BAB IV
PEMBUATAN ALAT

4.1 Pembuatan Prototype

a. Pembuatan Perangkat Keras (Hardware)

b. Pembuatan Perangkat Lunak (Software)

4.2 Alat dan Bahan

Tabel 4.1 Bahan Pembuatan Tugas Akhir

No	Komponen	Spesifikasi	Jumlah
1	Mikrokontroler	ESP32 DevKit V1	1
2	Panel Surya Polikristalin	50 Wp, Vmp 19.12V, Imp 2.45 A, Voc 22.68V, Isc 2.62 A.	1
2	Sensor Cahaya	BH1750	1
3	Sensor Tegangan & Arus	INA219	1
4	Power Supply	PSU 12V	1
5	Buck Converter	MP1584	2
6	Resistor Kapur	15w 10 ohm	1
7	LCD	20x4 I2c	1
8	Keypad	I2C 4x4	1
9	Connector	JST XH 4p 2,54 mm	3
10	Connector	JST XH 5p 2,54 mm	1
11	Connector	JST XH 6p 2,54 mm	1
12	Terminal Block	Pitch 5.08 mm	3
13	PCB	Fiber	1

Tabel 4.2 Alat yang digunakan

No	Nama Alat	Jumlah
1	Multimeter	1

No	Nama Alat	Jumlah
2	Solder	1
3	Penyedot Timah	1
4	Tang Potong	1
5	Bor tangan	1
6	Obeng (-)	1
7	Obeng (+)	1
8	Cairan Pelarut	1
9	Gunting	1
10	Lux Meter	1

4.3 Pembuatan Perangkat Keras

4.3.1 Perancangan Mekanik

Perancangan mekanik dilakukan untuk menghasilkan prototype sistem monitoring kinerja panel surya yang dapat menempatkan seluruh komponen secara terintegrasi dan mudah digunakan selama proses monitoring. Pada tahap ini, perancangan difokuskan pada penempatan panel surya, project box, sensor BH1750, serta dudukan penyangga agar sistem dapat bekerja secara stabil dan aman ketika digunakan di lingkungan luar ruangan.

Panel surya polikristalin 50 WP dipasang pada dudukan dengan sudut kemiringan tertentu agar permukaan panel dapat menerima paparan cahaya matahari secara optimal. Selanjutnya, project box dipasang pada bagian samping dudukan panel dan digunakan sebagai tempat penyimpanan komponen elektronik, seperti ESP32 DevKit V1, sensor INA219, buck converter MP1584, serta rangkaian pendukung lainnya. Penggunaan project box bertujuan untuk melindungi komponen dari debu, benturan, maupun gangguan lingkungan yang dapat memengaruhi kinerja sistem. Sensor BH1750 ditempatkan pada bagian samping panel surya dengan posisi yang menghadap ke arah datangnya cahaya matahari. Penempatan ini dilakukan agar sensor dapat mengukur intensitas cahaya yang diterima panel secara langsung sehingga nilai lux yang diperoleh dapat merepresentasikan kondisi radiasi matahari

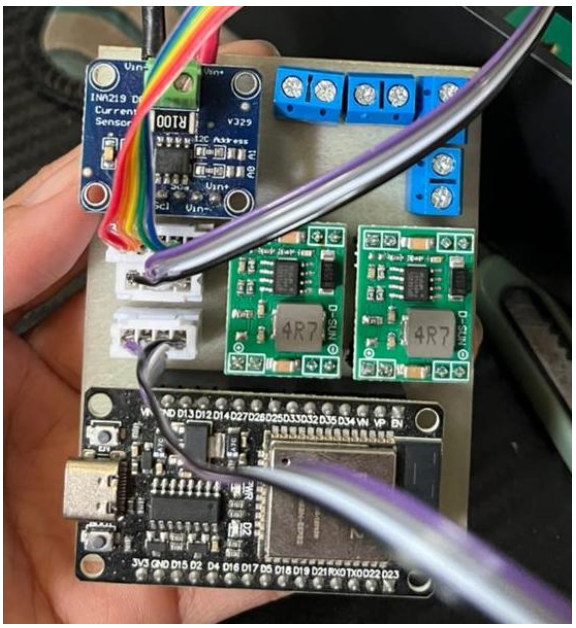
pada permukaan panel surya. Sementara itu, sensor INA219 ditempatkan di dalam project box karena sensor ini digunakan untuk mengukur parameter kelistrikan berupa tegangan dan arus keluaran panel melalui rangkaian kabel yang terhubung secara langsung.

4.3.2 Perancangan Elektrik

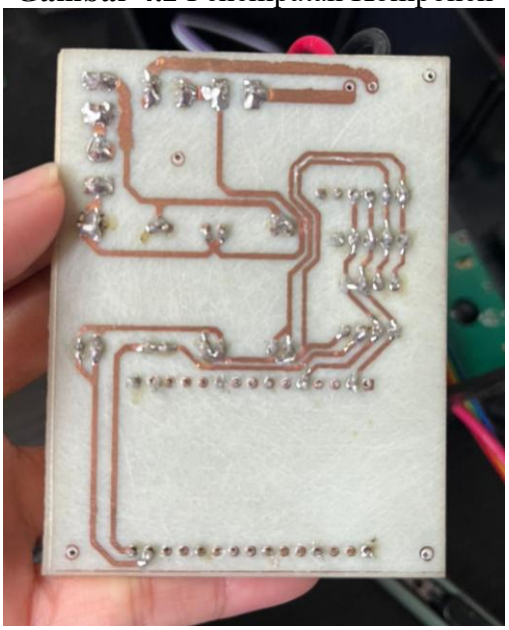
Setelah tahap pembuatan perangkat keras selesai, langkah selanjutnya adalah melakukan perancangan prototype secara menyeluruh yang mencakup integrasi seluruh komponen elektronik pada sistem monitoring kinerja panel surya. Perancangan ini bertujuan untuk memastikan bahwa setiap komponen dapat bekerja secara optimal sesuai dengan fungsi masing-masing, sehingga sistem mampu melakukan monitoring intensitas cahaya, tegangan, arus, daya, dan efisiensi panel surya secara real-time. Selain itu, perancangan juga mempertimbangkan aspek kemudahan penggunaan, keamanan penempatan komponen, serta keandalan sistem selama proses monitoring berlangsung. Posisi dan tata letak komponen yang digunakan pada sistem menjadi acuan dalam proses perancangan dan pencetakan PCB

Gambar 4.1 Design Alat

agar seluruh rangkaian dapat terhubung dengan baik dan bekerja sesuai dengan perancangan yang telah dibuat.



