Science Olympiad 2022

Detector Building Design Log

May 5, 2022

Detector Building Design Log

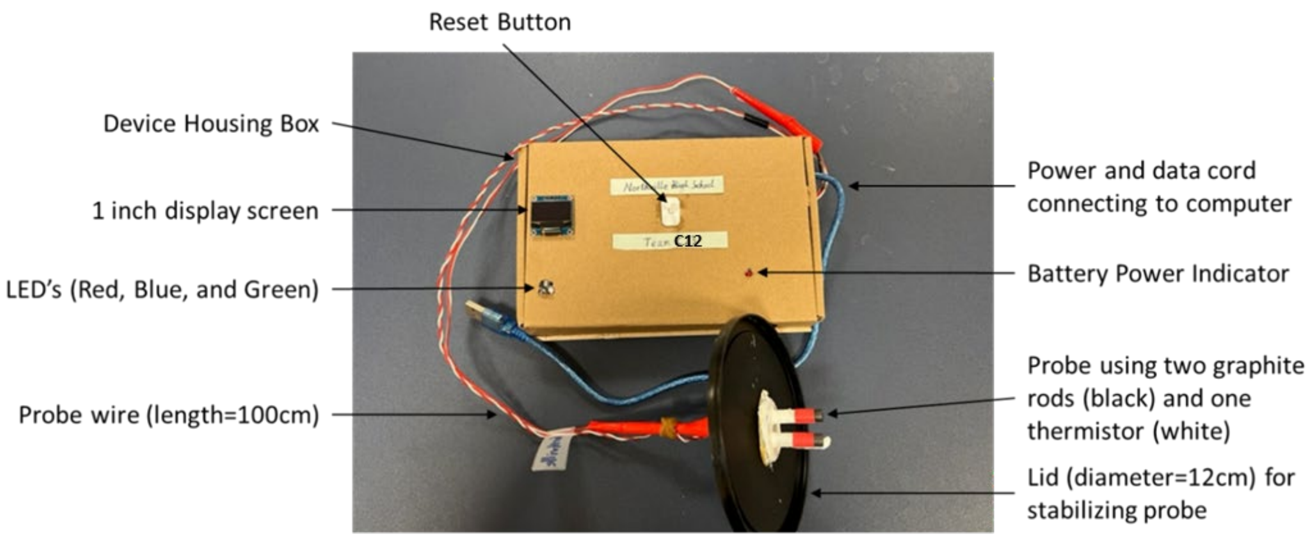
May 5, 2022

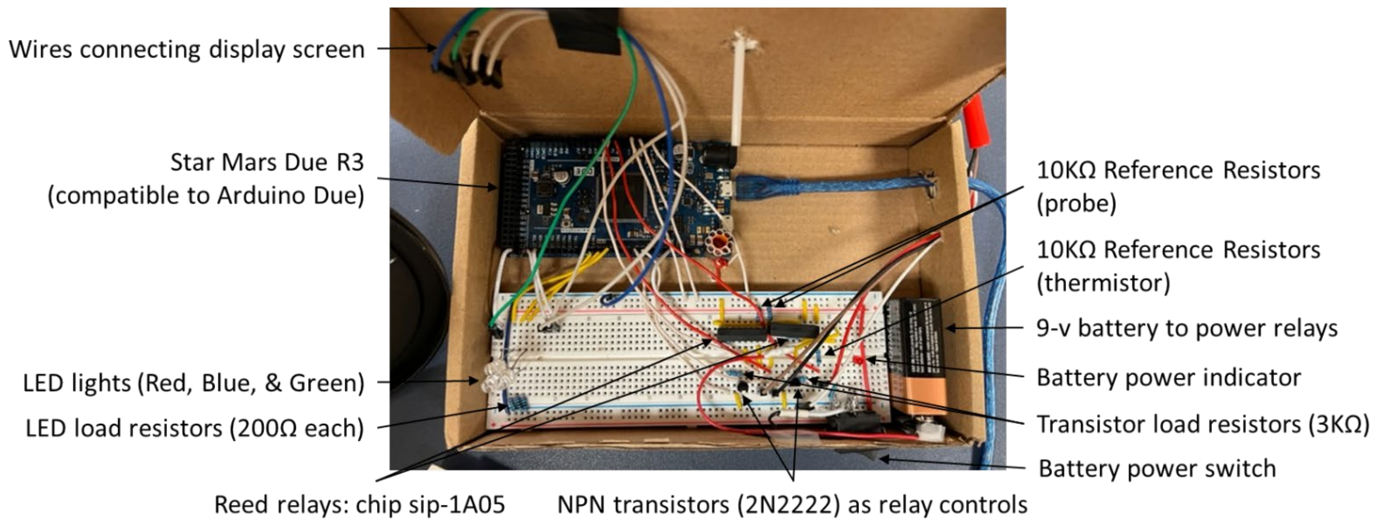
Northville High School VARSITY Team, Team # C 12

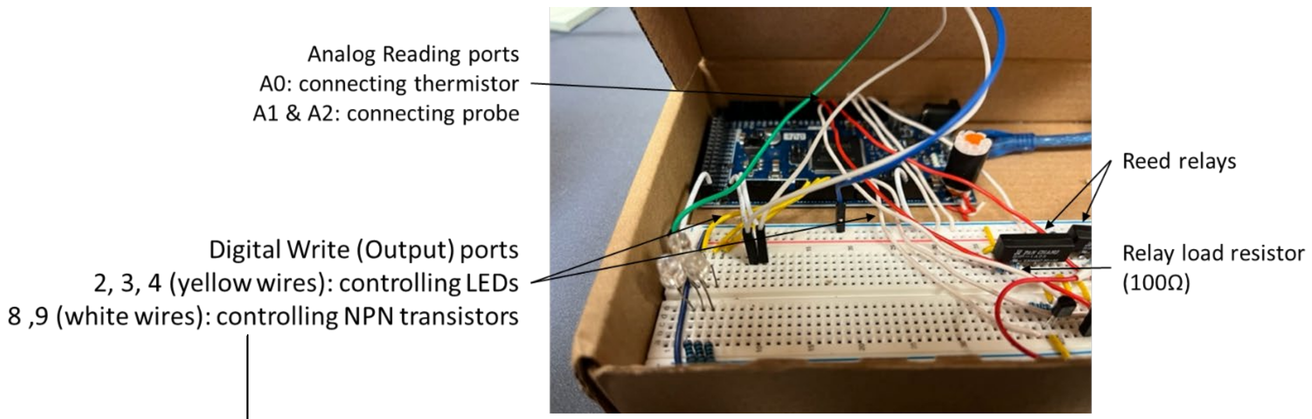
Note: The physical design is featured with: 1) using a Star Mars Due R3 (compatible to Arduino Due) microcontroller; using a thermistor to measure temperature; 2) using two reed relays (chip model Sip-1A05) to ground probe ends and suppress any capacitance built up between measurements; 3) using two NPN transistors as high-frequency switches to control the relays; 4) using a 9-v battery to power relays; 5) using digital output pin 8&9 to power the probe in sequence to alter the current between the two probe ends, not only suppressing any capacitance but also resulting in two independent measurements; and 6) the device is packed in a box for mobility and reliability.

