

BAB IV

PEMBUATAN ALAT

4.1. Pembuatan Perangkat Keras

Pembuatan perangkat keras dilakukan berdasarkan rancangan sistem monitoring beban transformator distribusi yang telah disusun sebelumnya. Sistem monitoring terdiri atas dua unit utama, yaitu unit transmitter dan unit receiver, serta satu unit pendukung berupa box trafo untuk simulasi variasi tegangan. Unit transmitter digunakan untuk membaca dan mengolah parameter kelistrikan melalui PZEM-004T serta mengirimkan data melalui komunikasi LoRa. Unit receiver digunakan untuk menerima data melalui komunikasi LoRa, kemudian meneruskannya ke Google Spreadsheet dan Telegram. Sementara itu, unit box trafo simulasi variasi tegangan digunakan untuk menyediakan kondisi tegangan undervoltage, normal, dan overvoltage selama proses pengoperasian dan pengujian prototipe.

Tahap pembuatan perangkat keras mencakup pemilihan komponen, perakitan rangkaian, pemasangan modul pada panel, dan penyusunan jalur koneksi antarkomponen. Setiap bagian dirakit sesuai fungsi kerja yang telah ditentukan pada perancangan. Penempatan komponen dilakukan agar proses pengukuran, pengiriman data, penerimaan data, dan pencatatan hasil monitoring dapat berjalan secara berurutan. Susunan perangkat juga disesuaikan dengan kebutuhan pengujian prototipe pada skala laboratorium.

4.1.1 Alat dan Bahan

Pembuatan prototipe memerlukan alat kerja dan bahan utama yang mendukung proses perakitan perangkat keras. Alat digunakan untuk kegiatan pemotongan, penyambungan, pengukuran, pemasangan komponen, dan pemeriksaan rangkaian. Bahan digunakan sebagai komponen penyusun sistem, mulai dari pengendali utama, sensor, modul komunikasi, catu daya, hingga media penampil data.

Pemilihan alat dan bahan disesuaikan dengan kebutuhan sistem yang terdiri atas unit transmitter, unit receiver, dan unit pendukung simulasi variasi tegangan.

Pada unit transmitter, komponen digunakan untuk membaca, mengolah, menampilkan, dan mengirimkan parameter kelistrikan. Pada unit receiver, komponen digunakan untuk menerima data LoRa serta meneruskan data menuju media penyimpanan dan notifikasi. Adapun unit simulasi variasi tegangan menggunakan dua transformator, rotary selector switch, sensor arus PZCT-02, stopkontak, dan box akrilik sebagai perangkat pendukung pengujian. Daftar alat dan bahan yang digunakan dalam pembuatan perangkat keras disajikan pada Tabel 4 – 1 dan Tabel 4 – 2.

Tabel 4 - 1 Alat yang Digunakan

No	Nama Alat	Jumlah
1	Multimeter	1
2	Clampmeter	1
3	Obeng Set	1
4	Tang Potong	1
5	Tang Kupas	1
6	Tang Kombinasi	1
7	Solder	1
8	Bor	1
9	Gerinda	1
10	Alat Tulis	1

Tabel 4 - 2 Bahan yang Digunakan

No	Nama Bahan	Jumlah
1	Box Panel	1
2	ESP32 30 pin	2
3	LCD I2C 16x2	1
4	LoRa Ra-02 SX1278	2
5	PZEM-004T	1
6	PZCT-02	1
7	Terminal Blok 4 Sisi	1

No	Nama Bahan	Jumlah
8	PSU 5V 2A	2
9	Terminal Block 2 Pin	2
10	JST Connector 4 Pin	4 Pasang
11	JST Connector 3 Pin	2 pasang
12	Box Plastik Hitam Uk. X6	1
13	Push Button	1
14	Saklar	2
15	Baut M3 (5mm)	Sesuai kebutuhan
16	Baut M3 (10mm)	Sesuai kebutuhan
17	Baut M3 (15mm)	Sesuai kebutuhan
18	Spacer M3 (8mm)	Sesuai kebutuhan
19	Mur M3	Sesuai kebutuhan
20	Timah Solder	Sesuai kebutuhan
21	PCB Polos	2
22	Pin Header Female	Sesuai kebutuhan
23	Resistor 10k	1
24	Cover LCD	1
25	Antena LoRa 433 MHz	2
26	Skun Kabel	6
27	Ferric Chloride (FeCl ₃)	Sesuai kebutuhan
28	Kabel NYAF 2.5mm	2 meter
29	Transformator <i>center tap</i> satu fasa 5A	2 Buah
30	Rotary selector switch 4 posisi	1 Buah
31	Stopkontak	1 Buah
32	Akrilik 3mm	Sesuai kebutuhan

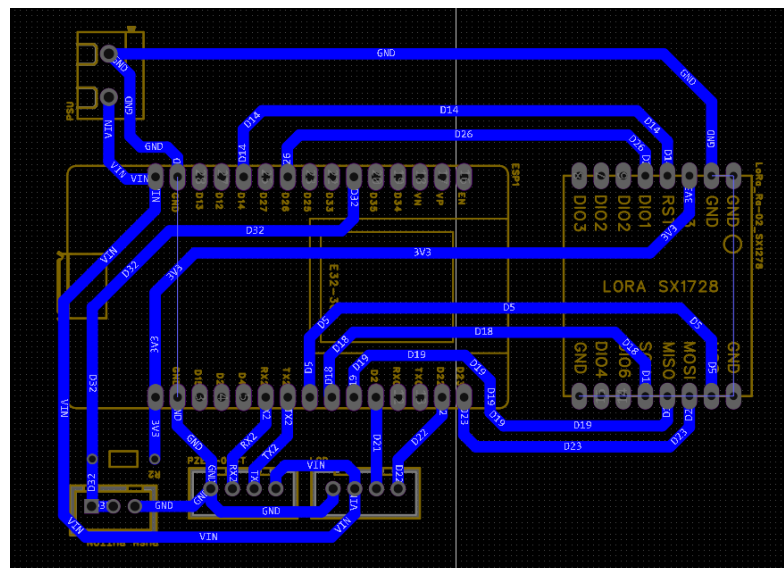
4.1.2 Pembuatan Rangkaian Elektrik

4.1.2.1 Pembuatan Rangkaian Elektrik Transmitter

Perancangan perangkat elektrik unit transmitter dilakukan untuk menyusun hubungan antarkomponen dalam satu papan rangkaian. Unit ini berfungsi sebagai pusat pembacaan sensor, pengolahan data, tampilan lokal, dan pengiriman data melalui LoRa. Komponen utama yang digunakan meliputi ESP32, LoRa Ra-02 SX1278, konektor sensor, konektor LCD, resistor, dan terminal catu daya. Susunan tersebut disesuaikan dengan kebutuhan kerja transmitter pada sistem monitoring beban transformator distribusi.

4.1.2.1.1 Perancangan Layout PCB Transmitter

Perancangan jalur PCB dilakukan menggunakan aplikasi EasyEDA. Aplikasi ini digunakan untuk membuat skematik rangkaian dan layout PCB unit transmitter. Setiap komponen ditempatkan sesuai kebutuhan koneksi dan kemudahan perakitan. Pin header ESP32 diletakkan sebagai bagian utama karena seluruh proses kendali dilakukan oleh mikrokontroler tersebut.



Gambar 4. 1 Layout PCB Transmitter

Routing jalur PCB dibuat berdasarkan hubungan antarkomponen pada unit transmitter. Jalur komunikasi SPI digunakan untuk menghubungkan ESP32 dengan modul LoRa Ra-02 SX1278. Jalur komunikasi UART disiapkan untuk koneksi menuju sensor PZEM-004T. Jalur I2C disiapkan untuk koneksi LCD. Jalur input

tombol dan jalur catu daya juga dibuat sesuai kebutuhan rangkaian.

Penempatan komponen dilakukan dengan mempertimbangkan kerapian jalur dan jarak antarkoneksi. Pin header LoRa ditempatkan dekat dengan jalur SPI agar koneksi lebih pendek. Konektor JST digunakan sebagai penghubung perangkat eksternal dengan PCB. Terminal block dua pin digunakan sebagai masukan catu daya. Resistor ditempatkan pada jalur tombol sebagai bagian dari rangkaian input digital.

Hasil akhir dari proses ini berupa layout PCB yang siap dicetak pada media kertas transfer. Layout tersebut memuat pola jalur tembaga, titik pengeboran, dan posisi setiap komponen. Pemeriksaan layout dilakukan sebelum pencetakan untuk menghindari kesalahan jalur. Pemeriksaan ini mencakup kesinambungan jalur, posisi pad, dan kesesuaian koneksi dengan rancangan transmitter.

4.1.2.1.2 Percetakan dan Penyolderan Komponen pada PCB Transmitter

Percetakan PCB dilakukan menggunakan PCB tembaga polos sebagai media dasar rangkaian. Setelah layout PCB selesai dibuat, pola jalur dicetak pada kertas CTS menggunakan printer laser. Sisi kertas yang memiliki tinta kemudian ditempelkan pada sisi tembaga PCB. Posisi kertas dibuat sejajar agar pola jalur tidak bergeser saat proses pemindahan.

PCB yang telah ditemplei kertas kemudian dimasukkan ke dalam mesin press. Proses pengepresan dilakukan selama kurang lebih 10 menit. Tahap ini bertujuan memindahkan pola tinta laser ke permukaan tembaga. Setelah proses selesai, kertas dilepas secara perlahan. Jalur yang telah menempel diperiksa agar tidak terdapat bagian yang terputus.

Tahap selanjutnya dilakukan dengan proses pelarutan tembaga. PCB dicelupkan ke dalam larutan ferric chloride. PCB digoyangkan secara perlahan hingga tembaga yang tidak tertutup pola jalur larut seluruhnya. Setelah proses pelarutan selesai, permukaan PCB dibersihkan dari sisa kertas dan tinta. Bagian pad kemudian dilubangi sesuai titik kaki komponen.

Penyolderan dilakukan setelah seluruh lubang komponen selesai dibuat. Komponen yang dipasang terdiri atas pin header ESP32, pin header LoRa, dua konektor JST empat pin, satu konektor JST tiga pin, satu resistor, dan satu terminal

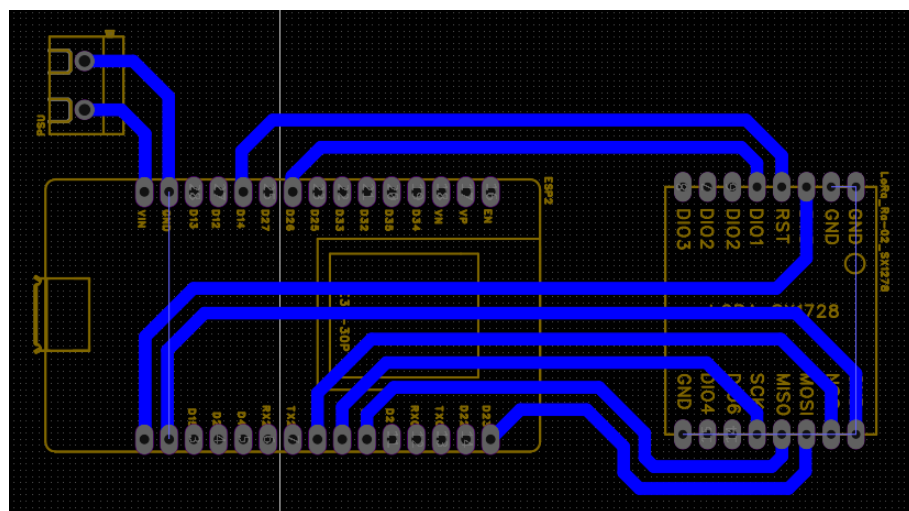
block dua pin. Setiap kaki komponen disolder pada pad yang telah tersedia. Hasil solder diperiksa untuk memastikan tidak terdapat jalur yang terhubung singkat. Pemeriksaan akhir dilakukan sebelum PCB dipasang pada unit transmitter.

4.1.2.2 Pembuatan Rangkaian Elektrik Receiver

Perancangan perangkat elektrik unit receiver dilakukan untuk membentuk rangkaian penerima dan pengolah data dari unit transmitter. Unit ini menggunakan ESP32 sebagai pengendali utama dan LoRa Ra-02 SX1278 sebagai media komunikasi nirkabel. Data yang diterima dari transmitter kemudian diproses oleh ESP32 sebelum dikirimkan ke Google Spreadsheet dan Telegram melalui koneksi WiFi.

