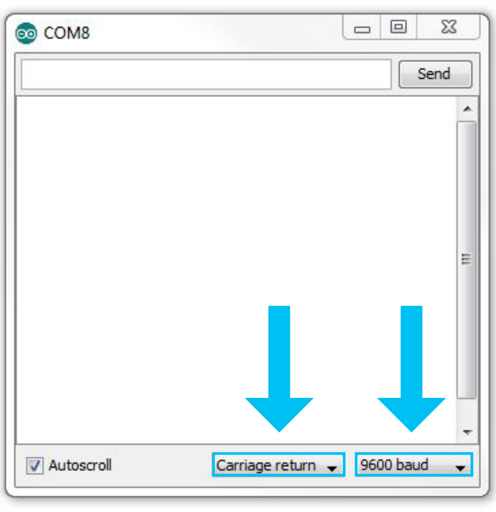
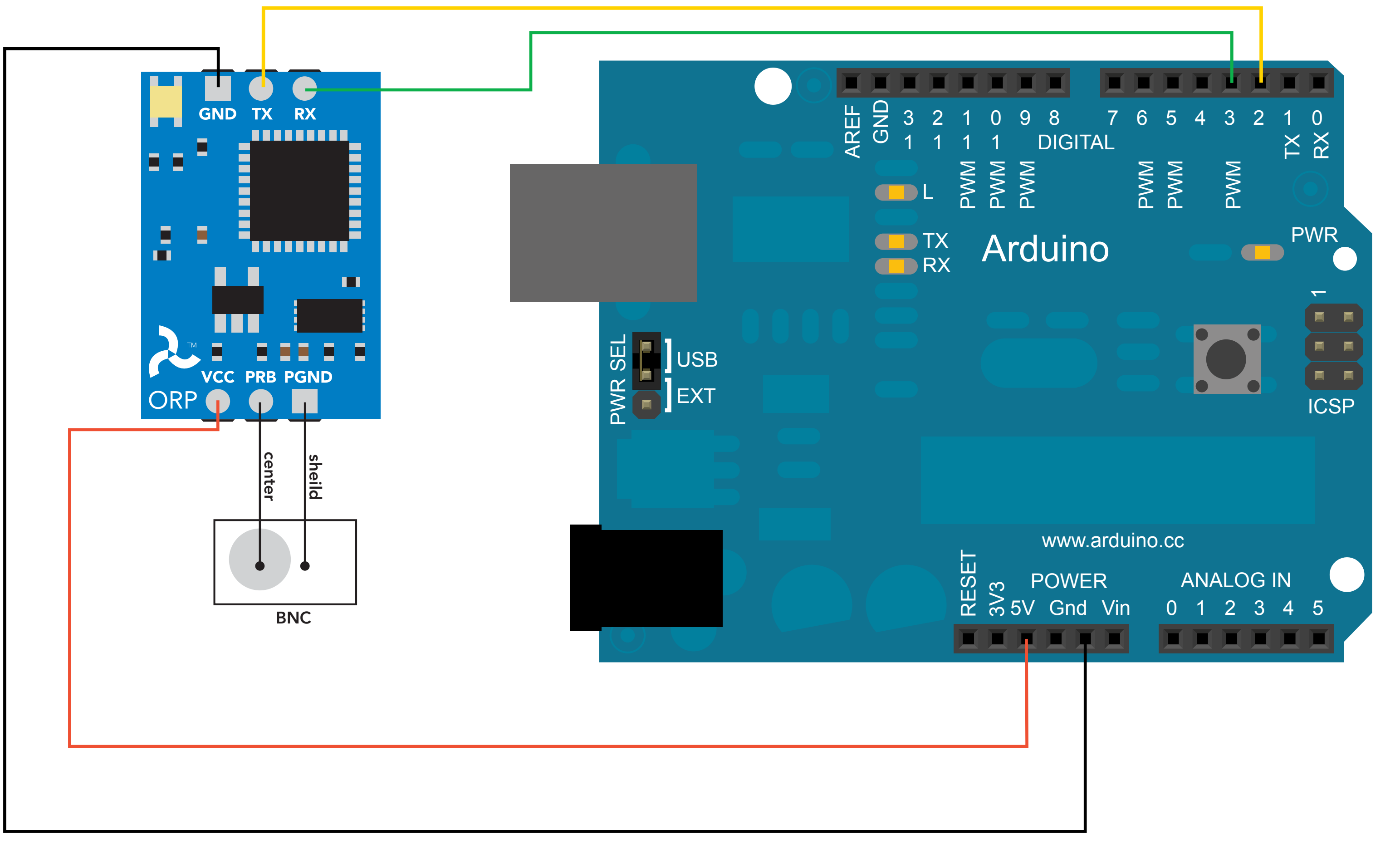
AtlasScientific
Environmental Robotics

