LAMPIRAN

Program/*Coding* Arduino

```
//Arduino code
#include <EEPROM.h>
#include "SerialTransfer.h"
#include <SoftwareSerial.h>
#include "GravityTDS.h"
#include <OneWire.h>
#include <DallasTemperature.h>
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
#define TOM 7
SoftwareSerial s(2,3);
LiquidCrystal_I2C lcd(0x27, 20, 4);
#define ONE_WIRE_BUS 4
#define TdsSensorPin A1

GravityTDS gravityTds;

unsigned long last,chek;
float tdsValue = 0;
OneWire oneWire(ONE_WIRE_BUS);
DallasTemperature dallasTemperature(&oneWire);
float ph_act,PH;
float calibration_value = 21.63;
int phval = A0;
unsigned long int avgval;
int buffer_arr[10],temp;
int count,button,cek;
```

```
int t = 30;
int  set = 31; // penggatian waktu per detik
char error;
SerialTransfer myTransfer;
bool runn = false;
bool Status = false;
bool cal = false;
struct STRUCT {

  float ph;
  float tds;
  char S;
} testStruct;



void setup()
{
  Serial.begin(115200);; // Dubugging on hardware Serial 0
  s.begin(9600);
  myTransfer.begin(s);
  pinMode(TOM, INPUT_PULLUP);
  lcd.init();
  lcd.begin(20, 4);
  lcd.backlight();
  lcd.setCursor(0, 0);
  lcd.print("     Pengecekan      ");
  lcd.setCursor(0, 1);
  lcd.print("   PH, TDS & Suhu  ");
    gravityTds.setPin(TdsSensorPin);
```

```
    gravityTds.setAref(5.0);  //reference voltage on ADC, default 5.0V on Arduino
UNO
    gravityTds.setAdcRange(1024);  //1024 for 10bit ADC;4096 for 12bit ADC
    gravityTds.begin();
  delay(2000);
lcd.clear();


button = digitalRead(TOM);
if(button == LOW){
      cal = true;
    }
    while(cal == true){
   // gravityTds.setTemperature(dallasTemperature.getTempCByIndex(0));
    gravityTds.setTemperature(23.8);// set the temperature and execute temperature
compensation
    gravityTds.update();  //sample and calculate
    tdsValue = gravityTds.getTdsValue();  // then get the value
        lcd.setCursor(0, 0);
        lcd.print("calibrasi TDS");
        lcd.setCursor(0, 1);
        lcd.print("TDS: ");
        lcd.print(tdsValue);
        delay(1000);
    button = digitalRead(TOM);
    if(button == LOW){
      cal = false;
    }
    }
        lcd.clear();
    pH();
```

```
    TDS();
    lcd.setCursor(0, 3);
    lcd.print("Count : Ready");

    lcd.setCursor(0, 0);
    LCD();
}


void loop()
{
  button = digitalRead(TOM);
  if(button == LOW){
        lcd.setCursor(0, 3);
        lcd.print("Count :      ");
        count = 0;
        runn = true;
  }

  while((s.available()> 0) && (Status == true)){
        //error = ' ';
        error = s.read();
        //Serial.println(error);
  if(error == 'D'){
        lcd.setCursor(0, 3);
        lcd.print("Conect: ");
        lcd.print("DISCONNECT");
        Serial.print("error: ");
        Serial.println(error);
```

```
      error = ' ';
if(error == 'S'){
      lcd.setCursor(0, 3);
      lcd.print("Status: ");
      lcd.print("OK       ");
      Serial.print("error: ");
      Serial.println(error);
      error = ' ';
      Status = false;
      delay(1000);

  }
}

if(runn == true){
  pH();
  PH = ph_act;
  cek = 1;
}
while(cek == 1){
  lcd.setCursor(0, 3);
  lcd.print("Count : ");
  lcd.print(count);
     pH();
     if(PH - ph_act >= 1 || PH - ph_act <= -1 ){
     PH = ph_act;
     count = 0;
     }
     TDS();
```

```
if(count == 5){
    lcd.clear();
    LCD();
