Using the TekScope IVI-COM Driver from C# .NET

Introduction

This document describes the step-by-step procedure for using the TekScope IVI-COM driver from a .NET environment using C#. Microsoft .Net supports COM interoperability allowing .Net components to easily call COM components.

In this simple exercise, you will learn how to import the TekScope IVI-COM driver from the C# environment and to generate “Runtime Callable Wrappers”. You will also learn how to use these wrappers to build a simple user interface (UI) to connect your computer to an oscilloscope and get current record length.

Requirements

The following software must be installed on your oscilloscope:

- TekVISA.
- IVI shared components.
- TekScope IVI-COM driver.
- Visual Studio .NET.

Step-by-step Development

Step 1: Create the new C# project

- Open Visual Studio .NET and, from the **Start** page, choose **New Project**.
- Select **Visual C# Project** from the tree view on the left.
- Select **Windows Application** as the project template.
- Set the name of the application to *TekScopeIVITest*, as shown in Figure 1, and click **OK** to create the project.

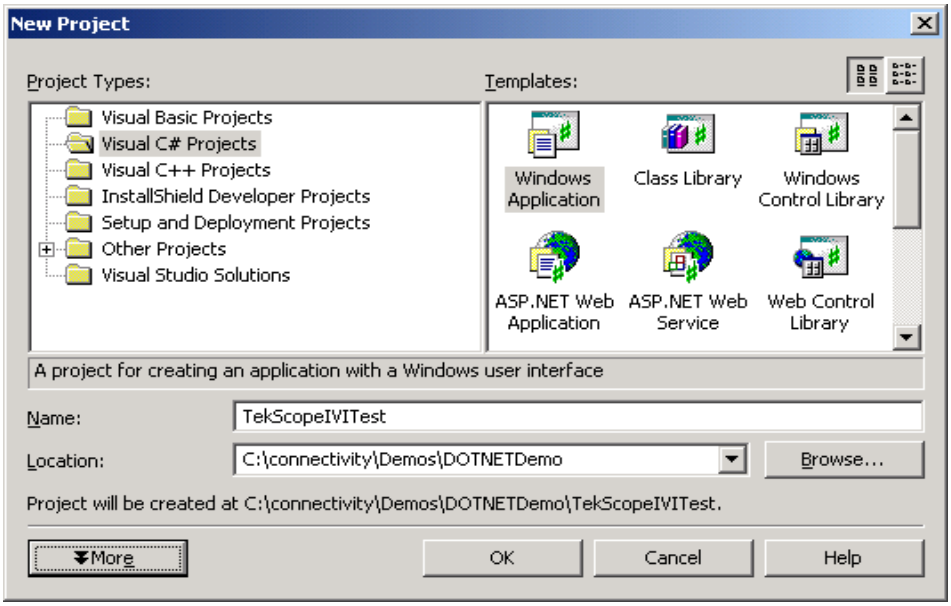

Figure 1: Start the new project

Step 2: Design the Form

From the "Solution Explorer" window, double click on the Form1.cs file to open the default form in the editor. Add two buttons to the form, as shown in Figure 2, and set the properties of those button controls, as given in Table 1.

Table 1: Button Control Properties

Control Type	Property	Value
Button	Name	button1
	Text	Record length?
Button	Name	button2
	Text	Close

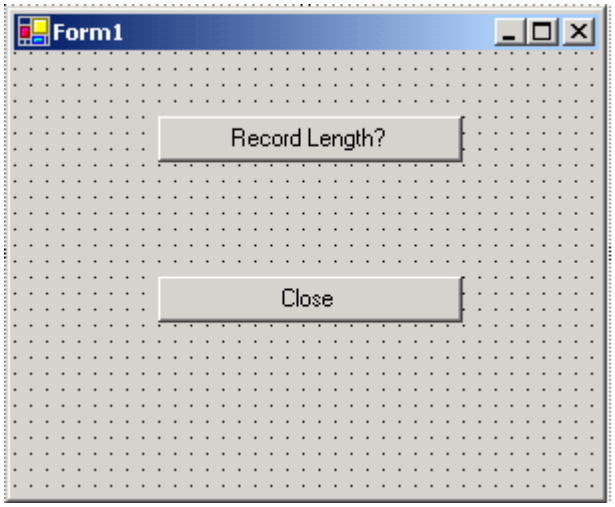

Figure 2: Form with two buttons

Step 3: Generate the Runtime Callable Wrappers for the driver

In this step you will generate the Runtime Callable Wrappers (RCWs) for the TekScope IVI-COM driver. Visual studio .Net provides two different ways to generate this. One is using the *tlbimp.exe* tool and the other is using the studio **Project** > **Add Reference…** option. Here we will use the second method.

