LAMPIRAN A
Program Sistem

```
#include <LiquidCrystal_I2C.h>
#include <Wire.h>
LiquidCrystal_I2C lcd(0x27, 16, 2);

#define VIN A0 // define the Arduino pin A0 as voltage input (V in)

// Sensor Arus
const float sensitivity = 0.066; // for ACS712ELCTR-05B-T
const float VCC = 5.0;  // supply voltage is from 4.5 to 5.5V. Normally 5V.
const float QOV = 0.5 * VCC; // set quiescent Output voltage of 0.5V
const int ACS712_pin = A0;
float voltage; // internal variable for voltage
float cutOffLimit = 1.01; // Adjust this if needed

// Ultrasonic sensor pins
const int trigPin1 = 17;  // Front ultrasonic sensor trig pin
const int echoPin1 = 18;  // Front ultrasonic sensor echo pin
const int trigPin2 = 26;  // Rear ultrasonic sensor trig pin
const int echoPin2 = 27;  // Rear ultrasonic sensor echo pin

long duration1, duration2;
int distance1, distance2;
int safetyDistance = 10; // Distance in cm

// Receiver signal pins
double ch1_pin = 2;  // PWM pin
double ch2_pin = 3;  // PWM pin
double ch3_pin = 4;  // PWM pin

// Right motor driver pins
int R_EN_right = 7;
int L_EN_right = 8;
int R_PWM_right = 6; // PWM pin
int L_PWM_right = 5; // PWM pin

// Left motor driver pins
```

```
int R_EN_left = 11;
int L_EN_left = 12;
int R_PWM_left = 10; // PWM pin
int L_PWM_left = 9; // PWM pin

// Additional motor driver pins for the cleaner motor
int R_EN_cleaner = 13;
int L_EN_cleaner = 14;
int R_PWM_cleaner = 15; // PWM pin
int L_PWM_cleaner = 16; // PWM pin

// Rx threshold values - Update based on your TX values
// FWD
int Ch1Ch2_start_Fwd = 1530;
int Ch1Ch2_End_Fwd = 1980;
// BACK
int Ch1Ch2_start_Bac = 1460;
int Ch1Ch2_End_Bac = 960;

// Threshold values for cleaner motor control
int Ch3_start_Fwd = 1530;
int Ch3_End_Fwd = 1980;
int Ch3_start_Bac = 1460;
int Ch3_End_Bac = 960;

void setup()
{
   lcd.init();
   lcd.backlight();
   Serial.begin(9600);
   pinMode(ACS712_pin, INPUT);
   pinMode(trigPin1, OUTPUT);
   pinMode(echoPin1, INPUT);
   pinMode(trigPin2, OUTPUT);
   pinMode(echoPin2, INPUT);
   pinMode(ch1_pin, INPUT);
   pinMode(ch2_pin, INPUT);
   pinMode(ch3_pin, INPUT);
   pinMode(R_PWM_right, OUTPUT);
   pinMode(L_PWM_right, OUTPUT);
```