Gambar 4.2 Penempatan Komponen

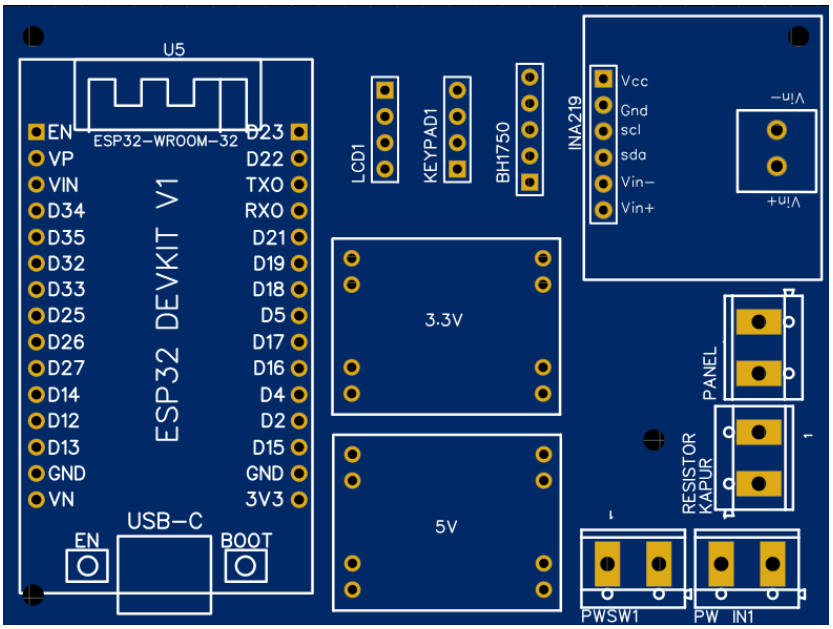


Gambar 4.3 Jalur PCB

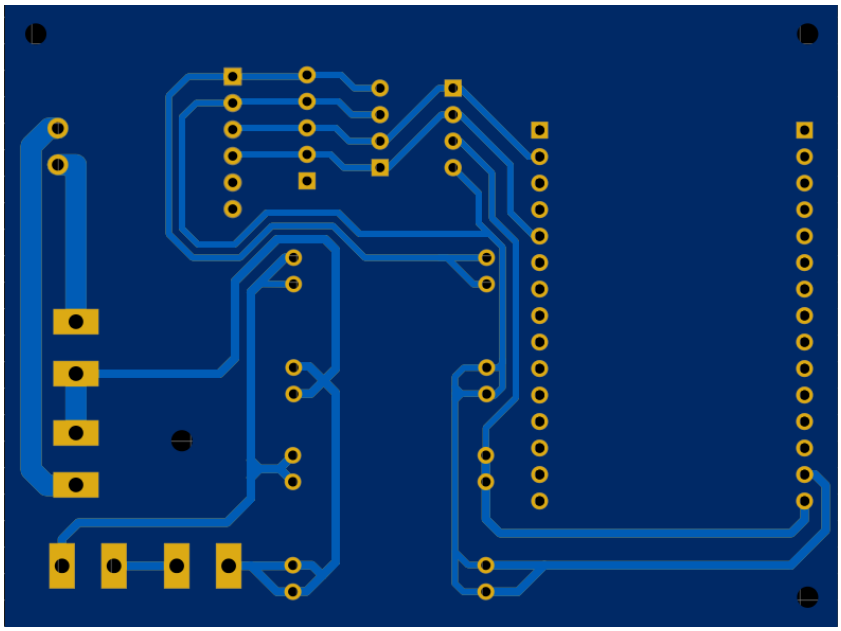
4.3.3 Perancangan PCB

Untuk meningkatkan kerapian, keandalan, dan efisiensi dalam pemasangan komponen elektronik, sistem monitoring kinerja panel surya berbasis Internet of Things (IoT) pada penelitian ini dirancang menggunakan Printed Circuit Board (PCB)

khusus. Penggunaan PCB bertujuan untuk mengintegrasikan seluruh komponen utama ke dalam satu papan rangkaian sehingga hubungan antar komponen menjadi lebih teratur, mempermudah proses perakitan, serta mengurangi kemungkinan kesalahan pemasangan kabel.



Gambar 4.4 Layout Depan PCB



Gambar 4.5 Layout Belakang PCB

Pada Gambar 4.4 dan Gambar 4.5 menunjukkan desain PCB yang digunakan pada sistem monitoring kinerja panel surya berbasis IoT menggunakan sensor BH1750 dan sensor INA219 berdasarkan radiasi matahari. PCB tersebut berfungsi sebagai media penghubung seluruh komponen utama sistem yang dikendalikan oleh mikrokontroler ESP32 DevKit V1. Melalui PCB ini, jalur distribusi catu daya, jalur komunikasi data, serta koneksi antar modul dapat diatur secara terintegrasi sehingga sistem dapat bekerja secara stabil dan lebih mudah dalam proses perawatan maupun pengembangan.

Pada bagian kiri PCB terdapat terminal input catu daya (Power In) yang menerima tegangan sebesar 12 Volt DC dari switching power supply. Tegangan tersebut kemudian didistribusikan menuju dua buah buck converter MP1584 yang berfungsi sebagai penurun tegangan. Buck converter pertama menghasilkan tegangan 5 Volt yang digunakan untuk mensuplai ESP32 DevKit V1 melalui pin VIN serta LCD I2C. Sementara itu, buck converter kedua menghasilkan tegangan 3,3 Volt yang digunakan untuk menyuplai sensor BH1750, sensor INA219, dan modul lain yang memerlukan tegangan kerja 3,3 Volt. Pemisahan jalur catu daya ini dilakukan agar masing-masing komponen memperoleh tegangan yang sesuai dengan spesifikasi kerjanya sehingga sistem dapat beroperasi dengan lebih aman dan stabil.

