DAFTAR PUSTAKA

Agung, I. G. A. P. R., Huda, S., dan Wijaya I. W. A. (2014). Speed control for DC motor with pulse width modulation (PWM) method using infrared remote control based on ATmega16 microcontroller. International Conference on Smart Green Technology in Electrical and Information Systems (ICSGTEIS), Bali, pp. 108-112.

Arduino.cc,. (n.d.). Arduino Nano. https://docs.arduino.cc/hardware/nano

Bose, B. K. dan Steigerwald, R. L. (1978). A DC Motor Control System for Electric Vehicle Drive. IEEE Transactions on Industry Applications, vol. IA-14, no.6, pp. 565–572.

Chen, B. Lu, G. Hao, R. Hu, L. dan Tian, X. (2009). Intelligent Speed and Mileage Measurement System for Vehicles Based on Hall Sensor. First International Workshop on Education Technology and Computer Science, pp. 272-274.

Electronics Tutorials. (n.d.). Pulse Width Modulation. https://www.electronics-tutorials.ws/blog/pulse-width-modulation.html

EVBox. (2021). History of the electric car [2023 update]. https://blog.evbox.com/electric-cars-history#going-electric

Gobor, A. (2019, Oktober 3). What is PWM and How Does It Work. ekwb. https://www.ekwb.com/blog/what-is-pwm-and-how-does-it-work/

Gupta, V. dan Deb, A. (2012). Speed control of brushed DC motor for low cost electric cars. IEEE International Electric Vehicle Conference.

Hertz, J. (2022, Agustus 29). H-bridge DC Motor Control Using Complementary PWM, Shoot-through, and Dead-time. All About Circuit. https://www.allaboutcircuits.com/technical-articles/h-bridge-dc-motor-control-complementary-pulse-width-modulation-pwm-shoot-through-dead-time-pwm/

Hirzel, T. (2023, April 26). Basics of PWM (Pulse Width Modulation). Arduino.cc. https://docs.arduino.cc/learn/microcontrollers/analog-output

Holtz, J. (1992). Pulsewidth modulation: a survey. IEEE Trans. Industrial Electronics, vol. IE-39, no.5 pp.410-420.

Hughes, A. (2006). Electric Motor. Dalam Electric Motors and Drives Fundamentals, Types and Applications (3rd ed). Oxford: Elsevier. pp. 32-37.

Junaidi, dan Prabowo, Y. D. (2018). Project Sistem Kendali Elektronik Berbasis Arduino. Lampung: AURA.

Kazimierczuk, M. K. (2008). Full-bridge PWM DC-DC Converter. Dalam Pulse-width Modulated DC-DC Power Converters. United Kingdom: Wiley. pp. 325-327.

Koca, Y. B., Aslan, Y., dan Gokce, B. (2014). Speed Control Based PID Configuration of a DC Motor for An Unmanned Agricultural Vehicle. International Conference on Electrical and Electronics Engineering (ICEEE), pp. 117-120.

Lab-Volt. (2014). Permanenet Magnet DC Motor (1st ed). Canada.

Nugraha, A. dan Isworo, H. (2018). Kinematika (Buku Ajar Kinematika HMKK533). Banjarmasin: Universitas Lambung Mangkurat. https://mesin.ulm.ac.id/assets/dist/bahan/Kinematika_full.pdf

Putri, R. A. (2023). Pengertian Rumus Serta Contoh Soal Kecepatan Linear dan Kecepatan Angular pada Gerak Melingkar. Materi Edukasi. https://www.materiedukasi.id/2023/05/pengertian-rumus-serta-contoh-soal-kecepatan-linear-dan-kecepatan-angular-pada-gerak-melingkar-2.html

Robu.in. (2020). Permanent Magnet DC Motor MY1020. https://robu.in/wp-content/uploads/2020/01/665828-MY1020.pdf

Sahdev, S. K. (2018). Electrical Machines. Cambridge: Cambridge University Press.

Theraja, B. L. dan Theraja, A. K. (2005). DC Motor. Dalam A Textbook of Electrical Technology Volume II (1st Multicolour ed). New Delhi: S. Chand.

U.S. Department of Energy. (1992). Batteries, DC Circuits, DC Generators, & DC Motors. Dalam DOE Fundamentals Handbook Electrical Science Volume 2 of 4. Washington, D.C.: DOE.

