

Virtual Serial Port Cookbook Errata 4/21/08

You can help...

You can help the community of folks reading this book by finding errors and sending them to error@smileymicros.com so that the author can put those errors in this errata section. Corrections will be included in later editions.

Typos:

Page 9, 1st paragraph: or is --> or if

Page 12, 3rd line: F232R to FT232R

Page 22 SimpleTerminalby needs a space

Page 47 the list of ComboBoxes truncates after the 'Baudrate'
It should include DataBit, Parity, Stop Bits, and Handshaking ComboBoxes as shown in the illustration on page 48.

Page 48 the word directory is omitted from the first sentence. Where it says "to the containing" it should say "to the directory containing".

Page 48 is unclear in the reference to the 'Imports' list. You create this list yourself. You can see how this is done in the source code for Chapter 5 available on our website.

Page 80 that we get [from] Windows

Page 97 tread --> thread

Page 50 immediately after the C# code insert:

```
#region Get Port Name
// We don't use the GetPortNames method of the SerialPort class
// because it doesn't work right the documentation says it reads
// a register key and if that key ain't right then the results
// ain't right. In my test case the key had an orphan port name in it,
// so I reverted to the DeviceInfo class that I'd made earlier before
// the SerialPort stuff came along.
//
// Oh, and DON'T LOOK AT the DeviceInfo class and after you do
// you'll see why I said don't
private static string[] GetPorts()
{
    string[] strArray = DeviceInfo.parseFriendlyPorts();

    return strArray;
}
```

```

// Select the port from the list box
private void listBoxPorts_SelectedIndexChanged(object sender, EventArgs
e)
{
    GetCOM(listBoxPorts.SelectedIndex);
}

// Set the selected port and display it in the label
private void GetCOM(int index)
{
    string[] strArray = DevInfo.DeviceInfo.parsePorts();

    if (strArray[index] != "")
    {
        SelectedPort = strArray[index];
        labelPort.Text = "Selected Port = " + SelectedPort;
    }
}
#endregion

```

Page 51 immediately after the VB code insert:

```

#Region "Get Port Name"
' We don't use the GetPortNames method of the SerialPort class
' because it doesn't work right the documentation says it reads
' a register key and if that key ain't right then the results
' ain't right. In my test case the key had an orphan port name in it,
' so I reverted to the DeviceInfo class that I'd made earlier before
' the SerialPort stuff came along.
'
' Oh, and DON'T LOOK AT the DeviceInfo class and after you do
' you'll see why I said don't
Private Shared Function GetPorts() As String()
Dim strArray As String() = DevInfo.DeviceInfo.ParseFriendlyPorts()
Return strArray
End Function

' Select the port from the list box
Private Sub listBoxPorts_SelectedIndexChanged(ByVal sender As Object,
wrap
ByVal e As EventArgs) Handles listBoxPorts.SelectedIndexChanged
GetCOM(listBoxPorts.SelectedIndex)
End Sub

' Set the selected port and display it in the label
Private Sub GetCOM(ByVal index As Integer)
Dim strArray As String() = DevInfo.DeviceInfo.ParsePorts()
If strArray(index) <> "" Then
SelectedPort_Renamed = strArray(index)
labelPort.Text = "Selected Port = " & SelectedPort_Renamed
End If
End Sub
#End Region

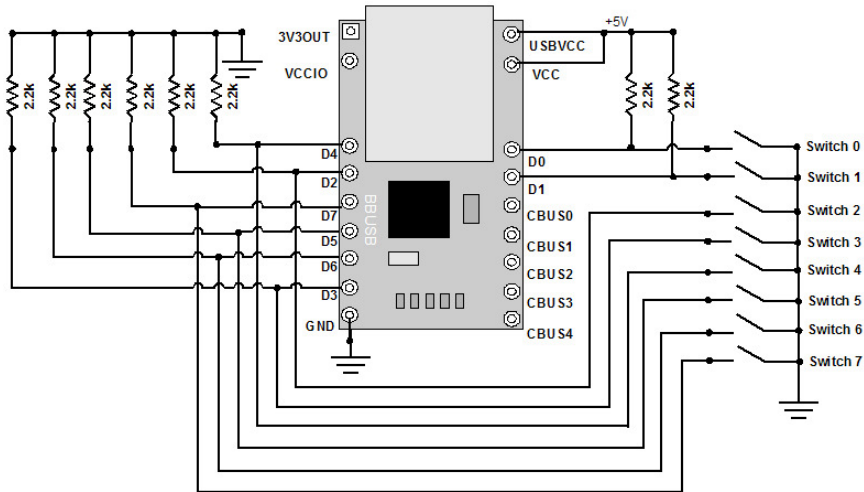
```

Error in pictures:

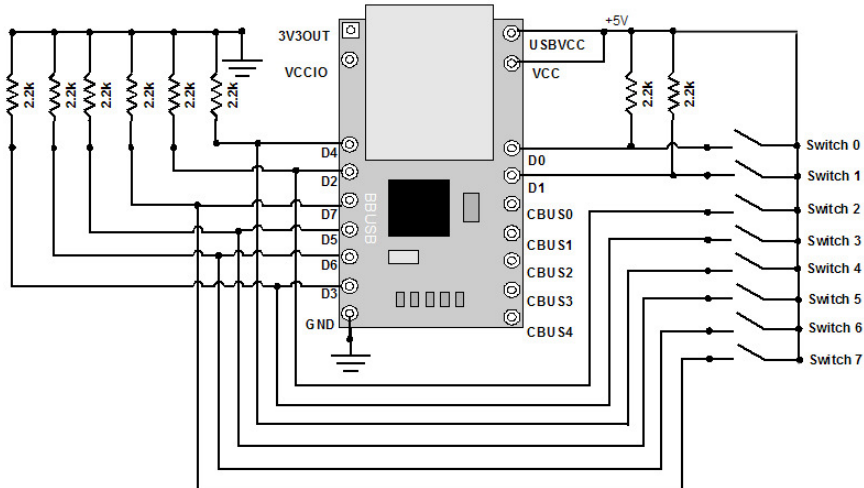
Page 19: The two photos show a 22uF capacitor (cylinder on the left power strip) that is not included in the kit and not needed for the experiment.