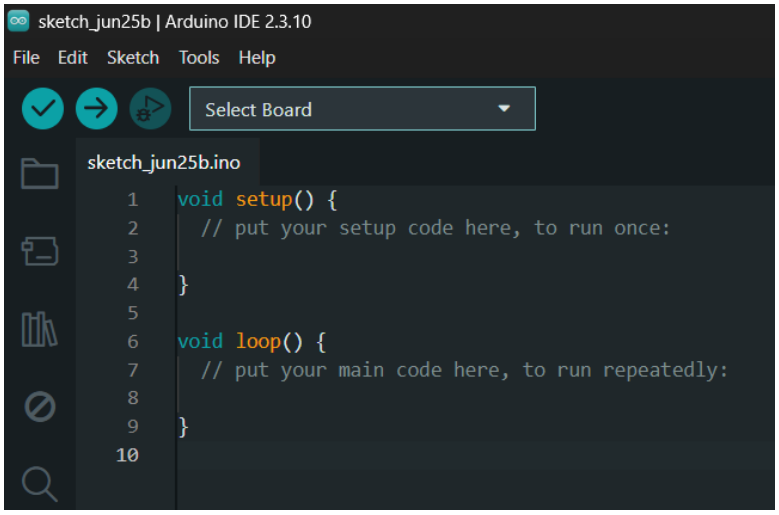
W/m^2

4.4 Pembuatan Perangkat Lunak (Software)

Pada tahap ini dilakukan pengembangan perangkat lunak (software) untuk mengimplementasikan algoritma sistem monitoring pada ESP32 DevKit V1 menggunakan Arduino IDE. Program yang dibuat berfungsi untuk membaca data sensor BH1750 dan INA219, menghitung parameter panel surya, menampilkan hasil

Gambar 4.6 Aplikasi Arduino IDE

monitoring pada LCD, serta mengirimkan data ke Telegram dan Google Spreadsheet. Dengan program tersebut, seluruh komponen dapat bekerja secara terintegrasi sesuai dengan fungsi sistem yang telah dirancang. Tampilan aplikasi Arduino IDE ditunjukkan pada Gambar 4.6 dan tampilan awal Arduino IDE ditunjukkan pada Gambar 4.7 Gambar 4.6 tampilan aplikasi Arduino IDE dan Gambar 4.7 tampilan awal Arduino IDE.



Gambar 4.7 Tampilan Awal Arduino IDE

4.4.1 Inisialisasi Sistem

Pengembangan perangkat lunak pada penelitian ini dilakukan menggunakan Arduino IDE sebagai software utama untuk menulis, melakukan Verify (Compile), serta meng-Upload program ke mikrokontroler ESP32 DevKit V1. Seluruh pustaka pendukung, seperti LiquidCrystal_I2C, BH1750_WE, Adafruit_INA219, I2CKeyPad, UniversalTelegramBot, ArduinoJson, dan ESP_Google_Sheet_Client, dipasang melalui Library Manager pada Arduino IDE sebelum sketch (.ino) dikompilasi. Pemilihan board ESP32 dilakukan pada menu Tools di Arduino IDE dengan memilih board ESP32 Dev Module sesuai dengan modul mikrokontroler yang digunakan pada alat.

```
#include <Arduino.h>
#include <LiquidCrystal_I2C.h>
#include <BH1750_WE.h>
#include <I2CKeyPad.h>
#include <Adafruit_INA219.h>
#include <WiFiClientSecure.h>
#include <UniversalTelegramBot.h>
#include <ArduinoJson.h>
#include <ESP_Google_Sheet_Client.h>
#include "time.h"

//set wifi
const char* ssid     = "Aira";
const char* password = "22112002";
int terhubung = LOW;

//bh1750
BH1750_WE myBH1750 = BH1750_WE(0x23);
float lightIntensity = 0;

//keypad
I2CKeyPad keyPad(0x21);
char keymap[19] = "123A456B789C*0#DNF";
char ch;

//LCD
LiquidCrystal_I2C lcd(0x27,20,4);

//INA219
Adafruit_INA219 ina219;
float busvoltage = 0;
float current_mA = 0;
float Daya = 0;
float tegangan_a = 0;
float tegangan_b = 0;
```

Gambar 4.8 Inisialisasi pustaka dan deklarasi perangkat sistem

```
  lcd.init();
  lcd.backlight();

  //BH1750
  if(!myBH1750.init()){
    Serial.println("Connection to the BH1750 failed");
    Serial.println("Check wiring and I2C address");
    lcd.clear();
    lcd.setCursor(0,0);
    lcd.print("BH1750 Error");
    while(1){}
  }
  //INA219
  if (! ina219.begin()) {
    Serial.println("Failed to find INA219 chip");
    lcd.clear();
    lcd.setCursor(0,0);
    lcd.print("INA219 Error");
    while (1) { delay(10); }
  }
  //keypad
  if (keyPad.begin() == false)
  {
    Serial.println("\nERROR: cannot communicate to keypad.\nPlease reboot.\n");
    lcd.clear();
    lcd.setCursor(0,0);
    lcd.print("KEYPAD Error");
    while(1);
  }
```

Gambar 4.9 Proses inisialisasi LCD, sensor BH1750, sensor INA219, dan keypad I2C pada fungsi setup()

```
127   //Wifi
128   Serial.print("Connecting to Wifi SSID ");
129     WiFi.mode(WIFI_STA);
130     WiFi.begin(ssid, password);
131     Serial.println("Starting...");
132
133   //TELEGRAM BOT
134   secured_client.setCACert(TELEGRAM_CERTIFICATE_ROOT);
135
136   //Gsheet
137   GSheet.printf("ESP Google Sheet Client v%s\n\n", ESP_GOOGLE_SHEET_CLIENT_VERSION);
138   GSheet.setPrerefreshSeconds(10 * 60);
139   GSheet.begin(CLIENT_EMAIL, PROJECT_ID, PRIVATE_KEY);
140   Serial.print("Gstate : ");
141   Serial.println(Gstate);
142   configTime(7 * 3600,0 , ntpServer);
```

