AtlasScientific
Biology • Technology

RS-232 Port Connector sample code

```
/*
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
This software was made to demonstrate how to quickly get your RS-232 connector running.
An Arduino Duemilanove board was used to test this code. Modify the code to fit your system.
Code efficacy was not considered, this is a demo only.
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

The soft serial port TX line (pin 3) goes to the RS-232 connector RX pin found in the Z block.
The soft serial port RX line (pin 2) goes to the RS-232 connector TX pin found in the Z block.

Data from the RS-232 connector is received and re-sent through the Arduinos hardware UART
TX line. Open TOOLS > serial monitor, set the serial monitor to the correct serial port and set
the baud rate to 38400. The data from your Atlas Scientific products will appear on the serial
monitor.

The RS-232 connector has 3 control pins; TX and RX are connected to the Z pins.
S0 (Arduino Duemilanove board pin 4)
S1 (Arduino Duemilanove board pin 5)
E (pull to GND)

E is the enable pin. This pin is active low; in this example, we don't use E. Instead we just pull E
to GND. S0 and S1 are the control pins that direct where the input/output comes from. There are
4 locations from where the input/output can be directed.

Y0 | S0=0
Y1 | S0=1
Y2 | S0=0
Y3 | S0=1
*/

S1=0
S1=0
S1=1
S1=1


#include <SoftwareSerial.h>                    //add the soft serial library
#define rxpin 2                                //set the RX pin to pin 2
#define txpin 3                                //set the TX pin to pin 3
SoftwareSerial myserial(rxpin, txpin);         //enable the soft serial port


String inputstring = "";             //a string to hold incoming data from the PC
String sensorstring = "";            //a string to hold the data from the Atlas Scientific product
short holding;                       //used to tell us the number of bytes that have been received
short i;                             //counter
short ec=0;                          //this will tell the Arduino to open the EC channel
short d_o=1;                         //this will tell the Arduino to open the D.O. channel
short orp=2;                         //this will tell the Arduino to open the ORP channel
short ph=3;                          //this will tell the Arduino to open the pH channel
short done=0;


void setup(){                        //set up the soft serial to run at 38400
myserial.begin(38400);               //set up the hardware serial port to run at 38400
Serial.begin(38400);                 //used to control the S0 pin
pinMode(4, OUTPUT);                  //used to control the S1 pin
pinMode(5, OUTPUT);                  //set aside some bytes for receiving data from the PC
inputstring.reserve(5);              //set aside some bytes for receiving data
sensorstring.reserve(30);
}


void loop() {                        //main loop


//Read from the E.C.
//____________________________________________________________________________


do{                                  //start a nested loop
  Open_channel(ec);                  //we call the sub open channel. The sub needs to pass a
                                     //val to indicate what channel to open
                                     //ec is set to 0. So, we are telling the function to open
                                     //channel 0


   myserial.print("r");              //every Atlas Scientific device will respond to the "r"
                                     //command. This instructs the device to take a reading
   myserial.print('/r');             //ALWAYS end a command with <CR>


   delay(1100);                      //let 1.1 sec pass

     if(myserial.available() > 3) {  //if we see that more than three bytes have been received
   holding=myserial.available();     //lets read how many bytes have been received


   for(i=1; i <= holding;i++){          //we make a loop that will read each byte we received
     sensorstring[i]= myserial.read();3 //and load that byte into the sensorstring array
   }


   Serial.print("EC:");                 // EC:


   for(i=1; i <= holding;i++){          //we now loop through the array
   Serial.print(sensorstring[i]);       //printing each byte we received
   }


   Serial.println("");                  //once we finished, we print a <CR><LF>
   break;                               //exit the nested loop

   }

}while (done==1);
done=0;


//____________________________________________________________________________
//Read the D.O.