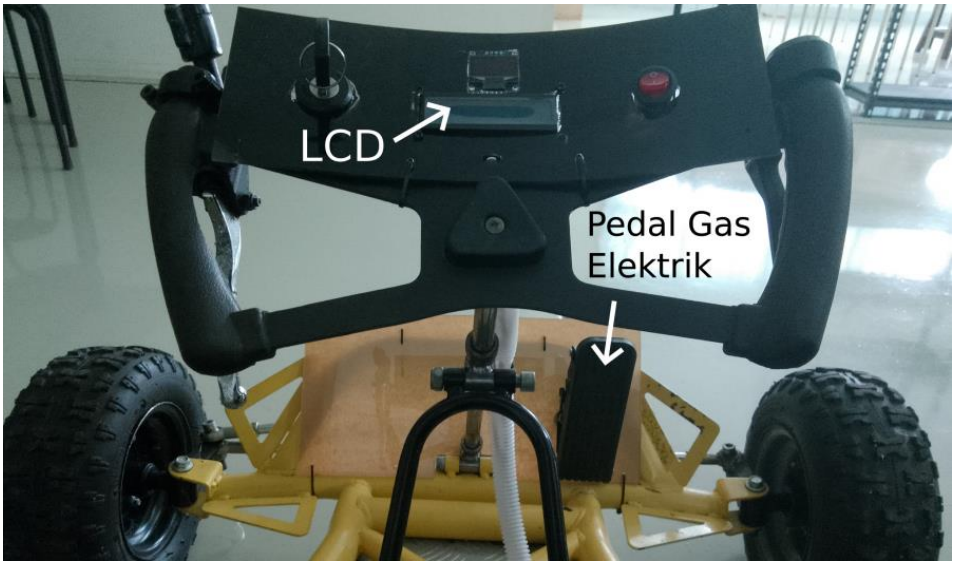
Vivekanand. (2019, Mei 6). Simple H-Bridge Motor Driver Circuit Using MOSFET. Circuit Digest. https://circuitdigest.com/electronic-circuits/simple-h-bridge-motor-driver-circuit-using-mosfet#:~:text=This%20circuit%20is%20called%20H,the%20speed%20can%20be%20controlled

Yurkevich, V. D. dan Stepanov, N. A. (2014). PWM speed control of DC motor based on singular perturbation technique. 6th International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT), pp. 434-440.

LAMPIRAN

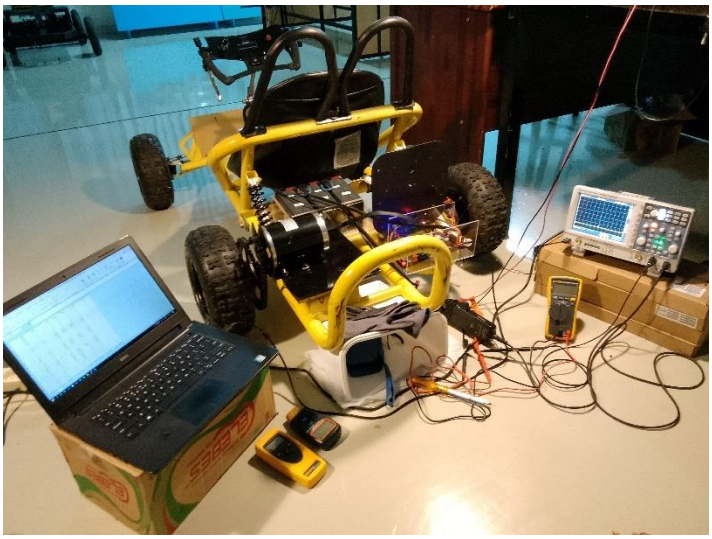
Lampiran 1. Tampak depan prototipe kendaraan listrik