Gambar 4.10 Konfigurasi koneksi WiFi, Telegram Bot, Google Spreadsheet dan sinkronisasi waktu menggunakan NTP

```
137   //Gsheet
138   GSheet.printf("ESP Google Sheet Client v%s\n\n", ESP_GOOGLE_SHEET_CLIENT_VERSION);
139   GSheet.setPrerefreshSeconds(10 * 60);
140   GSheet.begin(CLIENT_EMAIL, PROJECT_ID, PRIVATE_KEY);
141   Serial.print("Gstate : ");
142   Serial.println(Gstate);
143   configTime(7 * 3600,0 , ntpServer);
165 if (GSheet.ready()){
166   Serial.println("\nToken ready!");
167   FirebaseJson response;
168   Serial.println("\nUpdate spreadsheet values...");
169   Serial.println("------------------------------------------------------------");
170   FirebaseJson valueRange;
171   valueRange.add("majorDimension", "ROWS");
172   valueRange.set("values/[0]/[0]", "WAKTU"); // row 1/ column 1
173   valueRange.set("values/[0]/[1]", "Lux"); // row 2/ column 1
174   valueRange.set("values/[0]/[2]", "Tegangan"); // row 3/ column 1
175   valueRange.set("values/[0]/[3]", "Arus");
176   valueRange.set("values/[0]/[4]", "Power");
177   bool success = GSheet.values.append(&response /* returned response */, spreadsheetId /* spreadsheet Id to update */, "Sheet1!A1" /* range to update */, &valueRange /* data to update
178 }
```

Gambar 4.11 Inisialisasi Google Spreadsheet dan pembuatan header data pengukuran

4.4.2 Konfigurasi Sensor BH1750

Sensor BH1750 digunakan untuk mengukur intensitas cahaya matahari dalam satuan lux yang merepresentasikan radiasi matahari yang diterima oleh panel surya. Konfigurasi sensor dilakukan melalui objek BH1750_WE myBH1750 = BH1750_WE(0x23) yang menentukan alamat I2C sensor. Koneksi sensor kemudian diverifikasi pada tahap inisialisasi sistem.

Pembacaan data sensor dilakukan melalui fungsi BH1750(). Fungsi ini mengatur mode kerja sensor menggunakan setMode(OTH), memberikan waktu tunggu agar proses pengukuran selesai, kemudian membaca nilai intensitas cahaya dalam satuan lux melalui fungsi getLux(). Potongan program pembacaan data sensor BH1750 ditunjukkan pada Gambar 4.12.

```
427  void BH1750(){
428      //BH1750
429    myBH1750.setMode(OTH);
430    delay(140);
431    lightIntensity = myBH1750.getLux();
432
433  }
```

Gambar 4.12 Program pembacaan data sensor BH1750

Mode OTH (*One Time High Resolution*) dipilih agar sensor melakukan satu kali pengukuran dengan resolusi tinggi setiap kali fungsi dipanggil. Setelah mode pengukuran diaktifkan, program memberikan waktu tunggu selama 140 ms untuk memastikan proses konversi cahaya menjadi data digital telah selesai sebelum nilai lux dibaca oleh mikrokontroler.

Fungsi BH1750() dipanggil pada beberapa bagian program, seperti saat menu LCD menampilkan data intensitas cahaya, ketika sistem mengirim data ke Google Spreadsheet, dan saat pengguna meminta informasi melalui Telegram Bot. Dengan demikian, data lux yang diperoleh selalu merupakan hasil pengukuran terbaru sehingga dapat digunakan untuk memantau kondisi radiasi matahari yang diterima panel surya secara real-time.

4.4.3 Konfigurasi Sensor INA219

Sensor INA219 digunakan untuk mengukur tegangan dan arus pada keluaran panel surya. Sensor ini berkomunikasi dengan mikrokontroler melalui antarmuka I2C dan dikonfigurasi menggunakan pustaka Adafruit_INA219. Pembacaan data dilakukan melalui fungsi INA219() yang ditunjukkan pada Gambar 4.13.

```
542  void INA219(){
543    //INA219
544    tegangan_a = ina219.getBusVoltage_V();
545    tegangan_b = ina219.getShuntVoltage_mV();
546    current_mA = ina219.getCurrent_mA();
547    busvoltage = tegangan_a + (tegangan_b/1000);
548    Daya = 19.12*(current_mA/1000);
```

Gambar 4.13 Program pembacaan tegangan, arus, dan perhitungan daya menggunakan sensor INA219

Pada Gambar 4.13 merupakan program pembacaan tegangan, arus, dan perhitungan daya menggunakan sensor INA219. Program membaca nilai tegangan bus menggunakan fungsi getBusVoltage_V() dan tegangan shunt menggunakan fungsi getShuntVoltage_mV(). Tegangan shunt yang masih dalam satuan milivolt kemudian dikonversi ke volt dan dijumlahkan dengan tegangan bus untuk memperoleh nilai busvoltage. Nilai arus keluaran panel surya diperoleh melalui fungsi getCurrent_mA().

Perhitungan daya (Daya) pada program ini tidak menggunakan nilai busvoltage hasil pembacaan langsung dari sensor INA219, melainkan menggunakan nilai tegangan maksimum (Vmax) sebesar 19,12 V yang diambil dari spesifikasi *nameplate* panel surya, dikalikan dengan arus terukur (current_mA). Hal ini dilakukan karena resistor shunt (dummy load) yang digunakan bernilai kecil (10 Ω), sehingga menyebabkan panel beroperasi jauh dari titik daya maksimum (Maximum Power Point/MPP) dan tegangan yang terbaca oleh sensor INA219 menjadi sangat kecil, tidak merepresentasikan performa aktual panel pada kondisi kerja optimalnya. Dengan menggunakan tegangan maksimum dari datasheet panel sebagai referensi, nilai daya yang dihasilkan (Pout) menjadi lebih konsisten dan representatif terhadap kapasitas maksimum panel sesuai spesifikasi pabrikan.