4.1.2.2.1 Perancangan Layout PCB Receiver

Perancangan jalur PCB unit receiver dilakukan menggunakan aplikasi EasyEDA. Aplikasi ini digunakan untuk menyusun skematik rangkaian dan layout PCB. Komponen utama yang dimasukkan meliputi pin header ESP32, pin header LoRa Ra-02 SX1278, dan terminal block dua pin. Setiap komponen ditempatkan sesuai kebutuhan hubungan elektrik pada unit receiver.



terminal block menuju ESP32.

Penempatan komponen dilakukan dengan mempertimbangkan kerapian jalur dan kemudahan pemasangan. Pin header ESP32 ditempatkan sebagai pusat rangkaian. Pin header LoRa diletakkan dekat dengan jalur SPI agar sambungan lebih ringkas. Terminal block dua pin ditempatkan pada sisi tepi PCB agar kabel catu daya mudah dipasang.

Hasil perancangan layout kemudian diperiksa sebelum proses pencetakan. Pemeriksaan dilakukan pada kesinambungan jalur, jarak antarpad, dan posisi kaki komponen. Tahap ini diperlukan agar kesalahan koneksi dapat dikurangi sebelum PCB dibuat. Layout yang telah sesuai kemudian digunakan sebagai pola dasar pada proses percetakan PCB.

4.1.2.2.2 Percetakan dan Penyolderan Komponen pada PCB Receiver

Percetakan PCB receiver dilakukan menggunakan PCB tembaga polos. Setelah layout selesai dibuat, pola jalur dicetak pada kertas CTS menggunakan printer laser. Sisi kertas yang memiliki tinta ditempelkan pada sisi tembaga PCB. Posisi kertas diatur agar pola jalur menempel sesuai bentuk layout.

PCB yang telah ditemplei kertas kemudian dimasukkan ke dalam mesin press. Proses pengepresan dilakukan selama kurang lebih 10 menit. Tahap ini bertujuan memindahkan pola tinta dari kertas ke lapisan tembaga. Setelah proses selesai, kertas dilepas secara perlahan dari permukaan PCB. Jalur tembaga diperiksa untuk memastikan pola telah menempel dengan baik.

Tahap berikutnya dilakukan menggunakan larutan ferric chloride. PCB dicelupkan ke dalam larutan tersebut hingga bagian tembaga yang tidak tertutup pola jalur larut. PCB digoyangkan secara perlahan agar proses pelarutan berlangsung merata. Setelah tembaga yang tidak diperlukan hilang, PCB diangkat dan dibersihkan. Sisa kertas serta tinta yang masih menempel dihilangkan dari permukaan PCB.

Setelah proses pembersihan selesai, bagian pad komponen dilubangi sesuai titik pada layout. Komponen kemudian dipasang pada PCB sesuai posisi masing-masing. Komponen yang dipasang meliputi pin header ESP32, pin header LoRa, dan terminal block dua pin. Setiap kaki komponen disolder hingga menempel kuat

pada pad PCB. Hasil penyolderan diperiksa agar tidak terdapat hubungan singkat antarlahjur.

4.1.3 Pembuatan Perangkat Mekanik

4.1.3.1 Pembuatan Perangkat Mekanik Transmitter

Desain perangkat mekanik unit transmitter dilakukan menggunakan perangkat lunak SolidWorks. Perancangan ini difokuskan pada penempatan komponen, kerapian susunan perangkat, dan kemudahan proses perakitan. Unit transmitter dirancang menggunakan box panel karena perangkat ditempatkan pada sisi transformator dan berpotensi digunakan pada lingkungan luar ruangan. Pemilihan box panel bertujuan melindungi komponen dari debu, percikan air, dan gangguan fisik selama alat beroperasi.

Rancangan mekanik dibuat dengan memperhatikan posisi setiap komponen di dalam box panel. PCB utama, modul PZEM-004T, terminal block, dan power supply diletakkan pada dudukan plat yang berada di bagian dalam box panel. Susunan tersebut dibuat agar komponen utama memiliki posisi yang stabil dan mudah dijangkau saat pemeriksaan. LCD dan push button ditempatkan pada bagian bukaan box panel agar informasi pengukuran dapat dibaca dari luar.

Komponen mekanik pada unit transmitter disusun agar proses pemasangan dan perawatan dapat dilakukan dengan lebih mudah. Setiap bagian dirancang sesuai fungsi kerja masing-masing. Komponen yang digunakan pada konstruksi mekanik unit transmitter meliputi:

1. Box panel, berfungsi sebagai rumah utama perangkat transmitter dan pelindung komponen elektronik.
2. Dudukan plat bagian dalam, berfungsi sebagai tempat pemasangan PCB utama, PZEM-004T, terminal block, dan power supply.
3. PCB utama, berfungsi sebagai papan rangkaian untuk menghubungkan ESP32, LoRa, dan konektor pendukung.
4. Modul PZEM-004T, berfungsi sebagai unit pembacaan parameter kelistrikan pada sisi transmitter.
5. Power supply, berfungsi sebagai penyedia catu daya DC untuk perangkat elektronik.

6. Terminal block, berfungsi sebagai titik sambungan kabel masukan dan distribusi tegangan.
7. LCD, berfungsi sebagai penampil data pengukuran pada bagian luar box panel.
8. Push button, berfungsi sebagai tombol masukan untuk mengganti tampilan data pada LCD.

Perakitan alat dilakukan dengan memasang seluruh komponen sesuai posisi yang telah dirancang pada SolidWorks. PCB utama, PZEM-004T, terminal block, dan power supply dipasang pada dudukan plat menggunakan baut agar tidak mudah bergeser. LCD dan push button dipasang pada bagian bukaan box panel agar tetap dapat diakses tanpa membuka penutup panel. Setelah seluruh komponen terpasang, jalur kabel dirapikan agar tidak mengganggu sambungan antarkomponen. Desain mekanik ini dibuat agar unit transmitter mampu bekerja dengan susunan perangkat yang rapi, aman, dan mudah diperiksa saat pengujian.

4.1.3.2 Pembuatan Perangkat Mekanik Receiver

Desain perangkat mekanik unit receiver dilakukan menggunakan perangkat lunak SolidWorks. Perancangan ini difokuskan pada penempatan komponen, kerapian rangkaian, dan kemudahan akses saat alat digunakan. Unit receiver dirancang sebagai perangkat penerima data dari unit transmitter melalui komunikasi LoRa. Data yang diterima kemudian diproses oleh ESP32 sebelum dikirimkan ke Google Spreadsheet dan Telegram melalui jaringan internet. Fungsi tersebut disesuaikan dengan rancangan receiver pada sistem monitoring beban transformator distribusi.

Rancangan mekanik receiver menggunakan box plastik hitam berukuran X6 sebagai tempat penyimpanan komponen utama. Box plastik dipilih karena memiliki bobot ringan dan mudah digunakan pada perangkat portabel. Selain itu, material plastik dapat melindungi rangkaian dari debu, sentuhan langsung, dan gangguan fisik ringan. PCB utama dan power supply ditempatkan di bagian dalam box agar seluruh rangkaian tersusun rapi dan terlindung.

Saklar on/off diletakkan pada bagian luar box, tepatnya di sisi samping. Penempatan tersebut bertujuan memudahkan pengguna saat menghidupkan atau mematikan alat. Pengguna tidak perlu membuka box untuk mengakses sumber catu

daya. Posisi saklar juga dibuat agar tidak mengganggu susunan PCB utama dan power supply di bagian dalam box.

Komponen mekanik pada unit receiver disusun agar proses pemasangan dan perawatan dapat dilakukan dengan mudah. Setiap bagian ditempatkan sesuai fungsi kerjanya. Komponen yang digunakan pada konstruksi mekanik unit receiver meliputi:

1. Box plastik hitam X6, berfungsi sebagai rumah utama perangkat receiver dan pelindung komponen elektronik.
2. PCB utama, berfungsi sebagai papan rangkaian untuk menempatkan ESP32, LoRa, dan jalur koneksi pendukung.
3. Power supply, berfungsi sebagai penyedia catu daya DC untuk ESP32 dan modul pendukung.
4. Saklar on/off, berfungsi sebagai pemutus dan penghubung catu daya alat.
5. Terminal catu daya, berfungsi sebagai titik sambungan antara power supply dan rangkaian utama.
6. Modul LoRa Ra-02 SX1278, berfungsi sebagai penerima data nirkabel dari unit transmitter.
7. ESP32, berfungsi sebagai pengendali utama pada unit receiver.

Perakitan alat dilakukan dengan memasang PCB utama dan power supply pada bagian dalam box. Kedua komponen tersebut diposisikan agar jalur kabel tidak saling bertumpuk. Saklar dipasang pada sisi samping luar box agar mudah dijangkau saat alat dioperasikan. Setelah seluruh komponen terpasang, kabel catu daya dan jalur koneksi dirapikan mengikuti ruang kosong di dalam box. Pemeriksaan akhir dilakukan untuk memastikan setiap sambungan telah terpasang kuat dan tidak terjadi hubungan singkat. Desain mekanik ini dibuat agar unit receiver memiliki susunan perangkat yang ringkas, aman, dan mudah digunakan saat proses monitoring berlangsung.

4.1.3.3 Pembuatan Perangkat Mekanik Unit Box Trafo

Desain perangkat mekanik unit simulasi variasi tegangan dilakukan menggunakan website MakerCase dan perangkat lunak SolidWorks. Website MakerCase digunakan untuk membuat pola panel box beserta sambungan T-slot, sedangkan

SolidWorks digunakan untuk merancang bentuk tiga dimensi dan menentukan tata letak komponen secara keseluruhan. Perancangan ini difokuskan pada kesesuaian dimensi box, penempatan komponen, kerapian jalur pengawatan, kemudahan pengoperasian, serta kemudahan dalam melakukan pemeriksaan dan perawatan.

Unit simulasi variasi tegangan dibuat sebagai perangkat pendukung yang terpisah dari unit transmitter. Unit tersebut digunakan untuk menyediakan pilihan tegangan yang mewakili kondisi undervoltage, normal, dan overvoltage. Meskipun ditempatkan pada box yang berbeda, unit simulasi tetap terhubung secara elektrik dengan unit transmitter agar tegangan dan arus yang disalurkan menuju beban dapat diukur oleh PZEM-004T dan sensor arus PZCT-02.

Komponen mekanik pada unit simulasi variasi tegangan disusun agar proses pemasangan, pengoperasian, dan perawatan dapat dilakukan dengan mudah. Setiap bagian ditempatkan sesuai dengan fungsi kerjanya. Komponen yang digunakan pada konstruksi mekanik unit simulasi variasi tegangan meliputi:

1. Box akrilik, berfungsi sebagai rumah utama unit simulasi variasi tegangan dan melindungi komponen yang ditempatkan di dalamnya.
2. Baut dan mur M3, berfungsi menyatukan setiap panel akrilik melalui sistem sambungan T-slot.
3. Transformator pertama, berfungsi menurunkan tegangan sumber 220 V AC menjadi 12 V AC.
4. Transformator kedua, berfungsi menaikkan kembali tegangan 12 V AC serta menyediakan pilihan tegangan keluaran melalui terminal 110 V, 220 V, dan 240 V.
5. Rotary selector switch, berfungsi memilih tegangan keluaran sesuai dengan kondisi simulasi yang digunakan.
6. Stopkontak, berfungsi sebagai titik keluaran tegangan dan tempat penyambungan beban.
7. Sensor arus PZCT-02, berfungsi mendeteksi arus yang mengalir pada penghantar fasa menuju beban.
8. Terminal penghubung, berfungsi sebagai titik penyambungan kabel dan distribusi hubungan antarkomponen.

9. Kabel penghantar, berfungsi menghubungkan sumber tegangan, transformator, rotary selector switch, sensor PZCT-02, stopkontak, dan jalur pengukuran menuju unit transmitter.

Pembuatan perangkat mekanik diawali dengan membuat pola panel box menggunakan website MakerCase berdasarkan ukuran dan ketebalan material yang telah ditentukan. Pola tersebut kemudian digunakan sebagai acuan pemotongan lembaran akrilik menggunakan mesin CNC laser sehingga diperoleh panel-panel box beserta sambungan T-slot dan lubang pemasangan. Panel hasil pemotongan selanjutnya dirakit menggunakan baut dan mur M3 tanpa bahan perekat. Setelah box terbentuk, dua buah transformator dipasang pada bagian dasar sesuai tata letak yang telah dirancang menggunakan SolidWorks. Kedua transformator ditempatkan secara berurutan agar hubungan pada sisi 12 V dapat dibuat lebih ringkas dan dipasang menggunakan baut serta mur agar tetap stabil ketika unit dipindahkan maupun dioperasikan. Sensor arus PZCT-02 ditempatkan pada bagian dalam box, sedangkan rotary selector switch dan stopkontak dipasang pada panel luar agar dapat dioperasikan tanpa membuka penutup box.

