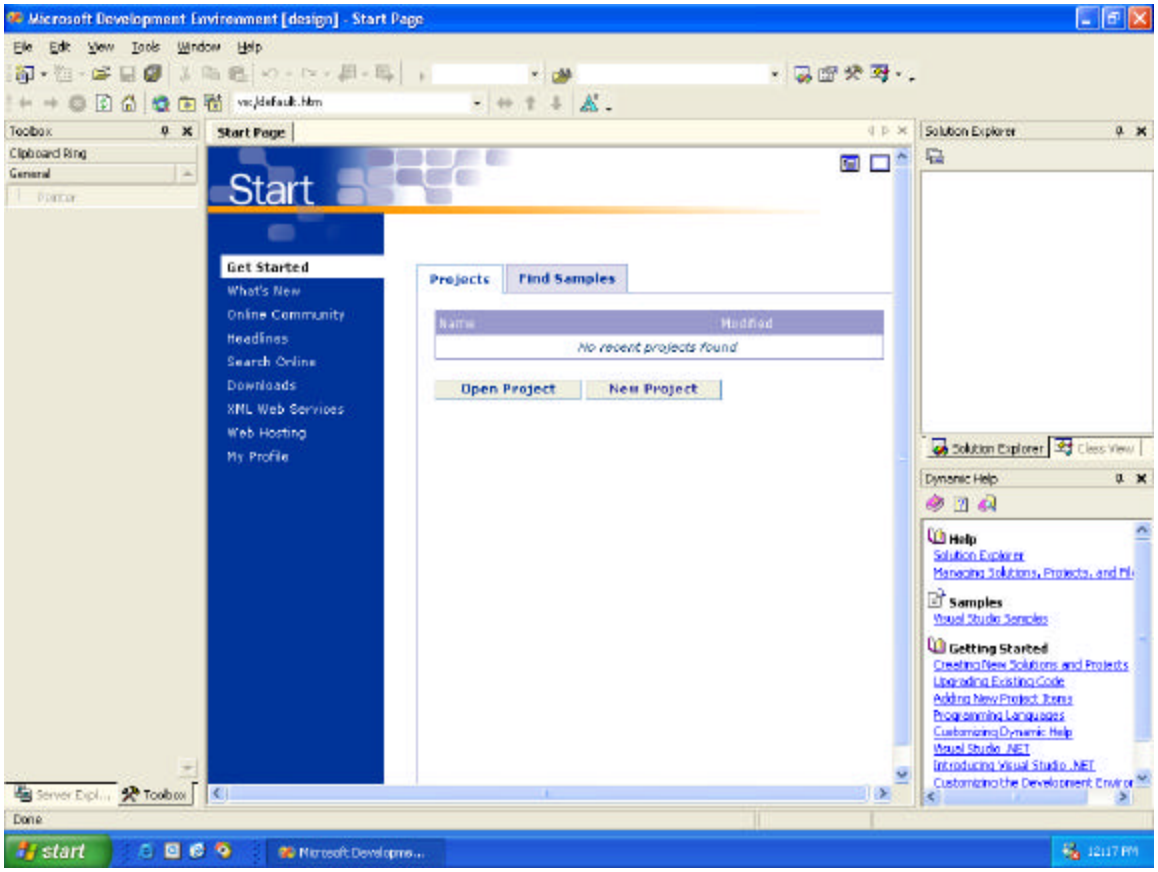
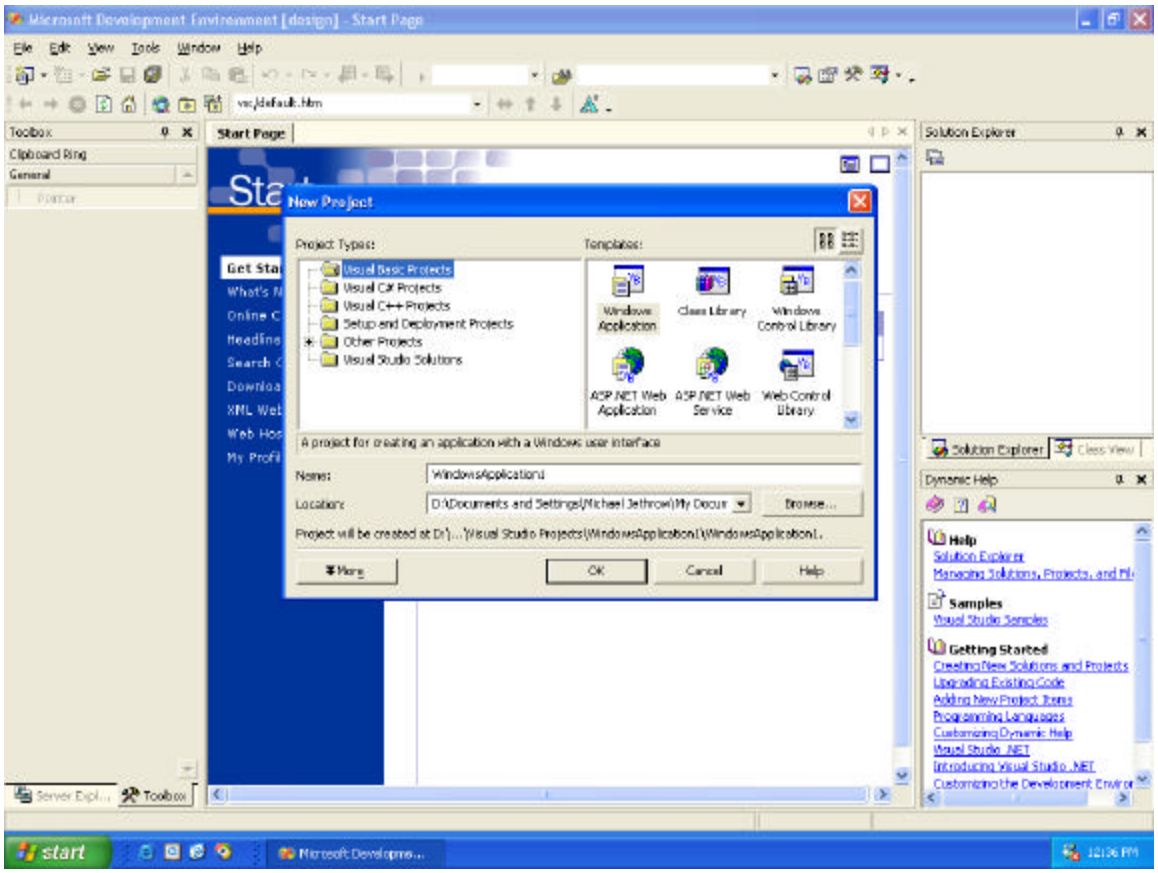
How to use Visual Basic.NET with DriverLINX