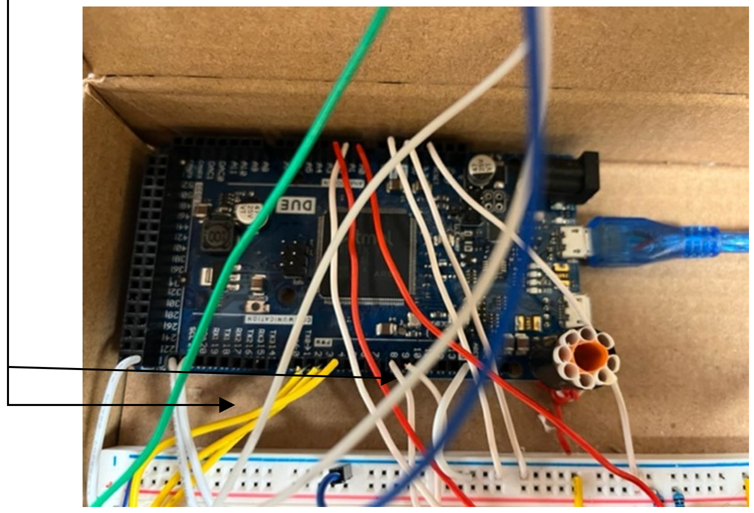
The codes are featured with: 1) oversampling: each measurement includes 250 samples; 2) denoising: noises in measurement readings, i.e., 20% below or above the average, are dropped; 3) measuring capacitance: each measurement includes readings of the first resistance (i.e., >0) and the subsequent five resistances gapped by 200 microseconds; a linear regression is conducted to calculate the initial resistance as well as the slope (increased resistance by 200 microseconds) induced by the capacitance. Both the initial resistance and the measured slope (indicating capacitance) are used in estimating the TDS; 4) using the ratio between the standard deviation and the average among five sequential measurements (5 X 250 or 1250 samples in total) as the criterion for the measurement to reach consistence; 5) the TDS is the weighted average (weighted by the effective sample size) based on the most consistent five sequential measurements from each of the two current directions; and 6) reporting the final TDS when either one threshold is met: a. reaching time limit of 90 seconds; or b. reaching consistence with the ratio of standard deviation and average is less than 0.0025.

4.b.i. Design photo


Reset Button
Device Housing Box
1 inch display screen
LED's (Red, Blue, and Green)
Probe wire (length=100cm)
Power and data cord connecting to computer
Battery Power Indicator
Probe using two graphite rods (black) and one thermistor (white)
Lid (diameter=12cm) for stabilizing probe
Northville High School
Team C12


Wires connecting display screen
Star Mars Due R3
(compatible to Arduino Due)
LED lights (Red, Blue, & Green)
LED load resistors (200Ω each)
Reed relays: chip sip-1A05
NPN transistors (2N2222) as relay controls
10KΩ Reference Resistors
(probe)
10KΩ Reference Resistors
(thermistor)
9-v battery to power relays
Battery power indicator
Transistor load resistors (3KΩ)
Battery power switch


Analog Reading ports
A0: connecting thermistor
A1 & A2: connecting probe
Digital Write (Output) ports
2, 3, 4 (yellow wires): controlling LEDs
8 ,9 (white wires): controlling NPN transistors
Reed relays
Relay load resistor
(100Ω)



Explanation: the detector device has a probe of two graphite rods as the resistance sensor, and one thermistor (Uxcell NTC Thermistor MF58 3950B 10K ohm) as the temperature sensor. The resistance is measured in two current directions altering between the probe ends. Star Mars Due R3 (compatible to Arduino Due) is selected as the microprocessor mainly for its 12-bit readings. Two reed relay chip (model sip-QA05) are used to ground the two probe ends in certain sequence in order to 1) alter current directions between the two probe ends, 2) when both probe ends are grounded, they discharge any capacitance built up during measurements. A 9V battery is used to power the relays to avoid draining too much power from the microprocessor. The relays are controlled by two NPN transistors (2N2222); digital pin 8 and 9 are used to send on/off signal (set to either HIGH or LOW at specific time) to the transistors to control the relays.

With such design, resistance is measured in two opposite directions, resulting in two independent measurements.

The detector cable is about 100cm long.

4.b.ii. Data table

Table 1. Saline Water TDS (in ppm) with Measured Resistance at 21°C

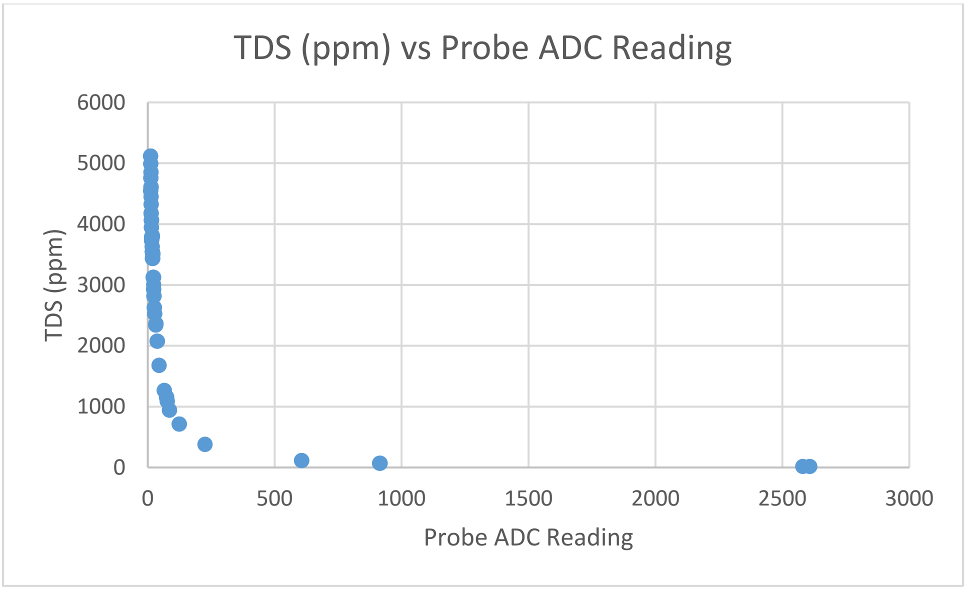
Measurement ID	Temperature (°C)	Probe ADC Reading	Voltage Drop (V)	Resistance (kΩ)	TDS
1	21.0	4.66897	0.02281999	0.04576	5115
2	21.0	7.31842	0.035769404	0.07353	3433
3	21.0	18.49521	0.090396921	0.18535	1675
4	21.0	14.85556	0.07260782	0.14712	2074
5	21.0	6.97322	0.034082209	0.06864	3727
6	21.1	6.96085	0.03402175	0.0685	3808
7	21.1	6.98274	0.034128739	0.06867	3727
8	21.1	18.57279	0.0907761	0.18531	1675
9	21.1	32.47625	0.15873045	0.32646	939
10	21.1	214.64614	1.049101369	2.64434	112
11	21.1	7.00003	0.034213245	0.0689	3627
12	21.1	18.77263	0.091752835	0.18691	1675
13	21.1	7.01587	0.034290665	0.0689	3627
14	21.1	8.9815	0.043897849	0.08859	3123
15	21.1	217.68161	1.063937488	2.70335	112
16	21.1	25.0739	0.122550831	0.25187	1263
17	21.1	6.98777	0.034153324	0.06875	3727
18	21.1	7.00285	0.034227028	0.06891	3627
19	21.1	9.35542	0.045725415	0.09141	3002
20	21.1	8.54199	0.041749707	0.08316	3510
21	21.1	12.28658	0.060051711	0.12185	2361
22	21.1	6.40938	0.031326393	0.06293	4278
23	21.1	6.85343	0.033496725	0.06749	3943
24	21.1	320.61791	1.567047458	4.56644	66
25	21.1	8.98745	0.043926931	0.08863	3123
26	21.1	24.66548	0.120554643	0.24671	1263

