

BAB IV

PEMBUATAN ALAT

4.1 Pembuatan *Prototype*

Pada tahap ini dilakukan proses perakitan seluruh komponen perangkat keras serta pemrograman mikrokontroler ESP32 sehingga membentuk sebuah sistem yang dapat bekerja sesuai dengan tujuan penelitian. Alat yang dirancang berfungsi sebagai sistem penstabil tegangan menggunakan *buck-boost converter* pada beban *sand battery* berbasis panel surya, serta dilengkapi dengan sistem monitoring tegangan dan arus secara *real-time* menggunakan ESP32, LCD 16×2 I2C, dan Google Spreadsheet.

Proses pembuatan alat diawali dengan pemasangan komponen sesuai wiring diagram, meliputi panel surya sebagai sumber tegangan, *buck-boost converter*, rangkaian *voltage divider* sebagai sensor tegangan, sensor arus ACS712, modul MP1584 sebagai catu daya ESP32, LCD 16×2 I2C, dan *heater* sebagai beban. Setelah seluruh rangkaian selesai dirakit, dilakukan pemrograman ESP32 menggunakan Arduino IDE untuk membaca data sensor, mengolah hasil pengukuran, menampilkan informasi pada LCD, serta mengirimkan data ke Google Spreadsheet melalui jaringan Wi-Fi. Sistem mampu melakukan memonitor kondisi tegangan dan arus secara terus-menerus selama alat beroperasi.

1. Perancangan Perangkat Keras (*Hardware*)

Perancangan *hardware* merupakan tahapan pembuatan perangkat keras yang bertujuan untuk merealisasikan sistem sesuai dengan rancangan yang telah dibuat pada Bab III. Pada tahap ini dilakukan pemilihan, penyusunan, dan penghubungan setiap komponen elektronik sehingga membentuk suatu sistem yang dapat berfungsi dengan baik. Perancangan *hardware* dilakukan berdasarkan kebutuhan alat, mulai dari sumber daya, sistem pengukuran, sistem *monitoring*, hingga beban yang digunakan pada penelitian.

Komponen utama yang digunakan dalam penelitian ini terdiri atas panel surya sebagai sumber tegangan DC, *buck-boost converter* sebagai penstabil tegangan, rangkaian *voltage divider* sebagai sensor pembaca tegangan, sensor

arus ACS712 sebagai pembaca arus beban, ESP32 sebagai pusat pengendali dan pengolah data, modul MP1584 sebagai penurun tegangan catu daya, LCD 16×2 I2C sebagai media tampilan, serta *heater* sebagai beban pada sistem *sand battery*. Seluruh komponen dirangkai sesuai dengan *wiring diagram* sehingga dapat bekerja sebagai satu sistem yang terintegrasi. Perancangan *hardware* yang baik akan mempengaruhi kinerja sistem dalam melakukan pembacaan, pengolahan, dan memonitor data secara akurat.

2. Perancangan Perangkat Lunak (*Software*)

Perancangan *software* merupakan tahapan pembuatan perangkat lunak yang bertujuan untuk mengatur cara kerja seluruh perangkat keras agar dapat beroperasi sesuai dengan fungsi yang diinginkan. Perangkat lunak pada penelitian ini dibuat menggunakan Arduino IDE dengan bahasa pemrograman C/C++, kemudian diunggah ke mikrokontroler ESP32 sebagai pusat pengendali sistem.

Program yang dibuat berfungsi untuk menginisialisasi seluruh perangkat yang digunakan, membaca data dari sensor tegangan dan sensor arus, mengolah hasil pembacaan sensor menjadi nilai tegangan dan arus yang sebenarnya, serta menampilkan hasil pengukuran pada LCD 16×2 I2C. ESP32 juga menghubungkan sistem dengan jaringan Wi-Fi untuk mengirimkan data ke Google Spreadsheet sehingga proses *monitoring* dapat dilakukan secara *real-time*. Seluruh proses pembacaan, pengolahan, penampilan, dan pengiriman data dilakukan secara berulang (*loop*) selama sistem beroperasi sehingga kondisi tegangan dan arus pada beban dapat dipantau secara terus-menerus.

