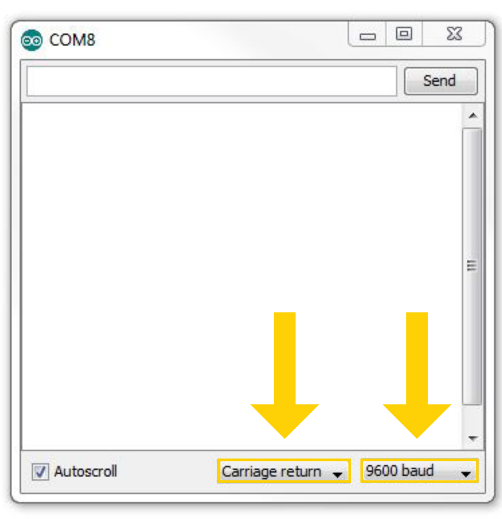
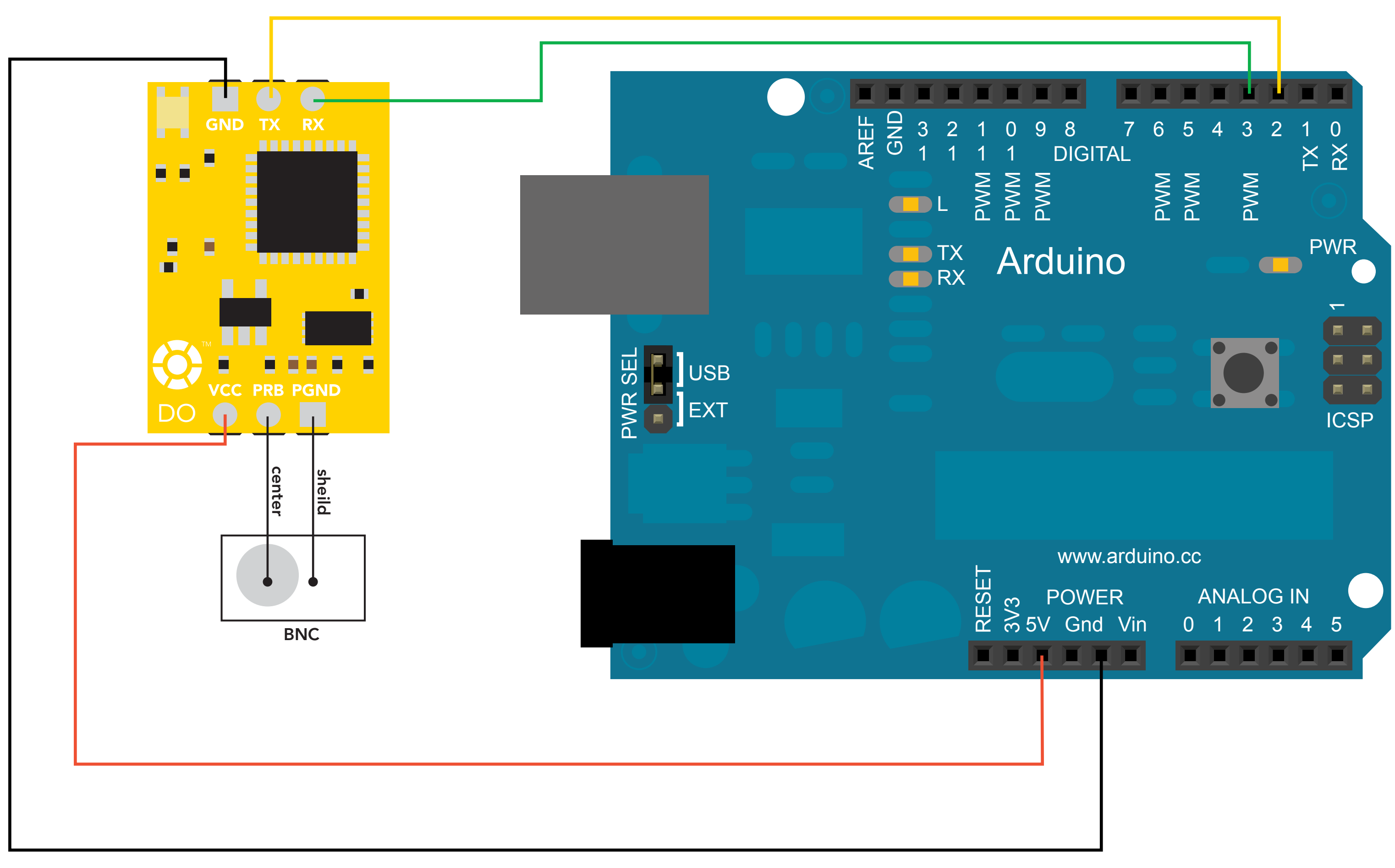
AtlasScientific™
Environmental Robotics