4.4.4 Perhitungan Efisiensi Panel Surya

Pada pengujian ini dilakukan verifikasi terhadap proses perhitungan daya masukan (Pin), daya keluaran (Pout), dan efisiensi panel surya yang diproses secara otomatis oleh mikrokontroler ESP32. Data yang digunakan berasal dari hasil pembacaan sensor BH1750 sebagai pengukur intensitas cahaya serta sensor INA219 sebagai pengukur arus keluaran panel surya. Nilai tegangan yang digunakan dalam perhitungan Pout bukan berasal dari pembacaan real-time sensor INA219, melainkan menggunakan nilai tegangan daya maksimum (Vmp) tetap sebesar 19,12 V sesuai spesifikasi *nameplate* panel surya. Seluruh proses perhitungan dilakukan secara real-time setiap kali sensor melakukan pembacaan data, sehingga nilai yang ditampilkan selalu menyesuaikan kondisi panel surya saat pengujian berlangsung. Implementasi proses perhitungan tersebut ditunjukkan pada Gambar 4.14.

```
//Rumus Pin
float Wpin = lightIntensity * 0.0079 * 0.3618;
  Wp = 0;
  if (Wpin >= 50){
    Wp = 50;
  }
  else{
    Wp = Wpin;
  }

eff = Daya/Wp *100;
```

Gambar 4.14 Potongan kode perhitungan Pin, Pout, dan efisiensi

Berdasarkan Gambar 4.14, program terlebih dahulu menghitung nilai daya masukan (Pin) yang pada kode diberi nama variabel Wpin (penamaan dibedakan dari istilah "Pin" untuk menghindari ambiguitas dalam penulisan program). Nilai Wpin diperoleh dari hasil kali antara intensitas cahaya (lightIntensity) dalam satuan lux yang terbaca oleh sensor BH1750, faktor konversi intensitas cahaya (lux) menjadi radiasi matahari (W/m²) sebesar 0,0079, dan luas permukaan aktif panel surya sebesar $0{,}67\ \text{m} \times 0{,}54\ \text{m} = 0{,}3618\ \text{m}^2$, sehingga ketiganya dikalikan langsung dalam satu baris perhitungan untuk memperoleh nilai daya masukan dalam satuan Watt.

Selanjutnya dilakukan pembatasan nilai menggunakan percabangan if: apabila Wpin bernilai lebih besar atau sama dengan 50 Watt, maka nilai Wp (Pin yang digunakan dalam perhitungan efisiensi) ditetapkan sebesar 50 Watt sesuai kapasitas maksimum panel surya yang digunakan pada penelitian ini. Sebaliknya, apabila Wpin masih di bawah 50 Watt, maka nilai Wp mengikuti nilai Wpin tersebut.

Setelah nilai daya masukan (Wp) diperoleh, mikrokontroler menghitung efisiensi panel surya menggunakan persamaan (2.4), yang diimplementasikan pada baris program *eff = Daya / Wp * 100;,* dengan Daya merupakan nilai daya keluaran (Pout) hasil perhitungan dari sensor INA219. Nilai efisiensi tersebut kemudian

digunakan sebagai parameter utama untuk ditampilkan pada LCD, Telegram, Google Spreadsheet, dan Serial Monitor.

4.4.5 Perancangan Tampilan LCD I2C 20x4

LCD I2C 20×4 digunakan sebagai antarmuka lokal untuk menampilkan informasi kondisi panel surya secara langsung. Tampilan LCD dikendalikan melalui fungsi LCD() yang dipanggil secara berulang pada fungsi loop(). Pengaturan halaman tampilan dilakukan menggunakan struktur enum Tampilan yang terdiri atas LCD_HOME, LCD_BH1750, LCD_INA219, LCD_UPGS, GS_HASIL, LCD_TELE, LCD_KIRIM_TELE, LCD_EFFISIENSI, LCD_TOLAK_MANUAL, dan LCD_MANUAL_COOLDOWN. Dua enum baru ditambahkan untuk mengakomodasi tampilan peringatan ketika pengguna menekan tombol 3 di luar kondisi yang diizinkan, yaitu LCD_TOLAK_MANUAL yang ditampilkan apabila tombol 3 ditekan saat jam pengiriman otomatis masih aktif, dan LCD_MANUAL_COOLDOWN yang ditampilkan apabila tombol 3 ditekan sebelum cooldown 1 menit selesai. Pada kondisi awal, LCD menampilkan menu utama sebagai panduan navigasi pengguna. Program tampilan menu utama ditunjukkan pada Gambar 4.15.

```
//Untuk Tampilan LCD
enum Tampilan{LCD_HOME, LCD_UPGS, LCD_TELE,LCD_KIRIM_TELE, LCD_INA219, LCD_BH1750, GS_HASIL, LCD_EFFISIENSI, LCD_TOLAK_MANUAL, LCD_MANUAL_COOLDOWN};
```

Gambar 4.15 Program tampilan menu utama pada LCD I2C 20×4.

Pada kondisi awal, LCD menampilkan menu utama sebagai panduan navigasi pengguna. Melalui menu utama tersebut, pengguna dapat memilih informasi yang ingin ditampilkan, seperti data intensitas cahaya dari sensor BH1750, data tegangan dan arus dari sensor INA219, pengiriman data ke Google Spreadsheet, maupun pengiriman data ke Telegram.

Ketika pengguna memilih menu BH1750 atau INA219, sistem akan memanggil fungsi pembacaan sensor yang sesuai dan menampilkan hasil pengukuran pada LCD. Program tampilan data sensor pada LCD ditunjukkan pada Gambar 4.16.

```
314     case LCD_BH1750:
315       BH1750();
316       lcd.setCursor(0,0);
317       lcd.print("BH1750");
318       lcd.setCursor(0,1);
319       lcd.print("Cahaya : ");
320       lcd.print(lightIntensity);
321       lcd.print(" Lux   ");
322       Serial.print("Lux : ");
323       Serial.println(lightIntensity);
324       lcd.setCursor(0,3);
325       lcd.print("0 untuk kembali");
326       BackHome = true;
327     break;
328     case LCD_INA219:
329       INA219();
330       lcd.setCursor(0,0);
331       lcd.print("INA219");
332       lcd.setCursor(0,1);
333       lcd.print("V:");
334       lcd.print(busvoltage);
335       lcd.print("V");
336       lcd.setCursor(0,2);
337       lcd.print("I:");
338       lcd.print(current_mA/1000);
339       lcd.print("A ");
340       lcd.print("P:");
341       lcd.print(Daya);
342       lcd.print("W   ");
343       lcd.setCursor(0,3);
344       lcd.print("0 untuk kembali");
345       Serial.print("Daya : ");
346       Serial.println(Daya);
347       BackHome = true;
348     break;
```

Gambar 4.16 Program tampilan data sensor pada LCD

Ketika pengguna memilih menu BH1750 atau INA219, sistem akan memanggil fungsi pembacaan sensor yang sesuai dan menampilkan hasil pengukuran pada LCD. Nilai intensitas cahaya, tegangan, arus, dan daya ditampilkan secara langsung menggunakan fungsi lcd.setCursor() dan lcd.print(). Selain itu, program menerapkan kondisi if(tampilan != lastMenu) sehingga fungsi lcd.clear() hanya

dijalankan saat terjadi perpindahan menu. Cara ini digunakan untuk menghindari efek berkedip (*flicker*) yang dapat mengganggu kenyamanan pengguna saat membaca informasi pada LCD.

