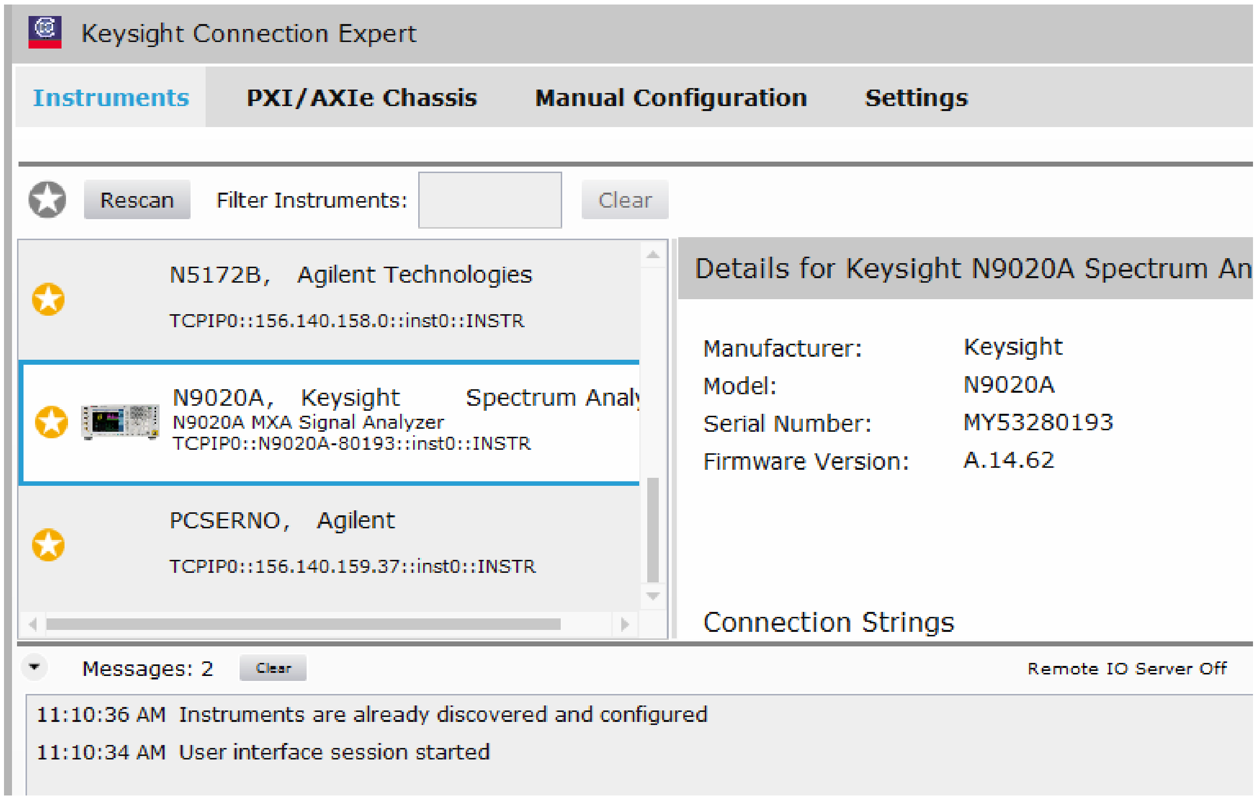
Visual Studio Community 2015 - Free

Visual Studio Community 2015 is a free, full-featured IDE with powerful coding productivity features, cross-platform mobile development tools for Windows, iOS and Android, tools for web and cloud development, and access to thousands of extensions. This edition of Visual Studio is available for free for individual developers, open source development, academic research, education and small professional teams. After installation, check out the Visual Studio Gallery. It provides quick access to tools, controls, and templates to help you get the most out of Visual Studio.