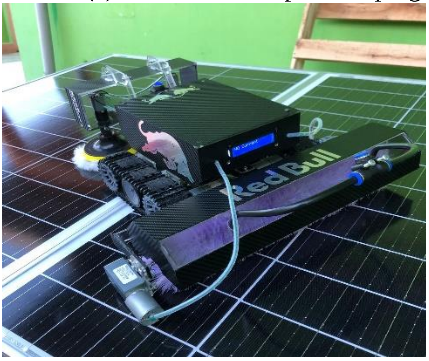
```
  pinMode(R_EN_right, OUTPUT);
  pinMode(L_EN_right, OUTPUT);
  pinMode(R_PWM_left, OUTPUT);
  pinMode(L_PWM_left, OUTPUT);
  pinMode(R_EN_left, OUTPUT);
  pinMode(L_EN_left, OUTPUT);
  pinMode(R_PWM_cleaner, OUTPUT);
  pinMode(L_PWM_cleaner, OUTPUT);
  pinMode(R_EN_cleaner, OUTPUT);
  pinMode(L_EN_cleaner, OUTPUT);
}

void loop()
{
  sensor_arus();
  checkUltrasonicSensors();

  // Receiver signal pins data
  double ch1 = pulseIn(ch1_pin, HIGH);
  double ch2 = pulseIn(ch2_pin, HIGH);
  double ch3 = pulseIn(ch3_pin, HIGH);

  Serial.print("ch1: ");
  Serial.println(ch1);
  Serial.print("\t");
  Serial.print("ch2: ");
  Serial.print(ch2);
  Serial.println("\t");
  Serial.print("ch3: ");
  Serial.print(ch3);
  Serial.println("\t");

  // Speed mapping for right and left motor drivers
  int spdFwd_RightLeft = map(ch1, Ch1Ch2_start_Fwd, Ch1Ch2_End_Fwd,
0, 255);
  int spdBac_RightLeft = map(ch1, Ch1Ch2_start_Bac, Ch1Ch2_End_Bac, 0,
255);

  // Speed mapping for cleaner motor
  int spdCleaner_FwdBac = map(ch3, Ch3_start_Fwd, Ch3_End_Fwd, 0,
255);
  int spdCleaner_BacFwd = map(ch3, Ch3_start_Bac, Ch3_End_Bac, 0, 255);

  digitalWrite(R_EN_right, HIGH);
  digitalWrite(L_EN_right, HIGH);
  digitalWrite(R_EN_left, HIGH);
  digitalWrite(L_EN_left, HIGH);
  digitalWrite(R_EN_cleaner, HIGH);
  digitalWrite(L_EN_cleaner, HIGH);

  // Determine if backward movement should be allowed
  bool backwardAllowed = distance1 <= safetyDistance;
  bool forwardAllowed = distance2 <= safetyDistance;

  if ((ch1 == 0) && (ch2 == 0) && (ch3 == 0))
  {
    stopAllMotors();
  }
  else if (forwardAllowed && backwardAllowed)
  {
    if ((ch1 > Ch1Ch2_start_Fwd) && (ch2 > Ch1Ch2_start_Fwd))
    {
      analogWrite(R_PWM_right, spdFwd_RightLeft);
      analogWrite(L_PWM_right, 0);
      analogWrite(R_PWM_left, spdFwd_RightLeft);
      analogWrite(L_PWM_left, 0);
    }
    else if ((ch1 > Ch1Ch2_start_Fwd) && (ch2 < Ch1Ch2_start_Bac))
    {
      analogWrite(R_PWM_right, 0);
      analogWrite(L_PWM_right, spdFwd_RightLeft);
      analogWrite(R_PWM_left, spdFwd_RightLeft);
      analogWrite(L_PWM_left, 0);
    }
    else if ((ch1 < Ch1Ch2_start_Bac) && (ch2 > Ch1Ch2_start_Fwd))
    {
      analogWrite(R_PWM_right, spdBac_RightLeft);
      analogWrite(L_PWM_right, 0);
      analogWrite(R_PWM_left, 0);
      analogWrite(L_PWM_left, spdBac_RightLeft);
    }
    else if ((ch1 < Ch1Ch2_start_Bac) && (ch2 < Ch1Ch2_start_Bac))
    {
        analogWrite(R_PWM_right, 0);
        analogWrite(L_PWM_right, spdBac_RightLeft);
        analogWrite(R_PWM_left, 0);
        analogWrite(L_PWM_left, spdBac_RightLeft);
    }
    else
    {
        stopAllMotors();
    }
}
else if (forwardAllowed)
{
    if ((ch1 > Ch1Ch2_start_Fwd) && (ch2 > Ch1Ch2_start_Fwd))
    {
        analogWrite(R_PWM_right, spdFwd_RightLeft);
        analogWrite(L_PWM_right, 0);
        analogWrite(R_PWM_left, spdFwd_RightLeft);
        analogWrite(L_PWM_left, 0);
    }
    else if ((ch1 > Ch1Ch2_start_Fwd) && (ch2 < Ch1Ch2_start_Bac))
    {