    lcd.setCursor(0, 3);
    lcd.print("Status: Send");
    Serial.println("Status: Send");
    s.write('X');
    delay(100);
    s.write('X');
    delay(100);
    s.write('X');
    delay(100);
    s.write('X');
    delay(100);
    s.write('X');
    delay(100);
    s.write('X');
    delay(100);
    s.write('X');
    delay(100);
    s.write('X');
    delay(100);
    s.write('X');
    delay(1000);
    testStruct.S = 'A';
    testStruct.ph = ph_act;
    testStruct.tds = tdsValue;
    myTransfer.txObj(testStruct, sizeof(testStruct));
    myTransfer.sendData(sizeof(testStruct));
    Serial.println("OK");
    cek = 0;
```

```
        runn = false;
        Status = true;
        delay(500);


}

if(PH - ph_act >= 0.06 || PH - ph_act <= -0.06 ){
  PH =  ph_act;
  count = 0;
}

  if( PH - ph_act >= 0.6 || PH - ph_act <= -0.6 ){
  PH =  ph_act;
  count = 0;
  }



  /*lcd.setCursor(0, 3);
  lcd.print("uno: ");
  lcd.print(runn);
  lcd.setCursor(8, 3);
  lcd.print(" node: ");
  if(t<10)lcd.print(" ");lcd.print(t);*/
  LCD();
  count++;
  Serial.println(count);
```

```
  }

}
```

Program NodeMCU ESP8266

```
#define _DEBUG_
#define _DISABLE_TLS_

#include "SerialTransfer.h"
#include <SoftwareSerial.h>
#include <ThingerESP8266.h>
#include <ESP8266WiFi.h>
#define USERNAME "pH_TDS"
#define DEVICE_ID "pH_TDS1"
#define DEVICE_CREDENTIAL "yfIq++60uNt@HS@f"

#define SSID "Khendra"
#define SSID_PASSWORD "wifipassword"

ThingerESP8266 thing(USERNAME, DEVICE_ID, DEVICE_CREDENTIAL);
SoftwareSerial s(D6,D5);
SerialTransfer myTransfer;
unsigned long chek,last;
int TDS;
float PH;
char Status,r;
bool kondisi;
struct STRUCT {
  float ph;
  float tds;
```

```
  char S;
} testStruct;
void setup() {
  Serial.begin(115200);
   s.begin(9600);
  myTransfer.begin(s);
  WiFi.begin(SSID, SSID_PASSWORD);
      while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  thing.add_wifi(SSID, SSID_PASSWORD);

  thing["pHTDS"] >> [](pson &out){
  out["PH"] = PH,2;
  out["TDS"] = TDS;
  };
}

void loop() {
  while((s.available()) && (kondisi == false)){
    r = ' ';
    r = s.read();
    Serial.println(r);
    if(r == 'X'){
          Serial.print("SENDING:....");
          Serial.println(r);
          kondisi = true;
     }
```

```
}
while (myTransfer.available()>0 && kondisi == true){
  myTransfer.rxObj(testStruct, sizeof(testStruct));
  Status = testStruct.S;
  PH = testStruct.ph,2;
  TDS = testStruct.tds + 0.5;
  Serial.print("NILAI TDS: ");
  Serial.print(testStruct.tds);
  Serial.println("ppm");
  Serial.print("NILAI pH: ");
  Serial.print(testStruct.ph,2);
  Serial.println("");
  Serial.println("");
  Serial.print("Status Terima: ");
  Serial.println(Status);
  chek = millis();
  last = chek;
  if(Status == 'A'){
    kondisi = false;
  }
}


    while (WiFi.status() != WL_CONNECTED) {
  delay(500);
  s.write('D');
  Serial.print(".");
}
 thing.handle();
chek = millis();
```

```
if((Status == 'A') && (chek - last >=60*1000)){
  last = chek;
  thing.write_bucket("buc_pHTDS","pHTDS");
    if(PH < 6 ){
  thing.call_endpoint("low_pHTDS","pHTDS");
  }
  if(PH >= 6 && PH <= 9){
  thing.call_endpoint("end_pHTDS","pHTDS");
  }
    if(PH > 9){
  thing.call_endpoint("high_pHTDS","pHTDS");
  }
  Status = ' ';
  Serial.print("Status Send: ");
  Serial.println(Status);
  s.write('S');
  }

}
```