LAMPIRAN

Lampiran 1 Program Rancang Bangun Alat di Arduino Ide

```
#include <Arduino.h>
#include "DHT.h"
#include <LiquidCrystal_I2C.h>
#include <FastLED.h>
#include "HardwareSerial.h"
//==========DEFINE
//LCD
#define LCD_ADDRESS 0x27
#define LCD_COLUMNS 16
#define LCD_ROWS 2
//PMS5003
#define PMS5003_RX  17
#define PMS5003_TX  16
//DHT22
#define DHT_PIN 4
#define DHTTYPE DHT22
//MQ135
#define MQ135_PIN 32
//LED
#define NUM_LEDS 1
#define DATA_PIN 25
CRGB leds[NUM_LEDS];
//BUZZER
#define BUZZER_PIN 19



//==========GLOBAL VARIABEL
//SENSOR DATA
```

```
double Temperature, Humidity;
uint16_t PM25;
float CO₂_PPM;

//PMS5003
struct pms5003data {
  uint16_t framelen;
  uint16_t pm10_standard, pm25_standard, pm100_standard;
  uint16_t pm10_env, pm25_env, pm100_env;
  uint16_t particles_03um, particles_05um, particles_10um, particles_25um,
particles_50um, particles_100um;
  uint16_t unused;
  uint16_t checksum;
};
#define PM25_BAIK   55
#define PM25_RINGAN 150
#define PM25_BAHAYA 250
int calibrationValue = 0;
//MQ135
const float a = 110.7432567;
const float b = -2.856935538;
const double ppmCalibration = 415.0;
float RO = 503994.0;
const float RLOAD = 10000.0;
float minPPM = 0;
float maxPPM = 0;
bool WarmUp = 0;
unsigned long warmUpMillis = 0;
const unsigned long intervalWarmUp = 1000UL*60UL*5;
#define PPM_BAIK 700
#define PPM_RINGAN 1000
```

```
#define PPM_BAHAYA 1500
//MENU
enum menuDisplay{
  MENU_PM25,
  MENU_CO2PPM,
  MENU_DHT,
  TOTAL_DISPLAY
};
int8_t display = 0;
char buff[17], buff1[17];
unsigned long lcdMillis = 0;
const unsigned long intervallcd = 1000UL*3;
//MQ135 CALIB
const uint8_t AVG_SIZE = 10;

float avgBuffer[AVG_SIZE];
uint8_t avgIndex = 0;
uint8_t avgCount = 0;
float avgSum = 0;
//POLLING
unsigned long previousMillis = 0;
const unsigned long interval = 1000UL; // 1000 ms = 1 detik
//==========CLASS DECLARATION
pms5003data PMS25;
HardwareSerial pm25Serial(2);
LiquidCrystal_I2C lcd(0x27,16,2);
DHT dht(DHT_PIN, DHTTYPE);

//==========FUNCTION
//LCD
void lcdBegin();
```

```
void displayCentered();
//FASTLED
void FastLEDBegin();
void updateLEDStatus();
//BUZZER
void buzzerBegin();
void updateBuzzer();
//PMS5003
void PMBegin();
bool PMStreamData(Stream *s);
void PMGetData();
int warnScalePM(float PM25);
//DHT
void DHTBegin();
void DHTReadData();
//MQ135
void MQ135Begin();
float findRO();
void AverageValuePush(float value);
float AverageValue();
void findMaxMinPPM();
float getPPM();
int warnScaleMQ(float PPM);

//==========MAIN FUNCTION
void displayData(int displayMenu);


void setup(){
  Serial.begin(115200);
  lcdBegin();
```

```
  FastLEDBegin();
  buzzerBegin();
  PMBegin();
  DHTBegin();
  MQ135Begin();
  findMaxMinPPM();
  display = 0;
  WarmUp = true;
}

void loop(){
  unsigned long currentMillis = millis();
  if (currentMillis - warmUpMillis >= intervalWarmUp)WarmUp = false;
  if (currentMillis - previousMillis >= interval){
    previousMillis = currentMillis;
    DHTReadData();
    PMGetData();
    CO2_PPM = getPPM();
    updateLEDStatus();
    updateBuzzer();
