DAFTAR PUSTAKA

[1] GAPITA, R. S. (2015). Perancangan Mesin Penggulung Kumparan Transformator Berbasis Mikrokontroller Atmega8535 (Doctoral dissertation, Riau Un4ersity).

[2] Yandri, V. R., & Desmiwarman, D. (2016). RANCANG BANGUN ALAT PENGGULUNG KAWAT EMAIL UNTUK KUMPARAN MOTOR MENGGUNAKAN MIKROKONTROLLER ATMEGA328 SEBAGAI UNIT PENGENDALI. Jurnal Teknik Elektro, 5(1).

[3] Ahyar, M., & Irdam, I. (2019). PERANCANGAN MESIN PENGGULUNG KUMPARAN MOTOR LISTRIK SISTEM OTOMATIS BERBASIS MIKROKONTROLER. Jurnal Keteknikan dan Sains (JUTEKS), 2(1), 8-13.

[4] Mukhlisin, M., & Irvawansyah, I. (2020). Alat Penggulung Belitan Motor dan Transformator Berbasis Arduino. Joule (Journal of Electrical Engineering), 1(2), 61-64.

[5] Irfan, D., Junaidi, J., & Surtono, A. (2021). Rancang Bangun Mesin Penggulung Lilitan Kawat Transformator Otomatis Berbasis Arduino Uno. Journal of Energy, Material, and Instrumentation Technology, 2(3), 51-57.

[6] Syahwil, M. (2020). Modifikasi Alat Penggulung Dinamo Sistem Manual Menjadi Otomatis Berbasis Arduino. Indonesian Journal of Laboratory, 3(1), 46-54.

[7] Simanjuntak, R. P., Siregar, A. S., & Panjaitan, A. (2019). RANCANG BANGUN ALAT PENGGULUNG TRANSFORMATOR SECARA OTOMATIS BERBASIS MIKROKONTROLER. Jurnal Ilmiah Dunia Ilmu Vol, 5(1).

[8] Adam, M., & Prabowo, A. (2019). Analisa Penambahan Trafo Sisip Sisi Distribusi 20 Kv Mengurangi Beban Overload Dan Jutah Tegangan Pada Trafo Bl 11 Rayon Tanah Jawa Dengan Simulasi Etab 12.6. 0. RELE (Rekayasa Elektrikal dan Energi): Jurnal Teknik Elektro, 1(2), 62-69.

[9] Septryanti, A., & Fitriyanti, F. (2017). Rancang Bangun Aplikasi Kunci Pintu Otomatis Berbasis Mikrokontroler Arduino Menggunakan *Smartphone* Android. CESS (Journal of Computer Engineering, Sistem and Science), 2(2), 59-63.

[10] Hanief, S., Jepriana, I. W., & Kom, S. (2020). Konsep Algoritme

dan Aplikasinya dalam Bahasa Pemrograman C++. Penerbit Andi.

[11] Artiyasa, M., Kusumah, I. H., Firmansyah, F., Efendi, M. A., & Iriyanto, M. (2020). Studi Perbandingan Platform Internet of Things (IoT) untuk Smart Home Kontrol Lampu Menggunakan NodeMCU dengan Aplikasi Web Thingspeak dan Blynk. Jurnal Fidelitiy, 2(1), 03-10.

[12] Prastyo, B., & Wahjudi, D. (2019). DESIGN OF TIME CONTROL IN GENERAL LIGHTING ROAD TO IMPROVE EFFICIENCY ELECTRICITY ENERGY BASED ON MICROCONTROLLER ATMEGA 328P. Teodolita (Media Komunikasi Ilmiah di Bidang Teknik), 20(1).

[13] Asriyadi, A., Fadliondi, F., & Ciksadan, C. (2022). Rancang Bangun Sistem Keamanan PorTabel Menggunakan GPS dan RFID Berbasis NodeMCU. RESISTOR (Elektronika Kendali Telekomunikasi Tenaga Listrik Komputer), 5(1), 7-14.

[14] Wajiansyah, A., & Supriadi, S. (2020, March). Rancang Bangun Prototipe Embedded Sistem Untuk Kendali Kecepatan Putaran Motor DC. In Prosiding-Seminar Nasional Teknik Elektro UIN Sunan Gunung Djati Bandung (pp. 343-352).

[15] Mauko, C., & Tunliu, S. (2019). KONTROLARAH GERAK WEB KAMERA (WEBCAM) BERBASIS WEB. J. Ilm. Flash, 2(2), 106.