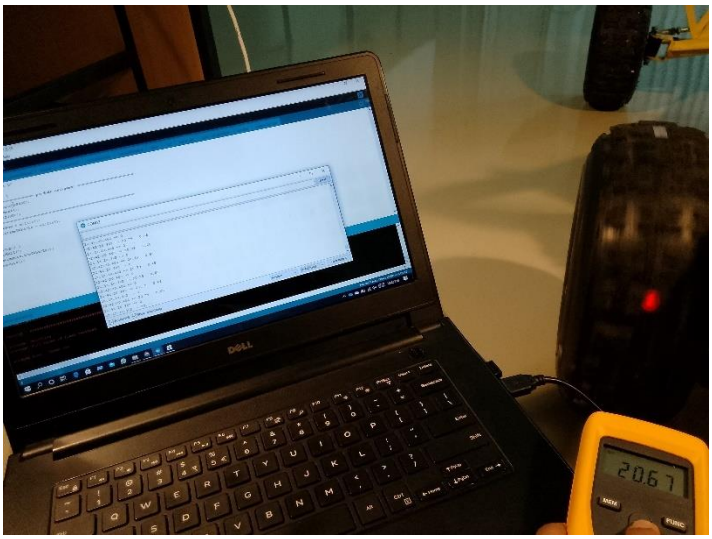
Lampiran 2. Perangkat I/O dan baterai

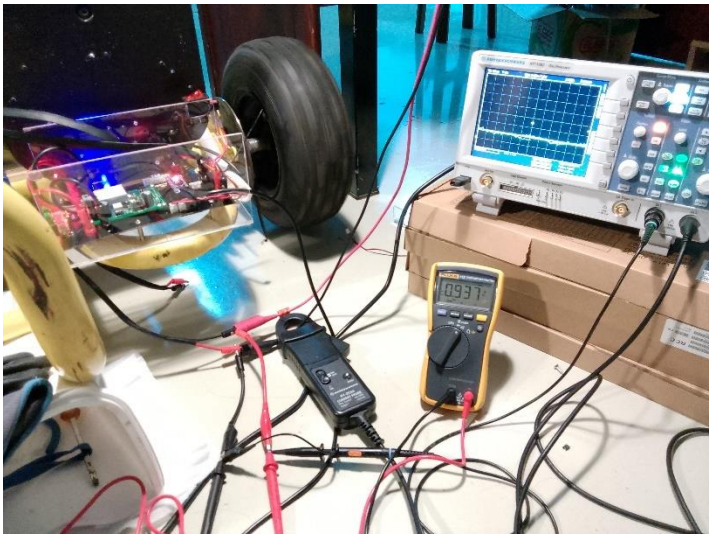


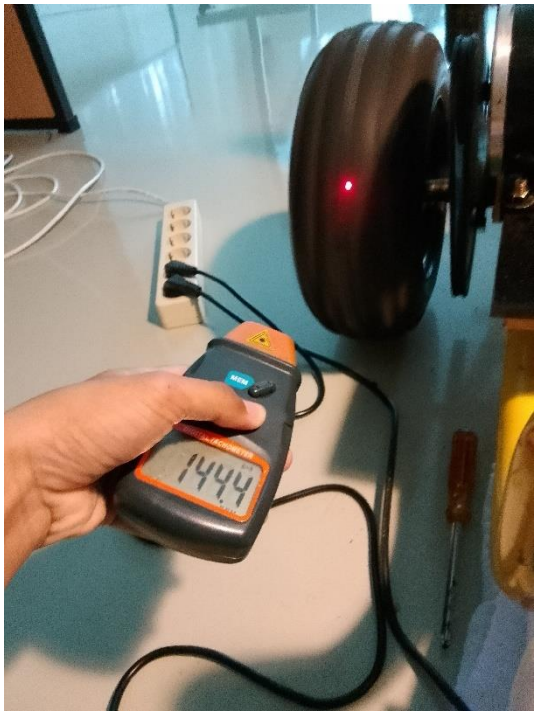
Lampiran 3 Gambar (*Setup*) pengambilan data