  }
  if (currentMillis - lcdMillis >= intervallcd){
    lcdMillis = currentMillis;
    display++;
    if(display > TOTAL_DISPLAY - 1) display = 0;
    lcd.clear();
  }
  displayData(display);

}
```

```
//==========FUNCTION
//LCD
void lcdBegin(){
  lcd.init();
  lcd.backlight();
}
void displayCentered(String text, int row){
  int len = text.length();
  int spaces = (LCD_COLUMNS - len) / 2;
  String padding = "";
  for (int i = 0; i < spaces; i++) {
    padding += " ";
  }
  lcd.setCursor(0, row);
  lcd.print(padding + text);
}
//FASTLED
void FastLEDBegin(){
  FastLED.addLeds<WS2812, DATA_PIN, GRB>(leds, NUM_LEDS);
  FastLED.setBrightness(80);
  leds[0] = CRGB::Black;
  FastLED.show();
}
void updateLEDStatus(){
  int pmStatus = warnScalePM(PM25);
  int ppmStatus = WarmUp ? 1 : warnScaleMQ(CO₂_PPM);

  int status = max(pmStatus, ppmStatus);

  switch(status){
    case 1: // Baik
```

```
      leds[0] = CRGB::Green;
      break;
    case 2: // Ringan
      leds[0] = CRGB::Yellow;
      break;
    case 3: // Sedang
      leds[0] = CRGB::Orange;
      break;
    case 4: // Bahaya
      leds[0] = CRGB::Red;
      break;
    default:
      leds[0] = CRGB::Blue;   // Error
      break;
  }

  FastLED.show();
}
//BUZZER
void buzzerBegin(){
  pinMode(BUZZER_PIN, OUTPUT);
  digitalWrite(BUZZER_PIN, HIGH);
}
void updateBuzzer(){
  int pmStatus  = warnScalePM(PM25);
  int ppmStatus = WarmUp ? 1 : warnScaleMQ(CO₂_PPM);
  if(pmStatus == 4 || ppmStatus == 4) digitalWrite(BUZZER_PIN, LOW);
  else digitalWrite(BUZZER_PIN, HIGH);
}
//PMS5003
void PMBegin(){
```

```
  Serial.println("==================================");
  Serial.println("    INITIALIZATION PM2.5 SENSOR     ");
  Serial.println("==================================");
  pm25Serial.begin(9600, SERIAL_8N1, PMS5003_TX, PMS5003_RX);
}
bool PMStreamData(Stream *s){
  if (! s->available()) {
    return false;
  }

  if (s->peek() != 0x42) {
    s->read();
    return false;
  }

  if (s->available() < 32) {
    return false;
  }

  uint8_t buffer[32];
  uint16_t sum = 0;
  s->readBytes(buffer, 32);

  for (uint8_t i=0; i<30; i++) {
    sum += buffer[i];
  }

  uint16_t buffer_u16[15];
  for (uint8_t i=0; i<15; i++) {
    buffer_u16[i] = buffer[2 + i*2 + 1];
    buffer_u16[i] += (buffer[2 + i*2] << 8);
```

```
  }

  memcpy((void *)&PMS25, (void *)buffer_u16, 30);

  if (sum != PMS25.checksum) {
    Serial.println("Checksum failure");
    return false;
  }

  return true;
}
void PMGetData(){
  Serial.println("====================================");
  Serial.println("             PM2.5 SENSOR           ");
  Serial.println("====================================");
  if (PMStreamData(&pm25Serial)) {
      Serial.print("PM 2.5: "); Serial.println(PMS25.pm25_env);
      PM25 = PMS25.pm25_env - calibrationValue;
  }
}
int warnScalePM(float PM25){
  if(PM25 < PM25_BAIK) return 1;
  else if(PM25 < PM25_RINGAN) return 2;
  else if(PM25 < PM25_BAHAYA) return 3;
  else return 4;
}
//DHT
void DHTBegin(){
  Serial.println("====================================");