Penyambungan kabel dilakukan sesuai dengan wiring diagram pada Bab III. Sisi 220 V transformator pertama dihubungkan ke sumber PLN, sedangkan sisi 12 V transformator pertama dihubungkan ke sisi 12 V transformator kedua. Terminal keluaran 110 V, 220 V, dan 240 V dari transformator kedua dihubungkan ke rotary selector switch, kemudian keluaran tegangan yang dipilih disalurkan menuju stopkontak melalui penghantar fasa yang melewati sensor arus PZCT-02. Setelah seluruh komponen terpasang, jalur pengawatan dirapikan dan dilakukan pemeriksaan terhadap posisi komponen, kekencangan baut dan mur, sambungan terminal, serta kesesuaian pengawatan dengan wiring diagram. Penggunaan sambungan T-slot membuat box dapat dibongkar dan dipasang kembali dengan mudah, sedangkan susunan komponen di dalamnya memungkinkan unit digunakan untuk menyediakan kondisi tegangan undervoltage, normal, dan overvoltage pada sistem monitoring.

4.2 Pembuatan Perangkat Lunak

Pembuatan perangkat lunak dilakukan untuk mengatur kerja sistem monitoring beban transformator distribusi. Tahap ini meliputi pemrograman ESP32 pada unit transmitter dan receiver menggunakan Arduino IDE. Selain itu, konfigurasi Google Spreadsheet dan Telegram juga dilakukan sebagai media pencatatan data dan pengiriman notifikasi. Program disusun agar proses pembacaan sensor, komunikasi LoRa, sinkronisasi waktu, penyimpanan data, dan pengiriman peringatan dapat berjalan sesuai rancangan sistem. Alur kerja perangkat lunak mengikuti perancangan transmitter dan receiver yang telah dijelaskan pada Bab III.

4.2.1. Pembuatan Perangkat Lunak Transmitter

Pembuatan perangkat lunak pada sisi transmitter difokuskan pada proses pembacaan data sensor dan pengiriman data ke unit receiver. Program transmitter dibuat menggunakan Arduino IDE dengan ESP32 sebagai pengendali utama. ESP32 diprogram untuk membaca data dari sensor PZEM-004T, menampilkan data pada LCD I2C, serta menunggu permintaan data dari receiver melalui LoRa. Mekanisme ini digunakan agar transmitter hanya mengirimkan data saat terdapat permintaan dari receiver. Dengan demikian, proses komunikasi dapat berlangsung lebih teratur dan sesuai dengan metode polling yang digunakan pada sistem.

4.2.1.1 Pemrograman Arduino IDE Transmitter

Pemrograman perangkat lunak transmitter dilakukan menggunakan Arduino IDE dengan bahasa pemrograman C++. Program ini berfungsi mengatur seluruh proses pembacaan parameter kelistrikan, pengolahan data, penampilan informasi pada LCD, serta komunikasi data dengan receiver melalui LoRa. ESP32 digunakan sebagai pengendali utama yang mengintegrasikan seluruh perangkat pada unit transmitter sehingga sistem mampu bekerja secara otomatis sesuai dengan rancangan yang telah ditetapkan.

Struktur program disusun secara modular untuk memudahkan proses pengembangan, pengujian, dan pemeliharaan sistem. Setiap fungsi dibuat berdasarkan tugas yang dijalankan oleh masing-masing perangkat. Program diawali dengan proses konfigurasi pin, inisialisasi sensor PZEM-004T, LCD I2C, dan modul LoRa Ra-02 SX1278. Setelah proses inisialisasi selesai, ESP32 melakukan pembacaan data sensor secara berkala, menampilkan hasil pembacaan pada LCD,

serta menunggu permintaan data dari receiver melalui komunikasi LoRa.

Pembagian program yang dikembangkan pada perangkat lunak transmitter meliputi beberapa bagian sebagai berikut:

1. Program untuk pengaturan port dan komunikasi serial ESP32.
2. Program untuk inisialisasi pin dan konfigurasi awal ESP32 transmitter.
3. Program untuk pembacaan parameter kelistrikan menggunakan sensor PZEM-004T.
4. Program untuk perhitungan persentase pembebanan transformator.
5. Program untuk penampilan data hasil pengukuran pada LCD I2C.
6. Program untuk pembacaan push button sebagai pengaturan antarmuka LCD.
7. Program untuk komunikasi dan pengiriman data melalui LoRa Ra-02 SX1278.
8. Program multitasking menggunakan FreeRTOS untuk menjalankan proses pembacaan sensor, tampilan LCD, dan komunikasi LoRa secara bersamaan.

Dengan pembagian tersebut, proses monitoring pada transmitter dapat dilakukan secara berkelanjutan. Data hasil pembacaan sensor akan selalu diperbarui, ditampilkan pada LCD, dan dikirimkan kepada receiver ketika terdapat permintaan data melalui mekanisme polling. Pendekatan ini memungkinkan sistem memperoleh data pengukuran yang selalu mutakhir tanpa membebani saluran komunikasi LoRa dengan pengiriman data secara terus-menerus.

4.2.1.2 Pengaturan Port pada Arduino IDE Transmitter

Pengaturan port pada Arduino IDE dilakukan sebelum program transmitter diunggah ke ESP32. Tahap ini bertujuan memastikan perangkat ESP32 terbaca oleh komputer dan dapat menerima program dengan benar. Pada Arduino IDE, jenis board disesuaikan dengan mikrokontroler yang digunakan, yaitu ESP32. Setelah board dipilih, port komunikasi serial yang terhubung dengan ESP32 ditentukan melalui menu port pada Arduino IDE. Tahap ini diperlukan agar proses kompilasi dan pengunggahan program dapat berjalan tanpa kesalahan komunikasi.

Pada program transmitter, komunikasi serial diaktifkan menggunakan fungsi `Serial.begin(115200)`. Fungsi ini digunakan sebagai media pemantauan data melalui serial monitor selama proses pengujian. Kecepatan komunikasi serial ditetapkan sebesar 115200 bps agar proses pembacaan status sistem dapat berjalan

stabil. Informasi yang ditampilkan melalui serial monitor meliputi status LoRa, data sensor PZEM-004T, dan data yang dikirim menuju receiver. Program pengaturan komunikasi serial pada transmitter ditunjukkan pada Gambar 4. 3.

```
void setup() {
  Serial.begin(115200);
  buttonBegin();
  lcdBegin();
  welcomeLCD();
  setupLoRa();
  menuPage = MENU_1;
}
```

Gambar 4. 3 Program Pengaturan Komunikasi Serial pada Arduino IDE Transmitter

Pengaturan port yang sesuai membuat ESP32 dapat berkomunikasi dengan Arduino IDE saat proses upload program dilakukan. Selain itu, pengaturan baud rate yang sama antara program dan serial monitor diperlukan agar data tampil dengan benar. Apabila baud rate tidak sesuai, karakter yang muncul pada serial monitor dapat tidak terbaca. Dengan demikian, pengaturan port dan komunikasi serial menjadi tahap awal sebelum program transmitter dijalankan pada perangkat keras.

4.2.1.3 Inisialisasi Pin dan Konfigurasi Awal ESP32 Transmitter

Inisialisasi pin dan konfigurasi awal ESP32 transmitter dilakukan untuk mendefinisikan seluruh jalur koneksi perangkat. Pada tahap ini, program menetapkan pin untuk modul LoRa, sensor PZEM-004T, LCD I2C, push button, dan identitas transformator. ESP32 berfungsi sebagai pengendali utama yang mengatur pembacaan sensor, tampilan LCD, serta komunikasi data melalui LoRa. Setiap pin ditentukan sesuai rancangan rangkaian transmitter yang telah dibuat pada perangkat keras.

Program transmitter menggunakan beberapa library pendukung. Library SPI.h dan LoRa.h digunakan untuk komunikasi modul LoRa Ra-02 SX1278. Library Wire.h dan LiquidCrystal_I2C.h digunakan untuk komunikasi LCD melalui jalur I2C. Library PZEM004Tv30.h digunakan untuk membaca data kelistrikan dari sensor

PZEM-004T. Pemanggilan library dan pendefinisian pin utama transmitter ditunjukkan pada Gambar 4. 4.

```
#include <Arduino.h>
#include <SPI.h>
#include <LoRa.h>
#include <Wire.h>
#include <PZEM004Tv30.h>
#include <LiquidCrystal_I2C.h>
//LORA
#define SS 5
#define RST 14
#define DIO0 26
const long frequencyLora = 433E6;
//PZEM004T
#define PZEM_RX_PIN 16
#define PZEM_TX_PIN 17
#define PZEM_ADDR 0xF8
#define PZEM_SERIAL Serial2
//LCD
#define I2C_ADDR 0x27
#define LCD_COLUMNS 16
#define LCD_ROWS 2
//PUSH BUTTON
#define BUTTON_PIN 32
```

Gambar 4. 4 Program Library dan Inisialisasi Pin Transmitter

Selain pendefinisian pin, program juga menetapkan identitas transformator dan kapasitas nominal yang digunakan. Identitas TRAFO_ID digunakan sebagai alamat transmitter saat receiver melakukan polling data. Nilai TRAFO_VA dan TRAFO_VOLT digunakan sebagai dasar perhitungan arus nominal dan persentase pembebanan. Dengan konfigurasi ini, transmitter hanya merespons permintaan data yang sesuai dengan identitas perangkat. Program identitas dan kapasitas transformator ditunjukkan pada Gambar 4. 5.

```
#define TRAFO_ID "TRAF01"
```

```
#define TRAFO_VA    1.1 * 1000.0
#define TRAFO_VOLT  220.0
```

Gambar 4. 5 Program Identitas dan Kapasitas Transformator

Objek perangkat kemudian dibuat agar setiap library dapat digunakan oleh ESP32. Objek lcd digunakan untuk mengatur tampilan LCD I2C. Objek pzem digunakan untuk membaca data sensor melalui komunikasi Serial2. Selain itu, dataMutex digunakan untuk mengatur akses data antara proses pembacaan sensor dan proses tampilan. Konfigurasi objek perangkat transmitter ditunjukkan pada Gambar 4. 6.

```
LiquidCrystal_I2C lcd(I2C_ADDR, LCD_COLUMNS, LCD_ROWS);
PZEM004Tv30 pzem(PZEM_SERIAL, PZEM_RX_PIN, PZEM_TX_PIN,
PZEM_ADDR);
SemaphoreHandle_t dataMutex;
```

Gambar 4. 6 Program Deklarasi Objek Perangkat Transmitter

Pada fungsi setup(), ESP32 menjalankan konfigurasi awal secara berurutan. Program mengaktifkan komunikasi serial, mengatur push button, menginisialisasi LCD, menampilkan tampilan awal, dan mengaktifkan LoRa. Setelah itu, program membuat dua task yang bekerja pada core berbeda. Task pertama digunakan untuk memeriksa permintaan data dari receiver. Task kedua digunakan untuk membaca sensor dan menampilkan data pada LCD. Program konfigurasi awal ESP32 transmitter ditunjukkan pada Gambar 4. 7.

```
void setup() {
    Serial.begin(115200);
    buttonBegin();
    lcdBegin();
    welcomeLCD();
    setupLoRa();
    menuPage = MENU_1;
    dataMutex = xSemaphoreCreateMutex();
    xTaskCreatePinnedToCore(
        taskIncoming,
        "Incoming",
        4096,
```

```

        NULL,
        2,
        NULL,
        0);

    xTaskCreatePinnedToCore(
        taskDisplay,
        "Display",
        4096,
        NULL,
        1,
        NULL,
        1);
}

```

Gambar 4. 7 Program Konfigurasi Awal ESP32 Transmitter

Konfigurasi awal tersebut membuat transmitter dapat menjalankan dua proses utama secara terpisah. Proses penerimaan polling LoRa dijalankan melalui `taskIncoming`. Proses pembacaan sensor dan tampilan LCD dijalankan melalui `taskDisplay`. Pembagian tugas ini digunakan agar transmitter tetap dapat membaca sensor saat menunggu permintaan data dari receiver. Dengan demikian, data yang dikirimkan melalui LoRa tetap berasal dari pembacaan terbaru pada sensor PZEM-004T.