Step 2) Verify Instrument Connectivity using Keysight IO Libraries

1) Choose a physical connection that works best for you. Many instruments today support remote connectivity over USB, LAN, GPIB, or PCIe. Determine which IO interface your instrument has and connect your instrument now. Consult your instrument's user's guide for additional instruction.

2) Launch Connection Expert, which is part of the Keysight IO libraries:
Start > All Programs > Keysight Connection Expert



N5172B, Agilent Technologies
TCPIP0::156.140.158.0::inst0::INSTR

N9020A, Keysight Spectrum Analy
N9020A MXA Signal Analyzer
TCPIP0::N9020A-80193::inst0::INSTR

PCSERNO, Agilent
TCPIP0::156.140.159.37::inst0::INSTR

Details for Keysight N9020A Spectrum An

Manufacturer:	Keysight
Model:	N9020A
Serial Number:	MY53280193
Firmware Version:	A.14.62

Connection Strings

Messages: 2 Clear

```
11:10:36 AM  Instruments are already discovered and configured
11:10:34 AM  User interface session started
```

Please reference the IO Libraries Documentation within the IO Libraries for help getting connected.

3) Once your instrument is properly configured, it will show up under "Instruments" tab. You can select the instrument entry and see its VISA address within the Details to the right. We will use this VISA address in our C# program.

Step 3) Communicate w/ Test Instruments using SCPI Commands in the C# environment

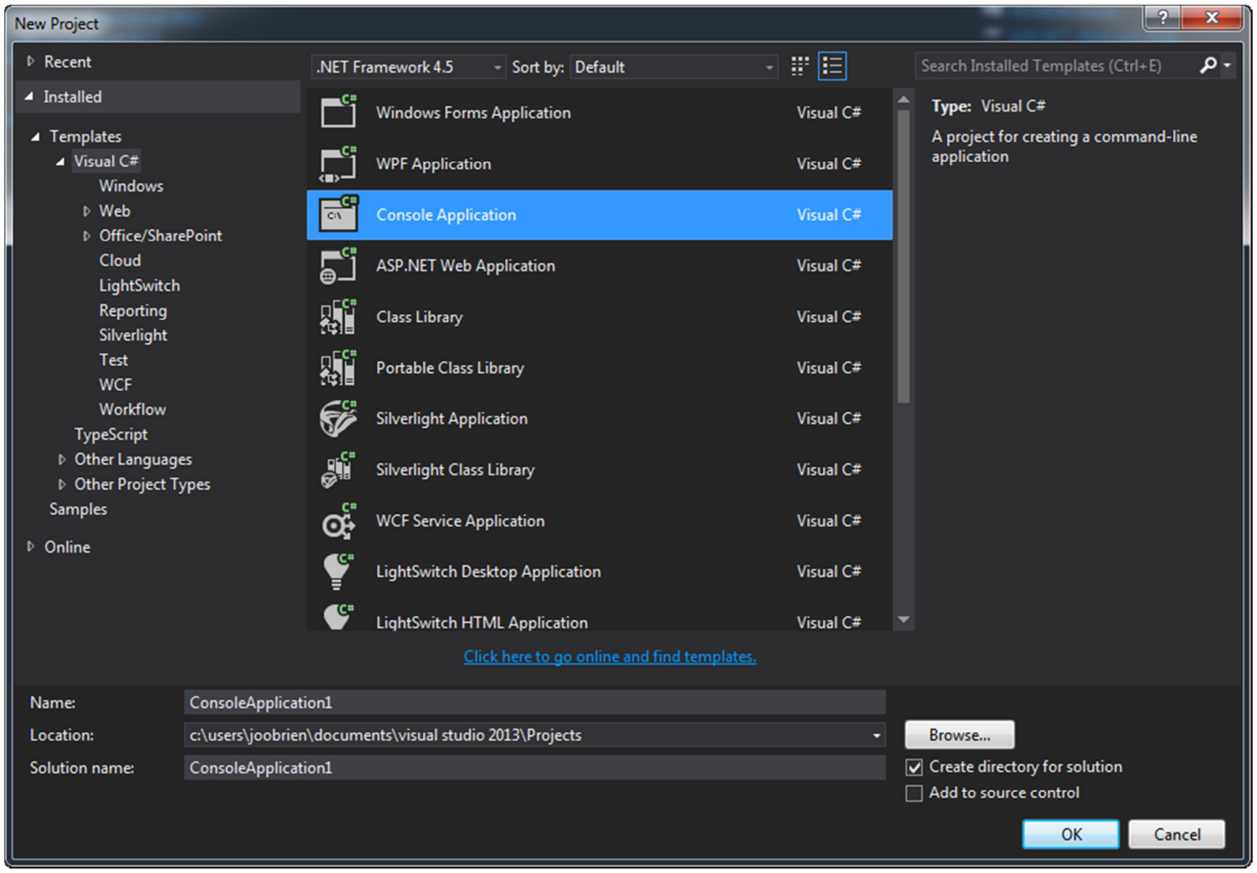
The VISA COM I/O API is a programming interface standardized by the IVI Foundation for communicating with instruments over GPIB, LAN, USB, RS-232, or other hardware interfaces. Keysight Technologies has an implementation of the VISA COM I/O standard that works with Keysight I/O hardware as well as the computer-standard I/O interfaces. VISA COM I/O is an update of the older VISA C API to work in and with Microsoft's COM technology.