4.2 Alat dan Bahan

Diperlukan beberapa alat dan bahan yang berfungsi untuk mendukung proses perakitan serta pengujian sistem. Alat yang digunakan berfungsi sebagai media dalam proses perakitan dan pengukuran, sedangkan bahan merupakan komponen utama yang digunakan untuk membangun sistem penstabil dan *monitoring* tegangan pada beban *sand battery* berbasis panel surya. Pemilihan alat

dan bahan disesuaikan dengan kebutuhan perancangan agar sistem dapat bekerja secara optimal sesuai dengan tujuan penelitian. Daftar alat yang digunakan dalam penelitian ini disajikan pada tabel berikut:

Tabel 4- 1 Tabel Alat

No.	Alat	Jumlah
1	Multimeter	1
2	Solder	1
3	Attractor	1
4	Tang Kombinasi	1
5	Obeng (+)	1
6	Obeng (-)	1
7	Bor	1
8	Mata Bor	1
9	Mesin Cutting Laser	1
10	Larutan FeCl	1
11	Gerinda	1

Adapun daftar bahan yang digunakan dalam penelitian ini disajikan pada tabel berikut:

Tabel 4- 2 Tabel Bahan

No.	Komponen	Jumlah
1	ESP32 DevKit V1	1
2	Modul MP1584	1
3	Adaptor 12 Volt	1
4	LCD 16x2 I2C	1
5	ACS712	1
6	Resistor 150K Ω	2
7	Resistor 10K Ω	3
8	Resistor 20K Ω	1
9	<i>Buck-Boost Converter</i>	1
10	Panel Surya	1
11	<i>Heater</i>	1

No.	Komponen	Jumlah
12	JST 4 Pin	1
13	Terminal Blok 2 Pin	3
14	PCB Single Layer	1
15	Kabel	1
16	Baut M3	8
17	Mur M3	8
18	Spacer	4
19	Duradus Box Panel	1

4.3 Pembuatan Perangkat Keras (*Hardware*)

Pembuatan perangkat keras (*hardware*) merupakan tahapan dalam proses pembuatan alat yang dilakukan dengan merealisasikan rancangan sistem menjadi sebuah perangkat fisik yang dapat beroperasi sesuai dengan fungsinya. Tahap ini meliputi proses pemilihan komponen, perakitan, pengkabelan, dan pemasangan setiap komponen berdasarkan desain yang telah dirancang sebelumnya. Tujuan dari pembuatan perangkat keras adalah membentuk suatu sistem yang terintegrasi sehingga seluruh komponen dapat saling berkomunikasi dan bekerja sesuai dengan fungsi masing-masing.

Pembuatan perangkat keras dilakukan dengan menghubungkan seluruh komponen elektronik sesuai dengan *wiring diagram* yang telah dirancang. Setiap komponen dipasang pada posisi yang tepat dan dihubungkan menggunakan kabel penghantar agar terbentuk rangkaian yang utuh. Setelah proses perakitan selesai, dilakukan pemeriksaan terhadap seluruh sambungan untuk memastikan tidak terjadi kesalahan pemasangan serta memastikan setiap komponen dapat bekerja dengan baik sebelum dilakukan proses pemrograman dan pengujian sistem.

4.3.1 Pembuatan PCB Kontroler

Pada pembuatan PCB kontroler terdapat 6 bagian yaitu, perancangan komponen, pembuatan *schematic* dan *layout* PCB pada aplikasi EasyEDA,

pembuatan jalur pada PCB, proses etching, pelubangan PCB, penyolderan komponen, dan pengetesan PCB.

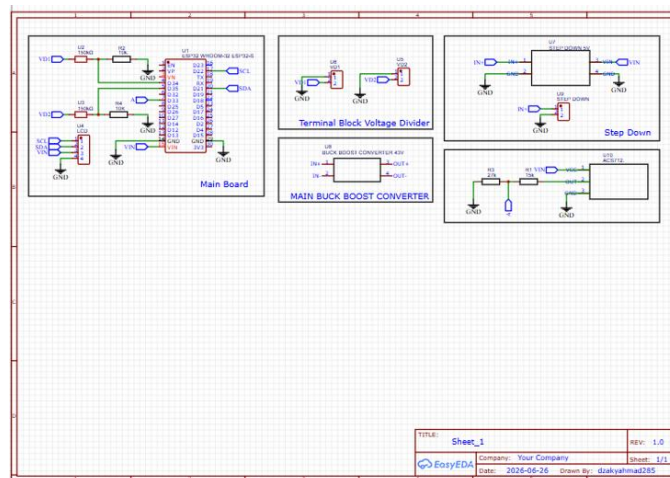
1. Perancangan Komponen

Perancangan komponen merupakan tahapan dalam proses pembuatan alat yang dilakukan dengan menentukan dan memilih komponen yang akan digunakan sesuai dengan kebutuhan sistem. Pemilihan komponen dilakukan berdasarkan fungsi, spesifikasi, serta kemampuan masing-masing komponen agar dapat bekerja secara optimal dan saling terintegrasi dalam membentuk sistem yang dirancang. Pada tahap ini juga dilakukan penyesuaian antara karakteristik komponen dengan kebutuhan alat sehingga diperoleh kinerja sistem yang baik dan sesuai dengan tujuan penelitian.