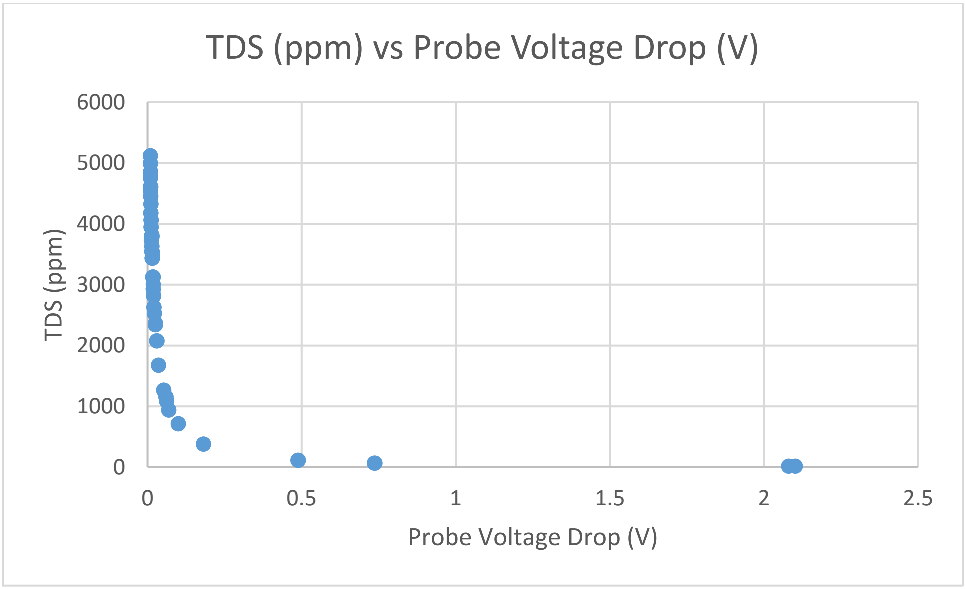
27	21.1	6.62338	0.032372336	0.06542	4060
28	21.1	6.66698	0.032585435	0.06617	4278
29	21.2	6.97029	0.034067889	0.06863	3808
30	21.2	7.03929	0.034405132	0.06891	3727
31	21.2	8.96927	0.043838074	0.08855	3123
32	21.2	5.88799	0.028778055	0.05776	4448
……					
129	21.5	4.75802	0.02325523	0.04684	5115
130	21.6	4.84263	0.023668768	0.04668	4990
131	21.6	28.52523	0.139419501	0.28725	1154
132	21.6	5.42559	0.026518035	0.05324	4850
133	21.6	5.80826	0.028388368	0.05704	4605
134	21.6	5.46839	0.026727224	0.05379	4756
135	21.6	5.78068	0.028253568	0.05686	4546
136	21.6	5.81884	0.028440078	0.05723	4605
137	21.6	5.82292	0.02846002	0.05743	4605
138	21.6	28.10661	0.13737346	0.2829	1154
139	21.7	5.6528	0.027628543	0.05576	4756
140	21.7	5.42881	0.026533773	0.05317	4756
141	21.7	5.42477	0.026514027	0.05367	4850
142	21.7	5.54864	0.027119453	0.05459	4850
143	21.8	5.33176	0.026059433	0.052	4756
144	21.9	6.18437	0.030226637	0.06079	4546

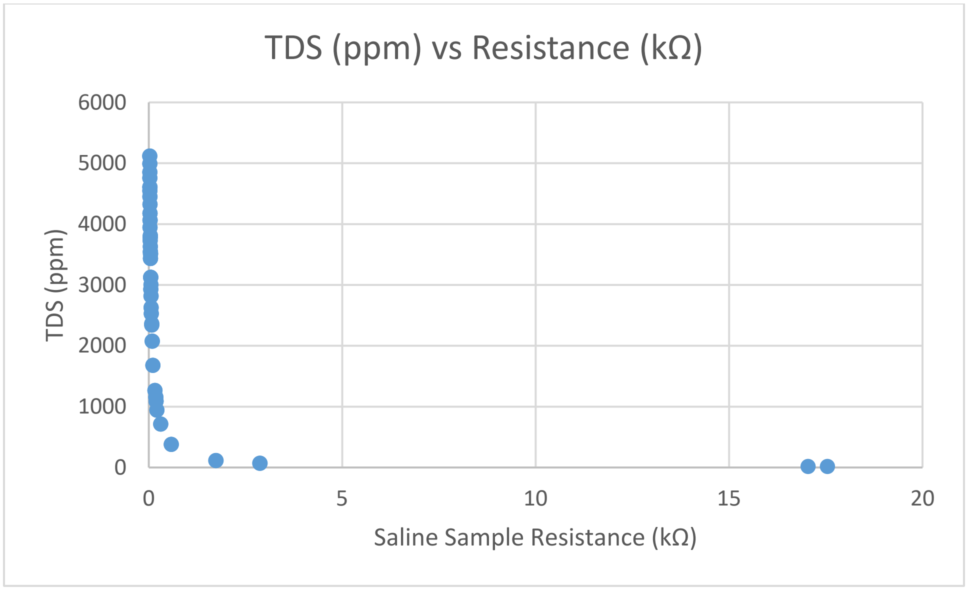
Note:

1. The above data are from one of the two independent measurements as explained in 4.b.i. at the temperature 21°C. Other measurement data are not presented here due to the scale. In total, there are 334 measurements covering reasonable room temperatures (16-22°C), each measurement includes four independent measures.
2. Each measurement is the average of 1250 measures (five consecutive samplings, with 250 measures in each sample), when the average converges (the standard deviation among the five consecutive samples is less than 0.25% of the overall average).

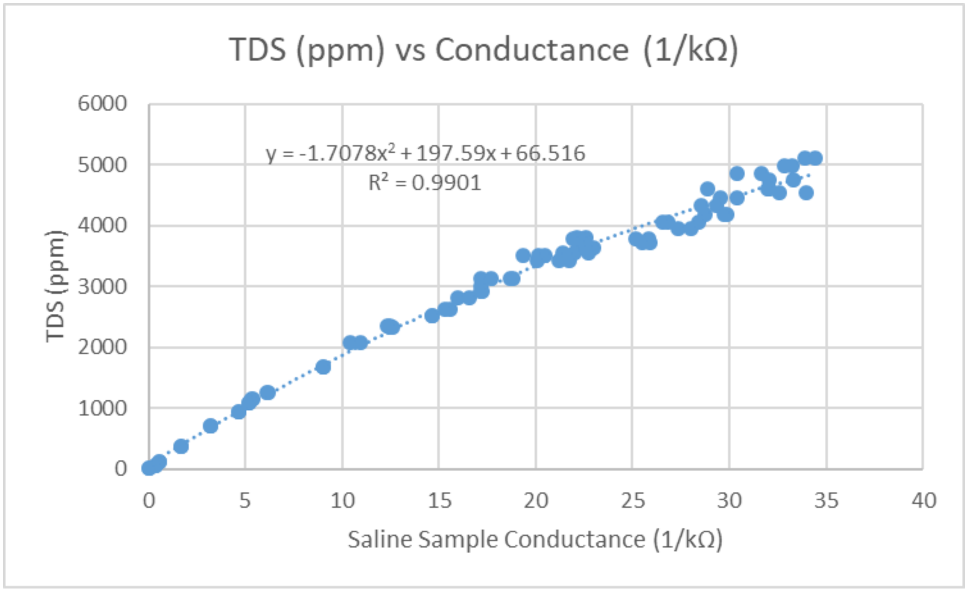
4.b.iii. Scatterplot


TDS (ppm) vs Probe ADC Reading
TDS (ppm)
6000
5000
4000
3000
2000
1000
0
0
500
1000
1500
2000
2500
3000
Probe ADC Reading


TDS (ppm) vs Probe Voltage Drop (V)
TDS (ppm)
6000
5000
4000
3000
2000
1000
0
0
0.5
1
1.5
2
2.5
Probe Voltage Drop (V)



4.b.iv. Function Graph of Mathematical Model for Calculating TDS



Discussion:

1. TDS has a non-linear relationship with ADC readings of the probe (which represents the measured resistance of the sampled solution). Research shows that TDS has direct linear relationship with conductance, the inverse of resistance, of the sampled saline solutions. As such, we use conductance, or the inverse of calculated resistance, as the main predictor. A simple linear regression gives a R^2 of 0.977, a second-order regression (as shown in the chart) gives a R^2 of 0.99.
2. The above chart shows the relationship between calculated conductance (1/1kΩ) and TDS (ppm). A close examination of the regression curve shows some minor deviation around 3000ppm. As such, the data set was

split into two categories: one above 3000ppm, another below 3000ppm. The according ADC readings are used as the cut-off line of categorization.