[16] Amarudin, A., Saputra, D. A., & Rubiyah, R. (2020). Rancang Bangun Alat Pemberi Pakan Ikan Menggunakan Mikrokontroler. Jurnal Ilmiah Mahasiswa Kendali Dan Listrik, 1(1), 7-13.

[17] Rahadian, D., Rahayu, G., & Oktavia, R. R. (2019). Teknologi Pendidikan: Kajian Aplikasi Ruangguru Berdasarkan Prinsip dan Paradigma Interaksi Manusia dan Komputer. Jurnal Petik, 5(1), 11-24.

[18] Muhajar, A. (2022). SISTEM KENDALI DAN *MONITORING* INFUS BERBASIS INTERNET OF THINGS (Doctoral dissertation, Un4ersitas Siliwangi).

[19] Purwanto, S. (2021). Rancang Bangun Electric Power Converter (Catu Daya) Untuk Alat Anodizing PorTabel. ENERGI & KELISTRIKAN, 13(2), 86-94.

[20] Pela, M. F., & Pramudita, R. (2021). Sistem *Monitoring* Penggunaan Daya Listrik Berbasis Internet Of Things Pada Rumah Dengan Menggunakan Aplikasi Blynk. Infotech: Journal of Technology Information, 7(1), 47-54.

[21] Jumrianto, J. (2021). Pengujian Sensit4itas Sensor Arus dan Sensor Tegangan pada Sistem Pengukuran Electrical Tracking Test. Journal of Sistem, Information Technology and Electronics Engineering (J-SITEE), 1(2), 30-39.

[22] Ghito, R. K., & Nurdiana, N. (2018, October). Rancang Bangun Smart Garden Sistem Menggunakan Sensor Soil Moisture Dan Arduino Berbasis Android (Studi Kasus: Di Gerai Bibit Narnea Cikijing). In Prosiding Industrial Research Workshop and National Seminar (Vol. 9, pp. 166-170

[23] Setiawan, S. I. A. (2011). Google SketchUp Perangkat Alternatif dalam Pemodelan 3D. Ultimatics: Jurnal Teknik Informatika, 3(2), 6-10.

[24] Noor, F. A., Ananta, H., & Sunardiyo, S. (2017). Pengaruh Penambahan Kapasitor Terhadap tegangan, arus, faktor daya, dan daya aktif pada beban listrik di minimarket. Jurnal Teknik Elektro, 9(2), 66-73.