Komponen yang digunakan pada penelitian ini meliputi panel surya sebagai sumber energi listrik DC, buck-boost converter sebagai penstabil tegangan, ESP32 sebagai mikrokontroler, rangkaian resistor voltage divider sebagai sensor pembaca tegangan, sensor arus ACS712 sebagai pembaca arus pada beban, modul MP1584 sebagai catu daya, LCD 16×2 I2C sebagai media penampil hasil pengukuran, heater sebagai beban sand battery, serta beberapa komponen pendukung seperti resistor, kabel jumper, terminal block, dan PCB. Seluruh komponen tersebut dipilih karena memiliki spesifikasi yang sesuai dengan kebutuhan sistem dan mampu mendukung proses monitoring tegangan dan arus secara real-time.

2. Pembuatan *Schematic* dan *Layout* PCB pada Aplikasi EasyEDA

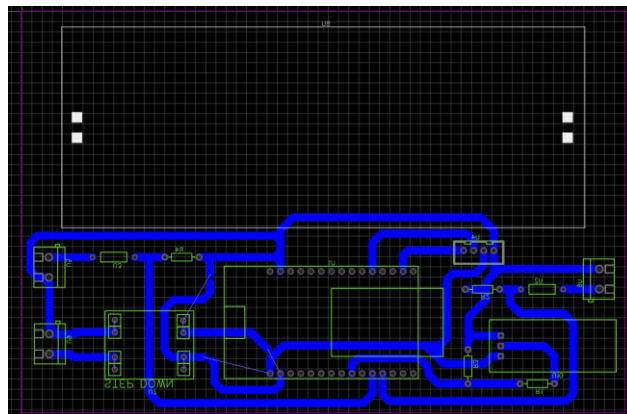
Proses pembuatan *schematic* dan *layout* PCB dilakukan menggunakan aplikasi EasyEDA. Perancangan *schematic* digunakan untuk menggambarkan hubungan listrik antar komponen sesuai dengan wiring diagram yang telah dirancang, sedangkan *layout* PCB dibuat untuk menentukan posisi komponen dan jalur konduktor pada papan PCB. Dengan menggunakan EasyEDA, proses perancangan menjadi lebih sistematis, meminimalkan kesalahan pengkabelan, serta mempermudah proses pembuatan dan perakitan perangkat keras.



Gambar 4. 1 Schematic Diagram

3. Pembuatan Jalur Pada PCB

Pada tahap ini dilakukan pencetakan jalur (*layout*) pada papan PCB.



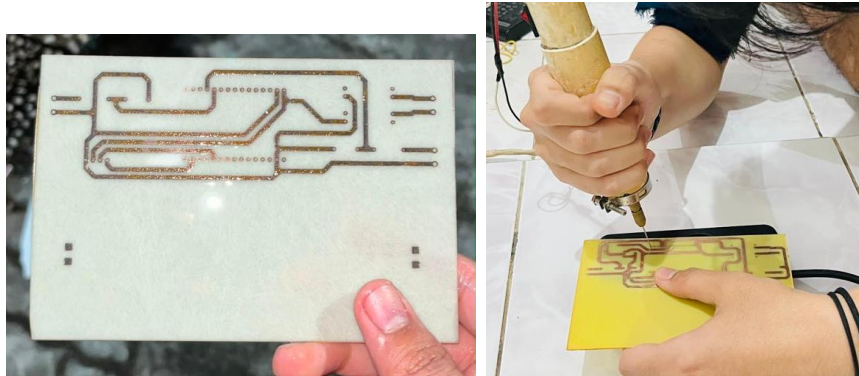
Gambar 4. 2 Desain Board

4. Proses *Etching* dan Pelubangan PCB

Proses *etching* bertujuan untuk menghilangkan lapisan tembaga yang tidak terlindungi oleh pola jalur rangkaian sehingga hanya menyisakan jalur konduktor sesuai dengan desain yang telah dibuat pada aplikasi EasyEDA. Proses *etching* dilakukan dengan merendam papan PCB ke dalam larutan *Ferric Chloride* (FeCl_3) hingga seluruh lapisan tembaga yang tidak diperlukan larut dan membentuk jalur rangkaian sesuai dengan desain.

Setelah proses *etching* selesai, papan PCB dibersihkan menggunakan air mengalir untuk menghilangkan sisa larutan kimia, kemudian dikeringkan. Selanjutnya dilakukan proses pelubangan PCB menggunakan

bor mini pada setiap titik pad komponen sesuai dengan ukuran kaki komponen yang akan dipasang. Proses pelubangan dilakukan dengan hati-hati agar posisi lubang tetap presisi dan tidak merusak jalur PCB yang telah terbentuk. Setelah seluruh lubang selesai dibuat, PCB siap digunakan untuk proses pemasangan dan penyolderan komponen elektronik.