4.b.v. Equation

Explanation:

Three factors are found important in assessing TDS, including conductance, capacitance, and temperature:

a. Conductance: As demonstrated in 4.b.iii, TDS is best predicted by the conductance, or the reverse of resistance, which can be calculated from the probe readings. In addition, as demonstrated in 4.b.iii, a second-order regression performs better than linear regression.
b. Capacitance: the saline solution is in fact an electrolyte with the property of capacitance. With a DC current, the resistance will increase with time (charging process of a capacitor). The changing conductivity under DC current reflects the capacitance induced by the ionized components in the sample, i.e., TDS.
c. Temperature: research shows temperature affects the conductance of saline solutions; the scale of the effect varies at different temperature.

In conclusion, the three factors of conductance (reverse of resistance), changing conductance (reflecting the capacitance), and temperature are included in the regression equation. Due to the interwoven nature of three factors jointly affecting the status of DC currents, their interactions are also included in addition to their individual effects.

The regression form is:

TDS = intercept + p1 * x1 + p2 * x1^2 + p3 * x2 + p4 * x2^2 + p5 * x3 + p6 * x1 * x2 + p7 * x1 *x3 + p8 * x2 * x3 + p9 * x1 * x2 * x3, where

x1: the reverse of resistance calculated from probe readings (resistance = probe readings /(4095 - probe readings) * 10 kΩ (the reference resistor));

x2: the changing conductance: the reverse of changing resistance measured over time during one specific measurement cycle.

X3: the temperature measured using the reading from a NTC thermistor.

The device design is featured with altering currents in two opposite directs; data set of each direction are split around 3000ppm. So four regressions are developed for two directions X two ppm categories.

An array is employed to store the parameters of the regression, as presented below:

```
double p0[4] = {32017.31, -25.18, -11937.594, 163.668};
double p1[4] = {0, 172.248, 0, 0};
double p2[4] = {-4.425, -10.255, 0.18, -13.344};
double p3[4] = {-402.199, 0, 0, 0};
double p4[4] = {-0.651, -0.108, 0.704, -0.087};
double p5[4] = {-27100.487, 9.709, 7084.839, -156.414};
double p6[4] = {0, 0, 0, 5.656};
double p7[4] = {12.324, 0, 673.04, 134.046};
double p8[4] = {383.241, 1.273, -21.866, 2.07};
double p9[4] = {3.127, 2.793, -5.027, -1.634};
```

For example, for current direction A□B and ADC reading < 33 (the according TDS is > 3000ppm), the actual regression will be:

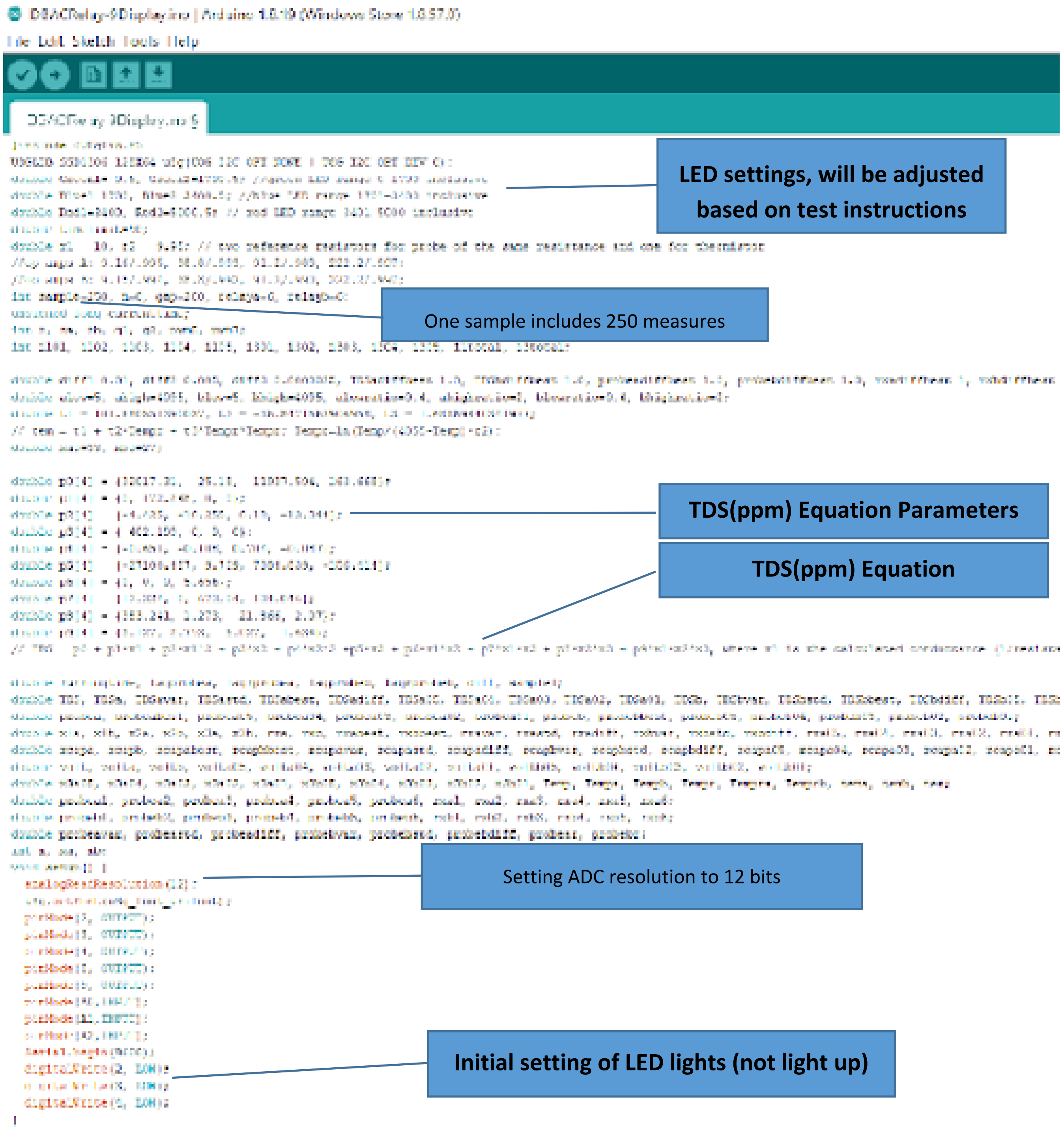
TDS = 31017.31 + 0 * x1 - 4.425 * x1^2 - 402.199 * x2 – 0.651 * x2^2 - 27100.487 * x3 + 0 * x1 * x2 + 12.324 * x1 * x3 + 383.241 * x2 * x3 + 3.127 * x1 * x2 * x3

The final TDS is the weighted average calculated from the two independent measurements.

In calibration, the average deviation from the true TDS (the absolute difference between our measures and the true TDS evenly distributed between 0 and 5000 ppm) is less than 47ppm at normal room temperatures (16-22°C).

