LAMPIRAN

LAMPIRAN A Coding Monitoring

```
#include <SPI.h> // library
#include <LCD_I2C.h> // library LCD
LCD_I2C lcd(0x27); //pemanggil LCD
#include <DHT.h>  //pemnggil library
#define VCC1 13  //fungsi pemanggil pin
#define VCC2 12  //pemanggil pin
#define GND2 7   //pemanggil pin
#define VCC3 5   //pemanggil pin
#define GND3 2   //pemanggil pin
#define DHTPIN 6  //pemanggil pin
#define DHTTYPE DHT22
DHT dht(DHTPIN, DHTTYPE);
const int ph_Pin = A0;
const int Pompa_Asam = 10;    // pemanggil pin pompa asam
const int Pompa_Basa = 9;     // pemanggil pin pompa basa
const int heater = 11;  // pemangil pin heater
const int sprayer = 8;  // pemanggil pin sprayer
const int kipasin = 4;  // pemanggil pin kipasin
const int kipasout = 3; // pemanggil pin kipasout
int nilai_analog_PH;
double Teganganph;
float Po = 0;
float PH_step;
int h;
float PH4 = 3.1;  // tegangan untuk pembacaan pH 4
float PH7 = 2.67; // tegangan untuk pembacaan pH 7

void setup()
{
  pinMode(VCC1, OUTPUT);
  digitalWrite(VCC1, HIGH);
  pinMode(VCC2, OUTPUT);
  digitalWrite(VCC2, HIGH);
  pinMode(GND2, OUTPUT);
  digitalWrite(GND2, LOW);
  pinMode(VCC3, OUTPUT);
  digitalWrite(VCC3, HIGH);
```

```
pinMode(GND3, OUTPUT);
digitalWrite(GND3, LOW);
pinMode(Pompa_Asam, OUTPUT);
digitalWrite(Pompa_Asam, HIGH);
pinMode(Pompa_Basa, OUTPUT);
digitalWrite(Pompa_Basa, HIGH);
pinMode(heater, OUTPUT);
digitalWrite(heater, HIGH);
pinMode(sprayer, OUTPUT);
digitalWrite(sprayer, HIGH);
pinMode(kipasin, OUTPUT);
digitalWrite(kipasin, HIGH);
pinMode(kipasout, OUTPUT);
digitalWrite(kipasout, HIGH);
pinMode (ph_Pin, INPUT);
Serial.begin(9600);
dht.begin();
lcd.begin();
lcd.backlight();
lcd.setCursor(5,1);
lcd.setCursor(0,0);
  lcd.print("ALAT KENDALI PID");
  delay(2000);
  lcd.clear();
  lcd.setCursor(0,0);
  lcd.print("Starting");
  delay(1000);
  lcd.setCursor(0,0);
  lcd.print("Starting .");
  delay(1000);
  lcd.setCursor(0,0);
  lcd.print("Starting ..");
  delay(500);
  lcd.setCursor(0,0);
  lcd.print("Starting ...");
  delay(500);
  lcd.setCursor(0,0);
  lcd.print("Starting ....");
  delay(500);
```

```
lcd.setCursor(0,0);
lcd.print("Starting .....");
delay(500);
lcd.setCursor(0,0);
lcd.print("Starting ......");
delay(500);
lcd.setCursor(0,0);
lcd.print("Starting .......");
delay(1500);
lcd.clear();
lcd.setCursor(0,1);
lcd.print("Reading Config");
delay(1500);
lcd.clear();
lcd.setCursor(0,0);
lcd.print("Reading Config");
lcd.setCursor(0,1);
lcd.print("Loading Config");
delay(1500);
lcd.clear();
lcd.setCursor(0,0);
lcd.print("Reading Config");
lcd.setCursor(0,1);
lcd.print("Config Loaded");
delay(1500);
lcd.clear();
lcd.setCursor(0,0);
lcd.print("Config Loaded");
lcd.setCursor(0,1);
lcd.print("Init Sensors");
delay(1500);
lcd.clear();
lcd.setCursor(0,0);
lcd.print("Init Sensors");
lcd.setCursor(0,1);
lcd.print("Sensors Online");
delay(1500);
lcd.clear();
lcd.setCursor(0,0);
```

```
    lcd.print("Sensors Online");
    lcd.setCursor(0,1);
    lcd.print("Successful Start");
    lcd.clear();
    delay(2500);
}

void loop()
{
  lcd.clear();
  lcd.setCursor(0,0);
  lcd.print("PH Air = ");
  lcd.print(Po,1);
  lcd.setCursor(0,1);
  lcd.print("Humidity = ");
  lcd.print(h,1);
  delay(5000);
}
```

LAMPIRAN B CODING *BUMP TEST*

```
#include <PID_v1.h>     //library PID
#include "DHT.h"        //library DHT