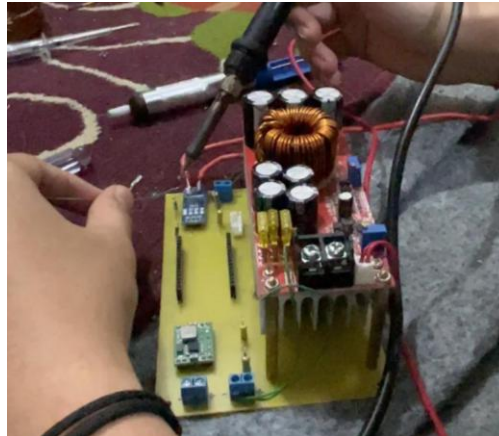


Gambar 4. 3 Proses *Etching* dan Pelubangan PCBC

5. Penyolderan Komponen

Pada tahap ini, seluruh komponen elektronik dipasang pada posisi yang telah ditentukan sesuai dengan layout PCB yang dirancang menggunakan EasyEDA. Penyolderan dilakukan untuk menghubungkan kaki komponen dengan jalur tembaga pada PCB sehingga terbentuk hubungan listrik yang baik dan setiap komponen dapat berfungsi sesuai dengan rancangan.

Proses penyolderan diawali dengan memasang komponen yang berukuran kecil, seperti *resistor*, dilanjutkan dengan pemasangan header pin, terminal *block*, konektor, dan komponen lainnya. Setiap sambungan dilakukan penyolderan menggunakan timah secukupnya agar menghasilkan sambungan yang kuat, rapi, dan tidak menyebabkan hubungan singkat (*short circuit*). Setelah seluruh komponen selesai dilakukan penyolderan, dilakukan pemeriksaan kembali terhadap kualitas sambungan dan posisi setiap komponen untuk memastikan tidak terdapat kesalahan pemasangan maupun jalur yang terputus. PCB yang telah selesai dirakit selanjutnya siap digunakan untuk proses pengujian rangkaian dan integrasi dengan komponen lainnya dalam sistem.



Gambar 4. 4 Penyolderan PCB

6. Pengetesan PCB

Setelah seluruh komponen selesai dipasang dan disolder pada PCB, tahapan berikutnya adalah proses pengujian PCB. Pengujian ini bertujuan untuk memastikan bahwa setiap jalur pada PCB telah terhubung dengan baik serta seluruh komponen berfungsi sesuai dengan rancangan. Pengujian juga dilakukan untuk mendeteksi adanya kesalahan pemasangan komponen, jalur yang terputus, maupun hubungan singkat (*short circuit*) yang dapat mengganggu kinerja sistem.

Proses pengujian diawali dengan melakukan pemeriksaan visual terhadap hasil penyolderan untuk memastikan tidak terdapat sambungan yang kurang baik atau timah yang menghubungkan dua jalur secara tidak sengaja. Selanjutnya dilakukan pengujian menggunakan multimeter pada beberapa jalur PCB untuk memeriksa kontinuitas dan memastikan tidak terjadi hubungan singkat. Setelah hasil pengujian menunjukkan bahwa rangkaian dalam kondisi baik, PCB diberikan catu daya dan dilakukan pengujian fungsi setiap komponen, seperti ESP32, sensor tegangan, sensor arus ACS712, LCD 16×2 I2C, serta modul MP1584. Apabila seluruh komponen bekerja sesuai dengan fungsinya, maka PCB dinyatakan layak untuk digunakan pada proses pengujian keseluruhan sistem.

4.3.2 Pembuatan Kontroler Utama

1. Penempatan Komponen Via *Software* 3D

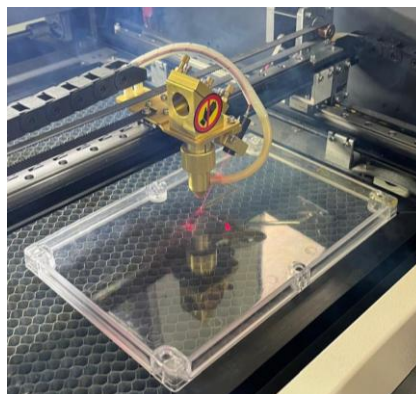
Penempatan komponen melalui *software* 3D merupakan tahap perancangan yang dilakukan untuk menentukan posisi dan tata letak setiap komponen pada alat secara virtual sebelum proses perakitan. Tahap ini bertujuan untuk memastikan seluruh komponen dapat ditempatkan secara tepat, rapi, dan sesuai dengan kebutuhan sistem, serta memudahkan proses perakitan dan mengurangi kemungkinan kesalahan dalam pemasangan komponen.