Pengambilan data pada kondisi tanpa beban





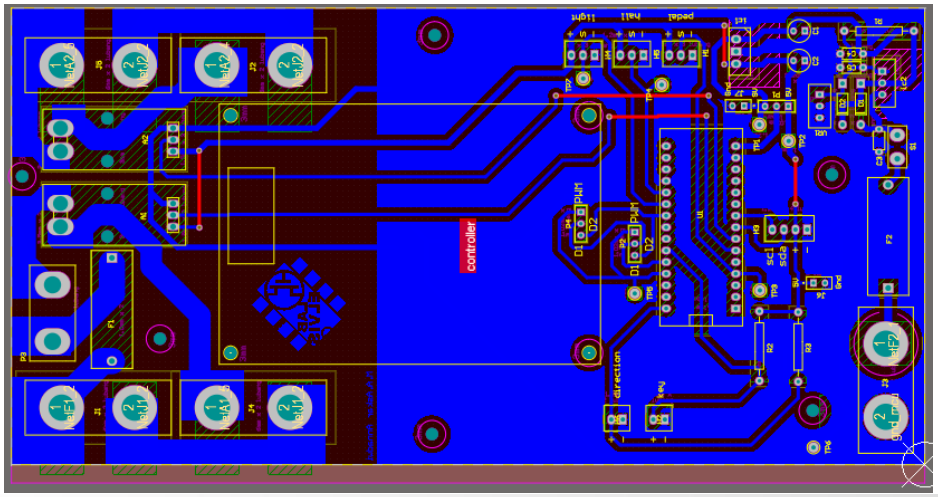


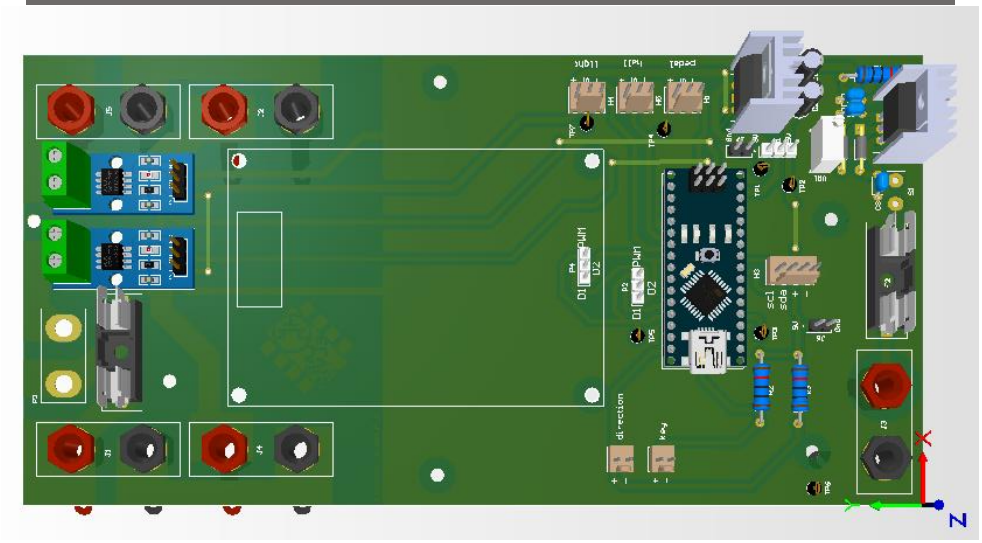


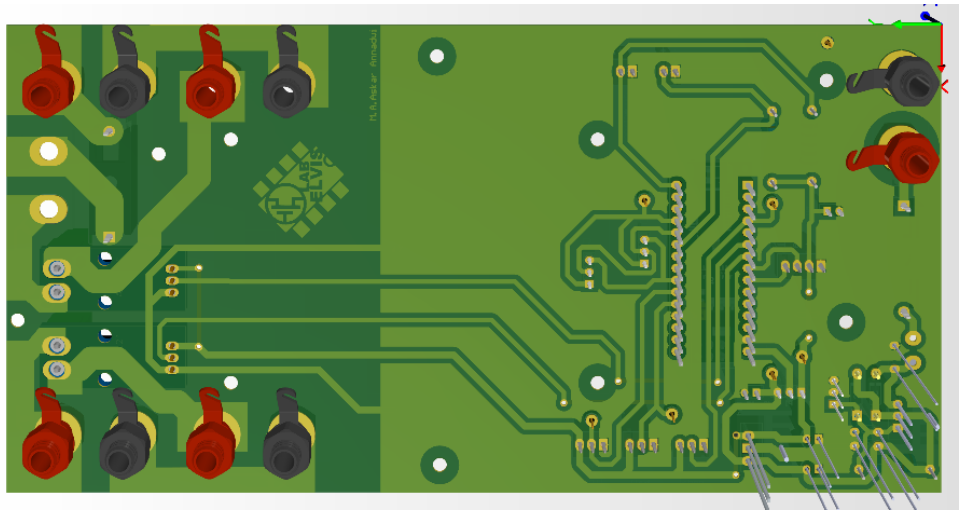
Pengambilan data pada kondisi dengan beban