#include <LCD_I2C.h>    //library LCD
#define VCC1 13         //pin pemanggil tampilan LCD
#define VCC2 12
#define GND2 7
#define VCC3 5
#define GND3 2
#include <iostream>
#include <chrono>
#include <thread>

// Simulated pH and humidity measurements
double getPHMeasurement() {
    // Replace this with actual code to get pH measurement
    return 6.8; // Simulated pH value
}

void adjustPHLevel(double changeAmount) {
    // Replace this with actual code to adjust pH level
}

double getHumidityMeasurement() {
    // Replace this with actual code to get humidity measurement
    return 50.0; // Simulated humidity value
}

void adjustHumidity(double changeAmount) {
    // Replace this with actual code to adjust humidity level
{

// Simple PID controller
class PIDController {
public:
    PIDController(double kp, double ki, double kd)
        : kp_(kp), ki_(ki), kd_(kd), prevError_(0), integral_(0)
}
```

```
    double calculate(double setpoint, double currentMeasurement) {
        double error = setpoint - currentMeasurement;
        integral_ += error;
        double derivative = error - prevError_;

        double controlOutput = kp_ * error + ki_ * integral_ + kd_
* derivative;

        prevError_ = error;

        return controlOutput;
    }

private:
    double kp_;
    double ki_;
    double kd_;
    double prevError_;
    double integral_;
};
// Perform a bump test
void performBumpTest(double setpoint, double changeAmount, double
testDuration,
                     double (*getMeasurement)(), void
(*adjustLevel)(double)) {
    double initialMeasurement = getMeasurement();
    double targetMeasurement = initialMeasurement + changeAmount;

    std::cout << "Initial Measurement: " << initialMeasurement <<
std::endl;

    // Apply the change
    adjustLevel(changeAmount);

    // Record measurements over time
    auto startTime = std::chrono::steady_clock::now();
    while
(std::chrono::duration_cast<std::chrono::seconds>(std::chrono::ste
ady_clock::now() - startTime).count() < testDuration) {
```

```
        double currentTime =
std::chrono::duration_cast<std::chrono::seconds>(std::chrono::stea
dy_clock::now() - startTime).count();
        double currentMeasurement = getMeasurement();
        std::cout << "Time: " << currentTime << "s | Measurement:
" << currentMeasurement << std::endl;
        std::this_thread::sleep_for(std::chrono::seconds(1));
    }

    // Return to initial level
    adjustLevel(-changeAmount);
}

int main({
    // Perform pH bump test with a change of 0.5 pH and a test
duration of 60 seconds
    performBumpTest(6.8, 0.5, 60, getPHMeasurement,
adjustPHLevel);
    // Perform humidity bump test with an increase of 10% humidity
and a test duration of 60 seconds
    performBumpTest(50.0, 10.0, 60, getHumidityMeasurement,
adjustHumidity);

    return 0;
}
```

LAMPIRAN C Coding PID

```
#include <PID_v1.h>     //library PID
#include "DHT.h"        //library DHT
#include <LCD_I2C.h>    //library LCD
#define VCC1 13         //pin pemanggil tampilan LCD
#define VCC2 12
#define GND2 7
#define VCC3 5
#define GND3 2

#define PH_SENSOR_PIN A0              // pH meter Analog output to
Arduino Analog Input 0

#define DHT_SENSOR_PIN 6            //DHT22 output
#define DHT_SENSOR_TYPE DHT22

#define POMPA_PH_UP_RELAY_PIN 9       // Pompa larutan basa