Gambar 4. 5 Desain Penempatan Komponen 3D

2. Melubangi *Box Controller*

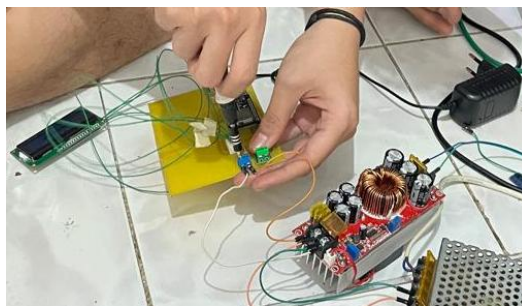
Proses pelubangan *box controller* merupakan tahap pembuatan lubang pada bagian tertentu dari *box* yang disesuaikan dengan ukuran dan posisi komponen yang akan dipasang. Proses ini dilakukan berdasarkan hasil perancangan dan penempatan komponen pada desain 3D agar setiap komponen dapat terpasang dengan tepat serta menghasilkan susunan alat yang rapi dan sesuai dengan kebutuhan sistem.



Gambar 4. 6 Proses Pelubangan PCB

3. Perakitan Komponen

Proses perakitan komponen merupakan tahap penyusunan dan pemasangan seluruh komponen yang telah dipersiapkan sesuai dengan desain dan tata letak yang telah ditentukan. Pada tahap ini, setiap komponen dipasang pada *box controller* dan dihubungkan sesuai dengan rangkaian yang telah dirancang. Proses perakitan dilakukan secara sistematis dan teliti untuk memastikan setiap komponen terpasang dengan baik serta dapat berfungsi sesuai dengan tujuan perancangan alat.



Gambar 4. 7 Proses Perakitan Komponen

4.4 Pembuatan Perangkat Lunak (*Software*)

Pembuatan perangkat lunak (*software*) pada alat ini dilakukan dengan menggunakan mikrokontroler ESP32 sebagai pusat pengendali utama sistem. Program dirancang untuk mengatur proses pembacaan tegangan *input* dan tegangan *output* pada *buck-boost converter*, serta pembacaan arus menggunakan sensor ACS712. Data hasil pengukuran kemudian ditampilkan secara langsung melalui LCD 16×2 I2C sebagai media monitoring lokal. Selain itu, sistem dilengkapi dengan koneksi Wi-Fi dan sinkronisasi waktu menggunakan NTP untuk mendukung proses pengiriman data hasil *monitoring* ke Google Spreadsheet secara berkala. Seluruh proses tersebut dikendalikan melalui program utama yang mencakup tahap inisialisasi perangkat, pembacaan sensor, pengolahan data, penampilan hasil pengukuran, serta pengiriman data secara otomatis.

4.4.1 Konfigurasi Sensor Arus ACS712

1. *Library* dan Konfigurasi Awal ACS712

Sensor arus ACS712 dikonfigurasi menggunakan library ACS712.h dan dihubungkan dengan pin ADC GPIO 33 pada ESP32. Tegangan referensi ADC ditetapkan sebesar 3,3 V. Objek sensor kemudian diinisialisasi melalui ACS712 ACS() dengan resolusi ADC sebesar 4095. Selain itu, digunakan *currentCalFactor* sebagai faktor koreksi hasil pembacaan serta *currentDeadband* sebesar 0,15 A untuk mengurangi pengaruh *noise* pada kondisi arus rendah.

```

6  #include <ACS712.h>
7
8  #define Vin 35
9  #define Vout 34
10 #define ACS_Pin 33
11
12 #define ADC_VREF 3.3
13 #define DIV_RTOP 10000.0
14 #define DIV_RBOTTOM 20000.0
15 #define ACS_VOLTS_EFF (ADC_VREF * (DIV_RTOP + DIV_RBOTTOM) / DIV_RBOTTOM)
16 ACS712 ACS(ACS_Pin, ACS_VOLTS_EFF, 4095, 100);
17
18 // Faktor koreksi hasil akhir: set 1.00 dulu, lalu bandingkan
19 // dengan tang ampere dan isi = (I_tang / I_terbaca)
20 float currentCalFactor = 1.00;
21 // Ambang buang noise (Ampere). Di bawah nilai ini dianggap 0 A
22 const float currentDeadband = 0.15;