[25] Novianto, Dwi. (2021, August). Menghitung kumparan primer dan sekunder

~Halaman Ini Sengaja Dikosongkan~

DAFTAR LAMPIRAN

LAMPIRAN A
PROGRAM CODEBLOCKS DAN ARDUINO IDE

1. PROGRAM CODEBLOCKS

a. PROGRAM RANGKAIAN KONTROL MOTOR STEPPER MESIN PENGGULUNG KUMPARAN TRANSFORMATOR BERBASIS MIKROKONTROLER ATMEGA 328

```
#include <avr/eeprom.h>
#include <stdlib.h>

#include "libs/keypad.h"
#include "libs/hd44780.h"
#include "libs/adc.h"
#include "libs/timer.h"

#define BUZZR PC_0
#define L_LCD PD_4
#define RLY_M PD_5
#define SENSR 2

#define cnt_flg 0
#define mtr_flg 1

/** used variables */
uint8_t flag = 0;
int     tmr0, tmr1, _break = 0;
char    buff[10];


/** Global Prototypes */
void IO_Init(void);
void ISR_Timer0(void);
void ISR_Timer1(void);
void LCD_BackLightOn(void);
void BUZZER_On(void);
void DISPLAY_WellcomeScreen(void);
```

```
void DISPLAY_CounterLimit(unsigned long);
void DISPLAY_SettingCounterLimit(void);
void Setting_CounterLimit(void);

int main(void) {
  unsigned long _limits = eeprom_read_dword((uint32_t*)0x00);
  unsigned long _count = 0;

  delay_ms(200);

  KEYPAD_SetUp(PD_0, PD_1, PD_2, PD_3, PC_5, PC_4, PC_3,
        P_NC);
  LCD_SetUp(PB_0, P_NC, PB_1, P_NC, P_NC, P_NC, P_NC,
        PB_2, PB_3, PB_4, PB_5);
  LCD_Init(2, 16, LCD_FONT_5X8);
  IO_Init();
  ADC_Init();
  TIMER_SetTime(0, 10000);
  TIMER_SetTime(1, 1000000);
  TIMER_AttachInterrupt(0, ISR_Timer0);
  TIMER_AttachInterrupt(1, ISR_Timer1);

  sei();

  flag = 0;

  DISPLAY_WellcomeScreen();
  DISPLAY_CounterLimit(_limits);

  while(1) {
    char c = KEYPAD_GetKey();
    if(c != ' ') {
      if(c == '*') {
        if(util_IsBitCleared(flag, mtr_flg)) {
          Setting_CounterLimit();
          _limits = eeprom_read_dword((uint32_t*)0x00);
          _count = 0;
          flag = 0;
          DISPLAY_CounterLimit(_limits);
```

```
      }
     }
     else if(c == '#') {
      if(_count >= _limits) {
       LCD_SetCursor(1, 7);
       LCD_DisplayString("       ");
       _count = 0;
      }
      else {
       if(util_IsBitCleared(flag, cnt_flg)) {
        util_BitSet(flag, cnt_flg);
        BUZZER_On();
       }
       util_BitToggle(flag, mtr_flg);
       if(util_IsBitSet(flag, mtr_flg)) GPIO_PinWrite(RLY_M, 1);
       else GPIO_PinWrite(RLY_M, 0);
      }
     }
     LCD_BackLightOn();
     KEYPAD_WaitRelease();
    }

    if(ADC_GetValue(SENSR) > 300) {
     if(++_count >= _limits) {
      flag = 0;
      BUZZER_On();
     }
     if(_count >= _limits-_break) GPIO_PinWrite(RLY_M, 0);

     while(ADC_GetValue(SENSR) > 300);
    }

    ultoa(_count, buff, 10);
    LCD_SetCursor(1, 7);
    LCD_DisplayString(buff);
   };

  return 0;
}
```

```
void IO_Init(void) {
  GPIO_PinDirection(BUZZR, OUTPUT);
  GPIO_PinDirection(L_LCD, OUTPUT);
  GPIO_PinDirection(RLY_M, OUTPUT);
  GPIO_PinDirection(SENSR, INPUT);

  GPIO_PinWrite(BUZZR, LOW);
  GPIO_PinWrite(L_LCD, LOW);
  GPIO_PinWrite(RLY_M, LOW);
}

void ISR_Timer0(void) {
  if(++tmr0 > 200) {
    TIMER_Stop(0);
    GPIO_PinWrite(BUZZR, LOW);
  }
}

void ISR_Timer1(void) {
  if(++tmr1 > 60) {
    TIMER_Stop(1);
    GPIO_PinWrite(L_LCD, LOW);
  }
}

void LCD_BackLightOn(void) {
  GPIO_PinWrite(L_LCD, HIGH);
  tmr1 = 0;
  TIMER_Start(1);
}

void BUZZER_On(void) {
  GPIO_PinWrite(BUZZR, HIGH);
  if(util_IsBitSet(flag, cnt_flg)) tmr0 = 170;
  else tmr0 = 0;
  TIMER_Start(0);
}
```

```
void DISPLAY_WellcomeScreen(void) {
  LCD_BackLightOn();
  LCD_SetCursor(0, 0);
  LCD_DisplayString("POLITEKNIK");
  LCD_SetCursor(1, 0);
  LCD_DisplayString("NEGERI CILACAP");
  for(int i=0; i<3; i++) {
    LCD_CmdWrite(CMD_LCD_VISIBLE);
    GPIO_PinWrite(BUZZR, HIGH);
    delay_ms(500);
    LCD_CmdWrite(CMD_LCD_BLANK);
    GPIO_PinWrite(BUZZR, LOW);
    delay_ms(500);
  }
  GPIO_PinWrite(BUZZR, LOW);
}

void DISPLAY_CounterLimit(unsigned long limit) {
  LCD_CmdWrite(CMD_DISPLAY_ON_CURSOR_OFF);
  ultoa(limit, buff, 10);

  LCD_Clear();
  LCD_DisplayString("Limit: ");
  LCD_DisplayString(buff);
  LCD_SetCursor(1, 0);
  LCD_DisplayString("Count:");
}

void DISPLAY_SettingCounterLimit(void) {
  LCD_CmdWrite(CMD_LCD_CLEAR);
  LCD_SetCursor(0, 1);
  LCD_DisplayString("Value is 10 to");
  LCD_SetCursor(1, 3);
  LCD_DisplayString("9.999.999");
  delay_sec(1);

  LCD_CmdWrite(CMD_LCD_CLEAR);
  LCD_CmdWrite(CMD_DISPLAY_ON_CURSOR_BLINK);
  LCD_SetCursor(0, 2);
```

```
  LCD_DisplayString("Insert Limit:");
  LCD_SetCursor(1, 4);
}

void Setting_CounterLimit(void) {
  unsigned long max;

displayMax:
  max = 0;
  char max_chr[] = "       ";
  int i = 0;

  DISPLAY_SettingCounterLimit();
  KEYPAD_WaitRelease();

  char c = KEYPAD_GetKey();
  KEYPAD_WaitRelease();
  while(c!='#') {
    if(c!='*') {
      if(c!=' ') {
        LCD_SetCursor(1, (4+i));
        LCD_DisplayChar(c);
        max_chr[i] = c;
        if(++i>7) goto displayMax;
      }
    }
    else if(c=='*') {
      if(--i<0) goto displayMax;
      LCD_SetCursor(1, (4+i));

      LCD_DisplayChar(' ');
      max_chr[i] = ' ';
      LCD_SetCursor(1, (4+i));
    }
    c = KEYPAD_GetKey();
    KEYPAD_WaitRelease();
  }
  LCD_Clear();
  max = atol(max_chr);
```

```
 if(max < 10) goto displayMax;

 eeprom_write_dword((uint32_t*)0x00, max);
}
```

