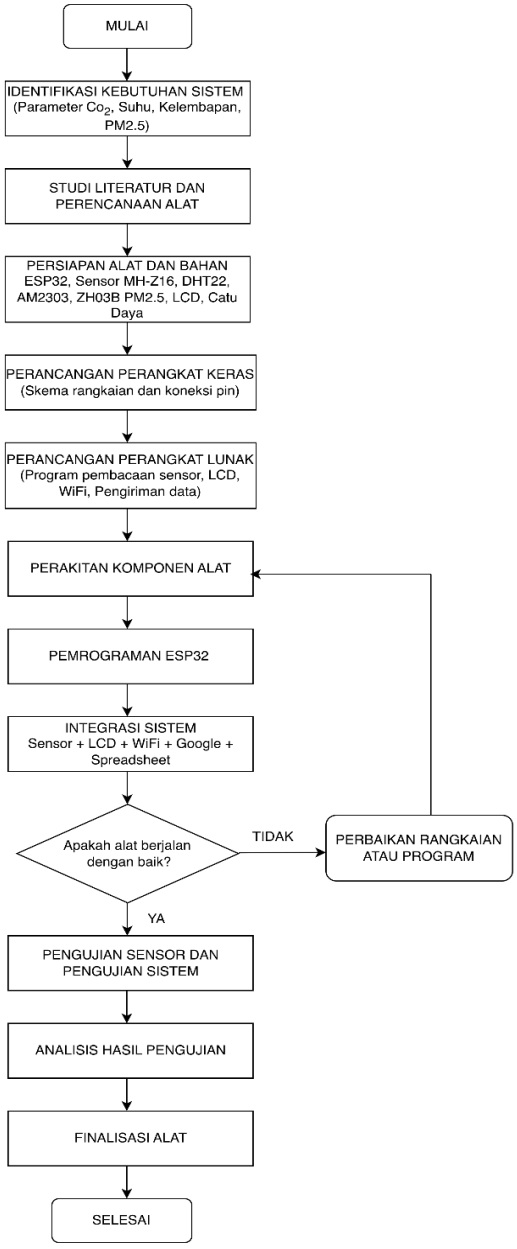
BAB IV
PEMBUATAN ALAT

DIAGRAM PROSEDUR PEMBUATAN ALAT



Gambar 4. 1 Diagram Prosedur Pembuatan Alat

Pembuatan alat “RANCANG BANGUN SISTEM MONITORING KUALITAS UDARA DALAM RUANGAN BERBASIS ESP32 DENGAN PARAMETER CO_2, SUHU, KELEMBAPAN, DAN PM2.5 DENGAN PENYIMPANAN DATA PADA GOOGLE SPREADSHEET” dibagi menjadi 2 bagian, yaitu :

4.1 Pembuatan Perangkat Keras (Hardware)

Pembuatan perangkat keras dilakukan dengan merangkai seluruh komponen yang digunakan pada sistem monitoring kualitas udara dalam ruangan. Komponen utama yang digunakan terdiri dari mikrokontroler ESP32, sensor karbon dioksida (CO_2) MH-Z16, sensor suhu dan kelembapan DHT22 AM2302, sensor partikulat PM2.5 ZH03B, LCD I2C 20x4, catu daya 5V, serta kabel jumper sebagai penghubung antar komponen.

4.1.1 Alat dan Bahan

Proses pertama dalam pembuatan sistem ini adalah mempersiapkan alat dan bahan yang digunakan. Daftar alat yang digunakan untuk membangun RANCANG BANGUN SISTEM MONITORING KUALITAS UDARA DALAM RUANGAN BERBASIS ESP32 DENGAN PARAMETER CO_2, SUHU, KELEMBAPAN, DAN PM2.5 DENGAN PENYIMPANAN DATA PADA GOOGLE SPREADSHEET dapat dilihat pada tabel 4-1.