Pengambilan data uji simpangan

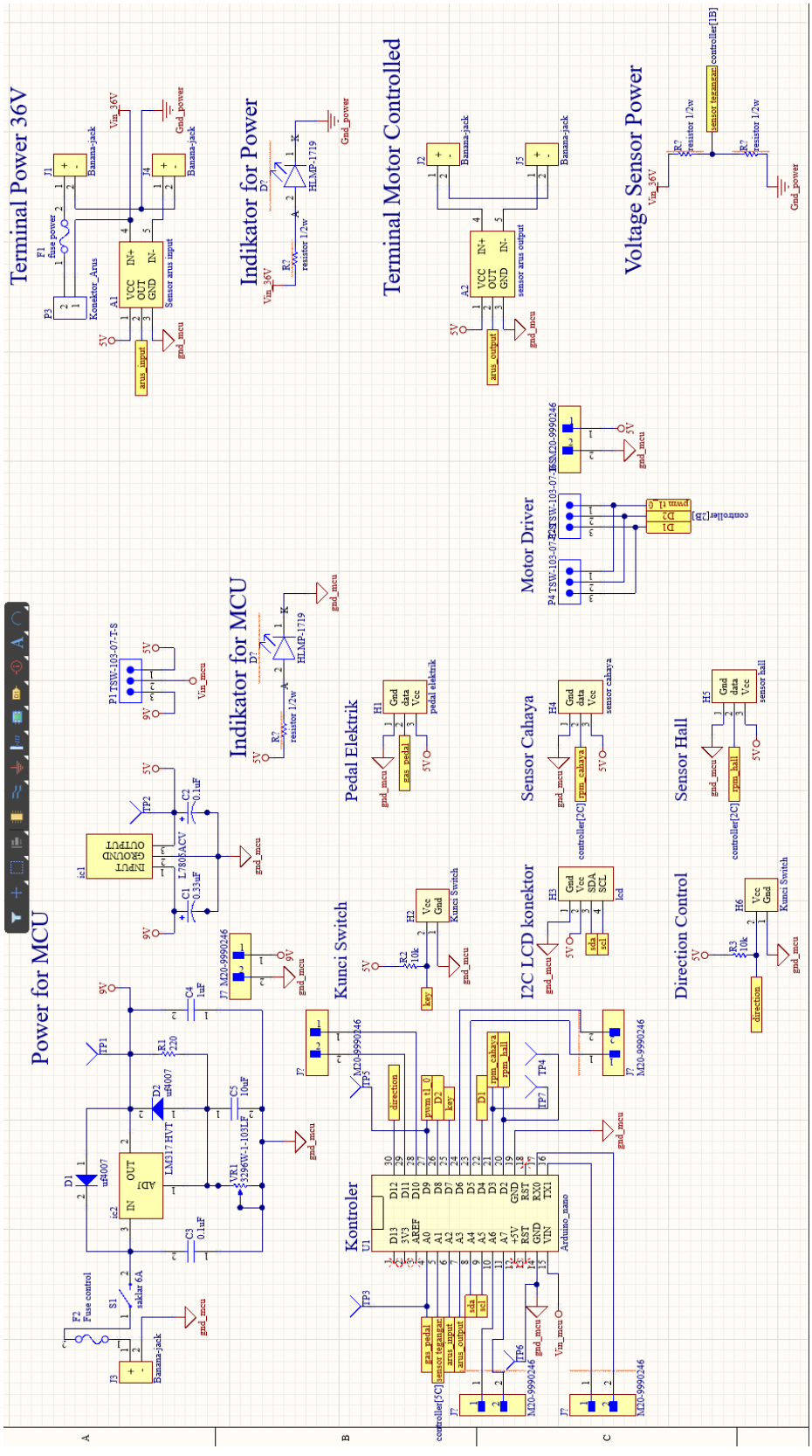
Lampiran 4 Layout desain PCB dan 3D







Lampiran 5 Skematik desain PCB



Lampiran 6 Program arduino

main.ino

```
#include "pInit.h"

int mOut, hall, rpm, pidSpeed;
float cIn, cOut, vIn;

void setup() {
  Serial.begin(115200);
  lcdStarting();
  pinAssigment();
  setFreq(20000);
  setLimitPID();
  currentTimeVehicle = millis();
}

void loop() {
  mOut = readPedal(0);
  goForwardManual(mOut);
  rpmComputation();
  lcd.setCursor(0, 0);
  lcd.print(mOut);
  lcd.setCursor(0, 1);
  lcd.print(rpmVehicle);
}
```

pInit.h

```
/*
  pin Assigment on board (PCB):
    - A0 : Gas pedal          - D2 : rpm hall sensor
    - A2 : arus input         - D3 : rpm hall sensor #2
    - A3 : arus output        - D4 : Direction 1 - driver
    - A4 : SDA                - D8 : Direction 2 - driver
    - A5 : SCL                - D9 : PWM signal
    - A1 : voltage input
    - D7  : key   (pull_up circuit)
    - D12 : direction throttle control (pull_up circuit)
*/
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
LiquidCrystal_I2C lcd(0x3f, 16, 2);

// ================== pin Declaration ======================
#define pedal A0
#define VolIn A1
#define arusIn A2
#define arusOut A3
#define key 7
#define throttle 12
#define rpmSensorMotor 3
#define rpmSensorVehicle 2
#define d1 4
#define d2 8
#define pwm 9

void pinAssigment() {
  pinMode(pedal, INPUT);
```

```
  pinMode(arusIn, INPUT);
  pinMode(arusOut, INPUT);
  pinMode(VolIn, INPUT);
  pinMode(key, INPUT);
  pinMode(throttle, INPUT);
  pinMode(rpmSensorMotor, INPUT);
  pinMode(rpmSensorVehicle, INPUT);
  pinMode(d1, OUTPUT);
  pinMode(d2, OUTPUT);
  pinMode(pwm, OUTPUT);
}
// ================== variabel Declaration ================
int pedalValue;

//inisiasi encoder
int pulsesVehicle;
const int rotationFlag = 1;
unsigned long timePasses, currentTimeVehicle, oldTimeVehicle = 0;
bool flagPulses = true;

//inisiasi speed computation
float rpmVehicle;
double myRpm;
float velocity;
const float wheelRadius = 0.15;  // in meter -; diameter 0.3m

//inisiasi sensor arus
float CurrentIn, CurrentOut;
const float sensitivitySensor = 0.1;  // 100mV/A -> 20A current
sensor sensitivity
const float calibrationSensorInMin = 0.015,
calibrationSensorInPlus = 0.0;
const float calibrationSensorOutMin = 0.015,
calibrationSensorOutPlus = 0.015;

//inisiasi sensor tegangan
float VoltageCalbIn;
const float ref_voltage = 5;
double R1 = 200000, R2 = 20000; // R1-200k R2-20k
float calibrationIndex = 0.1145;
int adc;
float vOO;

//inisiasi settingFrequency
const double crystal = 16000000; //16MHz
int topValue;
const float topVoltageLimit = 36.0;
float topPwmValue;
```

w_tab.ino

```
int readHall() {
  if ((digitalRead(rpmSensorVehicle) == LOW) && (flagPulses ==
false)) {
    flagPulses = true;
    pulsesVehicle += 1;
  }
  else if ((digitalRead(rpmSensorVehicle) == HIGH) && (flagPulses
== true)){
    flagPulses = false;
```

```
  }

}

void rpmComputation() {
  if (pulsesVehicle >= rotationFlag) {
    currentTimeVehicle = millis();
    timePasses = currentTimeVehicle - oldTimeVehicle;
    rpmVehicle = 60000.0 / timePasses * pulsesVehicle; // 60000 =
60*1000 -; millis()
    // rps = rpmVehicle / 60.0;
    // velocity = wheelRadius * rpmVehicle * 0.10472;  // 0.10472
equal to 2[phi] divided by 60; wheelRadius in meters -; m/s
    oldTimeVehicle = currentTimeVehicle;
    pulsesVehicle = 0;
  }
}
```