4.b.vi. Code with Equation (highlighted with tag in bold black) and

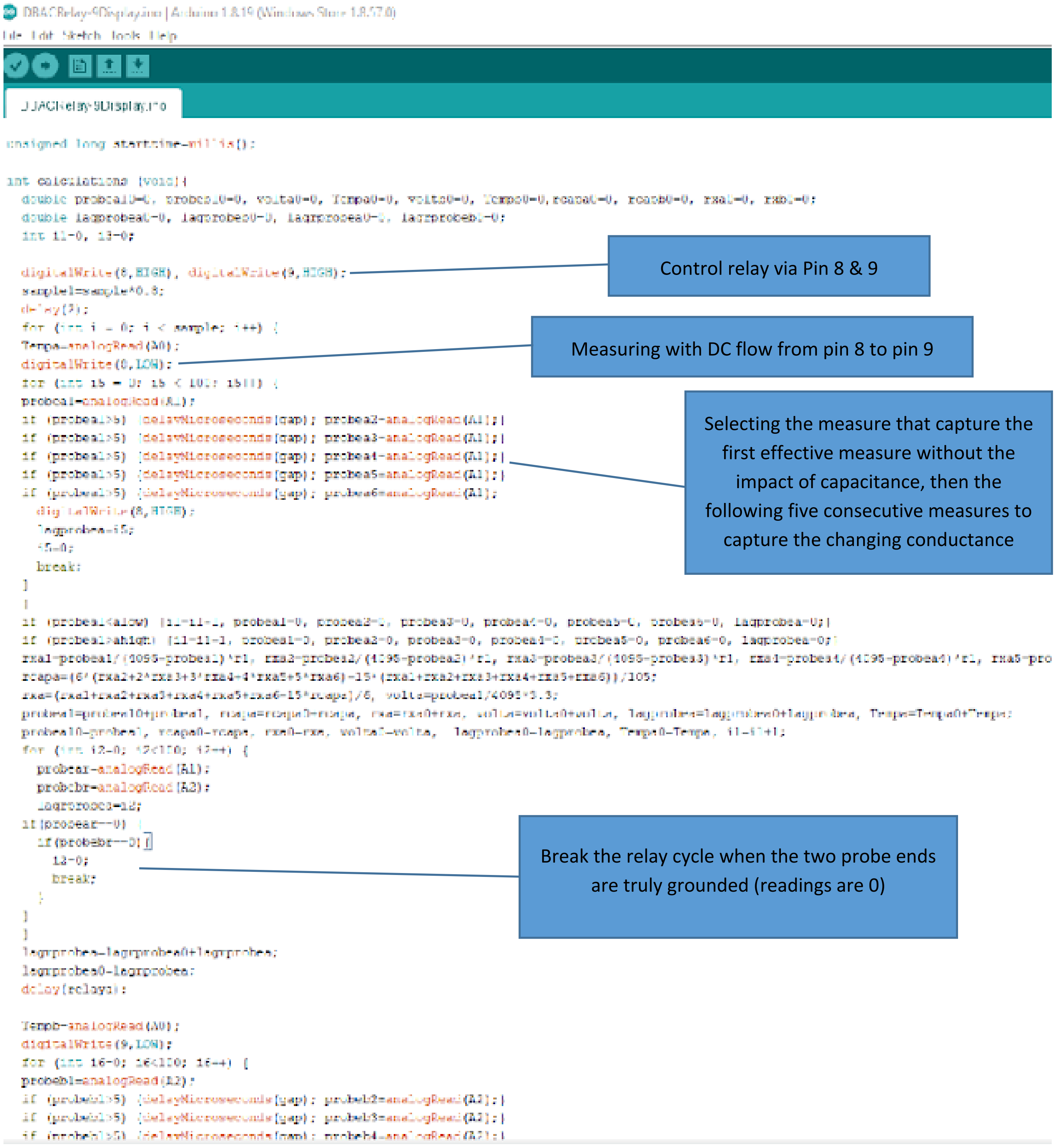
4.b.vii. Code of LED Settings (highlighted with tag in bold black)


DBACRelay-9Display.ino | Arduino 1.8.19 (Windows Store 1.8.57.0)
File Edit Sketch Tools Help
DBACRelay-9Display.ino §
LED settings, will be adjusted based on test instructions
One sample includes 250 measures
TDS(ppm) Equation Parameters
TDS(ppm) Equation
Setting ADC resolution to 12 bits
Initial setting of LED lights (not light up)

```
#include <U8glib.h>
U8GLIB_SSD1306_128X64 u8g(U8G_I2C_OPT_NONE | U8G_I2C_OPT_DEV_0);
double Green1=0.0, Green2=[illegible]; //green LED range [illegible] inclusive
double Blue1=1700, Blue2=3400.0; //blue LED range 1701-3400 inclusive
double Red1=3400, Red2=5000.0; // red LED range 3401-5000 inclusive
[illegible]
double r1 = 10, r2 = 9.95; // two reference resistors for probe of the same resistance and one for thermistor
//[illegible]
//[illegible]
int sample=250, n=0, gap=200, relaya=0, relayb=0;
unsigned long currentMillis;
int [illegible]
int I101, I102, I103, I104, I105, I301, I302, I303, I304, I305, I1total, I3total;

double diff1 [illegible]
double alow=5, ahigh=4095, blow=5, bhigh=4095, alowratio=0.4, ahighratio=2, blowratio=0.4, bhighratio=2;
double t1 = [illegible], t2 = [illegible], t3 = [illegible];
// tem = t1 + t2*Tempr + t3*Tempr*Tempr; Tempr=ln(Temp/(4095-Temp)*r2);
double [illegible]

double p0[4] = {22017.21, 25.11, 11907.506, 161.668};
double p1[4] = {[illegible]};
double p2[4] = {[illegible]};
double p3[4] = { 402.250, 0, 0, 0};
double p4[4] = {[illegible]};
double p5[4] = {-27106.127, 3.725, 7301.039, -226.414};
double p6[4] = {0, 0, 0, 5.658};
double p7[4] = {[illegible]};
double p8[4] = {553.241, 1.273, 21.566, 2.37};
double p9[4] = {[illegible]};
// TDS = p0 + p1*x1 + [illegible], where x1 is the calculated conductance [illegible]

double [illegible]
double TDS, TDSa, TDSavar, [illegible]
double [illegible]
double [illegible]
double [illegible]
double [illegible]
double [illegible]
double probea1, probea2, probea3, probea4, probea5, probea6, [illegible]
double [illegible]
double probeavar, probeastd, probeadiff, probebvar, probebstd, probebdiff, [illegible]
int [illegible]
void setup() {
  analogReadResolution(12);
  [illegible]
  pinMode(3, OUTPUT);
  pinMode(3, OUTPUT);
  pinMode(4, OUTPUT);
  pinMode(5, OUTPUT);
  pinMode(6, OUTPUT);
  pinMode(10, INPUT);
  pinMode(11, INPUT);
  pinMode(12, INPUT);
  Serial.begin(9600);
  digitalWrite(2, LOW);
  digitalWrite(3, LOW);
  digitalWrite(4, LOW);
}
```

4.b.vii. Code of Main Features

i. Code of managing current directions and probe measurements


DBACRelay-9Display.ino | Arduino 1.8.19 (Windows Store 1.8.57.0)
File Edit Sketch Tools Help
DBACRelay-9Display.ino

```
unsigned long starttime=millis();

int calculations (void){
  double probea10=0, probeb10=0, volta0=0, Tempa0=0, voltb0=0, Tempb0=0,rcapa0=0, rcapb0=0, rxa0=0, rxb0=0;
  double lagprobea0=0, lagprobeb0=0, lagrprobea0=0, lagrprobeb0=0;
  int i1=0, i3=0;

  digitalWrite(8,HIGH), digitalWrite(9,HIGH);
  sample1=sample*0.8;
  delay(3);
  for (int i = 0; i < sample; i++) {
  Tempa=analogRead(A0);
  digitalWrite(8,LOW);
  for (int i5 = 0; i5 < 100; i5++) {
  probea1=analogRead(A1);
  if (probea1>5) {delayMicroseconds(gap); probea2=analogRead(A1);}
  if (probea1>5) {delayMicroseconds(gap); probea3=analogRead(A1);}
  if (probea1>5) {delayMicroseconds(gap); probea4=analogRead(A1);}
  if (probea1>5) {delayMicroseconds(gap); probea5=analogRead(A1);}
  if (probea1>5) {delayMicroseconds(gap); probea6=analogRead(A1);
    digitalWrite(8,HIGH);
    lagprobea=i5;
    i5=0;
    break;
  }
  }
  if (probea1<alow) {i1=i1-1, probea1=0, probea2=0, probea3=0, probea4=0, probea5=0, probea6=0, lagprobea=0;}
  if (probea1>ahigh) {i1=i1-1, probea1=0, probea2=0, probea3=0, probea4=0, probea5=0, probea6=0, lagprobea=0;}
  rxa1=probea1/(4095-probea1)*r1, rxa2=probea2/(4095-probea2)*r1, rxa3=probea3/(4095-probea3)*r1, rxa4=probea4/(4095-probea4)*r1, rxa5=pro
  rcapa=(6*(rxa2+2*rxa3+3*rxa4+4*rxa5+5*rxa6)-15*(rxa1+rxa2+rxa3+rxa4+rxa5+rxa6))/105;
  rxa=(rxa1+rxa2+rxa3+rxa4+rxa5+rxa6-15*rcapa)/6, volta=probea1/4095*3.3;
  probea1=probea10+probea1, rcapa=rcapa0+rcapa, rxa=rxa0+rxa, volta=volta0+volta, lagprobea=lagprobea0+lagprobea, Tempa=Tempa0+Tempa;
  probea10=probea1, rcapa0=rcapa, rxa0=rxa, volta0=volta, lagprobea0=lagprobea, Tempa0=Tempa, i1=i1+1;
  for (int i2=0; i2<100; i2++) {
    probear=analogRead(A1);
    probebr=analogRead(A2);
    lagrprobea=i2;
  if(probear==0) {
    if(probebr==0){
      i2=0;
      break;
    }
  }
  }
  lagrprobea=lagrprobea0+lagrprobea;
  lagrprobea0=lagrprobea;
  delay(relays);

  Tempb=analogRead(A0);
  digitalWrite(9,LOW);
  for (int i6=0; i6<100; i6++) {
  probeb1=analogRead(A2);
  if (probeb1>5) {delayMicroseconds(gap); probeb2=analogRead(A2);}
  if (probeb1>5) {delayMicroseconds(gap); probeb3=analogRead(A2);}
  if (probeb1>5) {delayMicroseconds(gap); probeb4=analogRead(A2);}
```