Tabel 4- 1 Bahan yang digunakan

No	Bahan	Jumlah
1	Mikrokontoler ESP32	1
2	Sensor CO_2 Winsen MH-Z16	1
3	Sensor Suhu dan Kelembapan DHT22	1
4	Sensor PM2.5 Winsen (ZH03B)	1
5	Breadboard	1
6	Kabel Jumper	4

7	Adaptor/Catu Daya 5V	1
8	Kabel USB	1
9	Box Proyek	1
10	LCD 20x4 I2C	1

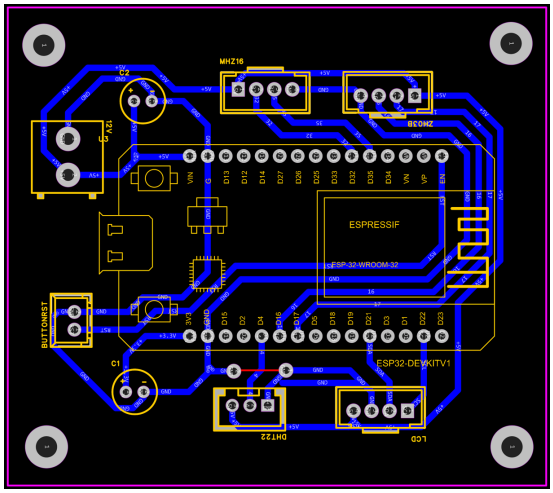
Tabel 4- 2 Alat yang Digunakan

No	Bahan	Jumlah
1	Solder	1
2	Multimeter	1
3	Gerinda mini	1
4	Bor	1

4.1.2 Pembuatan Perangkat Elektrik

Pembuatan Perangkat elektrik merupakan tahap penting dalam proses perakitan rancang bangun. Pada tahap ini, dilakukan perancangan dan perakitan seluruh komponen elektronik yang bertugas membaca data sensor, memproses informasi, serta mengirimkan hasil pengukuran ke media penyimpanan data. Seluruh komponen elektronik ini dirangkai dalam satu sistem berbasis mikrokontroller. Adapun langkah-langkah pembuatan perangkat elektrik adalah sebagai berikut :

1. Perancangan Skematik dan Layout PCB

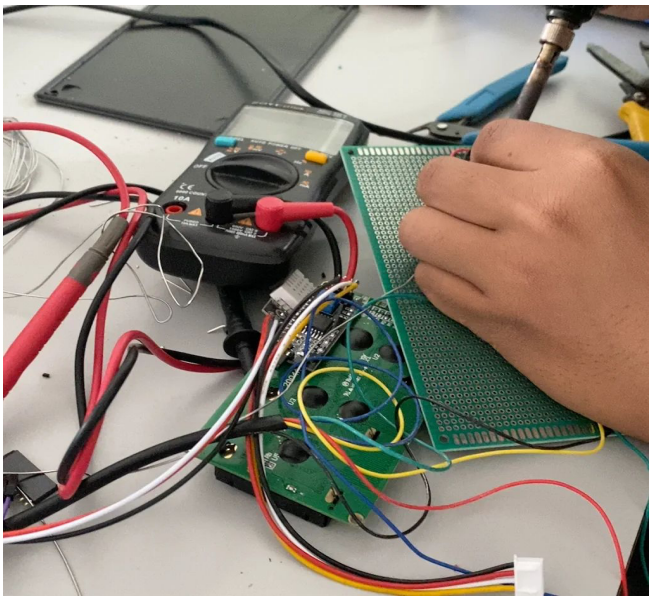


Gambar 4. 2 Perancangan Skematik dan Layout PCB

Perancangan sistem diawali dengan pembuatan skematik rangkaian elektronik menggunakan perangkat lunak EasyEDA, seperti ditunjukkan pada Gambar 4.1. Layout PCB dirancang secara modular dengan setiap sensor menggunakan konektor terpisah sehingga memudahkan proses pemasangan, pengujian, dan penggantian komponen. Penempatan ESP32 di bagian tengah PCB bertujuan untuk memperpendek jalur komunikasi ke setiap sensor, sehingga dapat mengurangi redaman sinyal dan potensi noise. Selain itu, konektor terpisah pada sensor DHT22, MH-Z16, ZH03B, dan LCD I2C memberikan fleksibilitas yang lebih baik dalam proses integrasi dan pemeliharaan sistem. Dari sisi elektrikal, jalur catu daya dan jalur sinyal dipisahkan untuk meningkatkan kestabilan pembacaan sensor. Penggunaan komunikasi digital berupa Single-Wire, UART, dan I2C juga membuat sistem lebih tahan terhadap gangguan dibandingkan komunikasi analog. Dengan demikian, desain PCB yang dirancang telah mendukung kebutuhan sistem monitoring kualitas udara berbasis ESP32 dari segi akurasi, kemudahan integrasi, dan keandalan operasional.

