CO2_PID_Controller_Arduino Source Code (Open access source code)

```
// Arduino IDE version used for development/compatibility reasons: 1.0.6
// https://www.arduino.cc/en/Main/OldSoftwareReleases#previous

//Acknowledgement to Andrew Pelling @ www.pellinglab.net for original version
//Modifications:  Danny Di Giacomo and Malcolm Brinn – University of Adelaide www.adelaide.edu.au

/*
 Humidifier and CO2 gas control via 12V normally closed solenoid and CO2 Sensor.
 CO2 Meter (SprintIR 0-20% GC-0017):
 http://www.co2meter.com/collections/co2-sensors/products/sprintir-100-percent-co2-sensor
 also need the latest COZIR library, download from this thread (Using Cozir 0-1-01.zip in this code):
 http://forum.arduino.cc/index.php?topic=91467.0
 CO2 supplied by BOC Gas
 Arduino UNO Pin Connections:
 Pin 2,3: Tx/Rx from CO2  //Pin 2 arduino goes to Tx on Sensor Pin 4
 Pin #5:
 Pin #6:
 Pin #7:
 Pin #8: Solenoid Relay
 Pin #9:
 SCL/SDA Pins: 12C Display
 */

/* CO2 CONTROL /////////
 Control works by reading the sensor (CO2) 3 times and averaging (to flatten out noise).
 The measurement is compared to a user-defined SETPOINT. If the CO2 is below the setpoint, the solenoid
 opens allowing the flow of CO2 into the incubator. Once the CO2 content comes within a user-defined
 THRESHOLD of the setpoint, a stepping cycle begins. The solenoid opens for a user defined DURATION and
 then closes. The open/closed cycle repeats until the setpoint is reached.

 The cycles allow the system to step up to the setpoints.
 Setpoints, thresholds and durations MUST be set according to your own system by TRIAL AND ERROR.
 May need to increase/decrease depending on size/shape of incubator, CO2 flow rate, etc.
 */

float CO2Setpoint = 1.0;   // Setpoint CO2 level in % 5
float CO2Threshold = 0.85; // Threshold to switch to stepping control in %/100
int SolenoidOnTime = 5000; // Duration time in milliseconds

// CO2 Sensors
#include "SoftwareSerial.h"
#include "cozir.h"
// Cozir library info/download: http://forum.arduino.cc/index.php?topic=91467.0

// Excel logging configuration
// http://www.robertovalgolio.com/sistemi-programmi/arduino-excel
#include <rExcel.h>
long idx = 0; // index
int outputTiming = 1000;   // packet sending timing in ms, determines the output timing
rExcel myExcel;            // class for Excel data exchanging
int save_loop_timer = 1;
/*
char worksheet[16];        // worksheet name
 char range[16];           // range set
 unsigned int row;         // row
 unsigned int column;      // column
 char value[16];           // written or read value
 */
```

```
// I2C LCD interface configuration
// http://forum.arduino.cc/index.php?topic=128635.0
// https://bitbucket.org/fmalpartida/new-liquidcrystal/downloads
// LiquidCrystal_V1.2.1.zip library used
#include <Wire.h>
#include <LCD.h>
#include <LiquidCrystal_I2C.h>
#define I2C_ADDR 0x3F
#define BACKLIGHT_PIN 3
#define En_pin 2
#define Rw_pin 1
#define Rs_pin 0
#define D4_pin 4
#define D5_pin 5
#define D6_pin 6
#define D7_pin 7
LiquidCrystal_I2C lcd(I2C_ADDR,En_pin,Rw_pin,Rs_pin,D4_pin,D5_pin,D6_pin,D7_pin);

////////* RELAY *////////
int Solenoid = 8;          // Relay for controlling 12V solenoid valve

////////* SPRINTIR COZIR CO2 SENSOR  *////////
SoftwareSerial nss(2, 3);  // Rx,Tx - Pin 3,4 on sensor
COZIR czr(nss);
float SingleCO2, CO2 = 0;
float multiplier = 0.001;  // 10/10000 (Hardware multiplier/ppm conversion).
float reading = 0;

void setup()
{
  Serial.begin(9600);                // Start serial port
  czr.SetOperatingMode(CZR_POLLING);  // Start the CO2 sensor and put into POLLING mode
  pinMode(Solenoid, OUTPUT);          // Sets pin for controlling solenoid relay
  digitalWrite(Solenoid, LOW);        // Set LOW (solenoid closed off)
  lcd.begin(16,2);
  lcd.setBacklightPin(BACKLIGHT_PIN,POSITIVE);
  lcd.clear();
  lcd.setBacklight(HIGH);
  lcd.home();
  pinMode(13, INPUT_PULLUP);
  // delay(500);                      // Wait
}

void loop()
{
  // Read CO2 sensor 3 times and determine the average.
  for (int i = 0; i < 3; i++) {
    SingleCO2 += czr.CO2() * multiplier;
  }
  CO2 = SingleCO2 / 3;
  SingleCO2 = 0;

  // Write CO2 % to LCD
  lcd.setCursor (0, 0);
  lcd.print("CO2 level: ");
  lcd.setCursor (10, 1);
  lcd.print(CO2, 2);
  lcd.print(" %");
```

```
  // If switch on GPIO 13 low, enable Excel logging
  if (digitalRead(13) == LOW) {

    static unsigned long  loopTime = 0;
    static unsigned long  time1 = 0;

    loopTime = millis();

    // Output Task
    // Arduino acts as client making requests to Excel
    // instructions performed each outputTiming ms
    if ((loopTime - time1) >= outputTiming) {
      time1 = loopTime;

      myExcel.write("Template", "A5", "%time%");            // write time to sheet "Template" cell A5
      myExcel.write("Template", "B5", CO2, 2);              // write CO2 %
      myExcel.writeIndexed("Template", idx+11, 1,"%date%");  // idx is zero initially, write date to row idx+11, column 1
      myExcel.writeIndexed("Template", idx+11, 2,"%time%");
      myExcel.writeIndexed("Template", idx+11, 3, idx);
      myExcel.writeIndexed("Template", idx+11, 4, CO2, 2);
      idx++;

      // Save spreadsheet every 60 seconds
      save_loop_timer++;
      if (save_loop_timer == 60) {
        myExcel.save();
        myExcel.write("Template", "E1", "Autosaved: ");
        myExcel.write("Template", "F1", "%time%");
        save_loop_timer = 1;
      }
    }
  }

  // Open/close the solenoid valve based on the current CO2 reading
  // Solenoid  opens for a duration of 'SolenoidOnTime' and then closes
  if (CO2 < CO2Setpoint && CO2 < CO2Threshold * CO2Setpoint) {
    digitalWrite(Solenoid, HIGH);
  }
  else if (CO2 < CO2Setpoint && CO2 >= CO2Threshold * CO2Setpoint) {
    digitalWrite(Solenoid, HIGH);
    delay(SolenoidOnTime);
    digitalWrite(Solenoid, LOW);
  }
  else if (CO2 > CO2Setpoint) {
    digitalWrite(Solenoid, LOW);
  }
  delay(10);
}
```