Microsoft has integrated robust support for COM components in the .NET environment. The COM interfaces tend to translate well into .NET equivalents via automated wrapper-generator tools that Microsoft provides. Due to this COM support in .NET, many programmers will find VISA COM to be an excellent choice for instrument communication in .NET. This article describes how to use Keysight's VISA COM I/O implementation with a C# example.

Getting Started:

The example shown here was developed using C# 2013 as part of Visual Studio 2013 Professional Edition.

Launch Visual Studio and Select New Project and Select under Templates "Visual C#"; Console Application; and Name the project "InstrumentID"


New Project

- Recent
- Installed
  - Templates
    - Visual C#
      - Windows
      - Web
      - Office/SharePoint
      - Cloud
      - LightSwitch
      - Reporting
      - Silverlight
      - Test
      - WCF
      - Workflow
    - TypeScript
    - Other Languages
    - Other Project Types
  - Samples
- Online

.NET Framework 4.5 Sort by: Default

Search Installed Templates (Ctrl+E)

Windows Forms Application	Visual C#
WPF Application	Visual C#
Console Application	Visual C#
ASP.NET Web Application	Visual C#
Class Library	Visual C#
Portable Class Library	Visual C#
Silverlight Application	Visual C#
Silverlight Class Library	Visual C#
WCF Service Application	Visual C#
LightSwitch Desktop Application	Visual C#
LightSwitch HTML Application	Visual C#

Type: Visual C#

A project for creating a command-line application

Click here to go online and find templates.