Arduino ORP Sample Code



```
//This code has intentionally has been written to be overly lengthy and includes
//unnecessary steps. Many parts of this code can be truncated. This code was written
//to be easy to understand. Code efficiency was not considered. Modify this code
//as you see fit. This code will output data to the Arduino serial monitor. Type commands
//into the Arduino serial monitor to control the ORP circuit. set the var Arduino_only to
//equal 1 to watch the Arduino take over control of the ORP circuit.
```

```
//As of 11/6/14 the default baud rate has changed to 9600.
//The old default baud rate was 38400.
```



```
#include <SoftwareSerial.h>          //we have to include the SoftwareSerial library, or else we can't use it.
#define rx 2                         //define what pin rx is going to be.
#define tx 3                         //define what pin Tx is going to be.


SoftwareSerial myserial(rx, tx);     //define how the soft serial port is going to work.
```

```
char ORP_data[20];                   //we make a 20 byte character array to hold incoming data from the ORP.
char computerdata[20];               //we make a 20 byte character array to hold incoming data from a pc/mac/other.
byte received_from_computer=0;       //we need to know how many characters have been received.
byte received_from_sensor=0;         //we need to know how many characters have been received.
byte received_from_sensor=0;         //if you would like to operate the ORP Circuit with the Arduino only and not use the serial monitor
byte arduino_only=0;                 //to send it commands set this to 1. The data will still come out on the serial monitor, so you can see
                                     //it working

byte startup=0;                      //used to make sure the Arduino takes over control of the ORP Circuit properly.
float ORP=0;                         //used to hold a floating point number that is the ORP.
byte string_received=0;              //used to identify when we have received a string from the ORP circuit.


void setup(){
   Serial.begin(9600);               //enable the hardware serial port
   myserial.begin(9600);             //enable the software serial port
   }
```

```
 void serialEvent(){                                                   //this interrupt will trigger when the data coming from the
                                                                       //serial monitor(pc/mac/other) is received.
          if(arduino_only!=1){                                         //if Arduino_only does not equal 1 this function will
           received_from_computer=Serial.readBytesUntil(13,computerdata,20); //be bypassed.
                                                                       //we read the data sent from the serial monitor
                                                                       //(pc/mac/other) until we see a <CR>. We also count how
           computerdata[received_from_computer]=0;                     //many characters have been received.
                                                                       //we add a 0 to the spot in the array just after the last
                                                                       //character we received. This will stop us from transmitting
                                                                       //incorrect data that may have been left in the buffer.
           myserial.print(computerdata);                               //we transmit the data received from the serial monitor
                                                                       //(pc/mac/other) through the soft serial port to
                                                                       //the ORP Circuit.
           myserial.print('\r');                                       //all data sent to the ORP Circuit must end with a <CR>.

     }

}
```

```
void loop(){

  if(myserial.available() > 0){                                 //if we see that the ORP Circuit has sent a character.
    received_from_sensor=myserial.readBytesUntil(13,ORP_data,20); //we read the data sent from ORP Circuit until we see a <CR>.
                                                                //We also count how many character have been received.
       ORP_data[received_from_sensor]=0;                        //we add a 0 to the spot in the array just after the last character
                                                                //we received. This will stop us from transmitting incorrect data
                                                                //that may have been left in the buffer.
       string_received=1;                                       //a flag used when the Arduino is controlling the ORP Circuit
                                                                //to let us know that a complete string has been received.
       Serial.println(ORP_data);                                //lets transmit that data received from the ORP Circuit to the
                                                                //serial monitor.
   }                                                            //If the var arduino_only is set to one we will call this function.
                                                                //Letting the Arduino take over control of the ORP Circuit
    if(arduino_only==1){Arduino_Control();}

}
```

```
void Arduino_Control(){

          if(startup==0){                  //if the Arduino just booted up, we need to set some things up first.
             myserial.print("c,0\r");      //take the ORP Circuit out of continues mode.
             delay(50);                    //on start up sometimes the first command is missed.
             myserial.print("c,0\r");      //so, let’s send it twice.
             delay(50);                    //a short delay after the ORP Circuit was taken out of continues mode is used to make sure
             startup=1;                    //we don’t over load it with commands.
                                           //startup is completed, let's not do this again during normal operation.
          }
```

```
  delay(800);                                 //we will take a reading ever 800ms. You can make this much longer or shorter if you like.
     myserial.print("R\r");                   //send it the command to take a single reading.
     if(string_received==1){                  //did we get data back from the ORP Circuit?
        ORP=atof(ORP_data);                   //many people ask us "how do I convert a sting into a float?" This is how...
        if(ORP>=500){Serial.println("high\r");} //This is the proof that it has been converted into a float.
        if(ORP<499.9){Serial.println("low\r");} //This is the proof that it has been converted into a float.
        string_received=0;}                   //reset the string received flag.
  }
```

```
//here are some functions you might find useful
//these functions are not enabled


/*
void cal_225(){                               //calibrate to a ORP of 225mv
  myserial.print("cal,225\r");}               //send the "cal,225" command to calibrate to a ORP of 225mv

void ORPFactoryDefault(){                     //factory defaults the ORP circuit
  myserial.print("X\r");}                     //send the "X" command to factory reset the device

void read_info(){                             //get device info
   myserial.print("I\r");}                    //send the "I" command to query the information

void ORPSetLEDs(byte enabled                  //turn the LEDs on or off
{
  if(enabled)                                 //if enabled is > 0
   myserial.print("L,1\r");                   //the LED's will turn ON
  else                                        //if enabled is 0
   myserial.print("L,0\r");                   //the LED's will turn OFF
}
*/
```

Click here to download the *.ino file

Atlas-Scientific.com

Copyright © Atlas Scientific LLC All Rights Reserved