#define POMPA_PH_DOWN_RELAY_PIN 10       // Pompa larutan asam
#define POMPA_SPRAYER_RELAY_PIN 8   // Pompa sprayer
#define KIPAS_IN_RELAY_PIN 4      // pin kipas in
#define KIPAS_OUT_RELAY_PIN 3    //pin kipas out

#define SUPPLY_VOLTAGE 5.0            // Tegangan supply
#define PH_SENSOR_OFFSET 0.3          // tegangan offset untuk
sensor pH
#define PH_SENSOR_CONSTANT 3.5        // konstanta konversi untuk
pH
#define ARRAY_LENGTH 40               // jumlah data sampel

#define PH_SAMPLING_INTERVAL 20       // interval waktu
pengambilan sampel sensor pH
#define KELEMBAPAN_SAMPLING_INTERVAL 20     // interval waktu
pengambilan sampel sensor kelembapan
#define PRINT_INTERVAL 800            // interval waktu print ke
serial monitor

#define PH4_CONSTANT 3.1      //tegangan pH=4
#define PH7_CONSTANT 2.67     //tegangan pH=7
```

```
LCD_I2C lcd(0x27);
DHT dht(DHT_SENSOR_PIN, DHT_SENSOR_TYPE);

double Voltage_pH;
double main_temperature = 25.0;  // temperatue suhu ruang

int pH_Array[ARRAY_LENGTH];   // Store the average value of the
sensor feedback
int pH_ArrayIndex = 0;
double pH_step;

// Set PID: Kelembapan
double x_Kelembapan = 64;          // batas bawah dari setpoint
Kelembapan dengan toleransi 5%
double y_Kelembapan = 66;          // batas atas dari setpoint
Kelembapan dengan toleransi 5%

// Define Variables we'll be connecting to
double Setpoint_Kelembapan, Input_Kelembapan, Output_Kelembapan;

// Specify the links and initial tuning parameters
double Kp_kipas = -5400;
double Ki_kipas = -225;
double Kd_kipas = -32400;
PID PID_Kelembapan_down(&Input_Kelembapan, &Output_Kelembapan,
&Setpoint_Kelembapan, Kp_kipas, Ki_kipas, Kd_kipas, REVERSE);

double Kp_sprayer = 2514.3;
double Ki_sprayer = 126;
double Kd_sprayer = 12571.5;
PID PID_Kelembapan_up(&Input_Kelembapan, &Output_Kelembapan,
&Setpoint_Kelembapan, Kp_sprayer, Ki_sprayer, Kd_sprayer, DIRECT);

// Set PID: pH
double x_pH = 6.305;     // batas bawah dari setpoint pH dengan
toleransi 3%
double y_pH = 6.695;     // batas atas dari setpoint pH dengan
toleransi 3
```

```
// Define Variables we'll be connecting to
double Setpoint_pH, Input_pH, Output_pH;

// Specify the links and initial tuning parameters
double Kp_pompa_down = -1187;
double Ki_pompa_down = -198;
double Kd_pompa_down = -1780.5;
PID PID_pH_down(&Input_pH, &Output_pH, &Setpoint_pH,
Kp_pompa_down, Ki_pompa_down, Kd_pompa_down, REVERSE);

double Kp_pompa_up = 1240;
double Ki_pompa_up = 207;
double Kd_pompa_up = 1860;
PID PID_pH_up(&Input_pH, &Output_pH, &Setpoint_pH, Kp_pompa_up,
Ki_pompa_up, Kd_pompa_up, DIRECT);

int WindowSize = 3000;
unsigned long windowStartTime;

bool pH_down_state, pH_up_state, Kelembapan_up_state,
Kelembapan_down_state = LOW;

// Setup routine runs once when you press reset:
void setup(void)
{


  pinMode(VCC1, OUTPUT);
  digitalWrite(VCC1, HIGH);
  pinMode(VCC2, OUTPUT);
  digitalWrite(VCC2, HIGH);
  pinMode(GND2, OUTPUT);
  digitalWrite(GND2, LOW);
  pinMode(VCC3, OUTPUT);
  digitalWrite(VCC3, HIGH);
  pinMode(GND3, OUTPUT);
  digitalWrite(GND3, LOW);

  pinMode(POMPA_PH_UP_RELAY_PIN, OUTPUT);
```

```
  pinMode(POMPA_PH_DOWN_RELAY_PIN, OUTPUT);
  pinMode(POMPA_SPRAYER_RELAY_PIN, OUTPUT);
  pinMode(KIPAS_IN_RELAY_PIN, OUTPUT);
  pinMode(KIPAS_OUT_RELAY_PIN, OUTPUT);

  Serial.begin(9600);                              // baud rate:9600