- Click **Project**, and then click **Add Reference**.
- In the **Add Reference** dialog box, click the **COM** tab.
- Search for the IVI TekScope driver from the list and click **Select,** as shown in Figure 3. The selected components will be added to the lower listview in the dialog box.
- Click **OK** to create RCWs for the driver in your Visual C# .NET project.

When you do this, you'll find that Visual C# .NET actually creates a DLL in your project's /Bin folder, with a name *Interop.TekScope_TektronixLib.dll*. This dll is a .Net assembly containing RCWs for the driver.

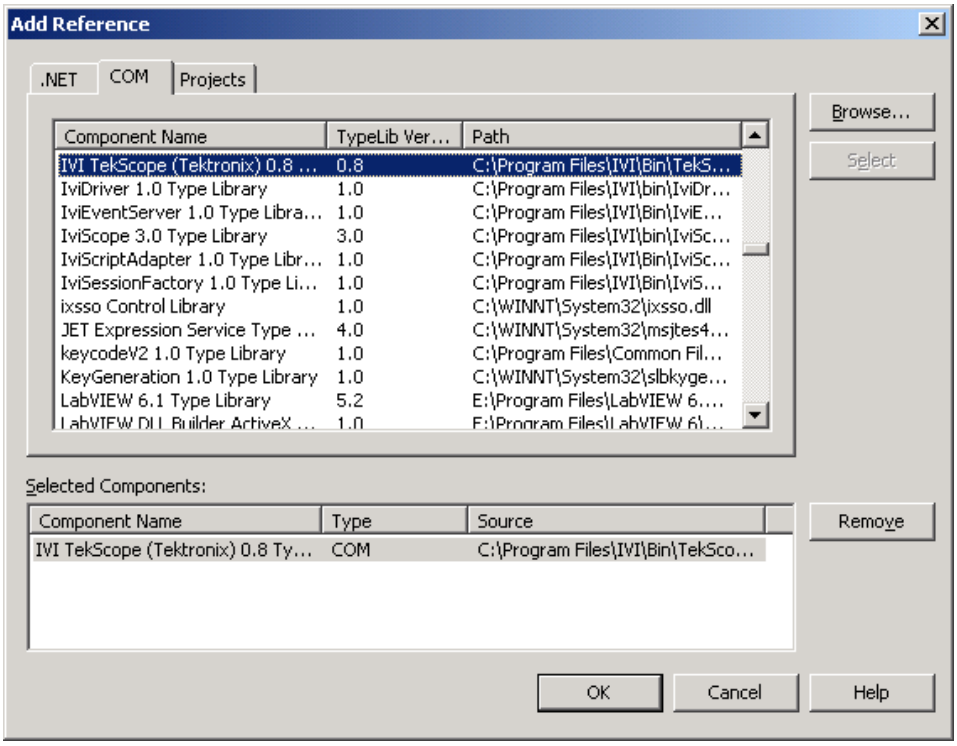

Figure 3: Search for the IviScope driver

Step 4: Write the code to use driver

Now you're ready to write code that uses the driver methods and properties available in the *TekScope_TektronixLib* class. Click **View**, click **Code**, and paste this code, as shown below, just after *Main()* function body.

```
//Declare a global driver object
private TekScope_TektronixLib.TekScope myScope;


private void button1_Click(object sender, System.EventArgs e)
{
        //Create the driver object
        myScope= new TekScope_TektronixLib.TekScopeClass();
        //Initialize the driver by connecting it to "GPIB8::1::INSTR"
        myScope.Initialize("GPIB8::1::INSTR",false,false,"");
        //Get the current record length
        int RecLen= myScope.Horizontal.RecordLength;
        //Format the value and display in a message box
        MessageBox.Show("Record Length is:"+RecLen.ToString());
}


private void button2_Click(object sender, System.EventArgs e)
{
        //Close the driver
        myScope.Close();
        //Terminate the application
        Application.Exit();
}
```

Note:

- Precede each line of C# code with a line of comment describing the functionality of the code.
- If you would like to generate your own code, then you can double click on the buttons and add code inside the function body.

- In the given sample code, the *Initialize()* function connects to *GPIB8::1::INSTR*. You can modify this depending on your VISA resource name for your oscilloscope.
- Visual Studio Intellisense will help you to browse through all properties and methods of the driver. Figure 4 shows C#'s Intellisense in action.