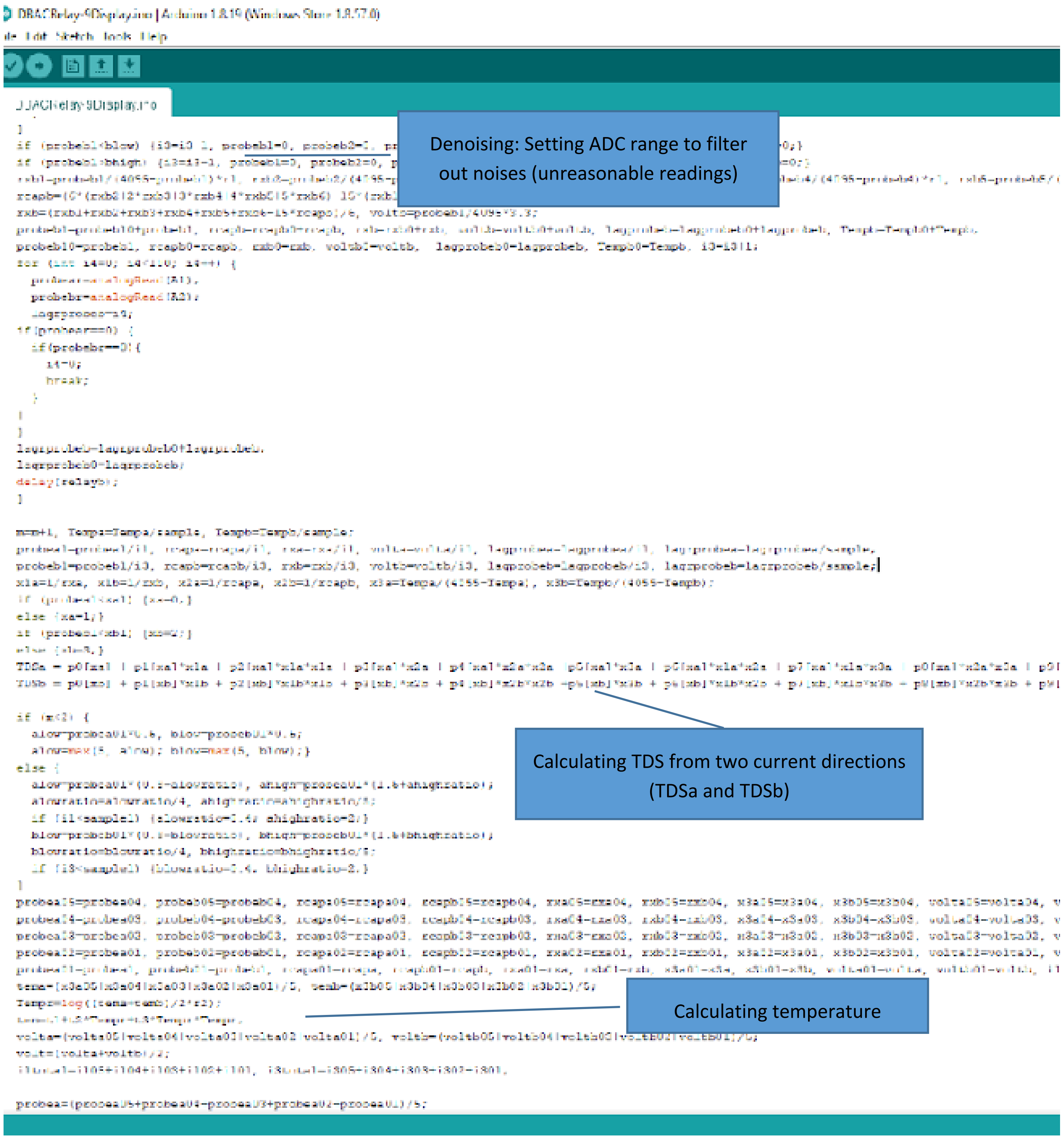
Control relay via Pin 8 & 9

Measuring with DC flow from pin 8 to pin 9

Selecting the measure that capture the first effective measure without the impact of capacitance, then the following five consecutive measures to capture the changing conductance

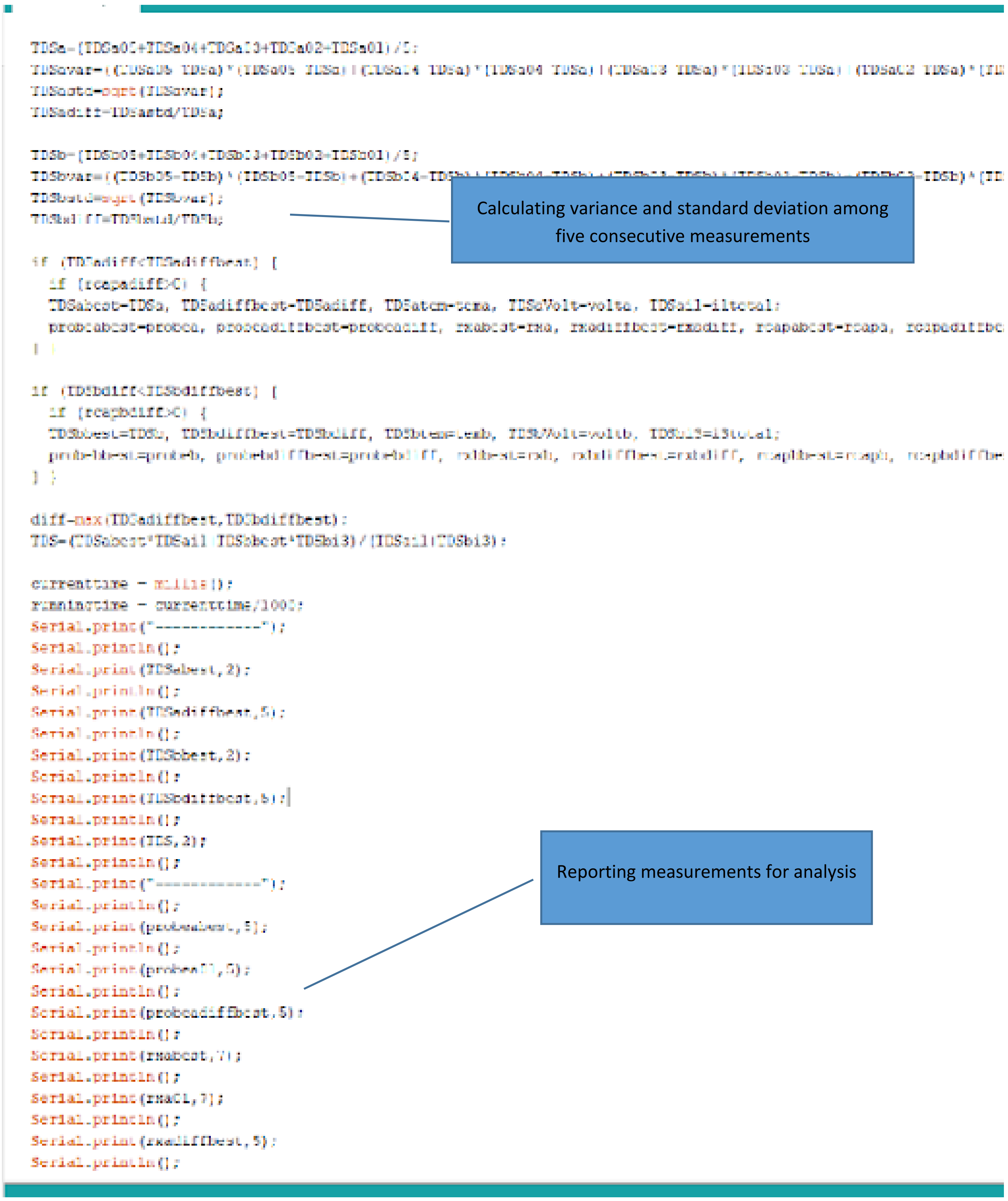
Break the relay cycle when the two probe ends are truly grounded (readings are 0)