```

Gambar 4. 8 *Library* dan Konfigurasi Awal ACS712

2. Variabel Pembacaan Arus

Variabel mA digunakan untuk menyimpan hasil pembacaan arus. CurrentSamples ditetapkan sebanyak 200 sampel. Penggunaan beberapa sampel bertujuan untuk memperoleh nilai pembacaan yang lebih stabil sebelum nilai arus digunakan oleh sistem.

```

165 //Read avg Current
166 float mA = 0;
167 int Current_Interval = 1;
168 unsigned long Cltm = 0;
169 long CurrentTotal = 0;
170 int CurrentSampleCount = 0;
171 const int CurrentSamples = 200;

```

Gambar 4. 9 Variabel Pembacaan Arus

3. Konfigurasi ADC dan Kalibrasi Titik Tengah ACS712

Konfigurasi ADC ESP32 dilakukan dengan resolusi 12 bit sehingga menghasilkan rentang nilai ADC 0–4095. Pin yang terhubung dengan

ACS712 kemudian diberikan pengaturan atenuasi ADC_11db. Sebelum proses pembacaan arus dilakukan, fungsi ACS.autoMidPoint() digunakan untuk melakukan kalibrasi titik tengah sensor pada kondisi tanpa arus. Nilai titik tengah hasil kalibrasi selanjutnya dapat dipantau melalui Serial Monitor.

```

384 // --- Setup ADC ESP32 untuk ACS712 ---
385 analogReadResolution(12); // 0..4095
386 analogSetPinAttenuation(ACS_Pin, ADC_11db); // range input ~0..3.3V
387
388 // --- Kalibrasi titik tengah (WAJIB tanpa arus / beban dilepas) ---
389 delay(500);
390 ACS.autoMidPoint();
391 Serial.print("ACS712 midpoint ADC: ");
392 Serial.println(ACS.getMidPoint());

```

Gambar 4. 10 Kalibrasi ACS712

4.4.2 Konfigurasi Pembaca Tegangan *Input Buck-Boost*

1. Konfigurasi Pembagi Tegangan dan Kalibrasi Input

Pada bagian ini dilakukan konfigurasi pembagi tegangan dan kalibrasi pembacaan tegangan input *buck-boost*. Nilai VDIV_RTOP ditetapkan sebesar 150 k Ω dan VDIV_RBOTTOM sebesar 10 k Ω . Kedua nilai tersebut digunakan untuk menentukan rasio pembagi tegangan melalui variabel VDIV_RATIO. Pembagi tegangan berfungsi untuk menurunkan tegangan input sebelum masuk ke pin ADC ESP32 sehingga nilai tegangan yang diterima masih berada pada rentang pembacaan yang aman.

Selanjutnya, dilakukan kalibrasi dua titik dengan menggunakan pasangan nilai VIN_X1, VIN_Y1, VIN_X2, dan VIN_Y2. Nilai VIN_X1 dan VIN_X2 merupakan hasil pembacaan tegangan oleh sistem, sedangkan VIN_Y1 dan VIN_Y2 merupakan nilai acuan hasil pengukuran. Data tersebut digunakan pada fungsi kalibrasi2titik() untuk membentuk persamaan linear ($y = mx + b$), sehingga hasil pembacaan tegangan dapat dikoreksi agar lebih mendekati nilai aktual. Selain itu, apabila hasil kalibrasi berada di bawah 0,15 V, nilai tersebut dianggap sebagai 0 V untuk mengurangi pengaruh *offset* atau *noise* pembacaan.

```

122 // ===== PEMBACAAN TEGANGAN =====
123 #define VDIV_RTOP 15000.0
124 #define VDIV_RBOTTOM 10000.0
125 #define VDIV_RATIO (VDIV_RBOTTOM / (VDIV_RTOP + VDIV_RBOTTOM))
126
127 // ---- Kalibrasi 2 titik (linear): y = m*x + b ----
128 #define VIN_X1 12.00
129 #define VIN_Y1 12.24
130 #define VIN_X2 22.79
131 #define VIN_Y2 23.97
132 #define VOUT_X1 2.27
133 #define VOUT_Y1 0.00
134 #define VOUT_X2 41.60
135 #define VOUT_Y2 42.60
136
137 double kalibrasi2titik(double x, double x1, double y1, double x2, double y2) {
138     double m = (y2 - y1) / (x2 - x1);
139     double b = y1 - m * x1;
140     double y = m * x + b;
141     if (y < 0.15) y = 0; // sisa offset kecil dianggap 0 V
142     return y;
143 }