b. PROGRAM RANGKAIAN COUNTER MESIN PENGGULUNG KUMPARAN TRANSFORMATOR BERBASIS MKROKONTROLER ATMEGA 328

```
#include <avr/eeprom.h>
#include <avr/io.h>
#include <stdlib.h>

#include "libs/bigfont.h"
#include "libs/keypad.h"
#include "libs/hd44780.h"
#include "libs/timer.h"

#define MOT   PD_4
#define LIM   PC_1
#define SENS  PC_2
#define ENBL  PD_5
#define DIR   PD_6
#define PLS   PD_7
#define D_ADR 0x0
#define L_ADR 0x8

uint16_t _d_pls, _l_pls;
char _buff[7];

void DEV_Winding(void);
void DEV_Reset(void);
void DEV_Init(void);

int main(void) {
  DEV_Init();

  KEYPAD_SetUp(PD_0, PD_1, PD_2, PD_3, PC_5, PC_4, PC_3,
       P_NC);
```

```
  while(1) {
    char _key = KEYPAD_GetKey();
    if(_key != ' ') {
      if(_key == '*') DEV_Reset();
      else if(_key == '#') DEV_Winding();
    };
  };
}

void Lcd_DispParameter(void) {
  LCD_SetCursor(1, 0);
  LCD_DisplayString("D:      L:");

  itoa(_d_pls, _buff, 10);
  LCD_SetCursor(1, 2);
  LCD_DisplayString(_buff);
  itoa(_l_pls, _buff, 10);
  LCD_SetCursor(1, 10);
  LCD_DisplayString(_buff);
}

void STEPPER_StepPulse(void) {
  GPIO_PinWrite(PLS, 1);
  delay_us(800);
  GPIO_PinWrite(PLS, 0);
  delay_us(500);
}

void STEPPER_Turn(char _direction) {
  if((_direction == 'i') || (_direction == 'o')) {
    GPIO_PinWrite(ENBL, 0);
    if(_direction == 'i') GPIO_PinWrite(DIR, 0);
    else if(_direction == 'o') GPIO_PinWrite(DIR, 1);
  } else GPIO_PinWrite(ENBL, 1);
  delay_ms(1);
}

void DEV_Winding(void) {
  KEYPAD_WaitRelease();
```

```
LCD_Clear();
LCD_DisplayString(" WINDING");

GPIO_PinWrite(MOT, 1);

int _dir = 1, _pls;

while(1) {
  _pls = 0;
  Lcd_DispParameter();
  itoa(_pls, _buff, 10);
  LCD_SetCursor(1, 2);
  LCD_DisplayString(_buff);
  LCD_DisplayString("     ");
  LCD_SetCursor(0, 11);
  if(_dir == 1) {
    LCD_DisplayString("--->");
    STEPPER_Turn('o');
  } else {
    LCD_DisplayString("<---");
    STEPPER_Turn('i');
  }

  while(_pls < _l_pls) {
    if(GPIO_PinRead(SENS)) {
      for(uint8_t i8=0; i8<_d_pls; i8++) {
        STEPPER_StepPulse();
        _pls++;
      }
      itoa(_pls, _buff, 10);
      LCD_SetCursor(1, 2);
      LCD_DisplayString(_buff);
      while(GPIO_PinRead(SENS));
    }
    if(KEYPAD_GetKey() == '*') goto stop;
  }
  _dir = -1*_dir;
};
```

```
stop:
  GPIO_PinWrite(MOT, 0);
  STEPPER_Turn('s');
  KEYPAD_WaitRelease();
  while(KEYPAD_GetKey() != '*');
  DEV_Reset();
  KEYPAD_WaitRelease();
}

uint16_t DEV_PerimeterSetupValue(uint8_t _perimetercode) {
reSetup:
  uint8_t _xpos;

  for(uint8_t i8=0; i8<7; i8++) _buff[i8] = '\0';
  uint16_t _limit, _value;
  uint8_t _invalid=0, i=0;