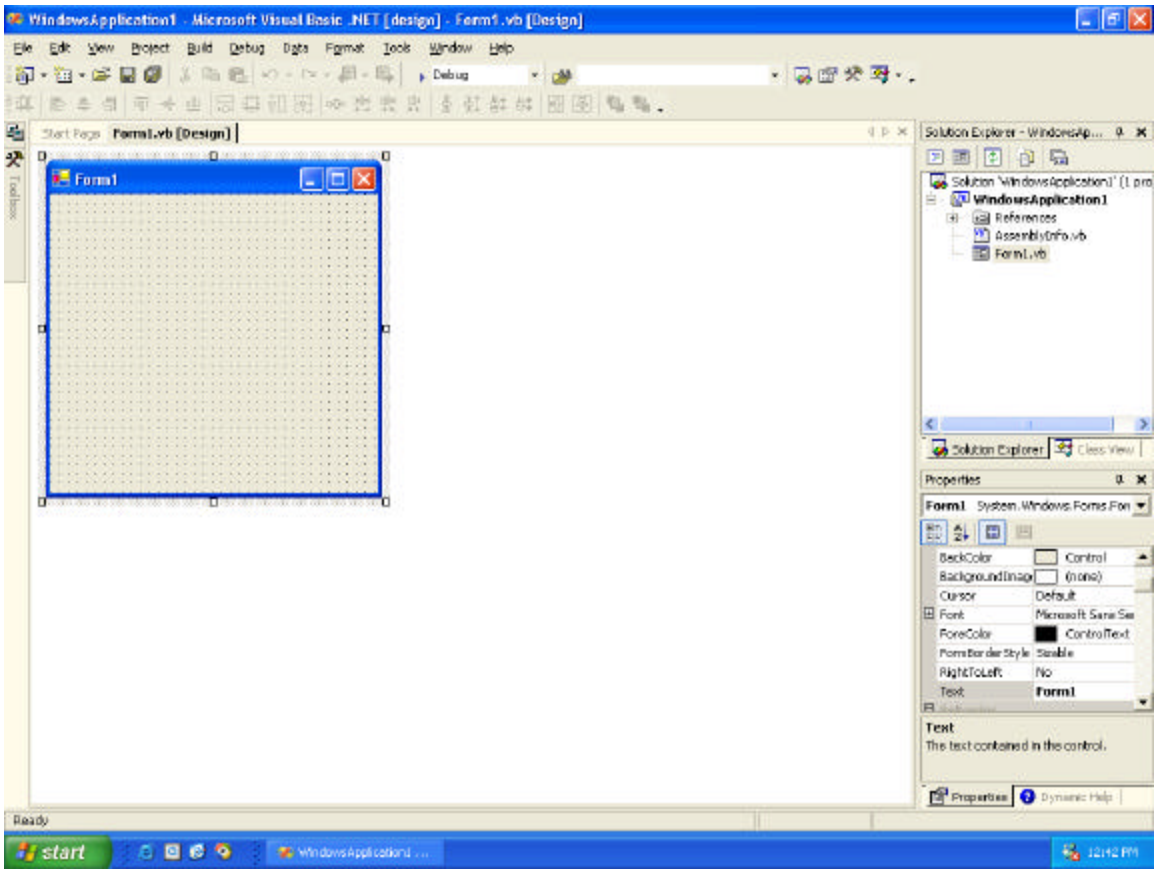
When you launch Microsoft VS.NET, you'll see the screen below:



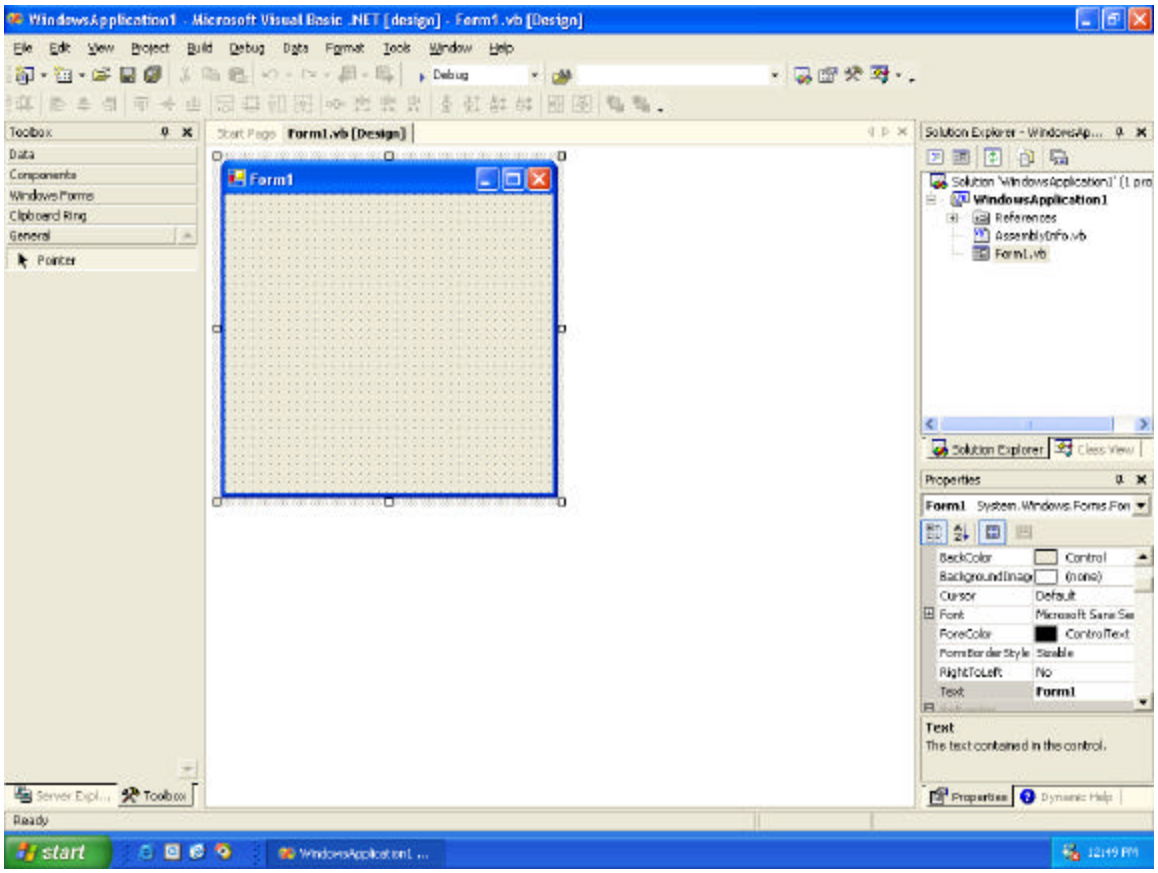
Click New Project. Choose the appropriate project for the language desired. In this case, we'll choose the Visual Basic Projects. Select the "Windows Application" for the Template type and then click OK.