        analogWrite(R_PWM_right, 0);
        analogWrite(L_PWM_right, spdFwd_RightLeft);
        analogWrite(R_PWM_left, spdFwd_RightLeft);
        analogWrite(L_PWM_left, 0);
    }
    else
    {
        stopAllMotors();
    }
}
else if (backwardAllowed)
{
    if ((ch1 < Ch1Ch2_start_Bac) && (ch2 > Ch1Ch2_start_Fwd))
    {
        analogWrite(R_PWM_right, spdBac_RightLeft);
        analogWrite(L_PWM_right, 0);
```

```
            analogWrite(R_PWM_left, 0);
            analogWrite(L_PWM_left, spdBac_RightLeft);
        }
        else if ((ch1 < Ch1Ch2_start_Bac) && (ch2 < Ch1Ch2_start_Bac))
        {
            analogWrite(R_PWM_right, 0);
            analogWrite(L_PWM_right, spdBac_RightLeft);
            analogWrite(R_PWM_left, 0);
            analogWrite(L_PWM_left, spdBac_RightLeft);
        }
        else
        {
            stopAllMotors();
        }
    }
    else
    {
        stopAllMotors();
    }

    // Cleaner motor control logic
    if ((ch3 > Ch3_start_Fwd) && (ch3 > Ch3_start_Bac))
    {
        analogWrite(R_PWM_cleaner, spdCleaner_FwdBac);
        analogWrite(L_PWM_cleaner, 0);
    }
    else if ((ch3 < Ch3_start_Bac) && (ch3 < Ch3_start_Fwd))
    {
        analogWrite(R_PWM_cleaner, 0);
        analogWrite(L_PWM_cleaner, spdCleaner_BacFwd);
    }
    else
    {
        analogWrite(R_PWM_cleaner, 0);
        analogWrite(L_PWM_cleaner, 0);
    }
}

void sensor_arus()
{
    float voltage_raw = (5.0 / 1023.0) * analogRead(VIN); // Read the voltage
from sensor
    voltage = voltage_raw - QOV + 0.012; // 0.000 is a value to make voltage
zero when there is no current
    float current = voltage / sensitivity;

    // Display current reading continuously on LCD
    lcd.clear();
    lcd.setCursor(0, 0); // set to line 1, char 0
    lcd.print("SENSOR ACS712");
    lcd.setCursor(0, 1); // set to line 1, char 0
    lcd.print("Current: ");
    lcd.setCursor(9, 1); // go to start of 2nd line
    lcd.print(current);
    lcd.setCursor(14, 1); // go to start of 2nd line
    lcd.print("A");

    Serial.print("V: ");
    Serial.print(voltage, 3); // print voltage with 3 decimal places
    Serial.print("V, I: ");
    Serial.print(current, 2); // print the current with 2 decimal places
    Serial.println("A");

    delay(1000);
}

void checkUltrasonicSensors()
{
    // Check front ultrasonic sensor
    digitalWrite(trigPin1, LOW);
    delayMicroseconds(2);
    digitalWrite(trigPin1, HIGH);
    delayMicroseconds(10);
    digitalWrite(trigPin1, LOW);

    duration1 = pulseIn(echoPin1, HIGH);
    distance1 = duration1 * 0.034 / 2;

    // Check rear ultrasonic sensor
    digitalWrite(trigPin2, LOW);
    delayMicroseconds(2);
    digitalWrite(trigPin2, HIGH);
    delayMicroseconds(10);
    digitalWrite(trigPin2, LOW);

    duration2 = pulseIn(echoPin2, HIGH);
    distance2 = duration2 * 0.034 / 2;
}

void stopAllMotors()
{
    analogWrite(R_PWM_right, 0);
    analogWrite(L_PWM_right, 0);
    analogWrite(R_PWM_left, 0);
    analogWrite(L_PWM_left, 0);
    analogWrite(R_PWM_cleaner, 0);
    analogWrite(L_PWM_cleaner, 0);
}
```