x_tab.ino

```
int readPedal(int mode) {
  int samplePedal = 0;
  for (int a = 0; a < 10; a++) {
    unsigned int pedalRaw = analogRead(pedal);
    samplePedal = samplePedal + pedalRaw;
  }
  pedalValue = samplePedal / 10;

    if (pedalValue < 285) return 0;
    else if (pedalValue >= 285 && pedalValue < 380) return 3;
    else if (pedalValue >= 681 && pedalValue < 760) return 13;
    else if (pedalValue >= 761 && pedalValue < 880) return 15;
    int mapPedal = map(pedalValue, 380, 680, 4, 12);
    return mapPedal;

  // ----------- pengujian Duty cycle Vs Kemiringan ------------
  //if (pedalValue < 380) return 0;
  //else if (pedalValue >= 380 && pedalValue < 450) return 80; //
supaya lebih halus; digunakan per duty cycle = 40%
  //else if (pedalValue >= 450 && pedalValue < 880) return 200; //
masukkan nilai pwm sesuai duty cycle
  // max pwm 399
  // duty cyle      pwm       tegangan
  // 5%             20        1.8         tidak jalan
  // 10%            40        3.6         tidak jalan
  // 20%            80        7.2         jalan
  // 30%            120       10.8        jalan
  // 40%            160       14.4        jalan
  // 45%            180       16.2
  // 50%            200       18          ~ stall

}

void computeCurrentIn() {
  int samplesCurrentIn = 0;
  for (int x = 0; x < 10; x++) {
    unsigned int AcsValueIn = analogRead(arusIn);
    samplesCurrentIn = samplesCurrentIn + AcsValueIn;
  }
  float AvgAcsIn = samplesCurrentIn / 10.0;
```

```
  float voltageAcsIn = (AvgAcsIn * (5 / 1024.0));
  CurrentIn = (voltageAcsIn - 2.5) / sensitivitySensor + 0.03;
}
void computeCurrentOut() {
  int sampleCurrentOut = 0;
  for (int y = 0; y < 10; y++) {
    unsigned int AcsValueOut = analogRead(arusOut);
    sampleCurrentOut = sampleCurrentOut + AcsValueOut;
  }
  float AvgAcsOut = sampleCurrentOut / 10.0;

  float voltageAcsOut = AvgAcsOut * (5 / 1024.0);
  CurrentOut = (voltageAcsOut - 2.5) / sensitivitySensor + 0.09;
}

float computeVoltageIn() {
  double samplesVoltageIn = 0.0;
  for (int z = 0; z < 10; z++) {
    unsigned int VolValueIn = analogRead(VolIn);
    samplesVoltageIn = samplesVoltageIn + VolValueIn;
  }
  float AvgVolIn = samplesVoltageIn / 10;
  float voltageSensorIn = (AvgVolIn * ref_voltage) / 1024.0;
  float VoltageIn = (voltageSensorIn / (R2 / (R1 + R2)));
  VoltageCalbIn = VoltageIn + (calibrationIndex * VoltageIn);

  return VoltageCalbIn;
}
```

y_tab.ino

```
void setFreq(double freq) {
  topValue = (crystal / (2 * freq)) - 1;

  TCCR1A = 0;           // undo the configuration done by...
  TCCR1B = 0;           // ...the Arduino core library
  TCNT1  = 0;           // reset timer
  TCCR1A = _BV(COM1A1)  // non-inverted PWM on ch. A
         | _BV(COM1B1)  // same on ch; B
         | _BV(WGM11);  // mode 10: ph. correct PWM, TOP = ICR1
  TCCR1B = _BV(WGM13)   // ditto
         | _BV(CS10);   // prescaler = 1
  ICR1   = topValue;  //topValue - 20kHz - 399
}

void myAnalogWrite(float myVoltageIn, float voltageOutput) {
  topPwmValue = topValue * topVoltageLimit / myVoltageIn;
  if (topPwmValue > topValue) topPwmValue = topValue;
  if (voltageOutput > topVoltageLimit) voltageOutput =
topVoltageLimit;
  float duty_v = voltageOutput * 100 / topVoltageLimit;
  float myADC = duty_v * topPwmValue / 100;
  adc = int(myADC);
  OCR1A = adc;

  vOO = adc * topVoltageLimit / topPwmValue;
}
```