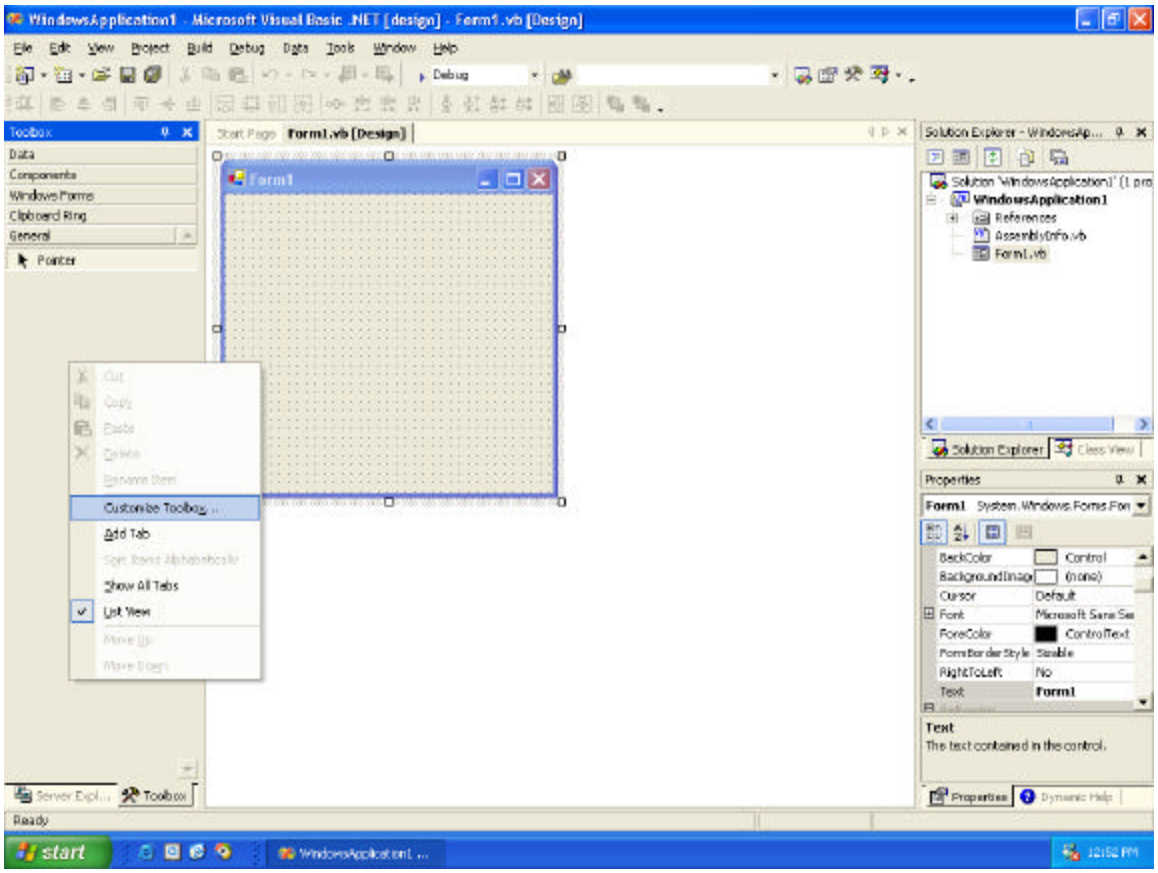
You should see the screen below:



Move cursor over Toolbox menu along the left hand side and pin it down (thumb tack). Select the General tab so the screen looks like below:



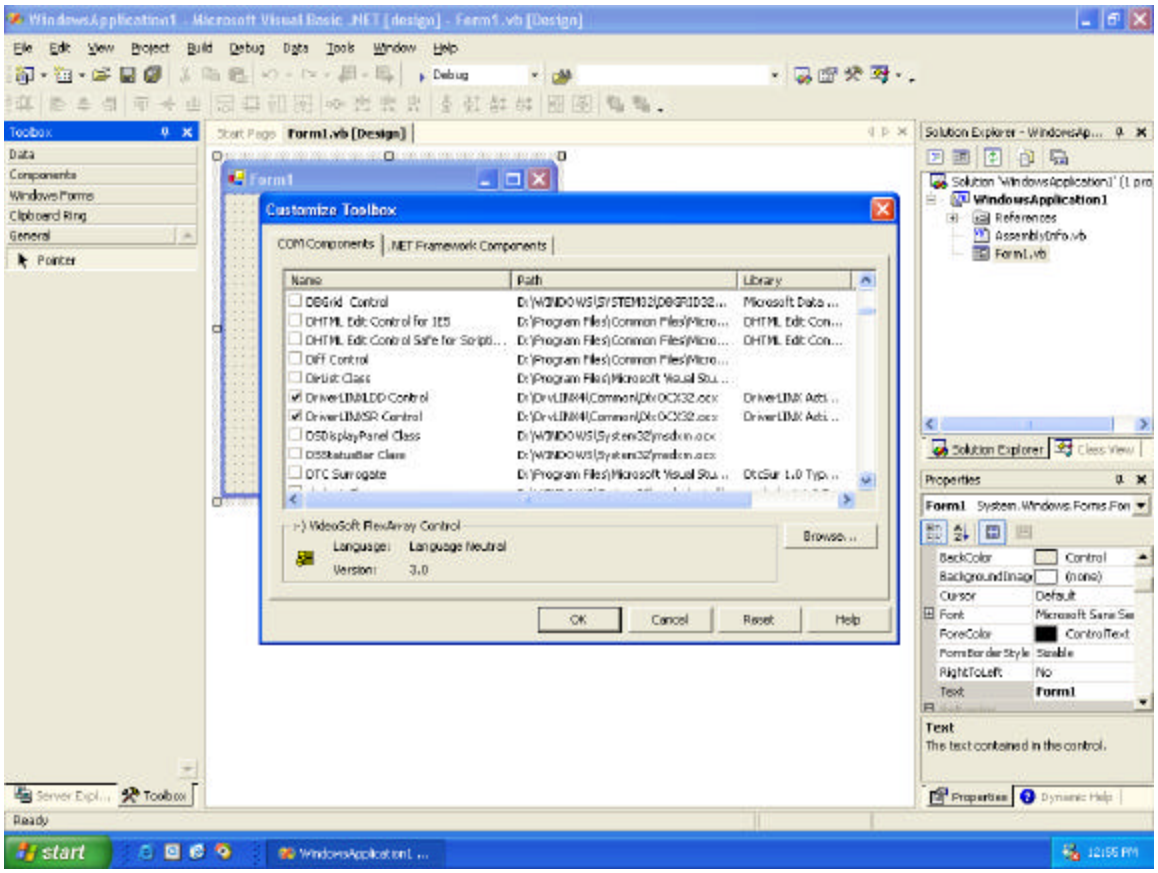
In order to use DriverLINX, the ActiveX controls need to be added to the Visual Basic project. To do this, move cursor over the Toolbox and click the right mouse button. Choose Customize Toolbox from the pop-up menu.