Selama proses pengiriman data ke Google Spreadsheet maupun Telegram, LCD juga digunakan untuk menampilkan informasi status proses, seperti "Uploading to SpreadSheet" dan "Data Terkirim!". Dengan demikian, LCD tidak hanya berfungsi sebagai media pemantauan data sensor, tetapi juga sebagai sarana pemberian informasi status sistem kepada pengguna.

4.4.6 Perancangan Navigasi Menu Keypad I2C

Navigasi menu pada sistem dilakukan menggunakan keypad I2C yang terhubung ke ESP32. Pembacaan tombol dilakukan melalui fungsi KEYPAD() yang memanfaatkan fungsi keyPad.getChar(). Program pembacaan tombol keypad ditunjukkan pada Gambar 4.17.

```
457  void KEYPAD(){
458      ch = keyPad.getChar();     //  note we want the translated char
459      Serial.println(ch);
460  }
```

Gambar 4.17 Program pembacaan input keypad I2C

Karakter tombol yang ditekan pengguna akan disimpan pada variabel ch dan digunakan sebagai dasar pengambilan keputusan pada sistem. Fungsi KEYPAD() dipanggil setiap kali fungsi LCD() dijalankan sehingga sistem dapat merespons penekanan tombol secara langsung.

Pengaturan perpindahan menu berdasarkan tombol keypad ditunjukkan pada Gambar 4.18.

```
243     else if(ch == 'D'){
244       tampilan = LCD_BH1750;
245     }
246     else if(ch == '#'){
247       tampilan = LCD_INA219;
248     }
249     // else if(ch == '*'){
250     //   tampilan = LCD_UPGS;
251     // }
252
253     if(tampilan != lastMenu){
254       lcd.clear();
255     }
256     switch (tampilan){
257       case LCD_HOME:
258         BackHome = false;
259         lcd.setCursor(0,0);
260         lcd.print("Monitoring Panel");
261         lcd.setCursor(0,1);
262         lcd.print("1.BH1750, 2.INA219");
263         lcd.setCursor(0,2);
264         lcd.print("3. Gsheet, 4. Tele");
265       break;
```

```
226   if (ch == 'C') {
227     if (saved_chat_id == "") {
228       // Belum ada chat_id → tampilkan pesan peringatan di LCD
229       lcd.clear();
230       lcd.setCursor(0, 0); lcd.print("Belum ada chat_id!");
231       lcd.setCursor(0, 1); lcd.print("Chat dulu via Tele");
232       lcd.setCursor(0, 3); lcd.print("3 untuk kembali");
233       tampilan = LCD_KIRIM_TELE;
234       lastMenu = tampilan;
235       return;
236     }
```

Gambar 4.18 Program navigasi menu menggunakan keypad I2C

Pada program tersebut, tombol 1 digunakan untuk menampilkan data sensor BH1750, tombol 2 untuk menampilkan data sensor INA219, tombol 4 untuk mengirim data ke Telegram Bot, tombol 5 untuk menampilkan nilai efisiensi, serta tombol 0 untuk kembali ke menu utama. Tombol 3 difungsikan sebagai tombol kirim manual ke Google Spreadsheet yang hanya dapat digunakan di luar jam pengiriman otomatis, yaitu pada rentang pukul 17.00–10.00 WIB. Apabila tombol 3 ditekan saat sistem berada dalam jam pengiriman otomatis (10.00–17.00), sistem akan menolak perintah dan menampilkan pesan peringatan pada LCD: 'Pengiriman Manual Dibatalkan! Msh jam operasional kirim otomatis aktif' sebagaimana ditunjukkan pada Gambar 4.19.

Gambar 4.19 Tampilan LCD saat Pengiriman Manual Ditolak karena Masih Jam Operasional

4.4.7 Konfigurasi dan Pengiriman Data ke Google Spreadsheet

Penyimpanan data historis dilakukan menggunakan layanan Google Spreadsheet melalui pustaka ESP_Google_Sheet_Client. Konfigurasi dilakukan menggunakan kredensial service account yang terdiri atas PROJECT_ID, CLIENT_EMAIL, dan PRIVATE_KEY, serta ID spreadsheet tujuan (spreadsheetId). Proses pengiriman data ditangani oleh fungsi UPGS() yang ditunjukkan pada Gambar 4.20.

```
bool jamKirimAktif = isJamKirim();
if(jamKirimAktif && (millis()-stimer) > duration){
  stimer = millis();
  UPGS();
  tampilan = LCD_UPGS;
}
else if(!jamKirimAktif && ch == '3'){
  if((millis()-stimer) > duration){
    // Sudah lebih dari 1 menit sejak upload manual terakhir, boleh kirim lagi
    stimer = millis();
    UPGS();
    tampilan = LCD_UPGS;
  }
  else{
    // Belum 1 menit sejak upload manual terakhir, tolak dulu
    tampilan = LCD_MANUAL_COOLDOWN;
    UpNotif = millis();
  }
}
else if(jamKirimAktif && ch == '3'){
  // User coba kirim manual padahal masih jam pengiriman otomatis
  tampilan = LCD_TOLAK_MANUAL;
  UpNotif = millis();
}
}
```

Gambar 4.20 Program pengiriman data ke Google Spreadsheet.

Sebelum data dikirim, sistem terlebih dahulu membaca ulang data sensor BH1750 dan INA219 agar data yang tersimpan merupakan data terbaru. Selanjutnya, nilai waktu hasil sinkronisasi NTP (waktuStr), intensitas cahaya, tegangan, arus, dan daya dimasukkan ke dalam objek FirebaseJson sebagai format data yang akan dikirim ke spreadsheet.