Name: ConsoleApplication1

Location: c:\users\joobrien\documents\visual studio 2013\Projects

Solution name: ConsoleApplication1

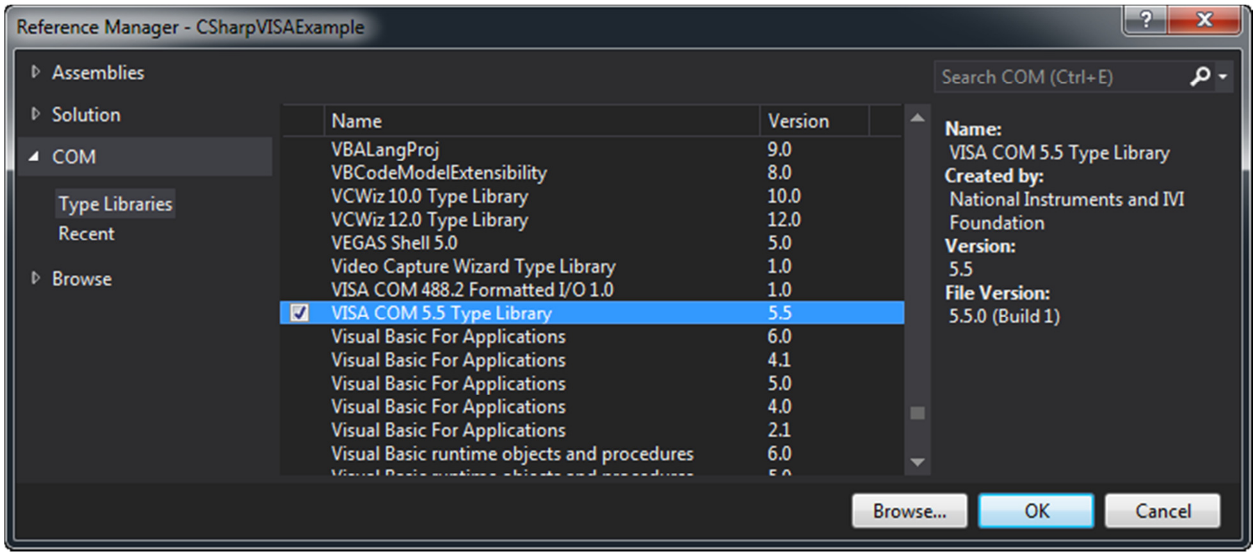
Browse...

☑ Create directory for solution

☐ Add to source control

OK Cancel

To use VISA COM I/O, you need to create a reference to it in your project. To add a COM reference to your project, right click “InstrumentID” in the Solution Explorer and Select Add – Reference. Select “VISA COM 5.5 Type Library.”


Reference Manager - CSharpVISAExample

- Assemblies
- Solution
- COM
  - Type Libraries
  - Recent
- Browse

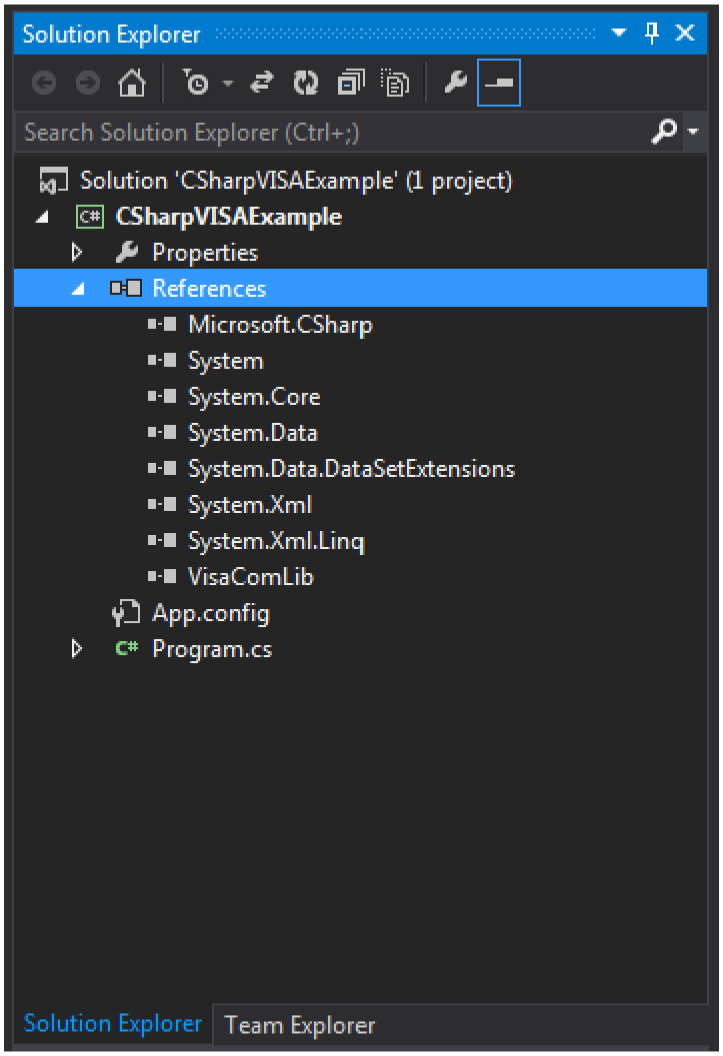
Search COM (Ctrl+E)