  Serial.println("    INITIALIZATION DHT22 SENSOR     ");
  Serial.println("====================================");
```

```
  dht.begin();
}
void DHTReadData(){
  Serial.println("====================================");
  Serial.println("              DHT22 SENSOR          ");
  Serial.println("====================================");
  Temperature = dht.readTemperature();
  Humidity = dht.readHumidity();
  Serial.print("TEMPERATURE : "), Serial.println(Temperature, 2);
  Serial.print("HUMIDITY : "), Serial.println(Humidity, 2);
}
//MQ135
void MQ135Begin(){
  pinMode(MQ135_PIN, INPUT);
  analogSetPinAttenuation(MQ135_PIN, ADC_11db);
}
float findRO(){
  float adcRaw = analogRead(MQ135_PIN);
  float RsValue = ((4095.0 * RLOAD)/adcRaw) - RLOAD;
  float newRO = RsValue * exp(log(a/ppmCalibration)/b);
  AverageValuePush(newRO);
  float ROAVG = AverageValue();
  return ROAVG;
}
void AverageValuePush(float value){
    if (avgCount < AVG_SIZE)
    {
        avgBuffer[avgIndex] = value;
        avgSum += value;
        avgCount++;
    }
```

```
    else
    {
       avgSum -= avgBuffer[avgIndex];
       avgBuffer[avgIndex] = value;
       avgSum += value;
    }

    avgIndex = (avgIndex + 1) % AVG_SIZE;
}
float AverageValue(){
  if (avgCount == 0) return 0.0;
  return avgSum / avgCount;
}
void findMaxMinPPM(){
  minPPM = pow((1000/a),(1/b));
  maxPPM = pow((10/a),(1/b));
}
float getPPM(){
  float adcRaw = analogRead(MQ135_PIN);
  float RsValue = ((4095.0 * RLOAD)/adcRaw) - RLOAD;
  float rSrO = RsValue/RO;
  if(rSrO < maxPPM && rSrO > minPPM){
    float ppmValue = a * pow((float)RsValue/(float)RO, b);
    AverageValuePush(ppmValue);
    float ppmValue_avg = AverageValue();
    return ppmValue_avg;
  }else return -1.0;
}
int warnScaleMQ(float PPM){
  if(PPM < PPM_BAIK) return 1;
  else if(PPM < PPM_RINGAN) return 2;
```

```
  else if(PPM < PPM_BAHAYA) return 3;
  else return 4;
}

//==========MAIN FUNCTION
void displayData(int displayMenu){
  int warnPM = 0;
  int warnPPM = 0;
  switch (displayMenu){
  case MENU_PM25:
    warnPM = warnScalePM(PM25);
    sprintf(buff,"PM25 : %01d", PM25);
    lcd.setCursor(0,0); lcd.print(buff);
    lcd.setCursor(0,1);
    if (warnPM == 1) lcd.print("BAIK");
    else if (warnPM == 2) lcd.print("RINGAN");
    else if (warnPM == 3) lcd.print("SEDANG");
    else if (warnPM == 4) lcd.print("BAHAYA");
    break;
  case MENU_CO2PPM:
    if(WarmUp) displayCentered("WARMUP MQ135", 0);
    else{
      warnPPM = warnScaleMQ(CO2_PPM);
      sprintf(buff,"PPM : %0.1f", CO2_PPM);
      lcd.setCursor(0,0); lcd.print(buff);
      lcd.setCursor(0,1);
      if (warnPPM == 1) lcd.print("BAIK");
      else if (warnPPM == 2) lcd.print("BURUK");
      else if (warnPPM == 3) lcd.print("SEDANG");
      else if (warnPPM == 4) lcd.print("BAHAYA");
```

```
    }
    break;
  case MENU_DHT:
    sprintf(buff,"TEMP : %0.1f", Temperature);
    sprintf(buff1,"HUM : %0.1f", Humidity);
    lcd.setCursor(0,0); lcd.print(buff);
    lcd.setCursor(0,1); lcd.print(buff1);
    break;
  }
}
```

