Supplement of Ocean Sci., 11, 313–321, 2015
http://www.ocean-sci.net/11/313/2015/
doi:10.5194/os-11-313-2015-supplement

Ocean Science
Open Access

Supplement of

Technical Note: A fully automated purge and trap GC-MS system for quantification of volatile organic compound (VOC) fluxes between the ocean and atmosphere

S. J. Andrews et al.

Correspondence to: S. J. Andrews (stephen.andrews@york.ac.uk)

1 2 1 2

1 Supplementary information

```
//Arduino code for auto purge and trap system. This code reads a relay trigger from the Unity2 and controls
//two valves which fill and empty the purge tube. Outputs to an LCD screen.
//it also runs isothermal heating control of the purge tube in the background.
//author Stephen J. Andrews

const int TDrelaypin = 7; // Pin attatched to TD sample valve relay
const int SolenoidApin = 13; // Pin attached to solenoid via a transistor and relay
const int SolenoidBpin = 8; //same as above
const int LM35Pin = A3; //analog in with LM35 temp senor attached
const int heaterPin = 10; //PWM output to SSR to control heating tape
const int tempsetpoint = 50; //what temp do you want the tube to be?
const int manualtriggerpin = 6;
const int omegaSSRout = 9;
const int peristaltic = 0;
const int floatswitch = A5;
const int omegatempvalue = 2;

float tempC;
float tempdiff;
int extrapower = 1;
boolean Running = false;
boolean linespurged = false;
boolean tubefilled = false;
boolean tubepurged = false;
boolean tubeflushed = false;
long countdownitem = 0;
long countdown = 0;
long count = 0;
long triggertime = 0;
long currenttime = 0;
long previoustemptime = 0;
int dataupdatespeed = 1000;
long filterpurgeinterval = 280000; //105000
long linepurgeinterval = 305000; //125000
long fillinterval = 351500; //156500 adjust this to set the time in ms for the tube to fill for. 46.5sec =30mL
long flushinterval = 2146500; //1325000 time(ms) from triggertime until the water is emptied to waste
long flushtime = 2166500; //1345000 time(ms) that it empties the water for (from triggertime)
int LM35value = 0; //value read from temp sensor
int heatervalue = 0; //PWM value to trim heater by
long POS = 0; //percentage of setpoint
int mappedLM35 = 0;

LiquidCrystal lcd(12, 11, 5, 4, 3, 2);

void setup() {
// set up the LCD's number of columns and rows:
lcd.begin(20, 4);
// initialize serial communications at 9600 bps:
analogReference(INTERNAL); //analog scale to 1.1V for better resolution!
pinMode(TDrelaypin, INPUT); //specify the pin modes
pinMode(SolenoidApin, OUTPUT);
pinMode(SolenoidBpin, OUTPUT);
pinMode(omegaSSRout, OUTPUT);
pinMode(peristaltic, OUTPUT);
pinMode(floatswitch, INPUT);
pinMode(manualtriggerpin, INPUT);
digitalWrite(SolenoidApin, LOW);
digitalWrite(SolenoidBpin, LOW);
digitalWrite(peristaltic, HIGH);
}
void loop() {
int floatswitchstate = digitalRead(floatswitch);
if (floatswitchstate == LOW) {digitalWrite(peristaltic, LOW);
lcd.setCursor(0,0);
lcd.print(" LEAK DETECTED!!!!! ");
}
if (floatswitchstate == HIGH) {
lcd.setCursor(0,0);
if (tempC < (tempsetpoint-1)){
lcd.print("Tube is below temp!!");}
else lcd.print("Automated Purge&Trap");}
```

```
LM35value = analogRead(LM35Pin); //read the value from the temp sensor
tempC = LM35value / 9.31;
tempdiff = tempsetpoint - tempC;
float temppercent = tempdiff * 2.5 * extrapower;
//mappedLM35 = map(LM35value, 0, 1023, 0, 255);
//scale 1024 analoge bits to 256 digital
//POS = mappedLM35 / tempsetpoint *100;
// if (POS < 80) {
if (tempC >= tempsetpoint) {
heatervalue = 0;
}
if (tempC < tempsetpoint) {
heatervalue = temppercent * 2.55;
}// }
// if (POS > 80) {
// heatervalue = (255/100)*(100 - POS);
// }
analogWrite(omegaSSRout, heatervalue); //write the PWM value to the heater pin
analogWrite(heaterPin, omegatempvalue);

int TDtrigger = digitalRead(TDrelaypin);
int manualtrigger = digitalRead(manualtriggerpin);
if ((manualtrigger == LOW) && (Running == false)) {TDtrigger = HIGH;}
if (TDtrigger == HIGH) {
count = count + 1;

if ((count == 1) && (Running == false)){
triggertime = millis();
lcd.setCursor(0,1);
lcd.print("Purging sample lines");
countdownitem = linepurgeinterval;
linespurged = false;
tubefilled = false;
tubepurged = false;
tubeflushed = false;
Running = true;
}}
else {count = 0 ;}

currenttime = millis();
if (Running == true){

if ((currenttime - triggertime > filterpurgeinterval) && (linespurged == false)) {
digitalWrite(SolenoidApin, HIGH);
lcd.setCursor(0,1);
lcd.print("Purging lines&filter");}

if ((currenttime - triggertime > linepurgeinterval) && (linespurged == false)) {
digitalWrite(SolenoidApin, HIGH);
digitalWrite(SolenoidBpin, HIGH);
//extrapower = 4;
lcd.setCursor(0,1);
lcd.print(" Filling purge tube ");
countdownitem = fillinterval;
linespurged = true;
}
// if ((currenttime - triggertime > linepurgeinterval) && (linespurged == false)) { // }
if ((currenttime - triggertime > fillinterval) && (tubefilled == false)) {
digitalWrite(SolenoidApin, LOW);
digitalWrite(SolenoidBpin, LOW);
lcd.setCursor(0,1);
lcd.print(" Purging tube ");
countdownitem = flushinterval;
tubefilled = true;
//extrapower = 4;
}
// if ((currenttime - triggertime > linepurgeinterval) && (linespurged == false)) {
// }
if ((currenttime - triggertime ¿ flushinterval) && (tubepurged == false)) {
digitalWrite(SolenoidApin, LOW);
digitalWrite(SolenoidBpin, HIGH);
lcd.setCursor(0,1);
lcd.print(" Evacuating tube ");
countdownitem = flushtime;
tubepurged = true;
}
// if ((currenttime - triggertime > linepurgeinterval) && (linespurged == false)) {
// }
if ((currenttime - triggertime > flushtime) && (tubeflushed == false)){
digitalWrite(SolenoidApin, LOW);
digitalWrite(SolenoidBpin, LOW);
```

```
Running = false;
lcd.setCursor(0,1);
lcd.print(" Waiting for sample ");
tubeflushed = true;
} }
if (currenttime - previoustemptime > dataupdate-
speed) {
previoustemptime = currenttime;
lcd.setCursor(0, 3);
lcd.print("Tube = ");
lcd.print(tempC,1);
lcd.print((char)223);
lcd.print("C ");
lcd.setCursor(14,3);
lcd.print(" P=");
lcd.print(temppercent,0);
lcd.print("% ");

if (Running == true) {
countdown = countdownitem - (currenttime - trig-
gertime);
lcd.setCursor(8,2);
lcd.print(countdown/1000);
lcd.print("sec ");
// lcd.setCursor(15,3);
// lcd.print("TRUE ");
}
else {lcd.setCursor(8,2);
lcd.print(" ");
// lcd.setCursor(15,3);
// lcd.print("FALSE");
}}}
```