An alphabetical listing of all the available ActiveX controls will be displayed. Scroll down to locate and select the DriverLINXSR Control and DriverLINXLDD Control.

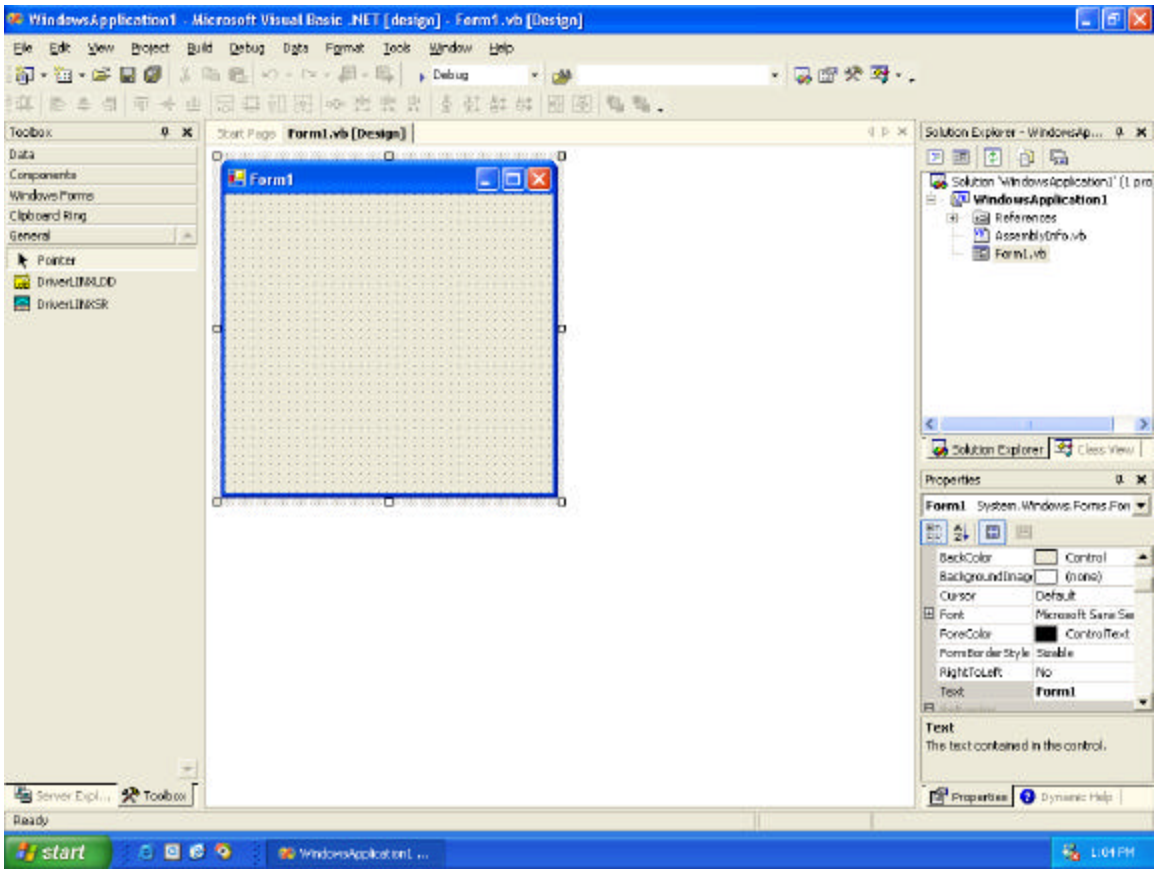
NOTE: The DriverLINXSR Control is required for application development. This is the Service Request control and is used to send and receive data to/from the board. The DriverLINXLDD Control is not required for a DriverLINX application. It is the Logical Device Descriptor and can be used to interrogate the features of your hardware such as how many Analog Input channels it has. Inclusion of this control in this example project is only shown for reference.

Make sure there is a check in front of the DriverLINXSR control and optionally in front of the DriverLINXLDD control too. Click the OK button.