ii. Code of denoising and calculating temperature, TDS, and others


DRACRelay-9Display.ino | Arduino 1.8.19 (Windows Store 1.8.57.0)
File Edit Sketch Tools Help
DRACRelay-9Display.ino
Denoising: Setting ADC range to filter out noises (unreasonable readings)
Calculating TDS from two current directions (TDSa and TDSb)
Calculating temperature

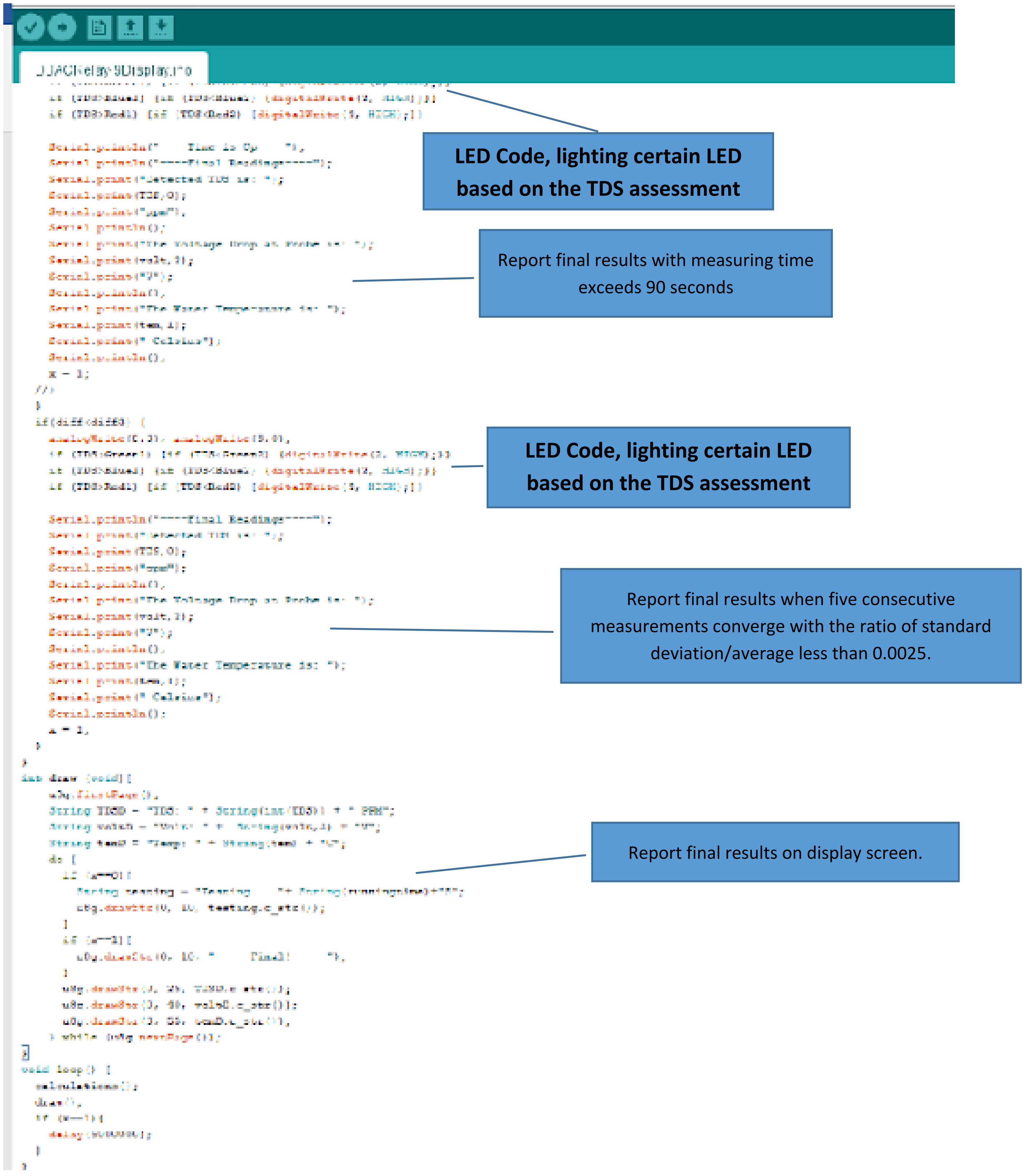
```
}
if (probebl<blow) {i3=i3-1, probebl=0, probeb2=0, pr[illegible] 0;}
if (probebl>bhigh) {i3=i3-1, probebl=0, probeb2=0, p[illegible] =0;}
rxbl=probebl/(4095-probebl)*r1, rxb2=probeb2/(4095-p[illegible] probeb4/(4095-probeb4)*r1, rxb5=probeb5/(
rcapb=(5*(rxb2+2*rxb3+3*rxb4+4*rxb5+5*rxb6)-15*(rxbl[illegible]
rxb=(rxbl+rxb2+rxb3+rxb4+rxb5+rxb6-15*rcapb)/6, voltb=probebl/4095*3.3;
probebl=probebl0+probebl, rcapb=rcapb0+rcapb, rxb=rxb0+rxb, voltb=voltb0+voltb, lagprobeb=lagprobeb0+lagprobeb, Tempb=Tempb0+Tempb;
probebl0=probebl, rcapb0=rcapb, rxb0=rxb, voltb0=voltb,  lagprobeb0=lagprobeb, Tempb0=Tempb, i3=i3+1;
for (int i4=0; i4<110; i4++) {
  probear=analogRead(A1);
  probebr=analogRead(A2);
  lagprobeb=i4;
if(probear==0) {
  if(probebr==0){
    i4=0;
    break;
  }
}
}
lagprobeb=lagprobeb0+lagprobeb;
lagprobeb0=lagprobeb;
delay(relayb);
}

m=m+1, Tempa=Tempa/sample, Tempb=Tempb/sample;
probeal=probeal/i1, rcapa=rcapa/i1, rxa=rxa/i1, volta=volta/i1, lagprobea=lagprobea/i1, lagprobea=lagprobea/sample;
probebl=probebl/i3, rcapb=rcapb/i3, rxb=rxb/i3, voltb=voltb/i3, lagprobeb=lagprobeb/i3, lagprobeb=lagprobeb/sample;
x1a=1/rxa, x1b=1/rxb, x2a=1/rcapa, x2b=1/rcapb, x3a=Tempa/(4095-Tempa), x3b=Tempb/(4095-Tempb);
if (probeal<xa1) {xa=0;}
else {xa=1;}
if (probebl<xb1) {xb=2;}
else {xb=3;}
TDSa = p0[xa] + p1[xa]*x1a + p2[xa]*x1a*x1a + p3[xa]*x2a + p4[xa]*x2a*x2a +p5[xa]*x3a + p6[xa]*x1a*x2a + p7[xa]*x1a*x3a + p8[xa]*x2a*x3a + p9[
TDSb = p0[xb] + p1[xb]*x1b + p2[xb]*x1b*x1b + p3[xb]*x2b + p4[xb]*x2b*x2b +p5[xb]*x3b + p6[xb]*x1b*x2b + p7[xb]*x1b*x3b + p8[xb]*x2b*x3b + p9[

if (m<2) {
  alow=probea01*0.6, blow=probeb01*0.6;
  alow=max(5, alow); blow=max(5, blow);}
else {
  alow=probea01*(0.8-alowratio), ahigh=probea01*(1.6+ahighratio);
  alowratio=alowratio/4, ahighratio=ahighratio/5;
  if (i1<sample1) {alowratio=0.4; ahighratio=2;}
  blow=probeb01*(0.8-blowratio), bhigh=probeb01*(1.6+bhighratio);
  blowratio=blowratio/4, bhighratio=bhighratio/5;
  if (i3<sample1) {blowratio=0.4, bhighratio=2.}
}
probea05=probea04, probeb05=probeb04, rcapa05=rcapa04, rcapb05=rcapb04, rxa05=rxa04, rxb05=rxb04, x3a05=x3a04, x3b05=x3b04, volta05=volta04, v
probea04=probea03, probeb04=probeb03, rcapa04=rcapa03, rcapb04=rcapb03, rxa04=rxa03, rxb04=rxb03, x3a04=x3a03, x3b04=x3b03, volta04=volta03, v
probea03=probea02, probeb03=probeb02, rcapa03=rcapa02, rcapb03=rcapb02, rxa03=rxa02, rxb03=rxb02, x3a03=x3a02, x3b03=x3b02, volta03=volta02, v
probea02=probea01, probeb02=probeb01, rcapa02=rcapa01, rcapb02=rcapb01, rxa02=rxa01, rxb02=rxb01, x3a02=x3a01, x3b02=x3b01, volta02=volta01, v
probea01=probeal, probeb01=probebl, rcapa01=rcapa, rcapb01=rcapb, rxa01=rxa, rxb01=rxb, x3a01=x3a, x3b01=x3b, volta01=volta, voltb01=voltb, i1
tema=(x3a05+x3a04+x3a03+x3a02+x3a01)/5, temb=(x3b05+x3b04+x3b03+x3b02+x3b01)/5;
Tempr=log((tema+temb)/2*r2);
[illegible]
volta=(volta05+volta04+volta03+volta02+volta01)/5, voltb=(voltb05+voltb04+voltb03+voltb02+voltb01)/5;
volt=(volta+voltb)/2;
i1total=i105+i104+i103+i102+i101, i3total=i305+i304+i303+i302+i301,

probea=(probea05+probea04+probea03+probea02+probea01)/5;
```

