EE 354 **October 14, 2009**
Notes on Serial I/O using C#

This is an example of a program which uses C# to communicate with the AT89C51AC3 board by way of the serial port at 19,200 baud. There are two programs to write: the first is in C# using Visual Studio and the second is in C using Keil µVision.

The C# program
Begin by creating a new project using Windows C# as a *forms application.* Create a form similar to that shown in Figure 1 which has a textbox named txtPort, three buttons named btn1, btn2, and btn3, an exit button named btnExit, and a serial port object named sptCom1. The serial port object does not show up on the form but is listed just below it on the form screen. The properties for the serial port should be taken as the defaults except for the name and the baud rate as shown in Figure 2.

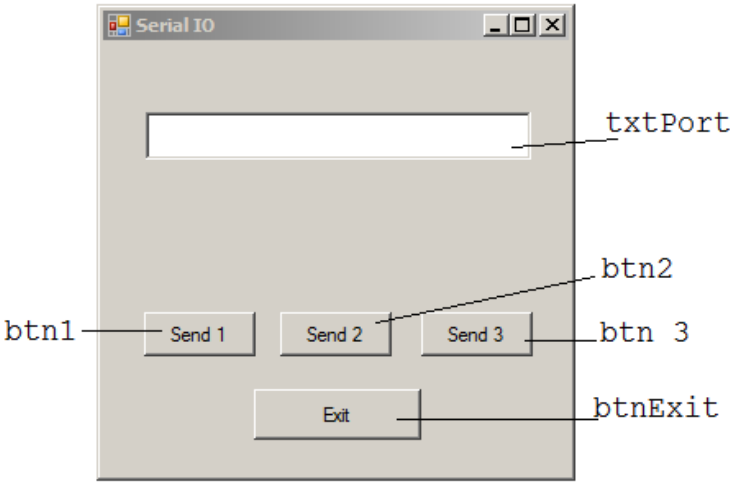

Figure 1
C# form for serial port project

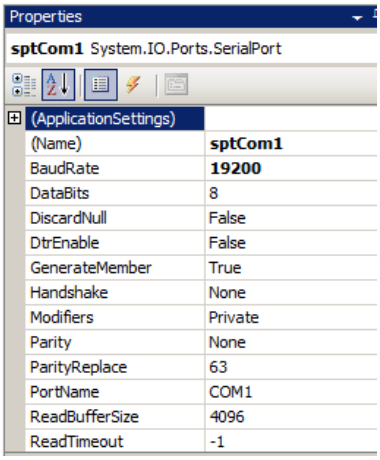

Figure 2
Set baud rate and name for the serial port in the properties window.

The listing for the C# code is given on the following page. The comments below refer to the line numbers in this listing:

1. Line 19 creates a delegate for the text box. The serial port is created on a different thread from the main program so the two cannot share a text box without creating this delegate that allows it to be done safely.
2. Line 20 creates a character buffer that is used for writing data to the serial port.
3. Lines 22 to 24 initialize all of the components and open the serial port. This runs when the program begins.
4. Lines 25 to 36 are the button events. btn1, btn2, and btn3 write to the serial port using the write command. The first argument is the buffer with the data, the second is the beginning element in the buffer and the third is the number of bytes to write.
5. Lines 39 to 43 are the receive interrupt for the serial port. The data that comes in is sent to the text box by way of the delegate created earlier.
6. Lines 46 to 56 handle the text box for both the serial port and the main program.

The µVision code listing is identical to that used to communicate with the PC via Tera Term. It is included for reference.