  if(_perimetercode == 0) {
    _limit = 250;
    _xpos = 2;
    LCD_SetCursor(1, _xpos);
  }
  else {
    _limit = 15000;
    _xpos = 10;
    LCD_SetCursor(1, _xpos);
  }
  LCD_DisplayString("0     ");

  LCD_SetCursor(1, _xpos);
  while(1){
    char c = KEYPAD_GetKeyPress();
    if(c != ' ') {
      if(c == '#') break;
      else if(c == '*') {
        _invalid = 1; break;
      }
      else {
        _buff[i++] = c;
```

```
      LCD_SetCursor(1, _xpos);
      LCD_DisplayString(_buff);
     }
    }
   }
   _value = atoi(_buff);

   if((_invalid != 0)) goto reSetup;
   if(_perimetercode == 0) { if((_value < 1) || (_value > _limit)) goto
        reSetup; }
   else if(_perimetercode == 1) { if((_value < _d_pls) || (_value >
        _limit)) goto reSetup; }

   return _value;
}

void DEV_Setup(void) {
   LCD_Clear();
   LCD_CmdWrite(CMD_DISPLAY_ON_CURSOR_BLINK);
   LCD_DisplayString("PARAMETER SETUP:");
   Lcd_DispParameter();

   KEYPAD_WaitRelease();

   _d_pls = DEV_PerimeterSetupValue(0);
   eeprom_update_word((uint16_t*)D_ADR, _d_pls);
   _l_pls = DEV_PerimeterSetupValue(1);
   eeprom_update_word((uint16_t*)L_ADR, _l_pls);

   LCD_CmdWrite(CMD_DISPLAY_ON_CURSOR_OFF);
   delay_ms(100);
}

void DEV_Reset(void) {
   _d_pls = eeprom_read_word((uint16_t*)D_ADR);
   _l_pls = eeprom_read_word((uint16_t*)L_ADR);

   for(uint8_t i8=0; i8<66; i8++) {
     if(KEYPAD_GetKey() != '*') goto pass;
```

```
    delay_ms(10);
  }
  DEV_Setup();

pass:
  if(GPIO_PinRead(LIM)) {
    STEPPER_Turn('i');
    while(GPIO_PinRead(LIM)) STEPPER_StepPulse();
  }
  STEPPER_Turn('s');

  LCD_Clear();
  LCD_DisplayString("    READY..!");
  Lcd_DispParameter();
}

void DEV_Init(void) {
  LCD_SetUp(PB_0, P_NC, PB_1, P_NC, P_NC, P_NC, P_NC,
        PB_2, PB_3, PB_4, PB_5);
  LCD_Init(4, 16, LCD_FONT_5X8);
  LCD_BuildChars();
  LCD_CmdWrite(CMD_DISPLAY_ON_CURSOR_OFF);

  LCD_SetCursorBigFont(0, 3);
  LCD_PrintBigText("PNC", 0);
  for(uint8_t i8=0; i8<15; i8++) {
    delay_ms(200);
    LCD_CmdWrite(CMD_LCD_SCROL_LEFT);
  }

  LCD_Clear();
  LCD_DisplayString(" CILACAP...");

  _d_pls = eeprom_read_word((uint16_t*)D_ADR);
  _l_pls = eeprom_read_word((uint16_t*)L_ADR);

  if(_d_pls<1 || _d_pls>25) {
    _d_pls = 250;
    eeprom_update_word((uint16_t*)D_ADR,_d_pls);
```

```
  }
  if(_l_pls<_d_pls || _l_pls>15000) {
    _l_pls = 15000;
    eeprom_update_word((uint16_t*)L_ADR,_l_pls);
  }
  DDRC &= ~((1<<2)|(1<<1));
  DDRD |= 0b11110000;

  PORTD |= (1<<5);

  if(GPIO_PinRead(LIM)) {
    STEPPER_Turn('i');
    while(GPIO_PinRead(LIM)) STEPPER_StepPulse();
  }
  STEPPER_Turn('s');

  delay_ms(500);

  DEV_Reset();
}
```