4.2.1.4 Pembacaan Sinyal Sensor PZEM-004T Transmitter

Pembacaan data sensor PZEM-004T dilakukan pada sisi transmitter untuk memperoleh parameter kelistrikan transformator. Sensor dihubungkan dengan ESP32 melalui komunikasi serial menggunakan `Serial2`. Data yang dibaca meliputi tegangan, arus, daya, frekuensi, dan faktor daya. Nilai tersebut kemudian digunakan untuk menghitung persentase pembebanan transformator. Program pembacaan data sensor PZEM-004T transmitter ditunjukkan pada Gambar 4. 8.

```

void readTransformerData(){
    voltage = pzem.voltage();
    current = pzem.current();
    power = pzem.power();
}

```

```

frequency = pzem.frequency();
pf = pzem.pf();

float ratedCurrent = (TRAFO_VA / TRAFO_VOLT);
loadPercentage = (current / ratedCurrent) * 100.0;
}

```

Gambar 4. 8 Program Pembacaan Data Sensor PZEM-004T Transmitter

Pada program tersebut, fungsi `readTransformerData()` digunakan sebagai fungsi utama untuk membaca seluruh data dari sensor PZEM-004T. Perintah `pzem.voltage()` digunakan untuk membaca nilai tegangan. Perintah `pzem.current()` digunakan untuk membaca nilai arus. Perintah `pzem.power()` digunakan untuk membaca nilai daya. Perintah `pzem.frequency()` digunakan untuk membaca frekuensi listrik. Perintah `pzem.pf()` digunakan untuk membaca nilai faktor daya. Seluruh nilai hasil pembacaan disimpan pada variabel yang telah dideklarasikan sebelumnya.

Nilai arus hasil pembacaan sensor kemudian digunakan untuk menghitung persentase pembebanan transformator. Perhitungan diawali dengan mencari arus nominal transformator menggunakan nilai kapasitas transformator dan tegangan nominal. Nilai kapasitas ditetapkan melalui variabel `TRAFO_VA`, sedangkan tegangan nominal ditetapkan melalui variabel `TRAFO_VOLT`. Setelah arus nominal diperoleh, nilai arus aktual dibandingkan terhadap arus nominal tersebut. Hasil perbandingan dikalikan 100 untuk memperoleh nilai pembebanan dalam persen dengan program yang ditunjukkan pada Gambar 4. 9.

```

#define TRAFO_VA    1.1 * 1000.0
#define TRAFO_VOLT  220.0

```

Gambar 4. 9 Program Penetapan Kapasitas dan Tegangan Nominal Transformator

Fungsi pembacaan sensor dijalankan secara berkala pada `taskDisplay`. Pembacaan dilakukan setiap 3000 ms atau 3 detik menggunakan fungsi `millis()`. Metode ini digunakan agar pembacaan sensor berlangsung teratur tanpa menghentikan proses lain pada ESP32. Selama proses pembacaan berlangsung, akses data diamankan menggunakan `dataMutex`. Mekanisme tersebut digunakan agar data tidak diakses bersamaan oleh proses tampilan LCD dan proses

komunikasi LoRa. Program pembacaan sensor secara berkala ditunjukkan pada Gambar 4. 10.

```
if (millis() - previousRead >= 3000){
    if (xSemaphoreTake(dataMutex, portMAX_DELAY)){
        readTransformerData();
        xSemaphoreGive(dataMutex);
    }
    previousRead = millis();
}
```

Gambar 4. 10 Program Pembacaan Sensor Secara Berkala pada Transmitter

Berdasarkan alur program tersebut, sensor PZEM-004T dibaca secara otomatis oleh ESP32 pada sisi transmitter. Data hasil pembacaan disimpan dalam variabel utama, lalu digunakan untuk tampilan LCD dan pengiriman data ke receiver. Dengan alur ini, data yang dikirimkan melalui LoRa merupakan data terbaru dari hasil pembacaan sensor.

4.2.1.5 Penampilan Data Sensor PZEM-004T Pada LCD Transmitter

LCD I2C digunakan sebagai media tampilan lokal pada unit transmitter. Komponen ini berfungsi menampilkan data hasil pembacaan sensor PZEM-004T secara langsung. Data yang ditampilkan meliputi tegangan, arus, frekuensi, faktor daya, daya, dan persentase pembebanan transformator. Penggunaan LCD bertujuan agar kondisi beban dapat diamati langsung pada lokasi pemasangan transmitter tanpa perangkat tambahan. Program inisialisasi LCD I2C transmitter ditunjukkan pada Gambar 4. 11.

```
void lcdBegin(){
    Wire.begin();
    lcd.init();
    lcd.backlight();
}
```

Gambar 4. 11 Program Inisialisasi LCD I2C Transmitter

Pada program tersebut, fungsi Wire.begin() digunakan untuk mengaktifkan komunikasi I2C pada ESP32. Fungsi lcd.init() digunakan untuk memulai kerja LCD. Fungsi lcd.backlight() digunakan untuk menyalakan lampu latar LCD.

Setelah proses ini selesai, LCD siap digunakan untuk menampilkan informasi dari sistem transmitter. Konfigurasi LCD telah ditetapkan menggunakan alamat I2C 0x27, jumlah kolom 16, dan jumlah baris 2. Program konfigurasi ukuran LCD ditunjukkan pada Gambar 4. 12.

```
#define I2C_ADDR    0x27
#define LCD_COLUMNS 16
#define LCD_ROWS    2

LiquidCrystal_I2C lcd(I2C_ADDR,LCD_COLUMNS,LCD_ROWS);
```

Gambar 4. 12 Program Konfigurasi Alamat dan Ukuran LCD

Setelah LCD berhasil diinisialisasi, sistem menampilkan pesan awal sebagai indikator bahwa transmitter telah aktif. Tampilan awal dibuat melalui fungsi `welcomeLCD()`. Pesan tersebut muncul selama 2 detik, kemudian layar dibersihkan menggunakan perintah `lcd.clear()`. Program tampilan awal LCD transmitter ditunjukkan pada Gambar 4. 13.

```
void welcomeLCD(){
    lcd.setCursor(0, 0);
    lcd.print("MONITORING BEBAN");
    lcd.setCursor(0, 1);
    lcd.print("    TRAFO-1.");
    delay(2000);
    lcd.clear();
}
```

Gambar 4. 13 Program Tampilan Awal LCD Transmitter

Penampilan data sensor dilakukan melalui fungsi `showDataOnLCD()`. Fungsi ini menerima nilai menu sebagai penentu halaman tampilan pada LCD. Setiap halaman menampilkan dua parameter pengukuran. Halaman pertama menampilkan tegangan dan arus. Halaman kedua menampilkan frekuensi dan faktor daya. Halaman ketiga menampilkan daya dan persentase pembebanan transformator. Program pembagian halaman tampilan LCD ditunjukkan pada Gambar 4. 14.

```
void showDataOnLCD(int menu){
```

```

switch(menu){
    case MENU_1:
        sprintf(buffer, "Volt: %0.2f V", voltage);
        sprintf(buffer2, "Amp: %0.2f A", current);
        break;
    case MENU_2:
        sprintf(buffer, "Freq: %0.2f Hz", frequency);
        sprintf(buffer2, "PF: %0.2f", pf);
        break;
    case MENU_3:
        sprintf(buffer, "Power: %0.2f W", power);
        sprintf(buffer2, "Load: %0.2f %%", loadPercentage);
        break;
}
}

```

Gambar 4. 14 Program Pembagian Halaman Tampilan LCD Transmitter

Pada program tersebut, perintah `switch(menu)` digunakan untuk memilih data yang akan ditampilkan. Variabel `buffer` digunakan untuk menyimpan teks pada baris pertama LCD. Variabel `buffer2` digunakan untuk menyimpan teks pada baris kedua LCD. Fungsi `sprintf()` digunakan untuk mengubah nilai numerik menjadi format teks dengan dua angka di belakang koma. Format ini membuat tampilan data lebih rapi dan mudah dibaca.

Data yang telah disusun dalam bentuk teks kemudian ditampilkan pada LCD. Perintah `lcd.setCursor(0, 0)` digunakan untuk mengatur posisi tampilan pada baris pertama. Perintah `lcd.setCursor(0, 1)` digunakan untuk mengatur posisi tampilan pada baris kedua. Program penampilan data pada baris LCD ditunjukkan pada Gambar 4. 15.

```

lcd.setCursor(0, 0);
lcd.print(buffer);
lcd.setCursor(0, 1);
lcd.print(buffer2);

```

Gambar 4. 15 Program Penampilan Data pada LCD Transmitter

Proses penampilan data LCD dijalankan secara berulang pada `taskDisplay`.

Pada bagian ini, ESP32 mengambil data sensor yang telah dibaca, lalu menampilkannya sesuai halaman menu aktif. Akses data dikendalikan menggunakan dataMutex agar data tidak dibaca bersamaan oleh proses lain. Program pembaruan tampilan LCD secara berulang ditunjukkan pada Gambar 4. 16.

```
if (xSemaphoreTake(dataMutex, portMAX_DELAY)){
    showDataOnLCD(menuPage);
    xSemaphoreGive(dataMutex);
}

vTaskDelay(pdMS_TO_TICKS(50));
```

Gambar 4. 16 Program Pembaruan Tampilan LCD Transmitter

Berdasarkan alur program tersebut, LCD transmitter bekerja sebagai tampilan lokal untuk hasil pembacaan sensor PZEM-004T. Data sensor dibaca oleh ESP32, disimpan pada variabel utama, kemudian ditampilkan berdasarkan halaman menu yang sedang aktif. Dengan mekanisme ini, proses pemantauan tegangan, arus, daya, frekuensi, faktor daya, dan persentase beban dapat dilakukan langsung pada unit transmitter.

4.2.1.6 Push Button Sebagai Pengaturan Antarmuka LCD Transmitter

Push button digunakan sebagai media interaksi antara pengguna dan unit transmitter. Komponen ini berfungsi untuk mengubah halaman tampilan LCD sehingga beberapa parameter hasil pengukuran dapat ditampilkan secara bergantian. Dengan penggunaan push button, seluruh data pengukuran tidak perlu ditampilkan dalam satu halaman, sehingga informasi yang ditampilkan pada LCD menjadi lebih jelas dan mudah dibaca. Program inisialisasi push button transmitter ditunjukkan pada Gambar 4. 17.

```
#define BUTTON_PIN 32

void buttonBegin(){
    pinMode(BUTTON_PIN, INPUT);
}
```

Gambar 4. 17 Program Inisialisasi Push Button Transmitter

Pada program tersebut, push button dihubungkan ke GPIO 32 ESP32. Fungsi pinMode() digunakan untuk mengatur pin sebagai masukan (input). Setelah proses inisialisasi selesai, ESP32 dapat membaca kondisi logika pin untuk mengetahui apakah tombol sedang ditekan atau tidak. Pembacaan kondisi push button dilakukan secara terus-menerus pada task tampilan LCD.

Proses pembacaan tombol dilakukan menggunakan fungsi digitalRead(). Saat tombol ditekan, ESP32 akan mendeteksi perubahan logika pada pin masukan. Program kemudian menunggu hingga tombol dilepaskan sebelum melanjutkan proses berikutnya. Metode ini digunakan agar satu kali penekanan tombol hanya menghasilkan satu kali perpindahan halaman LCD. Program pembacaan push button ditunjukkan pada Gambar 4. 18.

```
if (digitalRead(BUTTON_PIN) == LOW){
    while (digitalRead(BUTTON_PIN) == LOW){
        vTaskDelay(pdMS_TO_TICKS(5));
    }
    Serial.println("Button Pressed");
    menuPage++;
    if (menuPage >= TOTAL_MENU) menuPage = MENU_1;
    lcd.clear();
}
```

Gambar 4. 18 Program Pembacaan Push Button Transmitter

Ketika tombol ditekan, nilai variabel menuPage akan bertambah satu. Variabel ini digunakan sebagai penentu halaman yang akan ditampilkan pada LCD. Apabila nilai menuPage telah mencapai jumlah menu maksimum, program akan mengembalikannya ke halaman pertama. Dengan mekanisme tersebut, perpindahan tampilan berlangsung secara berurutan dan berulang. Alur perpindahan tampilan LCD ditunjukkan pada Gambar 4. 19.

MENU_1 → MENU_2 → MENU_3 → MENU_1

Gambar 4. 19 Alur Perpindahan Menu LCD Transmitter

Setiap halaman memiliki informasi yang berbeda. Halaman pertama digunakan untuk menampilkan tegangan dan arus. Halaman kedua digunakan untuk menampilkan frekuensi dan faktor daya. Halaman ketiga digunakan untuk

menampilkan daya dan persentase pembebanan transformator. Pemilihan halaman dilakukan melalui fungsi `showDataOnLCD()` berdasarkan nilai `menuPage` yang aktif. Program pemilihan halaman LCD ditunjukkan pada Gambar 4. 20.

```
switch(menu){
  case MENU_1:
    // Voltase dan Arus
    break;
  case MENU_2:
    // Frekuensi dan Faktor Daya
    break;
  case MENU_3:
    // Daya dan Persentase Beban
    break;
}
```

Gambar 4. 20 Program Pemilihan Halaman LCD Transmitter

Berdasarkan alur program tersebut, push button berfungsi sebagai pengatur antarmuka LCD pada unit transmitter. Setiap penekanan tombol akan mengubah halaman tampilan sehingga pengguna dapat melihat parameter pengukuran yang berbeda tanpa mengganggu proses pembacaan sensor maupun komunikasi LoRa. Dengan mekanisme ini, seluruh informasi hasil monitoring dapat ditampilkan secara bergantian pada LCD 16×2 secara lebih terorganisir.