```

Gambar 4. 11 Konfigurasi Pembagi Tegangan dan Kalibrasi *Input*

2. Proses Pembacaan Tegangan *Input Buck-Boost*

Pembacaan tegangan *input* dilakukan melalui pin Vin menggunakan fungsi `analogReadMilliVolts()`. Data diambil sebanyak 100 kali kemudian dirata-ratakan untuk memperoleh hasil yang lebih stabil. Nilai rata-rata tersebut dikonversikan ke dalam satuan volt, lalu dihitung kembali berdasarkan rasio pembagi tegangan. Setelah itu, nilai dikoreksi menggunakan fungsi `kalibrasi2titik()` sehingga diperoleh nilai akhir *Vinput* sebagai tegangan *input buck-boost*.

```

292 //Voltage Input
293 double t = 0;
294 for (int i = 0; i < VoltSamples; i++) {
295     t += analogReadMilliVolts(Vin);
296     delay(2);
297 }
298 Vp = (t / VoltSamples) / 1000.0;
299 Vinput = kalibrasi2titik(Vp / VDIV_RATIO, VIN_X1, VIN_Y1, VIN_X2, VIN_Y2);

```

Gambar 4. 12 Proses Pembacaan Tegangan *Input Buck-Boost*

4.4.3 Konfigurasi Pembaca Tegangan *Output Buck-Boost*

Pembacaan tegangan *output* dilakukan melalui pin Vout menggunakan fungsi `analogReadMilliVolts()`. Data diambil sebanyak 100 kali kemudian dirata-ratakan agar hasil pembacaan lebih stabil. Nilai tersebut selanjutnya dikonversikan ke dalam satuan volt, dihitung berdasarkan rasio pembagi tegangan, lalu dikoreksi menggunakan fungsi `kalibrasi2titik()`. Hasil akhirnya disimpan pada variabel *Voutput* sebagai nilai tegangan *output buck-boost*.

```

301 //Voltage Output
302 t = 0;
303 for (int i = 0; i < VoltSamples1; i++) {
304     t += analogReadMilliVolts(Vout);
305     delay(2);
306 }
307 Vp1 = (t / VoltSamples1) / 1000.0;
308 Voutput = kalibrasi2titik(Vp1 / VDIV_RATIO, VOUT_X1, VOUT_Y1, VOUT_X2, VOUT_Y2);
309
310 //Current
311 double c = ACS_MA_DC(CurrentSamples); // rata-rata internal library (mA)
312 mA = (c / 1000.0) * currentCalFactor; // -> Ampere
313 if (fabs(mA) < currentDeadband) mA = 0; // buang noise sekitar 0 A
314 }

```

Gambar 4. 13 Konfigurasi Pembaca Tegangan *Output Buck-Boost*

4.4.4 Konfigurasi LCD 16X2 12C

1. Proses Tampilan Data pada LCD

Fungsi LCD() digunakan untuk menampilkan hasil pengukuran secara bergantian pada LCD 16×2. Data yang ditampilkan meliputi tegangan *input*, tegangan *output*, arus, tanggal dan waktu, serta tampilan gabungan tegangan dan arus. Pergantian tampilan diatur menggunakan variabel *lcdPage*, kemudian setelah halaman terakhir selesai ditampilkan, nilai *lcdPage* dikembalikan ke 0 agar tampilan kembali ke halaman pertama dan terus berulang.

```

316 void LCD() {
317     {
318         lcd.clear();
319
320         switch (lcdPage) {
321             case 0:
322                 //Vinutput
323                 lcd.setCursor(0, 0);
324                 lcd.print("V input");
325                 lcd.setCursor(0, 1);
326                 lcd.print("V : ");
327                 lcd.print(Vinput);
328                 lcd.print(" V");
329                 break;
330
331             case 1:
332                 //Voututput
333                 lcd.setCursor(0, 0);
334                 lcd.print("V Output");
335                 lcd.setCursor(0, 1);
336                 lcd.print("V : ");
337                 lcd.print(Voututput);
338                 lcd.print(" V");
339                 break;
340
341             case 2:
342                 //Arus
343                 lcd.setCursor(0, 0);
344                 lcd.print("Arus");
345                 lcd.setCursor(0, 1);
346                 lcd.print("I : ");
347                 lcd.print(mA, 3);
348                 lcd.print(" A");
349                 break;
350
351             case 3:
352                 //Jam
353                 lcd.setCursor(0, 0);
354                 lcd.print(tanggalStr);
355                 lcd.setCursor(0, 1);
356                 lcd.print(jamStr);
357                 break;
358
359             case 4:
360                 //All
361                 lcd.setCursor(0, 0);
362                 lcd.print("V : ");
363                 lcd.print(Vinput);
364                 lcd.setCursor(0, 1);
365                 lcd.print("I : ");
366                 lcd.print(mA);
367                 lcd.print(" A");
368                 break;
369         }
370
371         lcdPage++;
372         if (lcdPage >= totalLcdPages) {
373             lcdPage = 0; // loop back to the
374         }
375     }
376 }