2. PROGRAM ARDUINO UNO

Program Mesin Pengujian Tegangan

```
/* Comment this out to disable prints and save space */
#define BLYNK_PRINT Serial

/* Fill-in your Template ID (only if using Blynk.Cloud) */
//#define BLYNK_TEMPLATE_ID   "YourTemplateID"
#define BLYNK_TEMPLATE_ID "TMPLYXi15xi3"
#define BLYNK_DEVICE_NAME "Monitoring Tegangan"
#defineBLYNK_AUTH_TOKEN
        "1xwtJF5GWFQROxyurf1JVvrlO1SekITL"

#include <ESP8266WiFi.h>
#include <BlynkSimpleEsp8266.h>
#include <PZEM004Tv30.h>

// You should get Auth Token in the Blynk.
// Go to the Project Settings (nut icon).
char auth[] = "1xwtJF5GWFQROxyurf1JVvrlO1SekITL";

// Your WiFi credentials.
// Set password to "" for open networks.
char ssid[] = "saya";
char pass[] = "yasudahlah";

PZEM004Tv30 pzem(12, 13); // 12 = D6 (rx), 13 = D7 (tx)
float Power, Energy, Voltase, Current;
const int input_sensor = A0; // nama lain dari A0 yaitu
        input_sensor
const int output_pwm = 2; // nama lain dari 2 yaitu output_pwm

//nilai default variabel data
int nilai_adc = 00;
int nilai_pwm = 00;
float voltageOut = 00;
float voltage_ac = 00;

void setup()
{
```

```
  // Debug console
  Serial.begin(9600);

  Blynk.begin(auth, ssid, pass);
  // You can also specify server:
  //Blynk.begin(auth, ssid, pass, "blynk-cloud.com", 80);
  //Blynk.begin(auth, ssid, pass, IPAddress(192,168,1,100), 8080);
}

void loop()
{
   // membaca nilai power
Power = pzem.power();
// jika gagal membaca power kW
if(isnan(Power))
{
  Serial.println("gagal membaca power");

}
else{
  Serial.print ("Power : ");
  Serial.print (Power);
  Serial.println ("kW");
}

// membaca nilai energy
Energy = pzem.energy();
// jika gagal membaca power kWh
if(isnan(Energy))
{
  Serial.println("gagal membaca energy");

}
else{
  Serial.print ("Energy : ");
  Serial.print (Energy);
  Serial.println ("kWh");
}
// membaca nilai Voltase
```

```
Voltase = pzem.voltage();
// jika gagal membaca Voltase
if(isnan(Voltase))
{
  Serial.println("gagal membaca Voltase");

}
else{
  Serial.print ("Voltase : ");
  Serial.print (Voltase);
  Serial.println ("V");
}
// membaca nilai current
Current = pzem.current();
// jika gagal membaca Current
if(isnan(Current))
{
  Serial.println("gagal membaca Current");

}
else{
  Serial.print ("Current : ");
  Serial.print (Current);
  Serial.println ("A");
}
 nilai_adc = analogRead(input_sensor); // Prosedur pembacaan
        analog pin
  // mapping pembacaan resolusi sensor
  nilai_pwm = map(nilai_adc, 00, 1023, 00 , 255);
  // mengubah nilai analog out
  analogWrite(output_pwm, nilai_pwm);
 voltageOut = nilai_adc * (5.0 / 1023.0);
 voltage_ac = map(nilai_adc, 00,560,00,27); //didapat dari
        pembacaan program adc diat

  // Mencetak hasil ke monitor serial:
  Serial.print("sensor = " );
  Serial.println(nilai_adc);
  Serial.print("output pwm= ");
```

```
  Serial.println(nilai_pwm);
  Serial.print("Voltage ADC = ");
  Serial.println(voltageOut);
  Serial.print("Voltage AC = ");
  Serial.println(voltage_ac);
  Serial.println();

//kirimdata ke Blynk
// Power = V0
// Energy = V1
// Voltase = V2
// Current = V3
Blynk.virtualWrite(V0, Power);
Blynk.virtualWrite(V1, Energy);
Blynk.virtualWrite(V2, Voltase);
Blynk.virtualWrite(V3, Current);
Blynk.virtualWrite(V4, voltage_ac);
delay(5000);
  Blynk.run();
}
```