2. Proses Penyolderan PCB

   Penyolderan diawali dengan membersihkan permukaan PCB dari debu, minyak, atau sisa proses pencetakan agar tidak memengaruhi kualitas sambungan solder. Selanjutnya, seluruh komponen dipasang sesuai dengan posisi pada layout PCB. Penyolderan dilakukan secara bertahap, dimulai dari komponen berukuran kecil dan berposisi rendah, kemudian dilanjutkan ke komponen yang lebih besar untuk mempermudah pemasangan dan menjaga kestabilan komponen. Timah solder dipanaskan menggunakan solder listrik hingga mencapai suhu kerja, kemudian diaplikasikan pada kaki komponen dan pad PCB secara bersamaan agar menghasilkan sambungan yang kuat dan memiliki kontak listrik yang baik. Setelah seluruh komponen terpasang, dilakukan pemeriksaan visual untuk memastikan tidak terdapat cacat penyolderan yang dapat mengganggu kinerja sistem.



Gambar 4. 3 Proses Penyolderan PCB

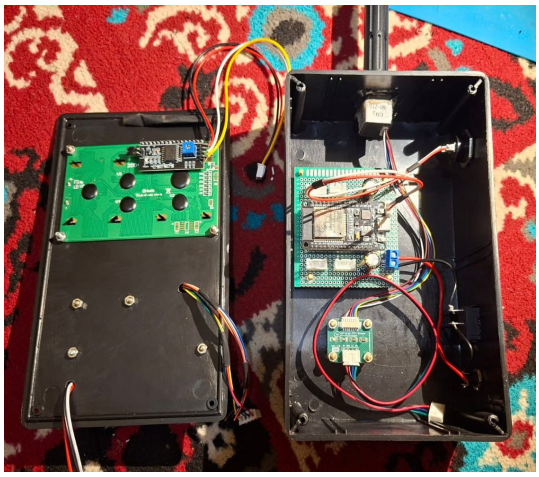
3. Proses pencetakan dan penempelan LCD pada tutup Box

   Setelah proses perakitan rangkaian elektronik selesai, tahap selanjutnya adalah pembuatan dan pemasangan LCD pada box alat sebagai media tampilan informasi sekaligus pelindung komponen elektronik. Proses diawali dengan menentukan posisi LCD I2C 20×4 pada bagian depan box berdasarkan ukuran modul dan kemudahan pembacaan oleh pengguna. Selanjutnya dilakukan penandaan, pemotongan, dan perapian lubang pada box agar sesuai dengan dimensi LCD. LCD kemudian dipasang pada bagian dalam box menggunakan baut, spacer, atau perekat sehingga terpasang kuat, stabil, dan sejajar dengan permukaan box. Setelah pemasangan selesai, dilakukan pengujian untuk memastikan LCD dapat menampilkan informasi hasil pengukuran, meliputi suhu, kelembapan, konsentrasi CO_2, konsentrasi PM2.5, tanggal, waktu, serta status pengiriman data dengan jelas. Selain memudahkan proses monitoring, pemasangan LCD pada box juga berfungsi melindungi modul dari debu, benturan, dan kerusakan mekanis selama alat dioperasikan.

Gambar 4. 4 Pembuatan LCD

4. Integrasi Sistem

   Integrasi sistem merupakan tahap penggabungan seluruh perangkat keras dan perangkat lunak sehingga membentuk sistem monitoring kualitas udara yang bekerja secara terpadu. Pada penelitian ini, sensor DHT22, MH-Z16, dan ZH03B dihubungkan ke ESP32 sebagai pusat pengendali untuk mengolah data hasil pengukuran. Data yang diperoleh kemudian ditampilkan pada LCD I2C sebagai media monitoring lokal serta dikirim melalui koneksi WiFi ke Google Spreadsheet menggunakan Google Apps Script sebagai media penyimpanan berbasis cloud.