Lampiran 2 Datasheet MQ-135

TECHNICAL DATA **MQ-135 GAS SENSOR**

FEATURES

Wide detecting scope — Fast response and High sensitivity
Stable and long life — Simple drive circuit

APPLICATION

They are used in air quality control equipments for buildings/offices, are suitable for detecting of NH3,NOx, alcohol, Benzene, smoke,CO_2 ,etc.

SPECIFICATIONS

A. Standard work condition

Symbol	Parameter name	Technical condition	Remarks
Vc	Circuit voltage	5V±0.1	AC OR DC
V_H	Heating voltage	5V±0.1	ACOR DC
R_L	Load resistance	can adjust	
R_H	Heater resistance	33Ω±5%	Room Tem
P_H	Heating consumption	less than 800mw	

B. Environment condition

Symbol	Parameter name	Technical condition	Remarks
Tao	Using Tem	-10℃-45℃	
Tas	Storage Tem	-20℃-70℃	
R_H	Related humidity	less than 95%Rh	
O_2	Oxygen concentration	21%(standard condition)Oxygen concentration can affect sensitivity	minimum value is over 2%

C. Sensitivity characteristic

Symbol	Parameter name	Technical parameter	Remark 2
Rs	Sensing Resistance	30KΩ-200KΩ (100ppm NH_3)	Detecting concentration scopeI 10ppm-300ppm NH_3 10ppm-1000ppm Benzene 10ppm-300ppm Alcohol
α (200/50) NH_3	Concentration Slope rate	≤0.65	
Standard Detecting Condition	Temp: 20℃±2℃ Vc:5V±0.1 Humidity: 65%±5% Vh: 5V±0.1		
Preheat time	Over 24 hour		

D. Structure and configuration, basic measuring circuit

	Parts	Materials
1	Gas sensing layer	SnO2
2	Electrode	Au
3	Electrode line	Pt
4	Heater coil	Ni-Cr alloy
5	Tubular ceramic	Al2O3
6	Anti-explosion network	Stainless steel gauze (SUS316 100-mesh)
7	Clamp ring	Copper plating Ni
8	Resin base	Bakelite
9	Tube Pin	Copper plating Ni

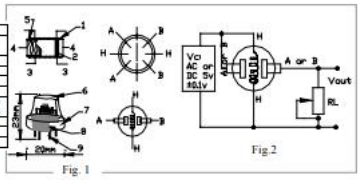
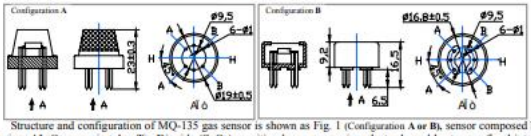


Fig.2

Fig. 1



Structure and configuration of MQ-135 gas sensor is shown as Fig. 1 (Configuration **A or B**), sensor composed by micro AL_2O_3 ceramic tube, Tin Dioxide (SnO_2) sensitive layer, measuring electrode and heater are fixed into a crust made by plastic and stainless steel net. The heater provides necessary work conditions for work of sensitive

Lampiran 3 Datasheet PMS5003

Digital universal particle concentration sensor

PMS5003 series data manual

Writer	Zhou Yong	Version	V2.3
Verifier	Zheng Haoxin	Date	2016-06-01

Main characteristics

- Zero false alarm rate
- Real-time response
- Correct data
- Minimum distinguishable particle diameter :0.3 micrometer
- High anti-interference performance because of the patent structure of six sides shielding
- Optional direction of air inlet and outlet in order to adapt the different design