4.2.1.7 Pengiriman Data Melalui LoRa Ra02-SX1278 Transmitter

Modul LoRa Ra-02 SX1278 digunakan sebagai media komunikasi nirkabel antara unit transmitter dan unit receiver. Pada sistem yang dirancang, komunikasi dilakukan menggunakan metode polling, yaitu receiver mengirimkan permintaan data kepada transmitter. Setelah permintaan diterima, transmitter akan mengirimkan data hasil pembacaan sensor yang tersimpan pada ESP32. Metode ini digunakan untuk menghindari pengiriman data yang tidak teratur serta mengurangi kemungkinan terjadinya tabrakan data apabila sistem dikembangkan menggunakan lebih dari satu transmitter.

Sebelum komunikasi dilakukan, modul LoRa harus diinisialisasi terlebih dahulu. Proses inisialisasi mencakup pengaturan pin komunikasi SPI dan penetapan

frekuensi kerja modul. Pada penelitian ini, LoRa dioperasikan pada frekuensi 433 MHz. Apabila proses inisialisasi gagal, sistem akan menghentikan program dan menampilkan pesan kesalahan pada LCD. Program inisialisasi LoRa transmitter ditunjukkan pada Gambar 4. 21

```
#define SS 5
#define RST 14
#define DIO0 26
const long frequencyLora = 433E6;

void setupLoRa(){
    LoRa.setPins(SS, RST, DIO0);

    if (!LoRa.begin(frequencyLora)) {
        while (true);
    }
}
```

Gambar 4. 21 Program Inisialisasi LoRa Ra-02 SX1278 Transmitter

Pada program tersebut, fungsi `LoRa.setPins()` digunakan untuk menentukan pin yang digunakan oleh modul LoRa. Fungsi `LoRa.begin()` digunakan untuk mengaktifkan modul sesuai frekuensi kerja yang telah ditentukan. Apabila proses inisialisasi berhasil, transmitter siap menerima permintaan data dari receiver melalui jaringan LoRa.

Setelah LoRa aktif, transmitter akan berada pada kondisi siaga sambil menunggu paket data masuk dari receiver. Pemeriksaan paket dilakukan menggunakan fungsi `LoRa.parsePacket()`. Apabila terdapat paket yang diterima, ESP32 akan membaca seluruh isi data yang dikirimkan receiver. Program pemeriksaan paket LoRa ditunjukkan pada Gambar 4. 22.

```
int packetSize = LoRa.parsePacket();

if (packetSize) {
    String incoming = "";

    while (LoRa.available()) {
```

```

        incoming += (char)LoRa.read();
    }
}

```

Gambar 4. 22 Program Penerimaan Paket LoRa Transmitter

Data yang diterima kemudian diperiksa menggunakan fungsi `handleRequest()`. Pada tahap ini, ESP32 akan membandingkan data permintaan dari receiver dengan identitas transmitter yang tersimpan pada variabel `TRAFO_ID`. Pengiriman data hanya dilakukan apabila identitas yang diterima sesuai dengan identitas transmitter. Program identifikasi permintaan receiver ditunjukkan pada Gambar 4. 23.

```

#define TRAFO_ID    "TRAFO1"

void handleRequest(String incoming){
    if (incoming == TRAFO_ID) {
        String data =
        sendDataToReceiver();
        lcd.clear();
        lcd.setCursor(0, 0);
        lcd.print("  LORA SENDING");
        sendLoRa(data);
    }
}

```

Gambar 4. 23 Program Validasi Permintaan Data Receiver

Apabila identitas sesuai, transmitter akan membentuk paket data yang berisi hasil pembacaan sensor PZEM-004T. Data yang dikirim meliputi tegangan, arus, daya, frekuensi, faktor daya, dan persentase pembebanan transformator. Seluruh data digabungkan dalam satu string menggunakan tanda koma sebagai pemisah antarparameter. Program pembentukan paket data ditunjukkan pada Gambar 4. 24.

```

String sendDataToReceiver(){
    String data = "DATA,";
    data += String(voltage,2) + ",";
    data += String(current,2) + ",";
    data += String(power,2) + ",";
}

```

```

        data += String(frequency,2) + ",";
        data += String(pf,2) + ",";
        data += String(loadPercentage,2);
        return data;
    }

```

Gambar 4. 24 Program Pembentukan Paket Data LoRa Transmitter

Berdasarkan program tersebut, paket data yang dikirim memiliki format yang ditunjukkan pada Gambar 4. 25.

DATA,220.50,1.25,275.60,50.00,0.98,25.00.

Gambar 4. 25 Format Paket Data yang Dikirim LoRa Transmitter

Format tersebut memudahkan receiver dalam memisahkan setiap parameter yang diterima. Parameter pertama menunjukkan tegangan, parameter kedua menunjukkan arus, parameter ketiga menunjukkan daya, parameter keempat menunjukkan frekuensi, parameter kelima menunjukkan faktor daya, dan parameter terakhir menunjukkan persentase pembebanan transformator.

Setelah paket data selesai dibentuk, transmitter mengirimkan data melalui LoRa menggunakan fungsi `LoRa.beginPacket()`, `LoRa.print()`, dan `LoRa.endPacket()`. Program pengiriman data LoRa ditunjukkan pada Gambar 4. 26.

```

void sendLoRa(String dataToSend){
    LoRa.beginPacket();
    LoRa.print(dataToSend);
    LoRa.endPacket();
    Serial.println("Sending: " + dataToSend);
}

```

Gambar 4. 26 Program Pengiriman Data Melalui LoRa

Alur pengiriman data pada transmitter dimulai ketika receiver mengirimkan identitas perangkat yang ingin dibaca. ESP32 kemudian memeriksa kesesuaian identitas, membaca data sensor terbaru, membentuk paket data, dan mengirimkan hasil pengukuran melalui LoRa. Setelah proses pengiriman selesai, transmitter kembali ke kondisi siaga sambil menunggu permintaan data berikutnya. Dengan mekanisme tersebut, komunikasi antara transmitter dan receiver dapat berlangsung secara teratur dan sesuai dengan metode polling yang digunakan pada sistem

monitoring beban transformator distribusi.

4.2.2 Pembuatan Perangkat Lunak Receiver

Pembuatan perangkat lunak pada sisi receiver difokuskan pada proses penerimaan data hasil monitoring dari unit transmitter, pengolahan data, penyimpanan data ke Google Spreadsheet, serta pengiriman notifikasi melalui Telegram Bot. Program dikembangkan menggunakan Arduino IDE dengan ESP32 sebagai pengendali utama. Selain berfungsi sebagai penerima data LoRa, ESP32 pada sisi receiver juga bertugas menyediakan konektivitas internet melalui WiFi sehingga data hasil monitoring dapat diakses secara daring.

Perangkat lunak receiver dirancang menggunakan metode polling, yaitu receiver secara aktif meminta data kepada transmitter berdasarkan interval waktu yang telah ditentukan. Data yang diterima kemudian diproses untuk menghasilkan informasi kondisi pembebanan transformator. Selanjutnya data tersebut dilengkapi dengan informasi waktu hasil sinkronisasi NTP sebelum dikirim ke Google Spreadsheet dan digunakan sebagai dasar pengiriman notifikasi Telegram. Dengan mekanisme tersebut, proses pemantauan transformator dapat dilakukan secara otomatis tanpa memerlukan intervensi pengguna secara langsung.

4.2.2.1 Pemrograman Arduino IDE Receiver

Pemrograman perangkat lunak receiver dilakukan menggunakan Arduino IDE dengan bahasa pemrograman C++. Program ini berfungsi mengatur seluruh proses komunikasi, pengolahan data, pencatatan data, dan pengiriman notifikasi pada sistem monitoring beban transformator distribusi. Struktur program disusun secara modular untuk memudahkan proses pengembangan, pengujian, dan pemeliharaan sistem. Setiap fungsi dibuat secara terpisah sesuai dengan tugas yang dijalankan pada receiver.

Secara umum, perangkat lunak receiver terdiri atas beberapa bagian utama. Bagian pertama digunakan untuk mengatur konfigurasi awal ESP32 dan modul LoRa. Bagian kedua digunakan untuk menghubungkan perangkat ke jaringan WiFi dan melakukan sinkronisasi waktu menggunakan Network Time Protocol (NTP). Bagian ketiga digunakan untuk menerima data hasil monitoring dari transmitter melalui komunikasi LoRa. Bagian keempat digunakan untuk mengirim dan

menyimpan data ke Google Spreadsheet. Bagian terakhir digunakan untuk melakukan monitoring kondisi transformator dan mengirimkan notifikasi peringatan melalui Telegram Bot apabila terdeteksi kondisi abnormal.

Pembagian program yang dikembangkan pada perangkat lunak receiver meliputi beberapa bagian sebagai berikut:

1. Program untuk pengaturan port dan komunikasi serial ESP32.
2. Program untuk inisialisasi pin dan konfigurasi awal ESP32 receiver.
3. Program untuk koneksi jaringan WiFi.
4. Program untuk sinkronisasi waktu menggunakan Network Time Protocol (NTP).
5. Program untuk komunikasi dan penerimaan data melalui LoRa Ra-02 SX1278.
6. Program untuk pengiriman dan penyimpanan data ke Google Spreadsheet.
7. Program untuk monitoring parameter transformator dan pengiriman notifikasi Telegram.
8. Program untuk menerima perintah dari pengguna melalui Telegram Bot.
9. Program multitasking menggunakan FreeRTOS untuk menjalankan proses polling dan monitoring secara bersamaan.

Dengan pembagian tersebut, setiap fungsi pada receiver dapat dijalankan secara terstruktur sesuai tugas masing-masing. Pendekatan ini memungkinkan proses penerimaan data, penyimpanan data, dan pengiriman notifikasi berlangsung secara berkelanjutan selama sistem memperoleh catu daya dan koneksi internet tersedia.

4.2.2.2 Pengaturan Port pada Arduino IDE Receiver

Pengaturan port pada Arduino IDE dilakukan sebelum program receiver diunggah ke mikrokontroler ESP32. Tahap ini bertujuan memastikan perangkat ESP32 dapat dikenali oleh komputer dan menerima program dengan baik. Selain sebagai media unggah program, port serial juga digunakan untuk menampilkan proses kerja sistem selama pengujian. Informasi yang ditampilkan meliputi status inisialisasi LoRa, koneksi WiFi, sinkronisasi waktu NTP, pengiriman data ke Google Spreadsheet, hingga proses komunikasi dengan Telegram Bot.

Pada program receiver, komunikasi serial diaktifkan melalui fungsi `Serial.begin(115200)`. Nilai baud rate sebesar 115200 bps digunakan agar proses

pengiriman data debugging berlangsung lebih cepat dan stabil. Serial monitor digunakan untuk menampilkan kondisi sistem secara real-time sehingga proses identifikasi kesalahan dapat dilakukan dengan lebih mudah. Program pengaturan komunikasi serial pada receiver ditunjukkan pada Gambar 4. 27.

```
void setup() {
  Serial.begin(115200);
  setupLoRa();
  secured_client.setCACert(TELEGRAM_CERTIFICATE_ROOT);
  setupWiFi();
  configTime(gmtOffset_sec, daylightOffset_sec, ntpServer);
  GsheetSetup();
  dataReceived = false;
}
```

Gambar 4. 27 Program Pengaturan Komunikasi Serial Receiver

Melalui serial monitor, pengguna dapat memantau seluruh proses kerja receiver mulai dari proses startup hingga sistem beroperasi normal. Informasi yang ditampilkan juga digunakan untuk memastikan setiap perangkat dan layanan eksternal telah terhubung dengan benar sebelum proses polling data dimulai. Dengan demikian, pengaturan port dan komunikasi serial menjadi tahap awal yang diperlukan sebelum receiver digunakan untuk menerima data dari transmitter.

4.2.2.3 Inisialisasi Pin dan Konfigurasi Awal ESP32 Receiver

ESP32 pada sisi receiver berfungsi sebagai pusat pengendali sistem monitoring. Mikrokontroler ini bertanggung jawab menerima data dari transmitter melalui LoRa, menghubungkan sistem ke jaringan internet, melakukan sinkronisasi waktu, mengirim data ke Google Spreadsheet, dan mengirim notifikasi melalui Telegram Bot. Sebelum seluruh fungsi tersebut dijalankan, dilakukan proses inisialisasi pin dan konfigurasi awal perangkat.