4.4.8 Konfigurasi Telegram Bot dan Command Handler (/status, /daya, /radiasi, /effisiensi)

Telegram Bot digunakan sebagai media pemantauan jarak jauh yang memungkinkan pengguna memperoleh informasi kondisi panel surya melalui aplikasi Telegram. Konfigurasi bot dilakukan menggunakan pustaka UniversalTelegramBot dengan token bot (BOT_TOKEN) dan komunikasi aman melalui WiFiClientSecure.

Pengecekan pesan masuk dilakukan secara berkala pada fungsi loop() menggunakan bot.getUpdates(). Program pengecekan pesan Telegram ditunjukkan pada Gambar 4.21.

```
if (text == "/Effisiensi")
{
  String pesan = "";
  pesan += "  *DATA EFISIENSI*  \n";
  pesan += "──────────────────────\n";
  pesan += " Pin        : `"; pesan += Pin; pesan += "`\n";
  pesan += " Pout       : `"; pesan += daya; pesan += "`\n";
  pesan += " Efisiensi : `"; pesan += EFF; pesan += "`\n";
  pesan += "──────────────────────\n\n";
  bot.sendMessage(chat_id, pesan, "Markdown");
}
}
}
```

Gambar 4.21 Program pengecekan pesan Telegram.

Pesan yang diterima kemudian diproses oleh fungsi handleNewMessages(). Pada fungsi ini, nilai chat_id pengirim disimpan ke dalam variabel saved_chat_id sehingga dapat digunakan kembali untuk pengiriman data melalui keypad. Program penanganan command Telegram ditunjukkan pada Gambar 4.22.

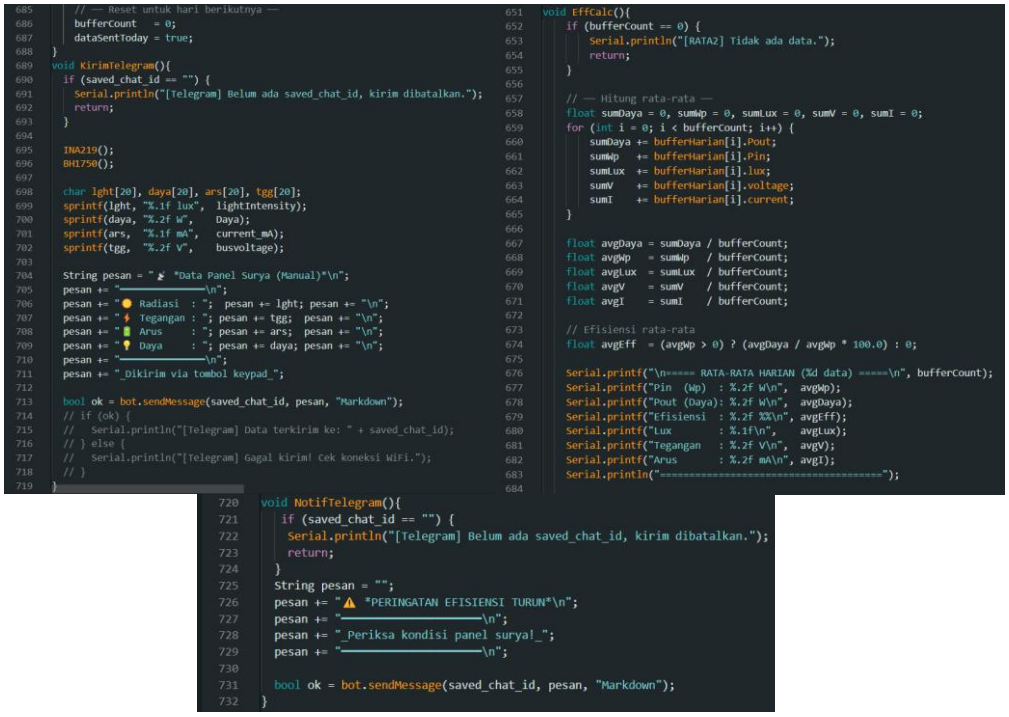
```
354  void handleNewMessages(int numNewMessages){
355    INA219();
356    BH1750();
357    //buat string
358    char lght[20];
359    sprintf(lght, "%.1f lux", lightIntensity);
360    char daya[20];
361    sprintf(daya, "%.1f W", Daya);
362    char ars[20];
363    sprintf(ars, "%.1f mA", current_mA);
364    char tgg[20];
365    sprintf(tgg, "%.1f V", busvoltage);
366
367    Serial.print("handleNewMessages ");
368    Serial.println(numNewMessages);
369
370    for (int i = 0; i < numNewMessages; i++)
371    {
372      String chat_id = bot.messages[i].chat_id;
373      saved_chat_id = chat_id;
374      String text = bot.messages[i].text;
375
376      String from_name = bot.messages[i].from_name;
377      if (from_name == "")
378        from_name = "Guest";
379
380      if (text == "/start")
381      {
382        String welcome = "Hola Amigos!, ini test bot \n";
383        welcome += "untuk monitoring effisiensi solar panel.\n\n";
384        welcome += "/status : untuk ambil data daya dan radiasi\n";
385        welcome += "/daya : untuk ambil data daya\n";
386        welcome += "/radiasi : untuk ambil data radiasi\n";
387        bot.sendMessage(chat_id, welcome, "Markdown");
388      }
389      if (text == "/daya") //kirim data daya ke tele
390      {
391        bot.sendMessage(chat_id, "Daya Panel Sekarang :", "");
392        bot.sendMessage(chat_id, daya, "");
393        bot.sendMessage(chat_id, "Arus Panel Sekarang :", "");
394        bot.sendMessage(chat_id, ars, "");
395        bot.sendMessage(chat_id, "Tegangan Panel Sekarang :", "");
396        bot.sendMessage(chat_id, tgg, "");
397      }
398
399      if (text == "/radiasi") //kirim data radiasi ke tele
400      {
401
402        bot.sendMessage(chat_id, "Radiasi Matahari Sekarang :", "");
403        bot.sendMessage(chat_id, lght, "");
404      }
405
406      if (text == "/status") //kirim data radiasi dan daya ke tele
407      {
408        bot.sendMessage(chat_id, "Radiasi Matahari Sekarang :", "");
409        bot.sendMessage(chat_id, lght, "");
410        bot.sendMessage(chat_id, "Daya Panel Sekarang :", "");
411        bot.sendMessage(chat_id, daya, "");
412        bot.sendMessage(chat_id, "Arus Panel Sekarang :", "");
413        bot.sendMessage(chat_id, ars, "");
414        bot.sendMessage(chat_id, "Tegangan Panel Sekarang :", "");
415        bot.sendMessage(chat_id, tgg, "");
416      }
417    }
418  }
```

Gambar 4.22 Program penanganan command Telegram Bot.