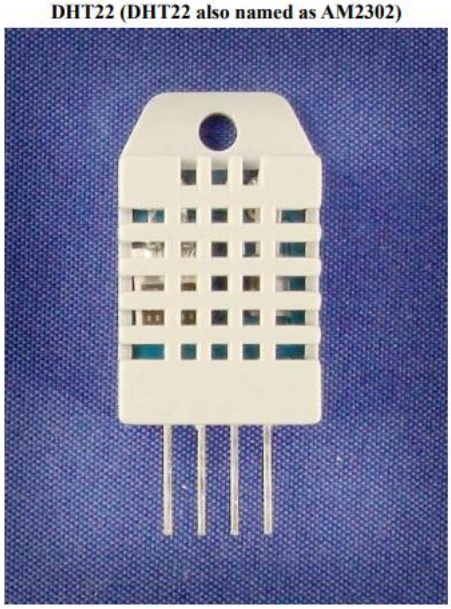
Lampiran 4 Datasheet DHT22

Aosong Electronics Co.,Ltd

Your specialist in innovating humidity & temperature sensors

Digital-output relative humidity & temperature sensor/module

DHT22 (DHT22 also named as AM2302)



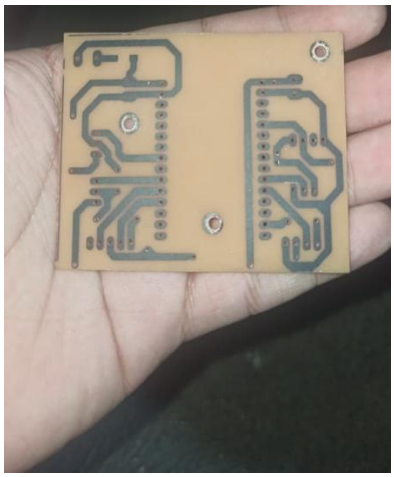
Capacitive-type humidity and temperature module/sensor

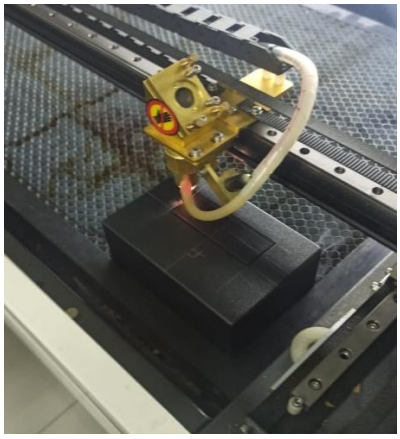
1

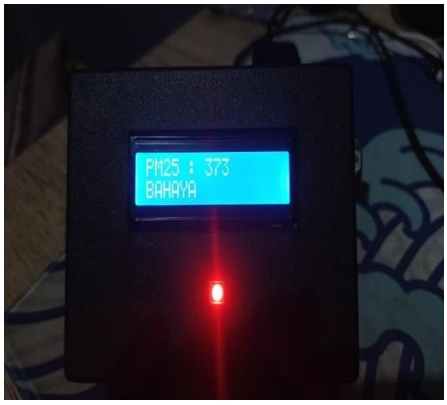
Thomas Liu (Business Manager)

Email: thomasliu198518@yahoo.com.cn

Lampiran 5 Proses Pembuatan







Lampiran 6 Data 1 (Ruangan Tertutup Tanpa Orang)

PMS5003 ((µg/m³)	MQ-135 (ppm)
	