```

Gambar 4. 14 Proses Tampilan Data pada LCD

2. Inisialisasi LCD pada setup()

Inisialisasi LCD dilakukan menggunakan fungsi `lcd.init()` untuk mengaktifkan dan menyiapkan LCD agar dapat digunakan. Selanjutnya, fungsi `lcd.backlight()` digunakan untuk menyalakan lampu latar LCD. Setelah proses inisialisasi selesai, perintah `lcd.setCursor(0, 0)` digunakan untuk menentukan posisi awal penulisan pada kolom pertama dan baris pertama, kemudian `lcd.print("Starting..")` menampilkan informasi bahwa sistem sedang melakukan proses awal.

```
394    lcd.init();
395    lcd.backlight();
396
397    lcd.setCursor(0, 0);
398    lcd.print("Starting..");
```

Gambar 4. 15 Inisialisasi LCD pada setup()

4.4.5 Pengiriman Data ke Google Spreadsheet

1. Inisialisasi Koneksi Google Spreadsheet

Fungsi `Gsheet_Boot()` digunakan untuk melakukan proses autentikasi dan menghubungkan ESP32 dengan layanan Google Spreadsheet. Fungsi `GSheet.ready()` digunakan untuk memeriksa kesiapan token autentikasi. Jika proses berhasil, sistem menampilkan informasi bahwa Google Spreadsheet telah terhubung.

```
183    // Mulai autentikasi Google
184    GSheet.setPrerefreshSeconds(10 * 60);
185    GSheet.begin(CLIENT_EMAIL, PROJECT_ID, PRIVATE_KEY);
186
187    lcd.clear();
188    lcd.setCursor(0, 0);
189    lcd.print("GSheet Connect");
190
191    unsigned long mulaiGsheet = millis();
```

Gambar 4. 16 Inisialisasi Koneksi Google Spreadsheet

2. Proses Pengiriman Data ke Google Spreadsheet

Fungsi Google Spreadsheet digunakan untuk mengirim data hasil monitoring ke Google Spreadsheet. Data yang dikirim terdiri dari waktu pengukuran, tegangan *input* (*Vinput*), tegangan *output* (*Voutput*), dan arus

(mA). Data dikirim menggunakan fungsi `GSheet.values.append()` sehingga setiap data baru ditambahkan pada baris berikutnya di Google Spreadsheet.

```

225 void Ghsheet() {
226     waktuStr = getFormattedTime();
227     Serial.println("Uploading ke GSheet!");
228
229     if (!GSheet.ready()) {
230         Serial.println("GSheet token expired atau belum siap, lewati upload");
231         return;
232     }
233
234     FirebaseJson response;
235     Serial.println("\nUpdate spreadsheet values...");
236     Serial.println("-----");
237     FirebaseJson valueRange;
238     valueRange.add("majorDimension", "ROWS");
239     valueRange.set("values/[0]/[0]", waktuStr); // row 1/ column 1
240     valueRange.set("values/[0]/[1]", Vinput); // row 2/ column 1
241     valueRange.set("values/[0]/[2]", Voutput); // row 3/ column 1
242     valueRange.set("values/[0]/[3]", mA);

```

Gambar 4. 17 Proses Pengiriman Data ke Google Spreadsheet

3. Pengaturan Pengiriman Setiap 5 Menit

Pengiriman data diatur berdasarkan waktu hasil sinkronisasi NTP. Kondisi `menit % kirimIntervalMenit == 0` membuat proses pengiriman dilakukan setiap kelipatan 5 menit, yaitu pada menit 00, 05, 10, 15, dan seterusnya. Variabel `lastMinuteSent` digunakan agar data tidak terkirim berulang kali pada menit yang sama.

```

270 struct tm timeinfo;
271 if (!getLocalTime(&timeinfo)) return;
272
273 int jam = timeinfo.tm_hour;
274 int menit = timeinfo.tm_min;
275
276 // Kirim ke Google Sheet tiap kelipatan 'kirimIntervalMenit' (setiap 5 menit)
277 if ((menit % kirimIntervalMenit == 0) && menit != lastMinuteSent) {
278     if (WiFi.status() == WL_CONNECTED) {
279         Ghsheet();
280     } else {
281         Serial.println("Gagal upload GSheet: WiFi terputus.");
282     }
283     lastMinuteSent = menit; // Set flag to prevent sending multiple times in the same minute
284 }

```

Gambar 4. 18 Pengaturan Pengiriman Setiap 5 Menit