Pada tahap awal program, beberapa library diaktifkan sesuai kebutuhan sistem. Library LoRa.h digunakan untuk komunikasi LoRa, WiFi.h digunakan untuk koneksi internet, ESP_Google_Sheet_Client.h digunakan untuk komunikasi dengan Google Spreadsheet, dan UniversalTelegramBot.h digunakan untuk

mengirim maupun menerima pesan Telegram. Selain itu, library time.h digunakan untuk memperoleh data waktu dari server NTP. Program pemanggilan library receiver ditunjukkan pada Gambar 4. 28.

```
#include <Arduino.h>
#include <SPI.h>
#include <LoRa.h>
#include <ESP_Google_Sheet_Client.h>
#include <WiFiClientSecure.h>
#include <UniversalTelegramBot.h>
#include <ArduinoJson.h>
#include <time.h>
#include <WiFi.h>
#include <GS_SDHelper.h>
```

Gambar 4. 28 Program Pemanggilan Library Receiver

Konfigurasi pin LoRa dilakukan dengan mendefinisikan pin SPI yang digunakan oleh modul Ra-02 SX1278. Selain itu, ditentukan pula frekuensi operasi sebesar 433 MHz yang digunakan untuk komunikasi antara transmitter dan receiver. Program konfigurasi pin LoRa ditunjukkan pada Gambar 4. 29.

```
#define SS 5
#define RST 14
#define DIO0 26
const long frequencyLora = 433E6;
```

Gambar 4. 29 Program Inisialisasi Pin LoRa Receiver

Selain konfigurasi LoRa, program juga mendefinisikan identitas transmitter yang akan dipanggil selama proses polling. Identitas ini digunakan receiver saat mengirim permintaan data melalui jaringan LoRa. Dengan metode ini, receiver hanya akan menerima data dari transmitter yang memiliki identitas sesuai. Program identitas transmitter ditunjukkan pada Gambar 4. 30.

```
#define TRAF0_ID_1 "TRAF01"
```

Gambar 4. 30 Program Identitas Transmitter pada Receiver

Sistem juga menggunakan beberapa objek utama yang berfungsi sebagai antarmuka komunikasi dengan layanan eksternal. Objek secured_client digunakan untuk komunikasi HTTPS, objek bot digunakan untuk Telegram Bot, sedangkan

objek `valueRange` dan `response` digunakan untuk proses pengiriman data ke Google Spreadsheet. Program deklarasi objek receiver ditunjukkan pada Gambar 4. 31.

```

FirebaseJson valueRange;
WiFiClientSecure secured_client;
UniversalTelegramBot bot(BOTtoken, secured_client);
FirebaseJson response;

```

Gambar 4. 31 Program Deklarasi Objek Sistem Receiver

Pada fungsi `setup()`, ESP32 melakukan konfigurasi awal seluruh perangkat dan layanan yang digunakan. Tahap ini dimulai dari inisialisasi LoRa, sertifikat Telegram, koneksi WiFi, sinkronisasi waktu NTP, dan konfigurasi Google Spreadsheet. Setelah seluruh proses berhasil dilakukan, sistem membuat dua task utama menggunakan FreeRTOS. Task pertama digunakan untuk proses polling data dan pengiriman ke Google Spreadsheet. Task kedua digunakan untuk monitoring Telegram dan pengiriman notifikasi peringatan. Program konfigurasi awal ESP32 receiver ditunjukkan pada Gambar 4. 32.

```

xTaskCreatePinnedToCore(
    taskPolling, "TaskPolling",
    8192,
    NULL,
    1,
    &taskPollingHandle,
    1
);

xTaskCreatePinnedToCore(
    taskTelegram,
    "TaskTelegram",
    8192,
    NULL,
    1,
    &taskTelegramHandle,
    0
);

```

Gambar 4. 32 Program Pembuatan Task pada ESP32 Receiver

Berdasarkan konfigurasi tersebut, receiver menjalankan dua proses utama secara bersamaan. Proses pertama bertugas melakukan polling data dari transmitter dan menyimpan hasil pengukuran ke Google Spreadsheet. Proses kedua bertugas memantau kondisi parameter trafo, mengirim notifikasi peringatan, serta merespons perintah yang dikirimkan melalui Telegram Bot. Pembagian tugas ini membuat sistem dapat melakukan monitoring secara berkelanjutan tanpa mengganggu proses komunikasi data antarperangkat.

4.2.2.4 Pembacaan Wifi Oleh ESP32 Receiver

Koneksi WiFi pada unit receiver digunakan sebagai media komunikasi antara ESP32 dengan layanan cloud yang meliputi Google Spreadsheet, server NTP, dan Telegram Bot. Berbeda dengan unit transmitter yang hanya berkomunikasi melalui LoRa, unit receiver harus terhubung ke jaringan internet agar dapat melakukan sinkronisasi waktu, penyimpanan data hasil monitoring, dan pengiriman notifikasi kepada pengguna. Oleh karena itu, proses pembacaan dan koneksi WiFi menjadi salah satu tahapan penting pada perangkat lunak receiver.

Sebelum proses koneksi dilakukan, program mendefinisikan nama jaringan (SSID) dan kata sandi WiFi yang akan digunakan. Informasi tersebut digunakan oleh ESP32 untuk terhubung ke access point yang tersedia. Program konfigurasi identitas jaringan WiFi ditunjukkan pada Gambar 4. 33.

```
#define WIFI_SSID "NAMA_WIFI"
#define WIFI_PASSWD "PASSWORD_WIFI"
```

Gambar 4. 33 Program Konfigurasi Identitas WiFi

Proses koneksi WiFi dilakukan melalui fungsi `setupWiFi()`. Pada fungsi ini, ESP32 terlebih dahulu mengaktifkan fitur reconnect otomatis menggunakan perintah `WiFi.setAutoReconnect(true)`. Fitur tersebut digunakan agar perangkat dapat mencoba terhubung kembali secara otomatis apabila koneksi internet terputus selama sistem beroperasi. Selanjutnya, ESP32 mulai melakukan koneksi menggunakan fungsi `WiFi.begin()`. Program koneksi WiFi ditunjukkan pada Gambar 4. 34.

```
void setupWiFi(){
```

```

    WiFi.setAutoReconnect(true);
    WiFi.begin(WIFI_SSID, WIFI_PASSWD)
}

```

Gambar 4. 34 Program Koneksi Awal WiFi

Setelah perintah koneksi dijalankan, ESP32 melakukan pemeriksaan status koneksi menggunakan fungsi `WiFi.status()`. Selama belum memperoleh status `WL_CONNECTED`, sistem akan terus melakukan pengecekan secara berkala. Pada proses ini, informasi status koneksi ditampilkan melalui serial monitor sehingga pengguna dapat mengetahui apakah perangkat berhasil terhubung ke jaringan atau tidak. Program pemeriksaan status koneksi WiFi ditunjukkan pada Gambar 4. 35.

```

while (WiFi.status() != WL_CONNECTED) {
    Serial.print(".");
    delay(1000);
    disconnectWifi++;
    if (disconnectWifi > 3) break;
}

```

Gambar 4. 35 Program Pemeriksaan Status Koneksi WiFi

Pada program tersebut, variabel `disconnectWifi` digunakan sebagai pencacah jumlah percobaan koneksi. Apabila ESP32 gagal terhubung setelah beberapa kali percobaan, program akan menghentikan proses menunggu dan menampilkan informasi kegagalan koneksi. Sebaliknya, apabila koneksi berhasil dilakukan, sistem akan menampilkan alamat IP yang diperoleh dari jaringan WiFi. Informasi ini dapat digunakan untuk memastikan bahwa perangkat telah terhubung ke jaringan internet dengan benar. Program penampilan status koneksi WiFi ditunjukkan pada Gambar 4. 36.

```

if(disconnectWifi > 3 && WiFi.status() != WL_CONNECTED){
    Serial.println();
    Serial.println("Failed to connect to Wi-Fi");
}else if(WiFi.status() == WL_CONNECTED){
    Serial.println();
    Serial.println("Wi-Fi connected successfully");
    Serial.print("Connected with IP: ");
}

```

```

Serial.println(WiFi.localIP());
delay(1000);
}

```

Gambar 4. 36 Program Status Koneksi WiFi

Setelah koneksi internet tersedia, sistem dapat mengakses layanan Google Spreadsheet dan Telegram yang digunakan pada penelitian ini. Koneksi WiFi juga digunakan untuk proses autentikasi akun Google Spreadsheet dan komunikasi HTTPS dengan Telegram Bot. Dengan demikian, seluruh fungsi Internet of Things pada sistem monitoring dapat berjalan melalui satu koneksi internet yang disediakan oleh ESP32 receiver.

4.2.2.5 Sinkronisasi Waktu Menggunakan Network Time Protocol (NTP)

Sinkronisasi waktu menggunakan Network Time Protocol (NTP) diterapkan pada unit receiver untuk memperoleh informasi tanggal dan waktu secara otomatis melalui jaringan internet. Informasi waktu ini digunakan sebagai penanda waktu (timestamp) pada setiap data hasil monitoring yang disimpan ke Google Spreadsheet maupun pada pesan notifikasi yang dikirimkan melalui Telegram. Penggunaan NTP dipilih karena mampu memberikan waktu yang selalu terbaru tanpa memerlukan modul RTC tambahan pada perangkat keras.

Konfigurasi NTP diawali dengan penentuan alamat server waktu, perbedaan zona waktu terhadap UTC, dan pengaturan daylight saving time. Pada penelitian ini digunakan server `id.pool.ntp.org` sebagai sumber referensi waktu. Nilai `gmtOffset_sec` diatur sebesar 7 jam untuk menyesuaikan waktu Indonesia bagian barat. Program konfigurasi server NTP ditunjukkan pada Gambar 4. 37.

```

const char* ntpServer = "id.pool.ntp.org";
const long  gmtOffset_sec = 7 * 3600;
const int   daylightOffset_sec = 0;

```

Gambar 4. 37 Program Konfigurasi Server NTP

Setelah parameter waktu ditentukan, ESP32 melakukan sinkronisasi dengan server NTP menggunakan fungsi `configTime()`. Fungsi ini dipanggil pada bagian `setup()` setelah koneksi WiFi berhasil dilakukan. Melalui proses ini, ESP32 memperoleh informasi waktu aktual yang disimpan pada sistem internal mikrokontroler. Program sinkronisasi waktu menggunakan NTP ditunjukkan pada

Gambar 4. 38.

```

    setupWiFi();
    configTime(
        gmtOffset_sec,
        daylightOffset_sec,
        ntpServer
    );

```

Gambar 4. 38 Program Sinkronisasi Waktu NTP

Setelah proses sinkronisasi dilakukan, sistem memeriksa ketersediaan data waktu menggunakan fungsi waitNtpTime(). Fungsi ini bekerja dengan memanggil getLocalTime() beberapa kali hingga waktu berhasil diperoleh dari server NTP. Selama proses berlangsung, status sinkronisasi ditampilkan pada serial monitor. Program pemeriksaan sinkronisasi waktu ditunjukkan pada Gambar 4. 39.

```

void waitNtpTime(){
    Serial.print("Menunggu NTP");
    struct tm timeinfo;
    for (int i = 0; i < 4; i++) {
        if (getLocalTime(&timeinfo)) break;
        Serial.print('.');
        delay(500);
    }
    Serial.println();
    delay(500);
}

```

Gambar 4. 39 Program Pemeriksaan Sinkronisasi Waktu NTP

Data waktu yang telah diperoleh kemudian diubah menjadi format epoch time melalui fungsi getEpochTime(). Nilai epoch time merupakan representasi waktu dalam bentuk detik sejak 1 Januari 1970 dan digunakan sebagai dasar pengolahan waktu pada sistem. Program pengambilan epoch time ditunjukkan pada Gambar 4. 40.

```

unsigned long getEpochTime(){
    struct tm timeinfo;
    if (!getLocalTime(&timeinfo)) return 0;
}

```

```

        return mktime(&timeinfo);
    }

```

Gambar 4. 40 Program Pengambilan Epoch Time

Nilai epoch time selanjutnya dikonversi menjadi format tanggal dan jam yang lebih mudah dibaca menggunakan fungsi `strftime()`. Hasil konversi disimpan pada variabel `waktuStr` sehingga dapat digunakan pada proses pencatatan data maupun pengiriman notifikasi. Format yang digunakan pada penelitian ini adalah DD/MM/YYYY HH:MM. Program konversi waktu menjadi format tanggal dan jam ditunjukkan pada Gambar 4. 41.

```

void timeToString(){
    unsigned long epochNow = getEpochTime();
    if (epochNow != 0) {
        time_t rawTime = epochNow;
        struct tm *timeinfo = localtime(&rawTime);

        strftime(
            waktuStr,
            sizeof(waktuStr),
            "%d/%m/%Y %H:%M",
            timeinfo
        );

        Serial.println(waktuStr);
    }
}

```

Gambar 4. 41 Program Konversi Data Waktu

Data waktu yang tersimpan pada variabel `waktuStr` digunakan sebagai timestamp pada setiap proses penyimpanan data ke Google Spreadsheet. Dengan adanya timestamp, setiap hasil monitoring dapat diketahui waktu pengambilannya secara tepat. Program penggunaan data waktu pada Google Spreadsheet ditunjukkan pada Gambar 4. 42.

```

timeToString();

```

```
valueRange.set("values/[0]/[0]", waktuStr);
```

Gambar 4. 42 Program Penggunaan Timestamp pada Google Spreadsheet

Selain digunakan untuk pencatatan data, informasi waktu juga dimanfaatkan pada pesan peringatan Telegram. Waktu pengiriman notifikasi dicantumkan pada isi pesan sehingga pengguna dapat mengetahui kapan kondisi gangguan atau pembebanan berlebih terjadi. Dengan demikian, sinkronisasi waktu menggunakan NTP menjadi bagian penting dalam sistem monitoring karena memastikan seluruh data dan notifikasi memiliki informasi waktu yang akurat dan seragam.