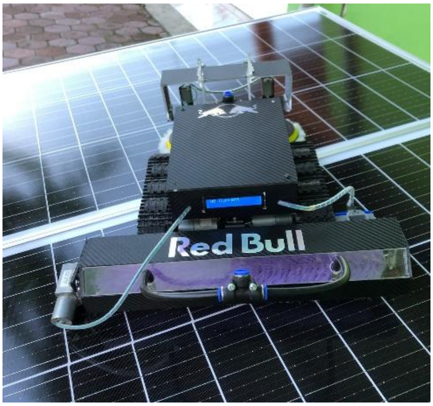
LAMPIRAN B

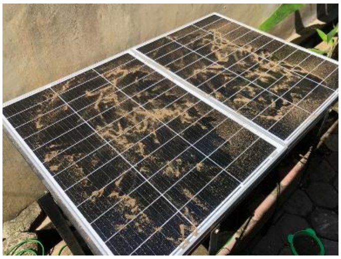
Hasil Alat dan pengujian

Gambar (a) Hasil Alat Tampak Samping

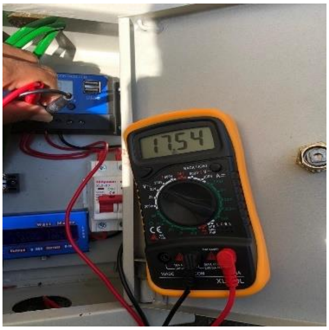


Gambar (b) Hasil Alat Tampak Depan

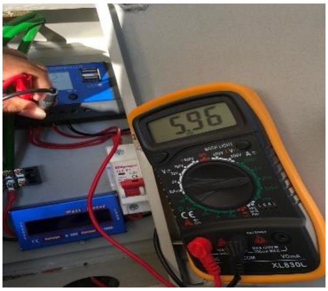




Gambar (c) Kondisi Panel Surya Sebelum Dibersihkan

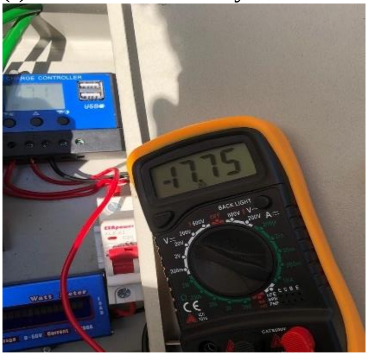


Gambar (d) Tegangan Terukur dalam Kondisi Kotor

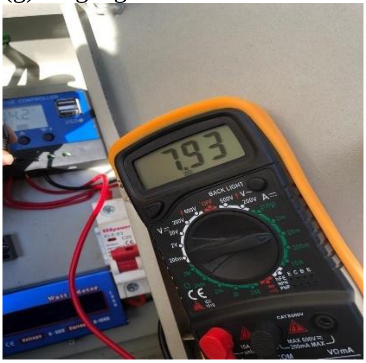


Gambar (e) Arus Terukur dalam Kondisi Kotor

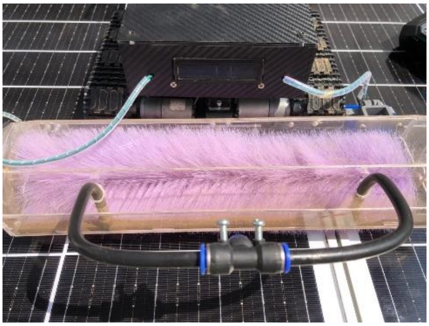
Gambar (f) Kondisi Panel Surya Setelah Dibersihkan



Gambar (g) Tegangan Terukur Setelah Dibersihkan



Gambar (h) Arus Terukur Setelah Dibersihkan


Gambar (i) Kondisi Robot setelah melakukan pembersihan

BIODATA PENULIS

Nama	:	Irfan Exzan Efendi
Tempat/Tanggal Lahir	:	Cilacap, 13 November 2002
Alamat	:	Jalan Pucang Rt 02 Rw 09, Gumilir, Cilacap, Cilacap Utara
Email	:	irfanexzanefendi@gmail.com
Telepon/HP	:	082135640259
Hobi	:	Membaca
Motto	:	Jadilah Dirimu Sendiri.

Riwayat Pendidikan :

• SD Negeri 06 Gumilir	Tahun 2009 – 2015
• SMP Negeri 7 Cilacap	Tahun 2015 – 2018
• SMK Boedi Oetomo	Tahun 2018 – 2021
• Politeknik Negeri Cilacap Prodi D3 Teknik Listrik	Tahun 2021 – 2024

Penulis telah mengikuti Seminar Tugas Akhir pada tanggal 12 Agustus 2024 sebagai salah satu persyaratan untuk memperoleh gelar Ahli Madya (A.Md).