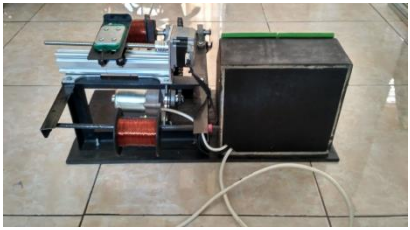

A-17

LAMPIRAN B
DESAIN MEKANIK

1. Mesin Penggulung Kumparan Transformator

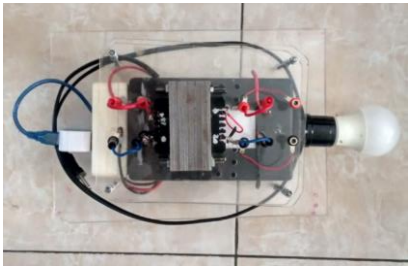
Gambar MekanikTampak Depan

Gambar Mekanik Tampak Samping

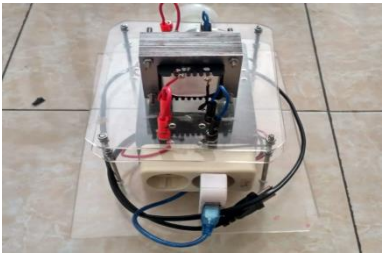


Gambar Mekanik Tampak Belakang

2. Alat Pengujian Tegangan Trasformator

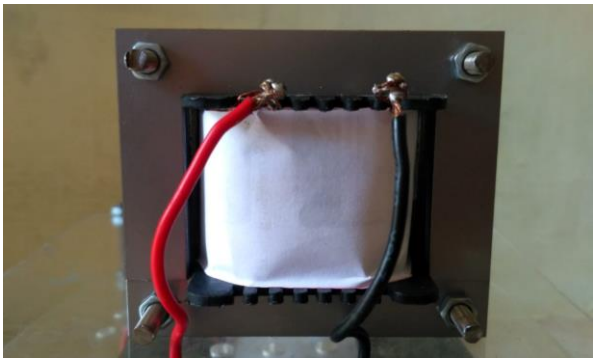


Gambar Mekanik Tampak Atas

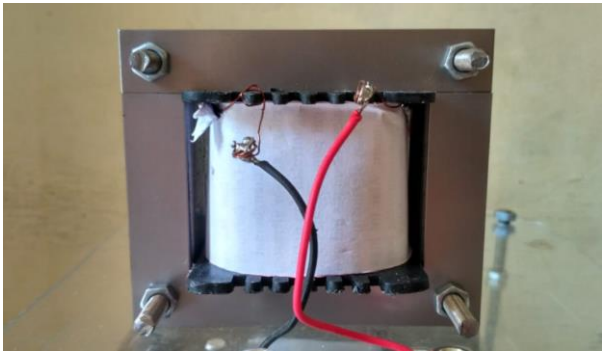


Gambar Mekanik Tampak Belakang

LAMPIRAN C
HASIL LILITAN TRANSFORMATOR



Gambar Transformator Step Down 27 V



Gambar Transformator Step Down 27 V

LAMPIRAN D
TABEL AMERICAN WIRE GUIDE

Table 1: American Wire Gauge (AWG) Cable / Conductor Sizes and Properties

AWG	Diameter [inches]	Diameter [mm]	Area [mm^2]	Resistance [Ohms / 1000 ft]	Resistance [Ohms / km]	Max Current [Amperes]	Max Frequency for 100% skin depth
0000 (4/0)	0.46	11.684	107	0.049	0.16072	302	125 Hz
000 (3/0)	0.4096	10.40384	85	0.0618	0.202704	239	160 Hz
00 (2/0)	0.3648	9.26592	67.4	0.0779	0.255512	190	200 Hz
0 (1/0)	0.3249	8.25246	53.5	0.0983	0.322424	150	250 Hz
1	0.2893	7.34822	42.4	0.1239	0.406392	119	325 Hz
2	0.2576	6.54304	33.6	0.1563	0.512664	94	410 Hz
3	0.2294	5.82676	26.7	0.197	0.64616	75	500 Hz
4	0.2043	5.18922	21.2	0.2485	0.81508	60	650 Hz
5	0.1819	4.62026	16.8	0.3133	1.027624	47	810 Hz
6	0.162	4.1148	13.3	0.3951	1.295928	37	1100 Hz
7	0.1443	3.66522	10.5	0.4982	1.634096	30	1300 Hz
8	0.1285	3.2639	8.37	0.6282	2.060496	24	1650 Hz
9	0.1144	2.90576	6.63	0.7921	2.598088	19	2050 Hz
10	0.1019	2.58826	5.26	0.9989	3.276392	15	2600 Hz
11	0.0907	2.30378	4.17	1.26	4.1328	12	3200 Hz
12	0.0808	2.05232	3.31	1.588	5.20864	9.3	4150 Hz
13	0.072	1.8288	2.62	2.003	6.56984	7.4	5300 Hz
14	0.0641	1.62814	2.08	2.525	8.282	5.9	6700 Hz
15	0.0571	1.45034	1.65	3.184	10.44352	4.7	8250 Hz
16	0.0508	1.29032	1.31	4.016	13.17248	3.7	11 k Hz