# Arduino D.O. 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 D.O. circuit. set the var Arduino_only
//to equal 1 to watch the Arduino take over control of the D.O. 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 DO_data[20];                    //we make a 20 byte character array to hold incoming data from the D.O.
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 arduino_only=0;                 //if you would like to operate the D.O. Circuit with the Arduino only and not use the serial monitor
                                     //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 D.O. Circuit properly.
byte string_received=0;              //used to identify when we have received a string from the D.O. circuit.
float DO_float=0;                    //used to hold a floating point number that is the D.O.
float sat_float=0;                   //used to hold a floating point number that is the percent saturation.
char *DO;                            //char pointer used in string parsing
char *sat;                           //char pointer used in string parsing


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


 void serialEvent(){                                                     //this interrupt will trigger when the data coming
      if(arduino_only!=1){                                               //from the serial monitor(pc/mac/other) is received.
        received_from_computer=Serial.readBytesUntil(13,computerdata,20);  //if Arduino_only does not equal 1 this function will
        computerdata[received_from_computer]=0;                          //be bypassed.
        myserial.print(computerdata);                                    //we read the data sent from the serial monitor
        myserial.print('\r');                                            //(pc/mac/other) until we see a <CR>. We also count
       }                                                                 //how 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.
                                                                         //we transmit the data received from the serial monitor
                                                                         //(pc/mac/other) through the soft serial port to
                                                                         //the D.O. Circuit.
                                                                         //all data sent to the D.O. Circuit must end with a <CR>.


void loop(){                                                      //if we see that the D.O. Circuit has sent a character.
                                                                  //we read the data sent from D.O. Circuit until we see a
    if(myserial.available() > 0){                                 //<CR>. We also count how many character have been received.
      received_from_sensor=myserial.readBytesUntil(13,DO_data,20);  //we add a 0 to the spot in the array just after the last character
      DO_data[received_from_sensor]=0;                            //we received. This will stop us from transmitting incorrect data
      string_received=1;                                          //that may have been left in the buffer.
                                                                  //a flag used when the Arduino is controlling the D.O. Circuit
      if((DO_data[0] >= 48) && (DO_data[0] <=57)){                //to let us know that a complete string has been received.
        pars_data();                                              //if DO_data[0] is a digit and not a letter
        }
      else
       Serial.println(DO_data);                                   //if the data from the D.O. circuit does not start with a number
       }                                                          //transmit that data to the serial monitor.
   }


 void pars_data()
{

 byte i;
 byte pars_flag=0;

 for(i=0;i<=received_from_sensor;i++)
    {
      if(DO_data[i]==','){pars_flag=1;}
    }


if(pars_flag){

 DO=strtok(DO_data, ",");                  //let's pars the string at each comma.
 sat=strtok(NULL, ",");                    //let's pars the string at each comma.

 Serial.print("DO:");                      //We now print each value we parsed sepratly.
 Serial.println(DO);                       //this is the DO value.
 //DO_float=atoi(DO);                      //Uncomment this line to turn the string into to floating pint value.
 }
                                           //We now print each value we parsed sepratly.
 Serial.print("%sat:");                    //this is the % sat value.
 Serial.println(sat);                      //Uncomment this line to turn the string into to floating pint value.
 //sat_float=atoi(sat);
 }

 else                                      //if the output is ony DO and not DO + % sat
 {
  Serial.print("DO:");                     //print out "DO:"
  Serial.println(DO_data);                 //printout that DO in Mg/L
 }


 }
```

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


/*
void cal_air(){                            //calibrate to the atmosphere
  myserial.print("cal\r");}                //send the "cal" command to calibrate to the atmosphere

void DOFactoryDefault(){                   //factory defaults the D.O. 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 DOSetLEDs(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**