	Name	Version
	VBALangProj	9.0
	VBCodeModelExtensibility	8.0
	VCWiz 10.0 Type Library	10.0
	VCWiz 12.0 Type Library	12.0
	VEGAS Shell 5.0	5.0
	Video Capture Wizard Type Library	1.0
	VISA COM 488.2 Formatted I/O 1.0	1.0
☑	VISA COM 5.5 Type Library	5.5
	Visual Basic For Applications	6.0
	Visual Basic For Applications	4.1
	Visual Basic For Applications	5.0
	Visual Basic For Applications	4.0
	Visual Basic For Applications	2.1
	Visual Basic runtime objects and procedures	6.0

Name:
VISA COM 5.5 Type Library
Created by:
National Instruments and IVI Foundation
Version:
5.5
File Version:
5.5.0 (Build 1)

Browse... OK Cancel

The VisaComLib should now show up in the Solution Explorer as follows:



Solution 'CSharpVISAExample' (1 project)
CSharpVISAExample
Properties
References
Microsoft.CSharp
System
System.Core
System.Data
System.Data.DataSetExtensions
System.Xml
System.Xml.Linq
VisaComLib
App.config
Program.cs

Once you have your references to VISA COM in your project, you are ready to create and use VISA COM I/O objects. Included is an example of a simple method that creates a resource and uses the VISA COM 488.2 Formatted I/O component to communicate with a Keysight N9020A Signal Analyzer.

Copy and Paste the following test code into the C# .NET IDE:

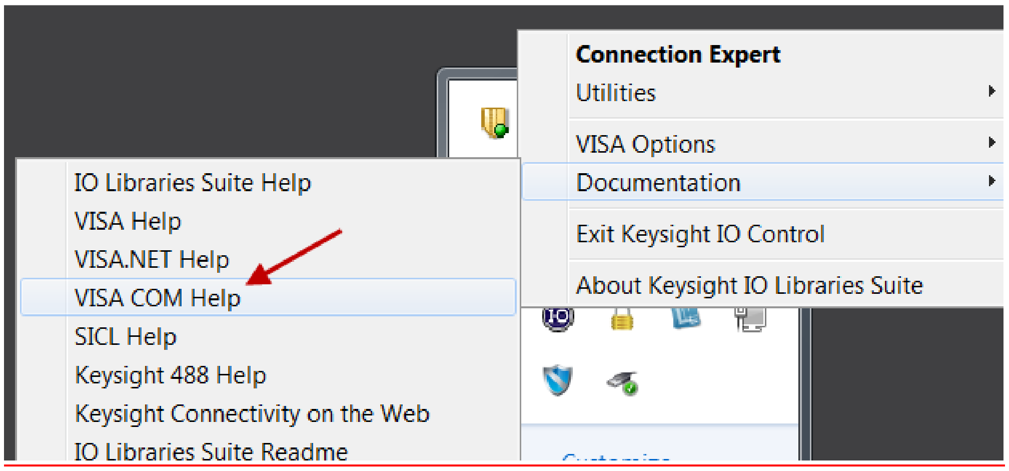
```
using System;
using Ivi.Visa.Interop;

namespace CSharpVISAExample
{
    class Program
    {
        static void Main(string[] args)
        {
            ResourceManager rm;
            FormattedIO488 instrument;
            String result;

            try
            {
                // Get a handle to the default resource manager
                rm = new ResourceManager();
                // Create an IO formatter
                instrument = new FormattedIO488();

                // Open a connection to the instrument and bind
                // it to the formatter. In the below line,
                // replace TCPIP0::.... ::INSTR with your
                // instrument address
                instrument.IO = (IMessage)rm.Open(
                    "TCPIP0::N9020A-80193::inst0::INSTR");

                // Once we're connected, make sure we
                // close things correctly
                try
                {
                    // Send the *IDN? query to the instrument...
                    instrument.WriteString("*IDN?", true);
                    // and read back the response.
                    result = instrument.ReadString();
                    Console.WriteLine("Instrument ID is: " + result);
                }
                finally
                {
                    // Close the connection to the instrument
                    instrument.IO.Close();
                    // and free the reference to the session.
                    instrument.IO = null;
                }
            }
            catch (Exception e)
            {
                Console.Error.WriteLine("Error occurred: " + e.Message);
            }
            Console.WriteLine("Press any key to quit.");
            Console.ReadKey(true);
        }
    }
}
```