C# Code Listing

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
/*********************************************************************
 * SerialIO
 * Run this program with the AT89C51AC3 board V1.0.  The test program in
 * the AT89C51AC3 board is named SerialIO.UV2.  This program prints
 * Ready!!! when it receives a ready signal from the board.  It then
 * prints echoed characters 1, 2, or 3 from the keyboard and transmits
 * those characters to the board.
*/
namespace SerialIO
  {public partial class frmSerialIO : Form
    {// This delegate enables asynchronous calls for setting
     // the text property on a TextBox control.
     delegate void SetTextCallback(string text);
     //Buffer of characters to be sent to the board.
     private char [] buffer = {'1', '2', '3'};
     public frmSerialIO()
       {InitializeComponent();
        sptCom1.Open();               //Open the serial port
       }
     private void btn1_Click(object sender, EventArgs e)
       {sptCom1.Write(buffer, 0, 1);  //Send buffer[0]
       }
     private void btn2_Click(object sender, EventArgs e)
       {sptCom1.Write(buffer, 1, 1);  //Send buffer[1]
       }
     private void btn3_Click(object sender, EventArgs e)
       {sptCom1.Write(buffer, 2, 1);  //Send buffer[2]
       }
     private void btnExit_Click(object sender, EventArgs e)
       {sptCom1.Close();              //Close the serial port
        Application.Exit();           //Exit the application
       }
     //This is the receive event.  It reads one line that MUST end with a CRLF and
     //  puts the data in the text box.
     private void sptCom1_DataReceived(object sender,
                         System.IO.Ports.SerialDataReceivedEventArgs e)
       {string sTmp;
        sTmp = sptCom1.ReadLine();   //Read one line (must contain CRLF)
        SetText(sTmp);               //Write line to the text box
       }
     //This is the text box routine.  It can be invoked by the serial port
     // thread or used by the main program.
     private void SetText(string text)
         {// InvokeRequired required compares the thread ID of the
          // calling thread to the thread ID of the creating thread.
          // If these threads are different, it returns true.
          if (txtPort.InvokeRequired)
              {SetTextCallback d = new SetTextCallback(SetText);
               this.Invoke(d, new object[] { text });
              }
          else
              {txtPort.Text = text;
              }
         }
    }
  }
```

µVision Code Listing

```
1  //SerialIO.c
2  /* This program works with either C# or Hyperterminal connected to the serial
3     port.  It can be run with the simulator as well.  (Use View to pull up the
4     serial window for simulation.)
5     The program transmits "Ready!!!" on reset and then waits for characters
6     to be received on the serial port at 19,200.  If the character is 1, 2,
7     or 3 a bit is set on port 1.
8  */
9  #include <reg51ac2.h>
10 code char msg [] = "Ready!!\r\n";
11 unsigned char c;
12 void main (void)
13   {unsigned char i;
14    CKCON = 0x01;      // x2 mode, T2 clock is same as crystal
15    SCON = 0X40;       // Mode 2, 8 bit uart transmit only, uses T2
16    RCLK=1;            // Turn on receive clock in T2CON
17    TCLK=1;            // Turn on transmit clock in T2CON
18    //Baud rate = fCrystal/(32*(65536 - (RCAP2H, RCAP2L))
19    // If double clocked, multiply fCrystal by 2.
20    RCAP2H=0XFF;       //19.2k baud @ 28.2076 Mhz
21    RCAP2L=0xA4;       //
22    TR2=1;             // TCON bit to start Timer 2
23    REN=1;             // Receive and transmit
24    RI = 0;            // Clear the receive interrupt flag
25    EA = 1;            //Turn on global interrupt flag
26    ES = 1;            //Turn no serial interrupt
27    i = 0;
28    while(msg[i] != 0) //Char string ends in 0
29      {TI = 0;         // Send out initial message
30       SBUF = msg[i];
31       i++;
32       while (TI == 0);   // Wait for write to be done
33      }
34    c = 0;
35    while(c == 0);        //Wait for a character to come in
36    while(1)              // strip off upper four bits of each
37       {switch (c & 0x0F) // character and set appropriate bit
38          {case 1:        // on P1 if 1, 2, or 3
39              P1 = 1;
40              break;
41           case 2:
42              P1 = 2;
43              break;
44           case 3:
45              P1 = 4;
46           default:
47              break;
48          }
49       }
50   }
51 //Serial interrupt that happens when character is received.
52 void SerialInt() interrupt 4 using 1
53   {if(RI)                           //If receiving data
54       {c = SBUF & 0x7F;             //strip off parity flag
55        SBUF = c;                    //Echo
56        while(TI == 0);              //Wait here until transmit complete
57        RI = 0;                      //Turn off receive interrupt flag
58        TI = 0;
59        SBUF = 0x0D;       //CR
60        while (TI == 0);   // Wait for write to be done
61        TI = 0;
62        SBUF = 0x0A;       //LF
63        while (TI == 0);   // Wait for write to be done
64        TI = 0;
65       }
66   }
```