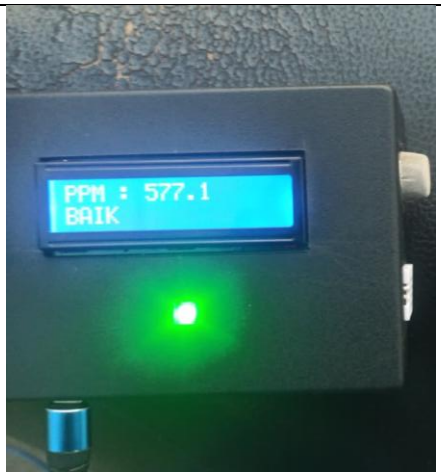
 PM25 : 40 BAIK	 PPM : 474.6 BAIK
 PM25 : 40 BAIK	 PPM : 467.4 BAIK
 PM25 : 42 BAIK	 PPM : 466.6 BAIK
 PM25 : 42 BAIK	 PPM : 471.2 BAIK

Lampiran 7 Data 2 (Ruangan Tertutup Dengan 3 Orang Merokok)

PMS5003 ((µg/m³)	MQ-135 (ppm)
	
	
	

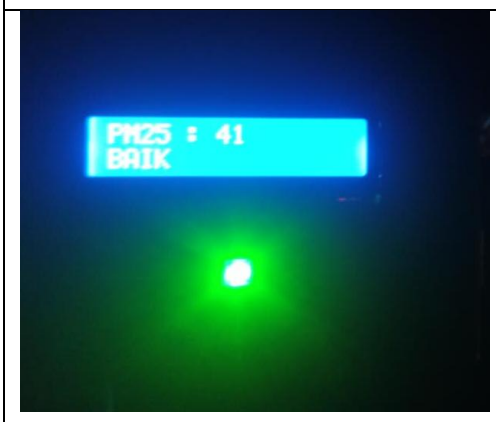
Lampiran 8 Data 3 (Tepi Jalan Raya dengan Lalu Lintas Padat)

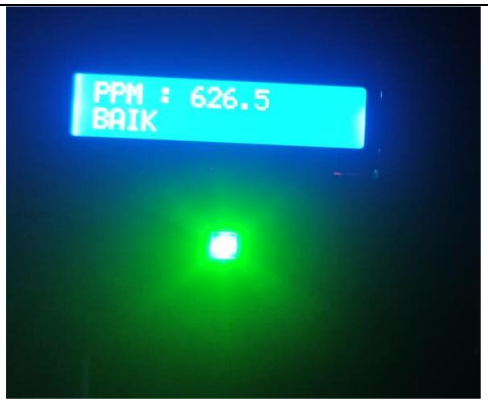
PMS5003 ((PM2.5)	MQ-135 (ppm)
	

 PM25 : 44 BAIK	 PPM : 574.8 BAIK
 PM25 : 43 BAIK	 PPM : 626.5 BAIK
 PM25 : 41 BAIK	 PPM : 617.3 BAIK

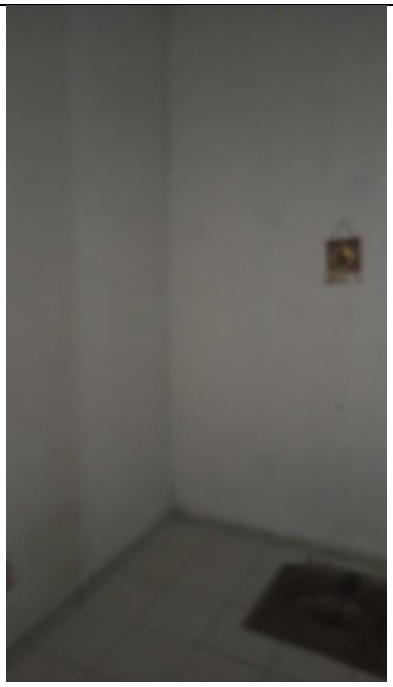
	

Lampiran 9 Data 4 (Jogging Track Undip)

PMS5003 ((µg/m³)	MQ-135 (ppm)
	
	

 PM25 : 42 BAIK	 PPM : 595.4 BAIK
 PM25 : 41 BAIK	 PPM : 604.2 BAIK
 PM25 : 43 BAIK	 PPM : 626.5 BAIK

Lampiran 10 Dokumentasi Pengambilan Data

Data 1	Data 2
	
Data 3	Data 4