The line `rm = new ResourceManager();` creates the Global Resource Manager (GRM), which can instantiate (create) any VISA COM resource installed on the system. Here you see it used to open a TCPIP resource at `"TCPIP0::N9020A-80193::inst0::INSTR"`. The line `instrument = new FormattedIO488();` creates an instance of the 488.2 Formatted I/O Class, which can help with parsing and writing out the data types most instruments use. Setting the IO property of the formatted I/O object prepares the object for reading and writing.

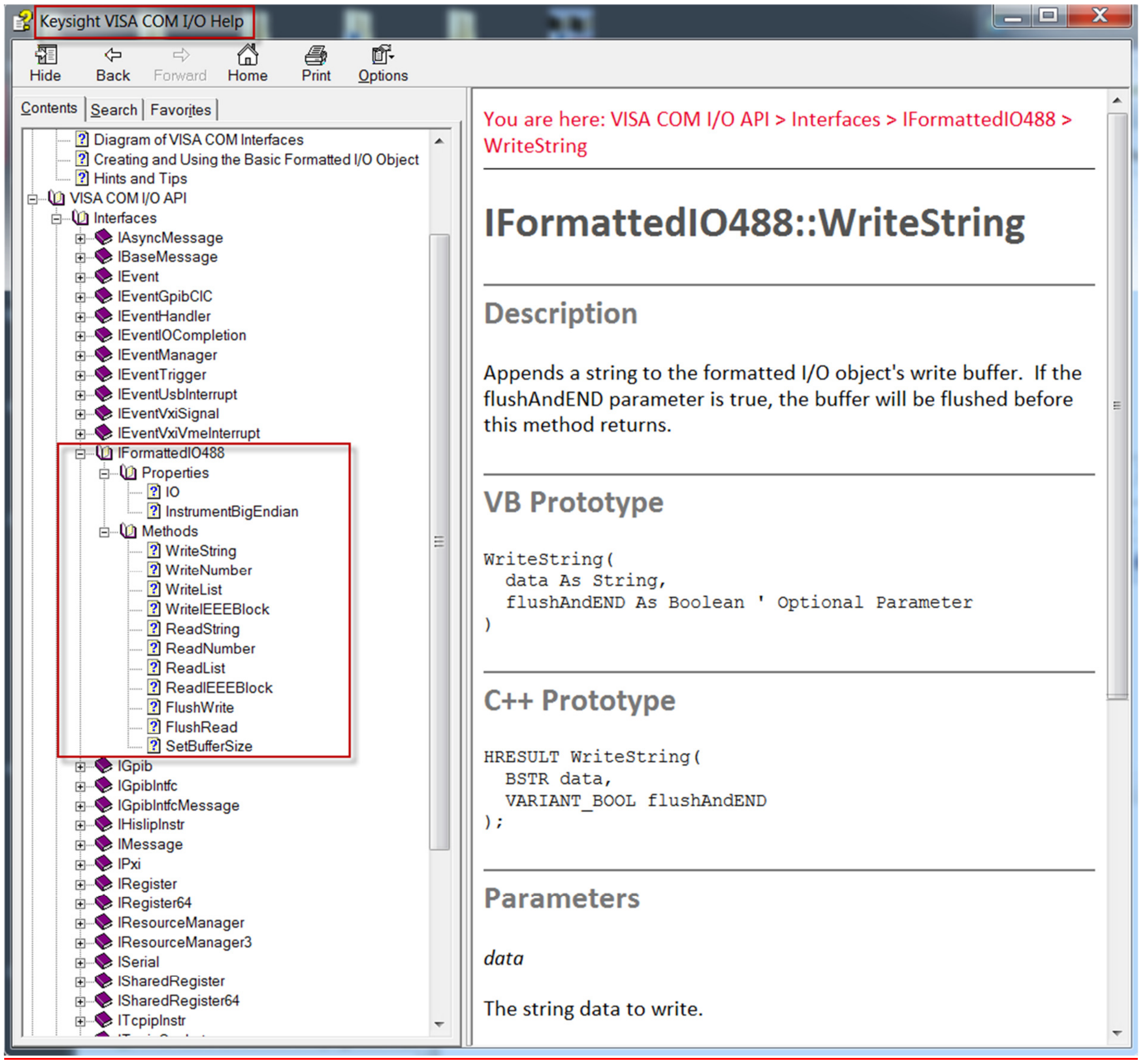
After creating the VISA COM I/O objects to be used, you see a call to WriteString(). This call sends the "*IDN?" string to the instrument. The next line uses the ReadString() method to parse the *IDN?" return value. The method returns a string.
The code in the Finally block is designed to clean up the I/O to be sure that the I/O session is closed immediately, all hardware I/O resources are released, and any valid COM objects are released. Call the Close() method on the Keysight VISA COM Formatted I/O session to cause the session to release any hardware I/O resources.