17	0.0453	1.15062	1.04	5.064	16.60992	2.9	13 k Hz
18	0.0403	1.02362	0.823	6.385	20.9428	2.3	17 kHz
19	0.0359	0.91186	0.653	8.051	26.40728	1.8	21 kHz
20	0.032	0.8128	0.518	10.15	33.292	1.5	27 kHz
21	0.0285	0.7239	0.41	12.8	41.984	1.2	33 kHz
22	0.0254	0.64516	0.326	16.14	52.9392	0.92	42 kHz
23	0.0226	0.57404	0.258	20.36	66.7808	0.729	53 kHz
24	0.0201	0.51054	0.205	25.67	84.1976	0.577	68 kHz
25	0.0179	0.45466	0.162	32.37	106.1736	0.457	85 kHz
26	0.0159	0.40386	0.129	40.81	133.8568	0.361	107 kHz
27	0.0142	0.36068	0.102	51.47	168.8216	0.288	130 kHz
28	0.0126	0.32004	0.081	64.9	212.872	0.226	170 kHz
29	0.0113	0.28702	0.0642	81.83	268.4024	0.182	210 kHz
30	0.01	0.254	0.0509	103.2	338.496	0.142	270 kHz
31	0.0089	0.22606	0.0404	130.1	426.728	0.113	340 kHz
32	0.008	0.2032	0.032	164.1	538.248	0.091	430 kHz
33	0.0071	0.18034	0.0254	206.9	678.632	0.072	540 kHz
34	0.0063	0.16002	0.0201	260.9	855.752	0.056	690 kHz
35	0.0056	0.14224	0.016	329	1079.12	0.044	870 kHz
36	0.005	0.127	0.0127	414.8	1360	0.035	1100 kHz
37	0.0045	0.1143	0.01	523.1	1715	0.0289	1350 kHz
38	0.004	0.1016	0.00797	659.6	2163	0.0228	1750 kHz
39	0.0035	0.0889	0.00632	831.8	2728	0.0175	2250 kHz
40	0.0031	0.07874	0.00501	1049	3440	0.0137	2900 kHz

BIODATA PENULIS

Nama	: Arief Rahman Hakim
Tempat/Tanggal Lahir	: Cilacap, 24 Mei 2001
Alamat	: Jalan Kauman Utara No. 72, Gunungreja RT 01 RW 02, Sidareja, Cilacap, Jawa Tengah. Kode Pos 53261
Telepon/HP	: +6282242617160
Hobi	: Sepak Bola
Motto	: "*Man Jadda Wa Jadda*"

Riwayat Pendidikan:

- SD Negeri 02 Gunungreja — Tahun 2007 - 2013
- SMP Negeri 1 Sidareja — Tahun 2013 - 2016
- SMA Negeri 1 Sidareja — Tahun 2016 - 2019
- Politeknik Negeri Cilacap — Tahun 2019 - 2022

Penulis telah mengikuti sidang Tugas Akhir pada tanggal 05 Agustus 2022 sebagai salah satu persyaratan untuk memperoleh gelar Ahli Madya (A.Md).