//____________________________________________________________________________


do{                                     //start a nested loop
  Open_channel(d_o);                    //we call the sub open channel. The sub needs to pass a
                                        //val to indicate what channel to open
                                        //D_O is set to 1. So, we are telling the function to open
                                        //channel 1


   myserial.print("r");                 //every Atlas Scientific device will respond to the "r"
                                        //command. This instructs the device to take a reading
   myserial.print('/r');                //ALWAYS end a command with <CR>


   delay(1100);                         //let 1.1 sec pass


   if(myserial.available() > 3) {       //if we see that more than three bytes have been received
   holding=myserial.available();        //lets read how many bytes have been received


   for(i=1; i <= holding;i++){          //we make a loop that will read each byte we received
     sensorstring[i]= myserial.read();  //and load that byte into the sensorstring array
   }


   Serial.print("DO:");                 // DO:
   for(i=1; i <= holding;i++){          //we now loop through the array
   Serial.print(sensorstring[i]);       //printing each byte we received
   }


   Serial.println("");                  //once we finished, we print a <CR><LF>
   break;                               //exit the nested loop

   }


}while (done==1);
done=0;


//____________________________________________________________________________
//Read the ORP
//____________________________________________________________________________



do{                                  //start a nested loop
  Open_channel(orp);                 //we call the sub open channel. The sub needs to pass a
                                     //val to indicate what channel to open
                                     //ORP is set to 2. So, we are telling the function to open
                                     //channel 2


   myserial.print("r");              //every Atlas Scientific device will respond to the "r"
                                     //command. This instructs the device to take a reading
   myserial.print('/r');             //ALWAYS end a command with <CR>


   delay(1100);                      //let 1.1 sec pass


     if(myserial.available() > 3) {  //if we see that more than three bytes have been received
   holding=myserial.available();     //lets read how many bytes have been received


   for(i=1; i <= holding;i++){          //we make a loop that will read each byte we received
     sensorstring[i]= myserial.read();  //and load that byte into the sensorstring array
   }


   Serial.print("ORP:");                //ORP:
   for(i=1; i <= holding;i++){          //we now loop through the array
   Serial.print(sensorstring[i]);       //printing each byte we received
   }


   Serial.println("");                  //once we finished, we print a <CR><LF>
   break;                               //exit the nested loop

   }


}while (done==1);
done=0;


//____________________________________________________________________________
//'Read the ph
//____________________________________________________________________________


do{                                   //start a nested loop
  Open_channel(ph);                   //we call the sub open channel. The sub needs to pass a


   myserial.print("r");               //every Atlas Scientific device will respond to the "r"
                                      //command. This instructs the device to take a reading
   myserial.print('/r');              //ALWAYS end a command with <CR>


   delay(1100);                       //let 1.1 sec pass


     if(myserial.available() > 3) {   //if we see that more than three bytes have been received
   holding=myserial.available();      //lets read how many bytes have been received


   for(i=1; i <= holding;i++){        //we make a loop that will read each byte we received
   sensorstring[i]= myserial.read();  //and load that byte into the sensorstring array
   }


   Serial.print("pH:");               //pH:
   for(i=1; i <= holding;i++){        //we now loop through the array
   Serial.print(sensorstring[i]);     //printing each byte we received
   }


   Serial.println("");                //once we finished, we print a <CR><LF>
   break;                             //exit the nested loop

   }

}while (done==1);
done=0;

//____________________________________________________________________________


void Open_channel(short channel){

      switch (channel) {


      case 0:                               //open channel Y0
      digitalWrite(4, LOW);                       //S0=0
      digitalWrite(5, LOW);                       //S1=0
      break;


      case 1:                               //open channel Y1
      digitalWrite(4, HIGH);                      //S0=1
      digitalWrite(5, LOW);                       //S1=0
      break;


      case 2:                               //open channel Y2
      digitalWrite(4, LOW);                       //S0=0
      digitalWrite(5, HIGH);                      //S1=1
      break;


      case 3:                               //open channel Y3
      digitalWrite(4, HIGH);                      //S0=0
      digitalWrite(5, HIGH);                      //S1=1
      break;

      }

   myserial.print('/r');          //the print cr was put in place to improve stability.
                                  //sometimes, when switching channels errant data
                                  //was passed. The print CR clears any incorrect data that was
                                  //transmitted to atlas scientific device.


return;
}
```