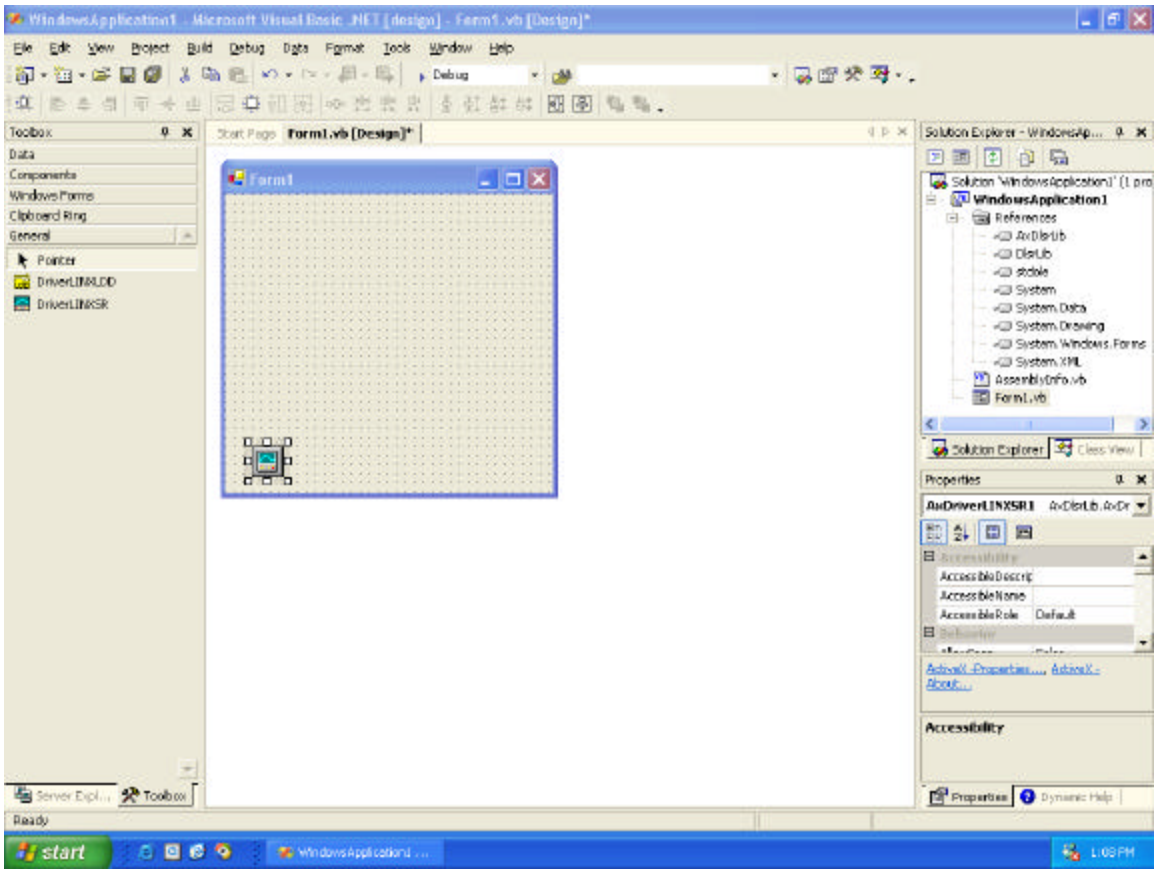


Notice two new ActiveX controls appear on the Toolbox.

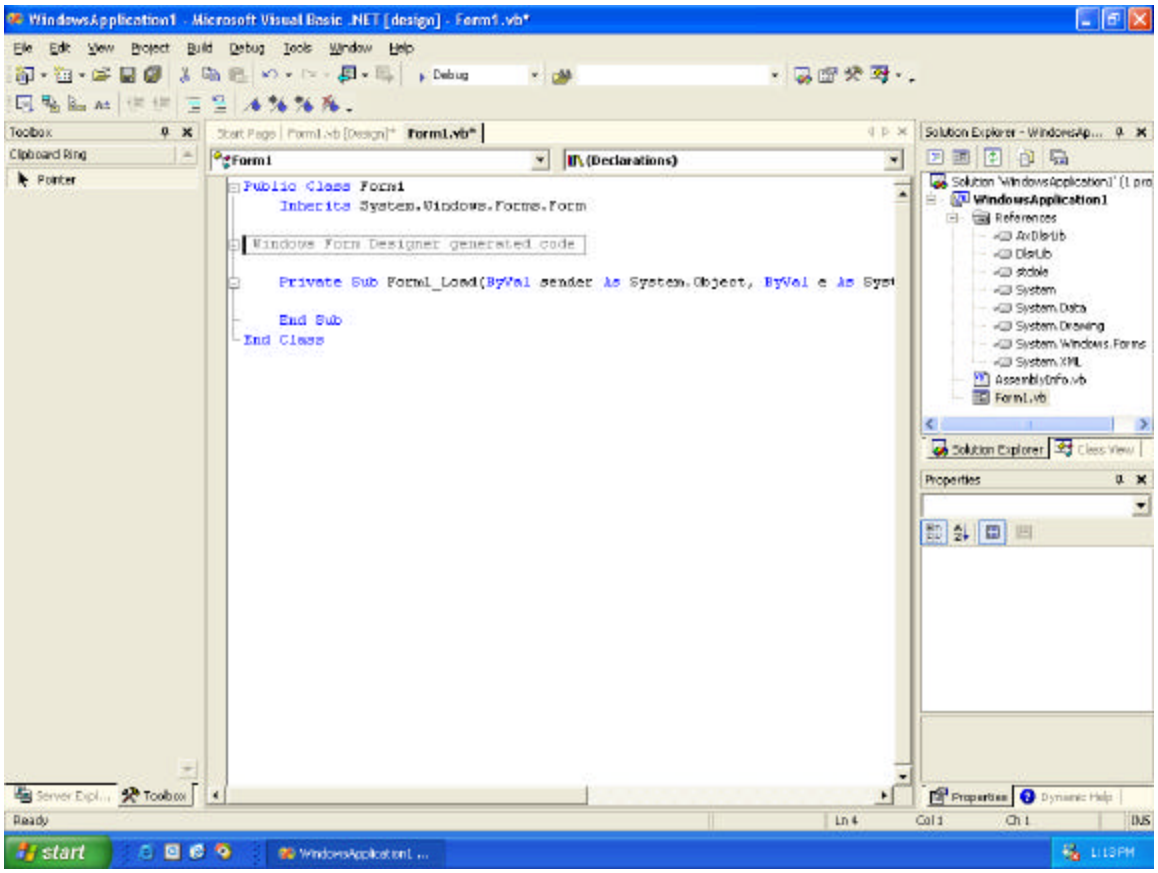
NOTE: if you did not select the General Tool box before this procedure, the controls will appear at the bottom of what ever toolbox was selected at that time.