  windowStartTime = millis();
  dht.begin();
  lcd.begin();
  lcd.backlight();

  lcd.setCursor(0, 0);
  lcd.print("S_Kelembapan: 65%");
  lcd.setCursor(0, 2);
  lcd.print("S_pH: 6.3");
  delay(2500);


  // initialize the variables we're linked to
  Setpoint_Kelembapan = 65;
  Setpoint_pH = 6.5;

  // tell the PID to range between 0 and the full window size
  PID_Kelembapan_down.SetOutputLimits(0, WindowSize);
  PID_Kelembapan_up.SetOutputLimits(0, WindowSize);

  // turn the PID on
  PID_Kelembapan_down.SetMode(AUTOMATIC);
  PID_Kelembapan_up.SetMode(AUTOMATIC);

  PID_Kelembapan_down.SetTunings(Kp_kipas, Ki_kipas, Kd_kipas);
  PID_Kelembapan_up.SetTunings(Kp_sprayer, Ki_sprayer,
Kd_sprayer);

  // tell the PID to range between 0 and the full window size
  PID_pH_down.SetOutputLimits(0, WindowSize);
  PID_pH_up.SetOutputLimits(0, WindowSize);

  // turn the PID on
```

```
  PID_pH_down.SetMode(AUTOMATIC);
  PID_pH_up.SetMode(AUTOMATIC);

  PID_pH_down.SetTunings(Kp_pompa_down, Ki_pompa_down,
Kd_pompa_down);
  PID_pH_up.SetTunings(Kp_pompa_up, Ki_pompa_up, Kd_pompa_up);

  //  Initial condition - Relay
  digitalWrite(POMPA_PH_UP_RELAY_PIN, pH_up_state);
  digitalWrite(POMPA_PH_DOWN_RELAY_PIN, pH_down_state);
  digitalWrite(POMPA_SPRAYER_RELAY_PIN, Kelembapan_up_state);
  digitalWrite(KIPAS_IN_RELAY_PIN, Kelembapan_down_state);
  digitalWrite(KIPAS_OUT_RELAY_PIN, Kelembapan_down_state);
}

// Loop routine runs over and over again forever
void loop(void)
{
  static unsigned long kelembapan_sampling_time = millis();
  static unsigned long ph_sampling_time = millis();
  static unsigned long processing_time = millis();

  if (millis() - kelembapan_sampling_time >
KELEMBAPAN_SAMPLING_INTERVAL)
  {
    Input_Kelembapan = dht.readHumidity();

    kelembapan_sampling_time = millis();
  }

  if (millis() - ph_sampling_time > PH_SAMPLING_INTERVAL) {
    pH_Array[pH_ArrayIndex++] = analogRead(PH_SENSOR_PIN);

    if (pH_ArrayIndex > ARRAY_LENGTH) {
      pH_ArrayIndex = 0;
    }

    Voltage_pH = (average_array(pH_Array, ARRAY_LENGTH) *
SUPPLY_VOLTAGE) / 1023.0;
```

```
    pH_step = (PH4_CONSTANT - PH7_CONSTANT) / 3;
    Input_pH = 7.00 + ((PH7_CONSTANT - Voltage_pH) / pH_step);

    ph_sampling_time = millis();
  }

  // Every 800 milliseconds, print a numerical, convert the state
of the LED indicator
  if (millis() - processing_time > PRINT_INTERVAL) {
    Serial.println("-> Input Value");
    Serial.print("Kelembapan: ");
    Serial.println(Input_Kelembapan);
    Serial.print("PH: ");
    Serial.println(Input_pH);
    Serial.println();
h
    PID_Kelembapan_down.Compute();
    PID_Kelembapan_up.Compute();

    PID_pH_down.Compute();
    PID_pH_up.Compute();
    //mengirim data ke serial port
    Serial.println("-> Output Value");
    Serial.print("Output (Kelembapan): ");
    Serial.println(Output_Kelembapan);
    Serial.print("Output (pH): ");
    Serial.println(Output_pH);
    Serial.println();

    lcd.clear();
    lcd.setCursor(0, 0);
    lcd.print("Kelembapan: ");
    lcd.print(Input_Kelembapan);

    lcd.setCursor(0, 2);
    // lcd.print((char)223);
    lcd.print("pH: ");
    lcd.print(Input_pH);
```

```
    if (millis() - windowStartTime > WindowSize) {
      // time to shift the Relay Window
      windowStartTime += WindowSize;