4.2.2.6 Penerimaan Data Melalui LoRa Ra02-SX1278 Receiver

Komunikasi LoRa pada sisi receiver digunakan untuk menerima data hasil pengukuran yang dikirimkan oleh unit transmitter. Sistem komunikasi yang digunakan menerapkan metode polling, yaitu receiver secara aktif mengirimkan permintaan data kepada transmitter berdasarkan identitas perangkat yang telah ditentukan. Setelah permintaan diterima oleh transmitter, data hasil pembacaan sensor PZEM-004T akan dikirimkan kembali menuju receiver melalui jaringan LoRa. Metode ini digunakan untuk mengontrol proses komunikasi agar berlangsung lebih teratur dan mengurangi kemungkinan terjadinya tabrakan data pada sistem dengan lebih dari satu transmitter.

Sebelum komunikasi dilakukan, modul LoRa Ra-02 SX1278 pada receiver harus diinisialisasi terlebih dahulu. Proses inisialisasi dilakukan dengan menentukan pin SPI yang digunakan serta frekuensi kerja modul sebesar 433 MHz. Setelah proses inisialisasi berhasil dilakukan, receiver siap melakukan komunikasi dengan transmitter. Program inisialisasi LoRa receiver ditunjukkan pada Gambar 4. 43.

```
void setupLoRa(){
    Serial.println("Preparing LoRa SX1278");
    delay(1000);
    LoRa.setPins(SS, RST, DIO0);
    if (!LoRa.begin(frequencyLora)) {
```

```

        Serial.println("LoRa init failed.");
        while (true);
    }
    Serial.println("LoRa init success!");
    delay(1000);
}

```

Gambar 4. 43 Program Inisialisasi LoRa Receiver

Setelah modul LoRa aktif, receiver akan memulai proses polling dengan mengirimkan identitas transmitter yang ingin dibaca. Pada penelitian ini digunakan identitas TRAF01 sebagai alamat transmitter. Identitas tersebut dikirim melalui LoRa menggunakan fungsi `sendRequest()`. Program pengiriman permintaan data ditunjukkan pada Gambar 4. 44.

```

#define TRAF0_ID_1    "TRAF01"

void sendRequest(String trafoId){
    LoRa.beginPacket();
    LoRa.print(trafoId);
    LoRa.endPacket();
    Serial.println("Request dikirim");
}

```

Gambar 4. 44 Program Pengiriman Permintaan Data ke Transmitter

Setelah permintaan data dikirimkan, receiver akan menunggu balasan dari transmitter. Penerimaan data dilakukan menggunakan fungsi `receiveData()`. Fungsi ini memanfaatkan `LoRa.parsePacket()` untuk memeriksa apakah terdapat paket data yang masuk. Apabila paket diterima, seluruh isi data akan dibaca dan disimpan ke dalam variabel string. Program penerimaan data LoRa ditunjukkan pada Gambar 4. 45.

```

String receiveData(int timeout) {
    long start = millis();
    while (millis() - start < timeout) {
        int packetSize = LoRa.parsePacket();
        if (packetSize) {
            String data = "";

```

```

        while (LoRa.available()) {
            data += (char)LoRa.read();
        }
        Serial.println("Receive Success.");
        return data;
    }
}

return "";
}

```

Gambar 4. 45 Program Penerimaan Data LoRa

Pada program tersebut, receiver memberikan waktu tunggu maksimum selama 5000 ms. Apabila dalam rentang waktu tersebut tidak terdapat balasan dari transmitter, sistem akan menganggap proses polling gagal dan menampilkan pesan timeout pada serial monitor. Sebaliknya, apabila data berhasil diterima, proses selanjutnya adalah melakukan ekstraksi setiap parameter yang terdapat pada paket data. Program proses polling ditunjukkan pada Gambar 4. 46.

```

void startPollingData() {
    sendRequest(TRAFO_ID_1);
    poolingSuccess = false;
    String data = receiveData(5000);
    if (data != "") {
        parseData(data);
        if(dataReceived == false) dataReceived = true;
        poolingSuccess = true;
    } else {
        Serial.println("Timeout (tidak ada respon)");
        poolingSuccess = false;
    }
    delay(1000);
}

```

Gambar 4. 46 Program Polling Data Receiver

Data yang diterima dari transmitter memiliki format string yang terdiri atas

beberapa parameter yang dipisahkan menggunakan tanda koma. Format data yang diterima ditunjukkan pada Gambar 4. 47.

DATA,220.50,1.25,275.60,50.00,0.98,25.00

Gambar 4. 47 Format Paket Data yang Diterima LoRa Receiver

Parameter tersebut secara berurutan merepresentasikan nilai tegangan, arus, daya, frekuensi, faktor daya, dan persentase pembebanan transformator. Proses pemisahan data dilakukan menggunakan fungsi `parseData()`. Program pemisahan data hasil penerimaan LoRa ditunjukkan pada Gambar 4. 48.

```
voltage_1 = data.substring(idx1 + 1, idx2).toFloat();
current_1 = data.substring(idx2 + 1, idx3).toFloat();
power_1 = data.substring(idx3 + 1, idx4).toFloat();
frequency_1 = data.substring(idx4 + 1, idx5).toFloat();
pf_1 = data.substring(idx5 + 1, idx6).toFloat();
loadPercentage_1 = data.substring(idx6 + 1).toFloat();
```

Gambar 4. 48 Program Ekstraksi Data Hasil Penerimaan LoRa Receiver

Setelah proses ekstraksi selesai, setiap parameter disimpan ke dalam variabel yang sesuai dan ditampilkan pada serial monitor untuk proses verifikasi. Data tersebut kemudian digunakan sebagai sumber informasi utama untuk pencatatan ke Google Spreadsheet dan analisis kondisi transformator. Program penampilan hasil penerimaan data ditunjukkan pada Gambar 4. 49.

```
Serial.printf("Voltage: %0.2f V\n", voltage_1);
Serial.printf("Current: %0.2f A\n", current_1);
Serial.printf("Power: %0.2f W\n", power_1);
Serial.printf("Frequency: %0.2f Hz\n", frequency_1);
Serial.printf("Power Factor: %0.2f\n", pf_1);
Serial.printf("Load Percentage: %0.2f%%\n", loadPercentage_1);
```

Gambar 4. 49 Program Verifikasi Data Hasil Penerimaan LoRa Receiver

Berdasarkan alur program tersebut, receiver terlebih dahulu mengirimkan permintaan data kepada transmitter, menerima paket data melalui LoRa, melakukan pemisahan setiap parameter pengukuran, kemudian menyimpan hasilnya ke dalam variabel sistem. Data yang telah diterima selanjutnya digunakan pada proses penyimpanan ke Google Spreadsheet dan pengiriman notifikasi Telegram. Dengan

mekanisme ini, receiver dapat memperoleh data monitoring transformator secara periodik sesuai interval polling yang telah ditentukan.

4.2.2.7 Pengiriman dan Penyimpanan Data ke Google Spreadsheets

Google Spreadsheet digunakan sebagai media penyimpanan data hasil monitoring beban transformator. Platform ini dipilih karena dapat diakses secara daring, mendukung pencatatan data secara real-time, serta tidak memerlukan server lokal tambahan. Pada sistem yang dirancang, seluruh data hasil pengukuran yang diterima oleh receiver akan dikirim dan disimpan secara otomatis ke Google Spreadsheet melalui koneksi internet yang tersedia pada ESP32.

Sebelum proses pengiriman data dilakukan, sistem harus melakukan autentikasi ke layanan Google Spreadsheet. Tahap ini menggunakan informasi akun layanan (service account) yang terdiri atas PROJECT_ID, CLIENT_EMAIL, dan PRIVATE_KEY. Data tersebut digunakan oleh library ESP Google Sheet Client untuk memperoleh hak akses ke spreadsheet yang telah ditentukan. Program konfigurasi autentikasi Google Spreadsheet ditunjukkan pada Gambar 4. 50.

```
#define PROJECT_ID "monitoring-trafo-499316"
#define CLIENT_EMAIL "monitoring-trafo@monitoring-trafo-
499316.iam.gserviceaccount.com"
const char PRIVATE_KEY[] PROGMEM = "-----BEGIN PRIVATE KEY-----
\nPRIVATE_KEY\n-----END PRIVATE KEY-----\n";
const char spreadsheetId[] =
"1LYxpLQVrhSDB4nqQ7RUwg3UJukk2JdIPz504LOevA8w";
```

Gambar 4. 50 Program Konfigurasi Autentikasi Google Spreadsheet

Setelah parameter autentikasi ditentukan, ESP32 melakukan inisialisasi layanan Google Spreadsheet melalui fungsi GsheetSetup(). Pada tahap ini, sistem mengaktifkan callback token, mengatur waktu refresh token, dan memulai proses autentikasi menggunakan kredensial yang telah disediakan. Program inisialisasi Google Spreadsheet ditunjukkan pada Gambar 4. 51.

```
void GsheetSetup(){
    GSheet.printf(
        "ESP Google Sheet Client v%s\n\n",
        ESP_GOOGLE_SHEET_CLIENT_VERSION
```

```

);
GSheet.setTokenCallback(tokenStatusCallback);
GSheet.setPrereshSeconds(10 * 60);
GSheet.begin(CLIENT_EMAIL, PROJECT_ID, PRIVATE_KEY);
delay(1000);
}

```

Gambar 4. 51 Program Inisialisasi Google Spreadsheet

Sebelum data dikirim, sistem terlebih dahulu memastikan bahwa layanan Google Spreadsheet telah siap digunakan. Pemeriksaan dilakukan menggunakan fungsi `GSheet.ready()`. Apabila proses autentikasi belum selesai, ESP32 akan menunggu hingga layanan siap menerima data. Langkah ini diperlukan agar pengiriman data tidak gagal akibat proses autentikasi yang belum selesai. Program pemeriksaan status Google Spreadsheet ditunjukkan pada Gambar 4. 52.

```

void sendToGoogleSheets(){
  if(WiFi.status() != WL_CONNECTED) WiFi.reconnect();
  while (1){
    bool ready = GSheet.ready();
    Serial.println("Waiting for GSheet to be ready...");
    if (ready) break;
  }
}

```

Gambar 4. 52 Program Pemeriksaan Status Google Spreadsheet

Data hasil monitoring kemudian disusun ke dalam objek `valueRange`. Data yang dikirim meliputi tanggal dan waktu pengambilan data, identitas transformator, tegangan, arus, daya, frekuensi, faktor daya, dan persentase pembebanan transformator. Setiap parameter ditempatkan pada kolom yang berbeda sehingga data lebih mudah dibaca dan dianalisis pada spreadsheet. Program penyusunan data monitoring ditunjukkan pada Gambar 4. 53.

```

valueRange.add("majorDimension", "COLUMNS");
valueRange.set("values/[0]/[0]", waktuStr);
valueRange.set("values/[1]/[0]", TRAFO_ID_1);
valueRange.set("values/[2]/[0]", voltage_1);
valueRange.set("values/[3]/[0]", current_1);
valueRange.set("values/[4]/[0]", power_1);

```

```

valueRange.set("values/[5]/[0]", frequency_1);
valueRange.set("values/[6]/[0]", pf_1);
valueRange.set("values/[7]/[0]", loadPercentage_1);