iii. Code of reporting raw measurement data


Calculating variance and standard deviation among five consecutive measurements
Reporting measurements for analysis

```
TDSa=(TDSa05+TDSa04+TDSa03+TDSa02+TDSa01)/5;
TDSavar=((TDSa05-TDSa)*(TDSa05-TDSa)+(TDSa04-TDSa)*(TDSa04-TDSa)+(TDSa03-TDSa)*(TDSa03-TDSa)+(TDSa02-TDSa)*(TD
TDSastd=sqrt(TDSavar);
TDSadiff=TDSastd/TDSa;

TDSb=(TDSb05+TDSb04+TDSb03+TDSb02+TDSb01)/5;
TDSbvar=((TDSb05-TDSb)*(TDSb05-TDSb)+(TDSb04-TD[illegible]-TDSb)*(TD
TDSbstd=sqrt(TDSbvar);
TDSbdiff=TDSbstd/TDSb;

if (TDSadiff<TDSadiffbest) {
  if (rcapadiff>0) {
  TDSabest=TDSa, TDSadiffbest=TDSadiff, TDSatem=tema, TDSaVolt=volta, TDSai1=i1total;
  probeabest=probea, probeadiffbest=probeadiff, rxabest=rxa, rxadiffbest=rxadiff, rcapabest=rcapa, rcapadiffbe
} }

if (TDSbdiff<TDSbdiffbest) {
  if (rcapbdiff>0) {
  TDSbbest=TDSb, TDSbdiffbest=TDSbdiff, TDSbtem=temb, TDSbVolt=voltb, TDSbi3=i3total;
  probebbest=probeb, probebdiffbest=probebdiff, rxbbest=rxb, rxbdiffbest=rxbdiff, rcapbbest=rcapb, rcapbdiffbe
} }

diff=max(TDSadiffbest,TDSbdiffbest);
TDS=(TDSabest*TDSai1+TDSbbest*TDSbi3)/(TDSai1+TDSbi3);

currenttime = millis();
runningtime = currenttime/1000;
Serial.print("------------");
Serial.println();
Serial.print(TDSabest,2);
Serial.println();
Serial.print(TDSadiffbest,5);
Serial.println();
Serial.print(TDSbbest,2);
Serial.println();
Serial.print(TDSbdiffbest,5);
Serial.println();
Serial.print(TDS,2);
Serial.println();
Serial.print("============");
Serial.println();
Serial.print(probeabest,5);
Serial.println();
Serial.print(probeaC1,5);
Serial.println();
Serial.print(probeadiffbest,5);
Serial.println();
Serial.print(rxabest,7);
Serial.println();
Serial.print(rxaC1,7);
Serial.println();
Serial.print(rxadiffbest,5);
Serial.println();
```

iv. LED code (highlighted with tag in bold black) and reporting final results



```
    [illegible]
    if (TDS>Blue1) {if (TDS<Blue2) {digitalWrite(3, HIGH);}}
    if (TDS>Red1) {if (TDS<Red2) {digitalWrite(4, HIGH);}}

    Serial.println("     Time is Up     ");
    Serial.println("-----Final Readings-----");
    Serial.print("Detected TDS is: ");
    Serial.print(TDS,0);
    Serial.print("ppm");
    Serial.println();
    Serial.print("The Voltage Drop at Probe is: ");
    Serial.print(volt,3);
    Serial.print("V");
    Serial.println();
    Serial.print("The Water Temperature is: ");
    Serial.print(tem,1);
    Serial.print(" Celsius");
    Serial.println();
    x = 1;
  //}
  }
  if(diff<diff0) {
    analogWrite(6,2); analogWrite(5,0);
    if (TDS>Green1) {if (TDS<Green2) {digitalWrite(2, HIGH);}}
    if (TDS>Blue1) {if (TDS<Blue2) {digitalWrite(3, HIGH);}}
    if (TDS>Red1) {if (TDS<Red2) {digitalWrite(4, HIGH);}}

    Serial.println("-----Final Readings-----");
    Serial.print("Detected TDS is: ");
    Serial.print(TDS,0);
    Serial.print("ppm");
    Serial.println();
    Serial.print("The Voltage Drop at Probe is: ");
    Serial.print(volt,3);
    Serial.print("V");
    Serial.println();
    Serial.print("The Water Temperature is: ");
    Serial.print(tem,1);
    Serial.print(" Celsius");
    Serial.println();
    x = 1;
  }
}
int draw (void){
    u8g.firstPage();
    String TDS0 = "TDS: " + String(int(TDS)) + " PPM";
    String volt0 = "Volt: " + String(volt,3) + "V";
    String tem0 = "Temp: " + String(tem) + "C";
    do {
      if (x==0){
        String testing = "Testing     "+ String(runningtime)+"s";
        u8g.drawStr(0, 10, testing.c_str());
      }
      if (x==1){
        u8g.drawStr(0, 10, "      Final!      ");
      }
      u8g.drawStr(0, 25, TDS0.c_str());
      u8g.drawStr(0, 40, volt0.c_str());
      u8g.drawStr(0, 55, tem0.c_str());
    } while (u8g.nextPage());
}
void loop() {
  calculations();
  draw();
  if (x==1){
    delay(600000);
  }
}
```

LED Code, lighting certain LED based on the TDS assessment

Report final results with measuring time exceeds 90 seconds

LED Code, lighting certain LED based on the TDS assessment

Report final results when five consecutive measurements converge with the ratio of standard deviation/average less than 0.0025.

Report final results on display screen.