z_tab.ino

```
void goForward(float myVin, int myVout) {
  digitalWrite(d1, LOW);
  digitalWrite(d2, HIGH);
  myAnalogWrite(myVin, myVout);
}

void goReverse(float myVin, int myVout) {
  digitalWrite(d1, HIGH);
  digitalWrite(d2, LOW);
  myAnalogWrite(myVin, myVout);
}

void lcdStarting() {
  lcd.begin();
  lcd.backlight();
}

void lcdShow() {
  lcd.setCursor(0, 0);
  lcd.print(vOO);
  lcd.setCursor(0, 1);
  lcd.print(rpmVehicle);
}

void showData(float data1, int data2){
  Serial.print(pedalValue); Serial.print("\t");
  Serial.print(data1); Serial.print("\t"); // tegangan baterai
  Serial.print(vOO); Serial.print("\t"); // tegangan output driver
  Serial.print(CurrentIn); Serial.print("\t"); // arus baterai
  Serial.print(CurrentOut); Serial.print("\t"); // arus output
driver
  Serial.print(rpmVehicle);  Serial.print("\t");
  float data_pin = data1 * CurrentIn;
  float data_pout = vOO * CurrentOut;
  Serial.print(data_pin); Serial.print("\t");
  Serial.print(data_pout);
  Serial.println();
}
```

Lampiran 7 Tabel data penelitian

Grafik akurasi Sensor		
voltage (V)	sensor (RPM)	takometer (RPM)
1	20.7	20.6
2	51.1	50.9
3	79.58	80.8
4	112.4	111.9
5	142.2	142
6	172.4	172.1
7	201	204
8	237.2	237.6
9	269	268.8
10	300.8	299.6
11	333.33	333.3
12	365.8	365.5
13	400.67	399.3
14	431	423
15	466.93	466.9
16	498.96	498.1
17	532.15	531.5
18	563	561
19	594	593
20	629	628.6
21	664	663
22	697	697
23	733	730
24	766	764

Pengujian tanpa beban											
	direct supply							motor driver			
tegangan input	tegangan terukur (V)	arus (A)	daya	kecepatan (RPM) (sensor)	kecepatan tacho	tegangan (V) (osiloskop)	arus (A) (osiloskop)	tegangan (V) (algoritma)	arus (A) (sensor)	kecepatan (RPM) (sensor)	daya
4	3.94	1.53	6.0282	112.4	111.9	3.84	1.31	3.99	1.59	107.87	6.3441
6	5.97	1.85	11.0445	172.4	172.1	6.05	1.15	5.91	1.7	169.85	10.047
9	8.8	2.32	20.416	269	268.8	8.93	2.2	8.93	2.26	264.9	20.1818
12	11.79	2.7	31.833	365.8	365.5	11.87	2.5	11.99	2.57	361.45	30.8143
15	14.67	3.07	45.0369	466.93	466.9	14.47	1.96	14.92	2.98	453.69	44.4616
18	17.62	3.44	60.6128	563	561	17.35	2.8	17.93	3.24	546.7	58.0932

Efisiensi motor driver				
waktu	Pin - 12v daya masuk	Pout - 12V daya keluar	Pin - 18v daya masuk	Pout - 18v daya keluar
1	33.69	33.54	66.65	56.59
2	38.02	35.46	72.13	64.91
3	42.8	36.48	66.81	60.78
4	34.08	27.9	64.75	52.66
5	36.68	31.67	59.15	56.62
6	35.34	28.79	66.31	59.32
7	26.56	25.57	69.24	65.68
8	36.04	32.99	79.14	65.67
9	38.16	37.71	64.87	58.17
10	32.47	30.82	68.29	64.41
11	34.06	32.68	66.07	59.44

Metode kontrol output				
	9v		36v	
tegangan baterai	tanpa metode kontrol	dengan metode kontrol	tanpa metode kontrol	dengan metode kontrol
37.8	9.473684	9	37.8	36
37.7	9.448622	8.97619	37.7	35.99925
37.6	9.423559	8.952381	37.6	35.99799
37.5	9.398496	8.928571	37.5	35.99624
37.4	9.373434	8.998496	37.4	35.99398
37.3	9.348371	8.974436	37.3	35.99123
37.2	9.323308	8.950376	37.2	35.98797
37.1	9.298246	8.926316	37.1	35.98421
37	9.273183	8.994987	37	35.97995
36.9	9.24812	8.970677	36.9	35.97519
36.8	9.223058	8.946366	36.8	35.96992
36.7	9.197995	8.922055	36.7	35.96416
36.6	9.172932	8.989474	36.6	35.95789
36.5	9.14787	8.964912	36.5	35.95113
36.4	9.122807	8.940351	36.4	35.94386
36.3	9.097744	8.915789	36.3	35.93609
36.2	9.072682	8.981955	36.2	35.92782
36.1	9.047619	8.957143	36.1	35.91905
36	9.022556	8.932331	36	36