```

Gambar 4. 53 Program Penyusunan Data Monitoring

Setelah seluruh data tersusun, ESP32 mengirimkan data ke Google Spreadsheet menggunakan fungsi `append()`. Metode ini dipilih karena setiap data baru akan otomatis ditambahkan pada baris berikutnya tanpa menghapus data sebelumnya. Dengan demikian, seluruh hasil monitoring dapat tersimpan sebagai data historis yang dapat digunakan untuk analisis lebih lanjut. Program pengiriman data ke Google Spreadsheet ditunjukkan pada Gambar 4. 54.

```

bool success = GSheet.values.append(
    &response /* returned response */,
    spreadsheetId /* spreadsheet Id to append */,
    "Sheet1!A1" /* range to append */,
    &valueRange /* data range to append */
);

```

Gambar 4. 54 Program Pengiriman Data ke Google Spreadsheet

Setelah proses pengiriman selesai, sistem melakukan pemeriksaan status pengiriman. Apabila proses berhasil dilakukan, respons dari server ditampilkan pada serial monitor dan data sementara yang tersimpan pada `valueRange` dihapus menggunakan fungsi `clear()`. Sebaliknya, apabila pengiriman gagal, sistem akan menampilkan penyebab kegagalan yang diperoleh dari fungsi `GSheet.errorReason()`. Program pemeriksaan hasil pengiriman data ditunjukkan pada Gambar 4. 55.

```

if (success){
    response.toString(Serial, true);
    valueRange.clear();
    Serial.println("AFTER APPEND");
}
else{
    Serial.println(GSheet.errorReason());
}

```

Gambar 4. 55 Program Pemeriksaan Status Pengiriman Data

Proses pengiriman data ke Google Spreadsheet dilakukan setelah receiver berhasil menerima data dari transmitter melalui LoRa. Pada fungsi taskPolling(), data yang telah diterima dan diproses akan langsung dikirim ke spreadsheet sebelum sistem melanjutkan ke proses berikutnya. Dengan mekanisme ini, setiap hasil monitoring yang berhasil diperoleh akan tersimpan secara otomatis pada platform cloud. Program pemanggilan fungsi pengiriman data ditunjukkan pada Gambar 4. 56.

```
if (poolingSuccess) {
    sendToGoogleSheets();
    xSemaphoreGive(newDataSignal);
}
```

Gambar 4. 56 Program Pemanggilan Pengiriman Data pada Task Polling

Berdasarkan alur program tersebut, data hasil monitoring yang diterima oleh receiver akan disusun, diverifikasi, dan dikirim secara otomatis ke Google Spreadsheet. Setiap data disimpan lengkap dengan identitas transformator dan waktu pengambilan data sehingga dapat digunakan sebagai media pemantauan kondisi beban transformator secara real-time maupun historis.

4.2.2.8 Monitoring dan Notifikasi Peringatan ke Bot Telegram

Bot Telegram digunakan sebagai media komunikasi antara sistem monitoring dengan pengguna. Fitur ini memungkinkan pengguna memperoleh informasi kondisi transformator secara langsung tanpa harus membuka Google Spreadsheet. Selain berfungsi sebagai media notifikasi otomatis, Telegram Bot juga digunakan untuk menampilkan data monitoring berdasarkan permintaan pengguna melalui perintah tertentu. Dengan mekanisme ini, kondisi transformator dapat dipantau dari jarak jauh menggunakan aplikasi Telegram yang terhubung ke internet.

Sebelum fitur Telegram digunakan, program terlebih dahulu mendefinisikan token bot dan identitas pengguna yang diizinkan menerima pesan. Token digunakan sebagai identitas bot pada server Telegram, sedangkan CHAT_ID digunakan untuk menentukan akun tujuan pengiriman notifikasi. Program konfigurasi Telegram Bot ditunjukkan pada Gambar 4. 57.

```
#define BOTtoken
"TOKEN_BOT"
#define CHAT_ID
"CHAT_ID"
```

Gambar 4. 57 Program Konfigurasi Telegram Bot

Agar komunikasi HTTPS dengan server Telegram dapat dilakukan secara aman, ESP32 melakukan konfigurasi sertifikat keamanan menggunakan objek `WiFiClientSecure`. Selanjutnya dibuat objek `UniversalTelegramBot` yang digunakan sebagai antarmuka komunikasi antara ESP32 dan Telegram Bot. Program inisialisasi Telegram Bot ditunjukkan pada Gambar 4. 58.

```
WiFiClientSecure secured_client;
UniversalTelegramBot bot(BOTtoken, secured_client);

secured_client.setCACert(TELEGRAM_CERTIFICATE_ROOT);
```

Gambar 4. 58 Program Inisialisasi Telegram Bot

Setelah proses inisialisasi selesai, sistem mengirimkan pesan awal sebagai indikator bahwa receiver telah berhasil aktif. Pesan ini dikirim ketika ESP32 selesai melakukan konfigurasi perangkat dan koneksi internet telah tersedia. Program pengiriman pesan awal ditunjukkan pada Gambar 4. 59.

```
bot.sendMessage(
    CHAT_ID,
    "Receiver Started",
    ""
);
```

Gambar 4. 59 Program Pengiriman Pesan Saat Sistem Aktif

Pada penelitian ini, Telegram digunakan sebagai sistem peringatan dini terhadap kondisi abnormal pada transformator. Parameter yang dipantau meliputi persentase pembebanan, tegangan rendah (undervoltage), dan tegangan lebih (overvoltage). Nilai ambang batas yang digunakan ditentukan melalui beberapa variabel program.

Nilai ambang batas untuk undervoltage dan over voltage ini diambil dari SPLN 1:1978 yang menetapkan tegangan maksimum +5% dan minimum -10% dari

tegangan nominal. Yang mana nilainya adalah 198 V – 231 V. sedangkan nilai ambang batas untuk persentasi beban trafo diambil dari SPLN 50:1997 yang menetapkan maksimum pembebanan untuk trafo adalah 80%. Program penetapan ambang batas ditunjukkan pada Gambar 4.60.

```
#define LOAD_PERCENTAGE_THRESHOLD 80.0f
#define UNDERVOLTAGE_THRESHOLD 198.0f
#define OVERVOLTAGE_THRESHOLD 231.0f
```

Gambar 4. 60 Program Penentuan Ambang Batas Notifikasi

Pemeriksaan kondisi transformator dilakukan melalui fungsi `checkThresholdAndNotify()`. Fungsi ini akan membandingkan data hasil monitoring dengan nilai ambang batas yang telah ditentukan. Apabila nilai pembebanan melebihi 80%, sistem akan membuat pesan peringatan dan mengirimkannya ke Telegram. Program pemeriksaan kondisi pembebanan ditunjukkan pada Gambar 4. 61.

```
if (data.loadPercentage > LOAD_PERCENTAGE_THRESHOLD) {
    if (!notifLoadSent) {
        String msg =
            "⚠️ *PERINGATAN BEBAN TINGGI* ⚠️ \n\n";
        msg +=
            "🏢 *Trafo* : " TRAFO_ID_1 "\n";
        msg +=
            "🕒 *Tanggal & Waktu* : " + String(waktuStr) + "\n";
        msg +=
            "📊 *Load* : " + String(data.loadPercentage, 2) + " %\n\n";
        msg +=
            "👤 Beban telah melebihi batas *80%*. \n";
        msg +=
            "⚡ Segera lakukan pemeriksaan.";
        sendToTelegram(msg, CHAT_ID);
        notifLoadSent = true;
    }
}
```

Gambar 4. 61 Program Pemeriksaan Beban Transformator

Selain memantau pembebanan, sistem juga memeriksa kondisi tegangan. Apabila tegangan yang diterima lebih rendah dari 198 V, sistem akan mengirimkan notifikasi undervoltage. Sebaliknya, apabila tegangan melebihi 231 V, sistem akan mengirimkan notifikasi overvoltage. Program pemeriksaan kondisi tegangan undervoltage ditunjukkan pada Gambar 4. 62 dan overvoltage pada Gambar 4. 63..

```
if (data.voltage < UNDERVOLTAGE_THRESHOLD) {
    if (!notifUnderVoltSent) {
        String msg = " ▽ *PERINGATAN UNDERVOLTAGE* ▽ \n\n";
        msg += " 🏠 *Trafo* : " TRAFO_ID_1 "\n";
        msg += " ⌚ *Tanggal & Waktu* : " + String(waktuStr) + "\n";
        msg += " ⚡ *Voltage* : " + String(data.voltage, 2) + " V\n\n";
        msg += " ⚠ Tegangan berada di bawah batas *198 V*.\n";
        msg += " 🛠 Mohon segera dilakukan pengecekan.";
        sendToTelegram(msg, CHAT_ID);
        notifUnderVoltSent = true;
    }
} else {
    notifUnderVoltSent = false;
}
```

Gambar 4. 62 Program Pemeriksaan Tegangan Undervoltage Transformator

```
if (data.voltage > OVERVOLTAGE_THRESHOLD) {
    if (!notifOverVoltSent) {
        String msg = " △ *PERINGATAN OVERVOLTAGE* △ \n\n";
        msg += " 🏠 *Trafo* : " TRAFO_ID_1 "\n";
        msg += " ⌚ *Tanggal & Waktu* : " + String(waktuStr) + "\n";
        msg += " ⚡ *Voltage* : " + String(data.voltage, 2) + " V\n\n";
        msg += " ⚠ Tegangan melebihi batas *231 V*.\n";
        msg += " 🛑 Segera lakukan pemeriksaan sistem.";
        sendToTelegram(msg, CHAT_ID);
        notifOverVoltSent = true;
    }
}
```

```

}
else {
    notifOverVoltSent = false;
}

```

Gambar 4. 63 Program Pemeriksaan Tegangan Overvoltage Transformator

Untuk mencegah pengiriman notifikasi secara berulang pada kondisi yang sama, sistem menggunakan beberapa variabel penanda seperti `notifLoadSent`, `notifUnderVoltSent`, dan `notifOverVoltSent`. Variabel tersebut akan berubah menjadi `true` setelah pesan berhasil dikirim. Dengan metode ini, satu kondisi gangguan hanya menghasilkan satu notifikasi sampai nilai parameter kembali ke kondisi normal.

Telegram Bot juga digunakan sebagai media monitoring interaktif. Pengguna dapat mengirimkan perintah `/cek` atau `/status` untuk memperoleh kondisi transformator secara langsung. Ketika perintah diterima, ESP32 akan mengambil data monitoring terbaru dan mengirimkannya kembali dalam bentuk pesan Telegram. Program pemeriksaan perintah pengguna ditunjukkan pada Gambar 4. 64.

```

if (text == "/cek" || text == "/status") {
    SensorData data;

```

Gambar 4. 64 Program Pemeriksaan Perintah Telegram

Data yang ditampilkan pada perintah tersebut meliputi tegangan, arus, daya, frekuensi, faktor daya, dan persentase pembebanan transformator yang ditampilkan dengan tambahan emoji yang membuat tampilan pesan lebih menarik. Seluruh data diambil dari hasil polling terakhir yang diterima receiver dari transmitter sehingga pengguna dapat memperoleh kondisi transformator secara aktual tanpa harus membuka spreadsheet.

Proses pemeriksaan notifikasi dan pesan Telegram dijalankan secara terpisah melalui `taskTelegram()`. Task ini bekerja secara kontinu untuk memeriksa data baru dari proses polling dan memantau pesan yang masuk dari pengguna. Program task Telegram ditunjukkan pada Gambar 4. 65.

```

void taskTelegram(void *pvParameters) {

```

```

for (;;) {
    // 1) Cek apakah ada data baru dari Task Polling -> evaluasi
    threshold
    if (xSemaphoreTake(newDataSignal, 0) == pdTRUE) {
        checkThresholdAndNotify();
    }

    // 2) Cek pesan masuk (manual check dari user)
    int numNewMessages = bot.getUpdates(bot.last_message_received
+ 1);
    while (numNewMessages) {
        handleNewMessages(numNewMessages);
        numNewMessages = bot.getUpdates(bot.last_message_received +
1);
    }

    vTaskDelay(pdMS_TO_TICKS(telegramCheckInterval));
}
}

```

Gambar 4. 65 Program Task Monitoring Telegram

Berdasarkan alur program tersebut, Telegram Bot berfungsi sebagai media monitoring dan sistem peringatan dini pada transformator distribusi. Receiver akan memantau nilai pembebanan dan tegangan secara otomatis, kemudian mengirimkan notifikasi apabila terdeteksi kondisi yang melebihi batas yang ditentukan. Selain itu, pengguna juga dapat meminta data monitoring secara langsung melalui perintah Telegram sehingga kondisi transformator dapat diketahui kapan saja melalui perangkat bergerak yang terhubung ke internet.