Gambar 4. 5 Integrasi Sistem

Hasil integrasi menunjukkan bahwa seluruh komponen dapat berkomunikasi dan berfungsi sesuai dengan rancangan, sehingga sistem mampu melakukan monitoring serta pencatatan data kualitas udara secara otomatis dan berkala dengan baik.

4.1.3 Pembuatan Perangkat Mekanik

Pembuatan mekanik alat merupakan tahapan yang dilakukan untuk menyediakan tempat pemasangan seluruh komponen elektronik sehingga sistem monitoring kualitas udara dapat beroperasi dengan baik, aman, dan memiliki tampilan yang rapi. Mekanik alat dirancang dalam bentuk box yang berfungsi sebagai pelindung komponen dari debu, benturan, serta gangguan lingkungan selama proses pengoperasian.

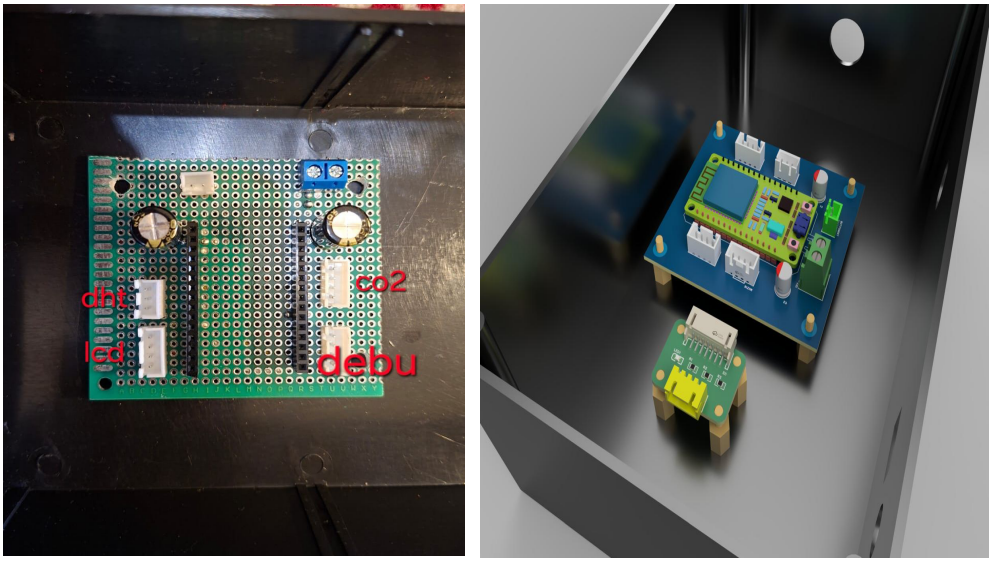
Proses pembuatan mekanik diawali dengan menentukan dimensi box berdasarkan ukuran komponen yang digunakan, seperti ESP32, PCB, LCD I2C 20×4, sensor DHT22, sensor MH-Z16, sensor ZH03B, serta adaptor catu daya. Perancangan dimensi dilakukan dengan mempertimbangkan kebutuhan ruang untuk pemasangan komponen, jalur kabel, ventilasi udara, dan kemudahan perawatan sistem. Secara keseluruhan, sistem ini terdiri dari beberapa komponen mekanik utama yang dirancang untuk menunjang kinerja alat, yaitu :

1. Box panel (Gambar 4.6), berfungsi sebagai tempat pemasangan dan pelindung seluruh komponen elektronik agar terhindar dari debu, benturan, dan gangguan lingkungan selama pengoperasian.