Additional Test and Measurement Resources

Finding the VISA COM documentation if you have Keysight IO Libraries Installed.



Common methods used in the VISA COM class



- Diagram of VISA COM Interfaces
- Creating and Using the Basic Formatted I/O Object
- Hints and Tips
- VISA COM I/O API
  - Interfaces
    - IAsyncMessage
    - IBaseMessage
    - IEvent
    - IEventGpibCIC
    - IEventHandler
    - IEventIOCompletion
    - IEventManager
    - IEventTrigger
    - IEventUsbInterrupt
    - IEventVxiSignal
    - IEventVxiVmeInterrupt
    - IFormattedIO488
      - Properties
        - IO
        - InstrumentBigEndian
      - Methods
        - WriteString
        - WriteNumber
        - WriteList
        - WriteIEEEBlock
        - ReadString
        - ReadNumber
        - ReadList
        - ReadIEEEBlock
        - FlushWrite
        - FlushRead
        - SetBufferSize
    - IGpib
    - IGpibIntfc
    - IGpibIntfcMessage
    - IHislipInstr
    - IMessage
    - IPxi
    - IRegister
    - IRegister64
    - IResourceManager
    - IResourceManager3
    - ISerial
    - ISharedRegister
    - ISharedRegister64
    - ITcpipInstr

You are here: VISA COM I/O API > Interfaces > IFormattedIO488 > WriteString

IFormattedIO488::WriteString

Description

Appends a string to the formatted I/O object's write buffer. If the flushAndEND parameter is true, the buffer will be flushed before this method returns.

VB Prototype

```
WriteString(
  data As String,
  flushAndEND As Boolean ' Optional Parameter
)
```

C++ Prototype

```
HRESULT WriteString(
  BSTR data,
  VARIANT_BOOL flushAndEND
);
```

Parameters

data

The string data to write.