Pengujian dengan beban			
tegangan (V)	kecepatan (RPM)	Arus (A)	Daya (W)
6	12.61	3.69	114.27
7.125	26.81	4.9	174.17
7.96875	51.47	6.72	237.41
9.09375	75.73	7.86	276.26
9.9375	82.99	8.78	308.37
11.0625	93.71	9.69	336.65
12.1875	101.05	10.89	380.59
13.03125	110.4	11.83	411.12
13.59375	122.89	13.43	457.3
14.15625	140.43	15.11	551.39
15	150.56	17.23	587.19

data *duty cycle* Vs Simpangan				
duty cycle	condong	simpangan (cm)	jarak (m)	sudut simpangan
20%	ke kanan	6	6.5	0.528
30%	ke kanan	18	6.5	1.586
40%	ke kanan	25.5	6.5	2.246
50%	ke kiri	-3	6.5	0.098

Sampel data pengujian dengan beban								
Waktu (s)	ADC	Vbat (V)	Vout (V)	I in (A)	I out (A)	Vel (rpm)	P in (W)	P out (W)
18.900	615	34.44	9.92	9.34	-24.13	85.53	321.76	-239.52
18.925	639	34.91	9.92	8.41	-23.5	85.53	293.56	-233.27
18.950	679	34.86	10.92	8.59	-24.12	85.53	299.41	-263.31
18.975	644	34.47	10.92	9.47	-24.14	85.53	326.36	-263.58
19.200	597	34.65	8.93	7.27	-20.79	93.71	251.8	-185.74
19.225	631	35.16	9.92	7.16	-24.18	93.71	251.9	-239.96
19.250	622	34.76	9.92	7.75	-23.54	93.71	269.56	-233.61
19.275	600	34.86	8.93	7.98	-23.96	93.71	278.35	-214
20.675	629	35.24	9.92	6.74	-21.29	93.71	237.48	-211.27
20.700	741	35.05	12.99	10.05	-24.11	93.71	352.45	-313.3
20.725	668	33.94	10.92	11.65	-24.19	93.71	395.29	-264.11
20.750	638	34.39	9.92	8.63	-24.23	93.71	296.91	-240.49
20.775	635	34.92	9.92	6.94	-22.08	93.71	242.34	-219.17

20.800	628	34.93	9.92	6.92	-22.55	93.71	241.91	-223.82
20.825	625	34.93	9.92	7.02	-22.34	140.43	245.32	-221.74
20.850	656	35	10.92	8.77	-24.19	140.43	306.79	-264.11
20.875	715	34.7	12.99	12.17	-24.11	140.43	422.2	-313.3
20.900	667	33.98	10.92	12.17	-24.23	140.43	413.46	-264.54
20.925	629	34.87	9.92	9.43	-23.72	140.43	328.96	-235.4
22.900	589	35.2	8.93	5.64	-18.94	140.43	198.36	-169.21
22.925	611	35.61	9.92	6.14	-22.81	140.43	218.6	-226.34
22.950	597	35.16	8.93	5.72	-16.18	140.43	201.07	-144.52
22.975	593	35.5	8.93	4.53	-17.71	140.43	160.73	-158.17
23.000	583	35.48	8.93	4.89	-15.96	101.05	173.61	-142.56
24.350	575	35.3	8.93	6.01	-23.27	101.05	212.03	-207.89
24.375	593	35.23	8.93	6.56	-21.42	101.05	231.04	-191.32
24.400	562	35.29	7.94	5.54	-21.77	101.05	195.59	-172.81
24.425	586	35.53	8.93	4.26	-17.52	101.05	151.29	-156.52
24.450	593	35.38	8.93	5.89	-19.95	101.05	208.35	-178.24
24.475	647	35.26	10.92	8.32	-24.24	101.05	293.43	-264.65
24.500	626	34.84	9.92	9.88	-24.21	101.05	344.33	-240.25
24.525	569	34.95	8.93	7.06	-22.86	101.05	246.77	-204.23
24.550	614	35.26	9.92	7.03	-23.84	101.05	247.93	-236.61
24.575	619	35.04	9.92	8.17	-24.26	101.05	286.46	-240.73
25.575	616	35.34	9.92	7.43	-24.3	138.33	262.5	-241.22
25.600	580	34.94	8.93	7.13	-24.31	138.33	249.28	-217.14
25.625	610	35.23	9.92	7.95	-24.32	138.33	280.11	-241.36
25.650	564	35.05	7.94	6.6	-21.7	138.33	231.27	-172.31
25.675	579	35.52	8.93	5.1	-19.61	138.33	181.27	-175.18
25.700	588	35.32	8.93	6.68	-24.25	138.33	235.94	-216.62
25.725	601	35.31	8.93	5.63	-23.49	138.33	198.65	-209.86
25.750	630	35.29	9.92	6.83	-23.98	138.33	241.08	-237.97
25.775	718	35.18	12.99	11.57	-24.17	138.33	407.14	-314
25.800	710	33.88	12.99	17.4	-24.12	138.33	589.53	-313.43
25.825	789	33.97	14.98	18.89	-24.12	138.33	641.63	-361.31
25.850	829	33.2	14.98	23.3	-24.04	138.33	773.42	-360