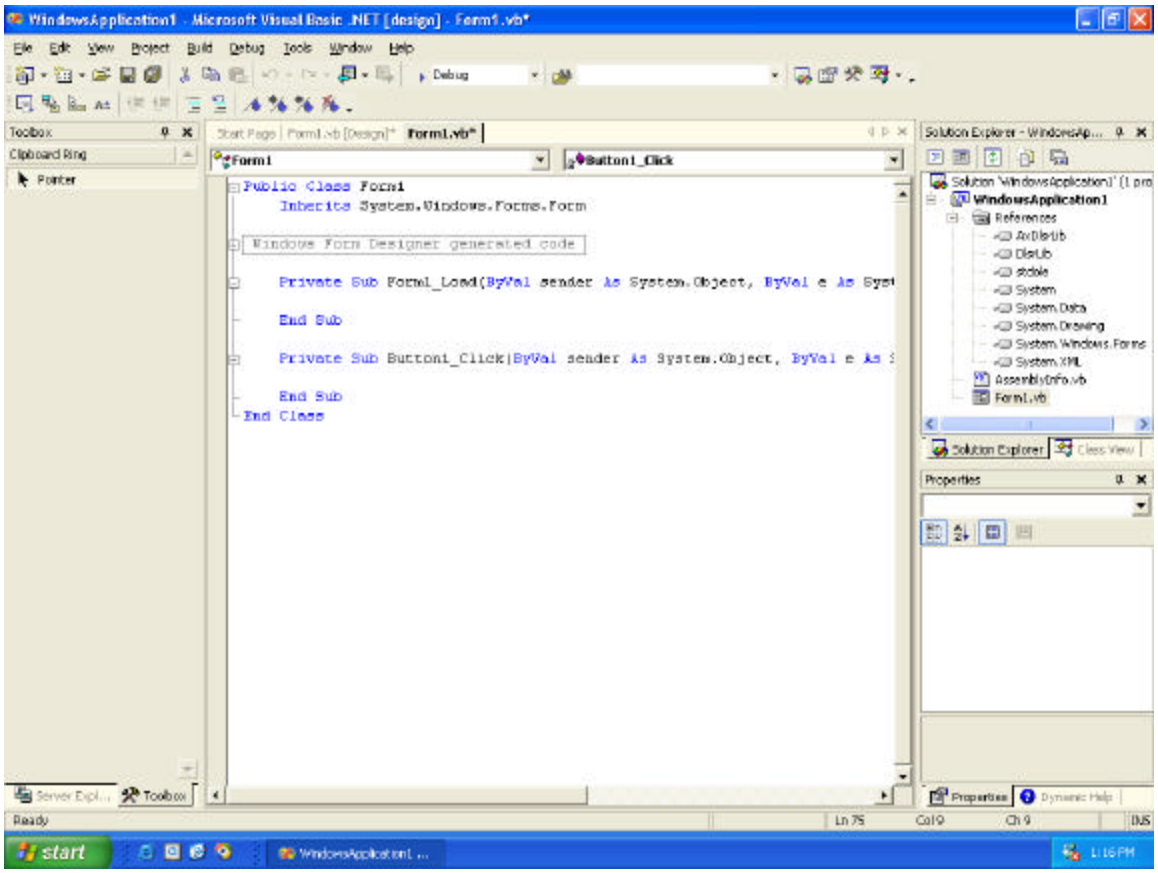
Load the Service Request control onto the form: click DriverLINXSR on the Toolbox and then click on the form.