Page 320: upper picture schematic shows the switches connected to GND, they should actually be connected to +5V as shown in the lower photo.

Incorrect picture:



Correct picture:



The whole receive region has been changed except for the last two functions, but I've included the whole region to simplify things:

C# Version:

```
#region Receive functions

// we want to have the serial port thread report back data received, but to display
// that data we must create a delegate function to show the data in the richTextBox

string InputData = String.Empty;
delegate void SetTextCallback(string text);

// note that this function runs in a separate thread and thus we must use a delegate in
// order to display the results in the richTextBox.
private void serialPort1_DataReceived(object sender, SerialDataReceivedEventArgs e)
{
    if (ReceiveASCII)
    {
        serialPort1.Encoding = System.Text.Encoding.ASCII;
    }
    else
    {
        serialPort1.Encoding = System.Text.Encoding.Unicode;
    }

    InputData = serialPort1.ReadExisting();
    if (InputData != String.Empty)
    {
        SetText(InputData);
    }
}

private void SetText(string text)
{
    if (richTextBoxReceive.InvokeRequired)
    {
        SetTextCallback d = new SetTextCallback(SetText);
        this.Invoke(d, new object[] { text });
    }
    else
    {
        if (ReceiveASCII) // Show in ASCII
        {
            this.richTextBoxReceive.Text += text;
        }
        else // Show in HEX
        {
            Encoding unicode = Encoding.Unicode;

            // Convert the string into a byte[].
            byte[] unicodeBytes = unicode.GetBytes(text);

            foreach (byte b in unicodeBytes)
            {
                if (b > 15)
                {
                    richTextBoxReceive.Text += "0x" + b.ToString("X") + ",";
                }
                else richTextBoxReceive.Text += "0x0" + b.ToString("X") + ",";
            }
        }
        ReceiveCount += text.Length;
        textBoxReceiveCount.Text = ReceiveCount.ToString();
    }
}
}
```

```

// This rigaramole is needed to keep the last received item displayed
// it kind of flickers and should be fixed
private void richTextBoxReceive_TextChanged(object sender, System.EventArgs e)
{
    moveCaretToEnd();
}

private void moveCaretToEnd()
{
    richTextBoxReceive.SelectionStart = richTextBoxReceive.Text.Length;
    richTextBoxReceive.SelectionLength = 0;
    richTextBoxReceive.ScrollToCaret();
}

#endregion

```

VB Version:

```

#Region "Receive functions"

' we want to have the serial port thread report back data received, but to display
' that data we must create a delegate function to show the data in the richTextBox

Private InputData As String = String.Empty
Delegate Sub SetTextCallback(ByVal text As String)

' note that this function runs in a separate thread and thus we must use a delegate in
' order to display the results in the richTextBox.
Private Sub serialPort1_DataReceived(ByVal sender As Object, ByVal e As
SerialDataReceivedEventArgs)
    If ReceiveASCII Then
        serialPort1.Encoding = System.Text.Encoding.ASCII
    Else
        serialPort1.Encoding = System.Text.Encoding.Unicode
    End If

    InputData = serialPort1.ReadExisting()
    If InputData <> String.Empty Then
        SetText(InputData)
    End If
End Sub

Private Sub SetText(ByVal text As String)
    If richTextBoxReceive.InvokeRequired Then
        Dim d As SetTextCallback = New SetTextCallback(AddressOf SetText)
        Me.Invoke(d, New Object() { text })
    Else
        If ReceiveASCII Then ' Show in ASCII
            Me.richTextBoxReceive.Text += text
        Else ' Show in HEX
            Dim [unicode] As Encoding = Encoding.Unicode

            ' Convert the string into a byte[].
            Dim unicodeBytes As Byte() = [unicode].GetBytes(text)

            For Each b As Byte In unicodeBytes
                If b > 15 Then
                    richTextBoxReceive.Text &= "0x" & b.ToString("X") & ", "
                Else
                    richTextBoxReceive.Text &= "0x0" & b.ToString("X") & ", "
                End If
            Next b
        End If
        ReceiveCount += text.Length
        textBoxReceiveCount.Text = ReceiveCount.ToString()
    End If
End Sub

' This rigaramole is needed to keep the last received item displayed

```

```
' it kind of flickers and should be fixed
Private Sub richTextBoxReceive_TextChanged(ByVal sender As Object,
                                         ByVal e As System.EventArgs)
    moveCaretToEnd()
End Sub

Private Sub moveCaretToEnd()
    richTextBoxReceive.SelectionStart = richTextBoxReceive.Text.Length
    richTextBoxReceive.SelectionLength = 0
    richTextBoxReceive.ScrollToCaret()
End Sub

#End Region
```