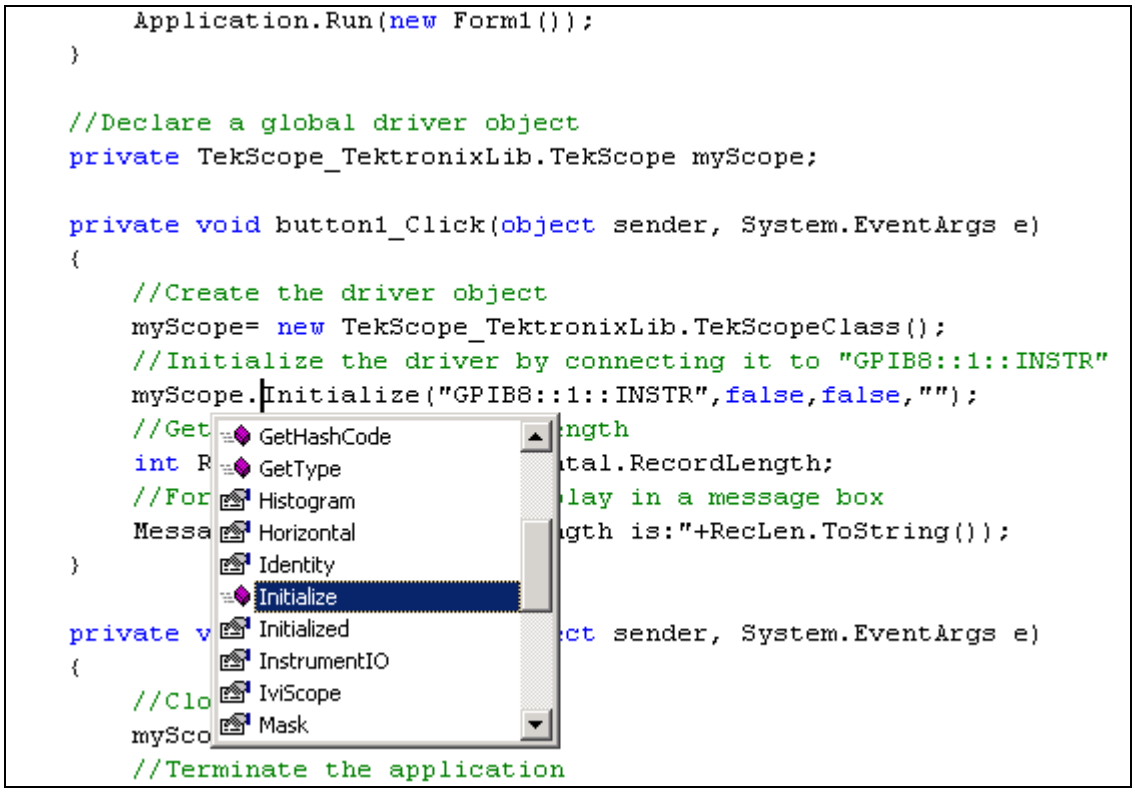

Figure 4: Intellisense in action

Step 5: Run the application

- Save the project using **File** > **Save all** menu.
- Build the application using **Build** > **Build Solution** menu.
- Run the application using **Debug** > **Start** menu. Figure 5 shows the application in running mode.

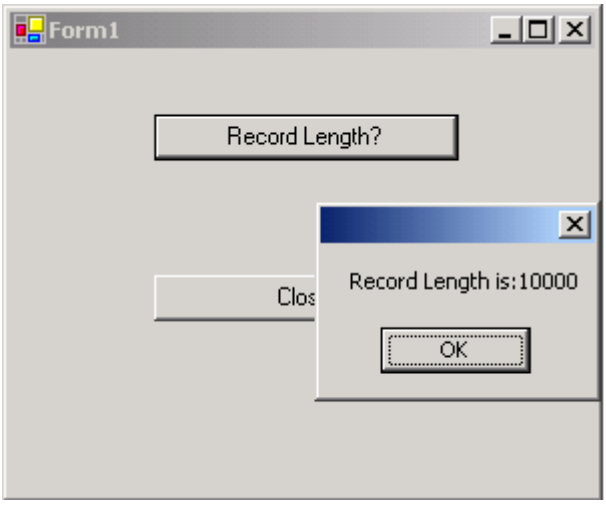

Figure 5: The application in running mode

Conclusion

You can also use other driver functions from C# and Visual Basic .Net.