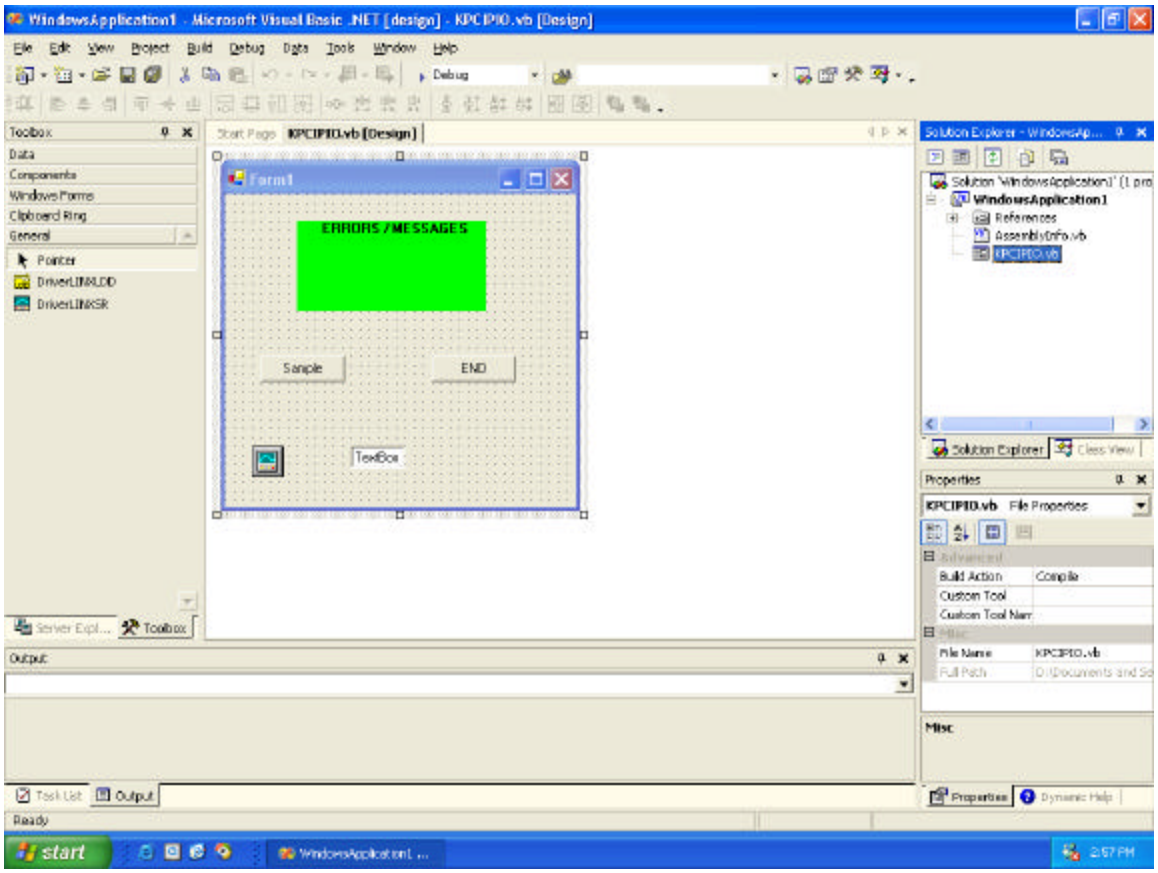
Now you can add buttons, textboxes, etc. in the same manner that you may already be familiar with from earlier versions of Visual Basic. To begin coding at Form Load, double click on Form1.



To get the click event procedure for a command button, add a Button to the form and double click on the button:



Here’s the resulting form for sampling and displaying one 8-bit digital port:



Here's the code that was used:

```
    Private Sub Form1_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load
        With AxDriverLINXSR1
            .Req_device = 0
            .Req_DLL_name = "kpcipio.dll"
            .Req_subsystem = DlsrLib.DL_SubsystemConstants.DL_DEVICE
            .Req_op = DlsrLib.Req_opConstants.DL_INITIALIZE
            .Req_mode = DlsrLib.Req_modeConstants.DL_OTHER
            .CtlRefresh()
            Label1.Text = .Message
        End With
    End Sub

    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button1.Click
        With AxDriverLINXSR1
            .Req_subsystem = DlsrLib.DL_SubsystemConstants.DL_DI
            .Req_mode = DlsrLib.Req_modeConstants.DL_POLLED
            .Req_op = DlsrLib.Req_opConstants.DL_START
            .Evt_Stp_type = DlsrLib.Evt_xxx_typeConstants.DL_NULLEVENT
            .Evt_Str_type = DlsrLib.Evt_xxx_typeConstants.DL_NULLEVENT
            .Evt_Tim_type = DlsrLib.Evt_xxx_typeConstants.DL_NULLEVENT
            .Sel_chan_format = DlsrLib.FormatConstants.DL_tNATIVE
            .Sel_chan_N = 1
            .Sel_chan_start = 0
            .Sel_chan_startGainCode = 0
            .Sel_buf_N = 0
            .CtlRefresh()
            TextBox1.Text = Str(.Res_Sta_ioValue)
            Label1.Text = .Message
        End With
    End Sub

    Private Sub Button2_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button2.Click
        AxDriverLINXSR1.Req_DLL_name = ""
        End
    End Sub
```