    }

    if (Output_Kelembapan < millis() - windowStartTime) {
      Kelembapan_up_state = LOW;
      Kelembapan_down_state = HIGH;
    } else {
      Kelembapan_up_state = HIGH;
      Kelembapan_down_state = LOW;
    }

    if (Input_Kelembapan > x_Kelembapan && Input_Kelembapan <
y_Kelembapan || Input_Kelembapan == 0) {
      Kelembapan_down_state  = LOW;
      Kelembapan_up_state  = LOW;
    }

    if (Output_pH < millis() - windowStartTime) {
      pH_up_state = LOW;
      pH_down_state = HIGH;
    } else {
      pH_down_state = LOW;
      pH_up_state = HIGH;
    }

    if (Input_pH > x_pH && Input_pH < y_pH) {
      pH_up_state = LOW;
      pH_down_state = LOW;
    }
//mengirim data ke port serial
//memindahkan baris data berikutnya
    Serial.println("-> Output");
    Serial.print("POMPA pH Up (1): ");
    Serial.println(pH_up_state);
    Serial.print("POMPA pH Down (2): ");
    Serial.println(pH_down_state);
    Serial.print("POMPA SPRAYER (3): ");
```

```
    Serial.println(Kelembapan_up_state);
    Serial.print("KIPAS IN (4): ");
    Serial.println(Kelembapan_down_state);
    Serial.print("KIPAS OUT (5): ");
    Serial.println(Kelembapan_down_state);

    Serial.println("\n");

    digitalWrite(POMPA_PH_UP_RELAY_PIN, pH_up_state); //keadaan on
off
    digitalWrite(POMPA_PH_DOWN_RELAY_PIN, pH_down_state);
    //keadaan on off
    digitalWrite(POMPA_SPRAYER_RELAY_PIN, Kelembapan_up_state);
    //keadaan on off
    digitalWrite(KIPAS_IN_RELAY_PIN, Kelembapan_down_state);
    //keadaan on off
    digitalWrite(KIPAS_OUT_RELAY_PIN, Kelembapan_down_state);
    //keadaan on off

    processing_time = millis();
  }
}

double average_array(int* arr, int number)
{
  int max_value, min_value;
  double avg;
  long amount = 0;

  if (number <= 0) {
    Serial.println("Error number for the array to averaging!\n");
    return 0;
  }

  // less than 5, calculated directly statistics
  if (number < 5) {
    for (int i = 0; i < number; i++) {
      amount += arr[i];
    }
```

```
    avg = amount / number;
  } else {
    if (arr[0] < arr[1]) {
      min_value = arr[0];     //pernyataan on off
      max_value = arr[1];     //pernyataan on off
    } else {
      min_value = arr[1];     //pernyataan on off
      max_value = arr[0];     //pernyataan on off
    }

    for (int i = 2; i < number; i++) {
      if (arr[i] < min_value) {
        amount += min_value;                  // arr < min
        min_value = arr[i];
      } else if (arr[i] > max_value) {
        amount += max_value;                  // arr > max
        max_value = arr[i];
      } else {
        amount += arr[i];                     // min <= arr <= max
      }
    }

    avg = (double) amount / (number - 2);
  }

  return avg;
}
```

LAMPIRAN D CODING KARAKTERISTIK KENDALI PID

```
t=data4(:,1);
sp=data4(:,2);
y=data4(:,3);

plot(t,sp,'k',t,y,'b')
hold on
grid on

ylabel('Nilai pH')
xlabel('waktu')

stepinfo(y,t,6.5)

steady_state_error = sp(end) - y(end)

t=Data1(:,1);
y=Data1(:,3);
sp=Data1(:,2);
t1=Data2(:,1);
y1=Data2(:,3);
sp1=Data2(:,2);
t2=Data3(:,1);
y2=Data3(:,3);
sp2=Data3(:,2);


plot(t,y,'k',t1,y1,'g',t2,y2,'b',t,sp,'r')
hold on
grid on
ylim([900, 1300])
ylabel('pH')
xlabel('Time (s)')
title('Control')
legend('Morning', 'Day', 'Afternooon')
```

LAMPIRAN E GAMBAR KOMPONEN AKUAPONIK