Berdasarkan program yang dibuat, tersedia lima command utama, yaitu /start, /daya, /radiasi, /status, dan /Effisiensi. Command /start digunakan untuk menampilkan pesan sambutan dan daftar command yang tersedia. Command /daya digunakan untuk menampilkan data tegangan, arus, Pin, dan Pout panel surya.

4.4.9 Logika Notifikasi Otomatis Penurunan Efisiensi

Pengujian ini dilakukan untuk memastikan sistem mampu menghitung efisiensi rata-rata harian berdasarkan data yang tersimpan pada buffer, kemudian mengirimkan hasil monitoring maupun notifikasi melalui aplikasi Telegram. Implementasi proses tersebut ditunjukkan pada Gambar 4.23.



```
    // — Reset untuk hari berikutnya —
    bufferCount   = 0;
    dataSentToday = true;
}
void KirimTelegram(){
  if (saved_chat_id == "") {
    Serial.println("[Telegram] Belum ada saved_chat_id, kirim dibatalkan.");
    return;
  }

  INA219();
  BH1750();

  char lght[20], daya[20], ars[20], tgg[20];
  sprintf(lght, "%.1f lux",  lightIntensity);
  sprintf(daya, "%.2f W",    Daya);
  sprintf(ars,  "%.1f mA",   current_mA);
  sprintf(tgg,  "%.2f V",    busvoltage);

  String pesan = "⚡ *Data Panel Surya (Manual)*\n";
  pesan += "──────────────\n";
  pesan += "🟡 Radiasi  : "; pesan += lght; pesan += "\n";
  pesan += "⚡ Tegangan : "; pesan += tgg;  pesan += "\n";
  pesan += "🔋 Arus     : "; pesan += ars;  pesan += "\n";
  pesan += "💡 Daya     : "; pesan += daya; pesan += "\n";
  pesan += "──────────────\n";
  pesan += "_Dikirim via tombol keypad_";

  bool ok = bot.sendMessage(saved_chat_id, pesan, "Markdown");
  // if (ok) {
  //   Serial.println("[Telegram] Data terkirim ke: " + saved_chat_id);
  // } else {
  //   Serial.println("[Telegram] Gagal kirim! Cek koneksi WiFi.");
  // }
}
```

```
void EffCalc(){
  if (bufferCount == 0) {
    Serial.println("[RATA2] Tidak ada data.");
    return;
  }

  // — Hitung rata-rata —
  float sumDaya = 0, sumWp = 0, sumLux = 0, sumV = 0, sumI = 0;
  for (int i = 0; i < bufferCount; i++) {
    sumDaya += bufferHarian[i].Pout;
    sumWp   += bufferHarian[i].Pin;
    sumLux  += bufferHarian[i].lux;
    sumV    += bufferHarian[i].voltage;
    sumI    += bufferHarian[i].current;
  }

  float avgDaya = sumDaya / bufferCount;
  float avgWp   = sumWp   / bufferCount;
  float avgLux  = sumLux  / bufferCount;
  float avgV    = sumV    / bufferCount;
  float avgI    = sumI    / bufferCount;

  // Efisiensi rata-rata
  float avgEff  = (avgWp > 0) ? (avgDaya / avgWp * 100.0) : 0;

  Serial.printf("\n===== RATA-RATA HARIAN (%d data) =====\n", bufferCount);
  Serial.printf("Pin  (Wp)  : %.2f W\n",  avgWp);
  Serial.printf("Pout (Daya): %.2f W\n",  avgDaya);
  Serial.printf("Efisiensi  : %.2f %%\n", avgEff);
  Serial.printf("Lux        : %.1f\n",    avgLux);
  Serial.printf("Tegangan   : %.2f V\n",  avgV);
  Serial.printf("Arus       : %.2f mA\n", avgI);
  Serial.println("=====================================");
}
```

```
void NotifTelegram(){
  if (saved_chat_id == "") {
    Serial.println("[Telegram] Belum ada saved_chat_id, kirim dibatalkan.");
    return;
  }
  String pesan = "";
  pesan += "⚠️ *PERINGATAN EFISIENSI TURUN*\n";
  pesan += "──────────────────\n";
  pesan += "_Periksa kondisi panel surya!_";
  pesan += "──────────────────\n";

  bool ok = bot.sendMessage(saved_chat_id, pesan, "Markdown");
}
```

Gambar 4.23 Implementasi Logika Notifikasi Otomatis Penurunan Efisiensi

Berdasarkan Gambar 4.22, fungsi EffCalc() diawali dengan pengecekan jumlah data yang tersimpan pada buffer menggunakan variabel bufferCount. Apabila tidak terdapat data (bufferCount == 0), proses perhitungan dihentikan dan sistem menampilkan pesan "Tidak ada data" pada Serial Monitor. Selanjutnya, sistem melakukan proses penjumlahan seluruh data hasil monitoring yang tersimpan pada array bufferHarian, meliputi daya keluaran (*Pout*), daya masukan (*Pin*), intensitas cahaya (lux), tegangan, dan arus. Nilai total masing-masing parameter kemudian dibagi dengan jumlah data (bufferCount) sehingga diperoleh nilai rata-rata harian. Setelah nilai rata-rata diperoleh, sistem menghitung efisiensi rata-rata menggunakan persamaan:

$$\eta = \frac{P_{out}}{P_{in}} \times 100\% \quad (2.4)$$

Perhitungan tersebut diimplementasikan pada program melalui variabel avgEff, yaitu dengan membandingkan rata-rata daya keluaran (avgDaya) terhadap rata-rata daya masukan (avgWp).