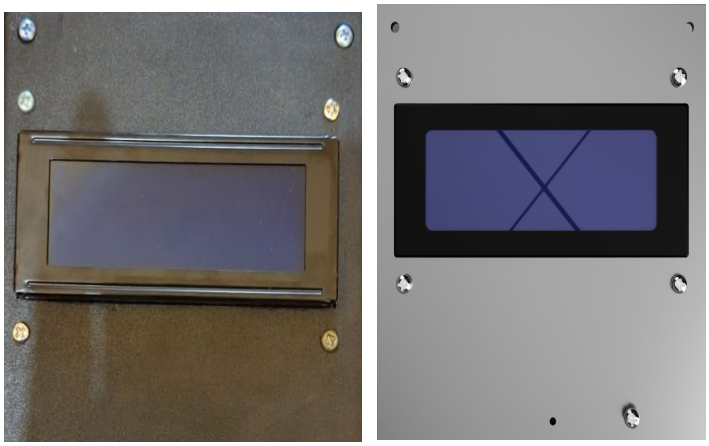
2. Dudukan PCB (Gambar 4.7), digunakan untuk menopang papan rangkaian utama sehingga posisinya tetap stabil dan terhindar dari kontak langsung dengan permukaan box.
3. Panel LCD 20×4 I2C (Gambar 4.8), berfungsi sebagai media tampilan untuk menampilkan data suhu, kelembapan, konsentrasi CO_2, PM2.5, serta informasi waktu langsung untuk tampilan LCD dan secara berkala untuk monitoring.
4. Ventilasi udara (Gambar 4.9), berfungsi sebagai jalur sirkulasi udara menuju sensor sehingga proses pengukuran dapat dilakukan dengan lebih akurat.
5. Lubang konektor catu daya (Gambar 4.10), digunakan sebagai jalur masuk adaptor 5V yang menyuplai kebutuhan daya seluruh sistem.
6. Tutup box (Gambar 4.11), berfungsi melindungi komponen elektronik dari debu dan benturan serta menjaga keamanan sistem selama beroperasi.
7. Dudukan sensor (Gambar 4.12), digunakan untuk menempatkan sensor DHT22, MH-Z16, dan ZH03B pada posisi yang sesuai agar pengukuran kualitas udara dapat dilakukan secara optimal.

Setiap komponen pada sistem dirancang dan ditempatkan secara terintegrasi di dalam box panel sehingga membentuk satu kesatuan sistem monitoring kualitas udara yang rapi dan mudah dioperasikan. Tata letak komponen seperti ESP32, PCB, LCD 20×4 I2C, sensor DHT22, sensor MH-Z16, sensor ZH03B, serta catu daya dirancang dengan mempertimbangkan kemudahan pemasangan, pengujian, dan pemeliharaan. Selain itu, desain mekanik yang modular memudahkan proses perakitan, penggantian komponen apabila terjadi kerusakan, serta memungkinkan pengembangan sistem di masa mendatang tanpa perlu melakukan perubahan besar pada struktur alat yang telah dibuat.

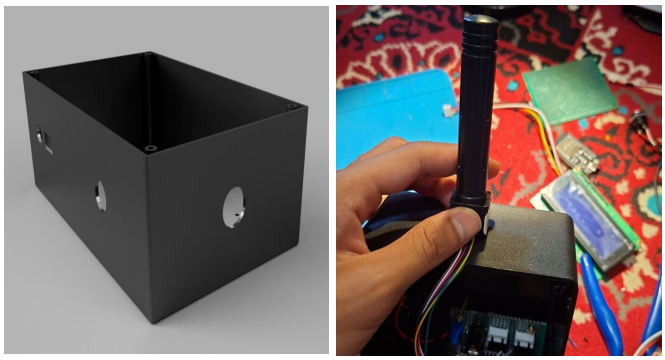
Gambar 4. 6 Box Panel



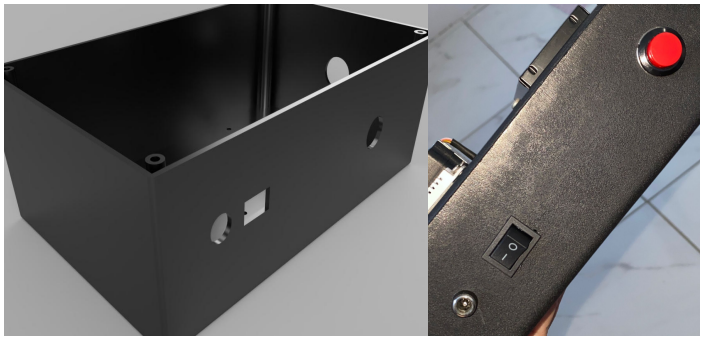
Gambar 4. 7 Dudukan PCB



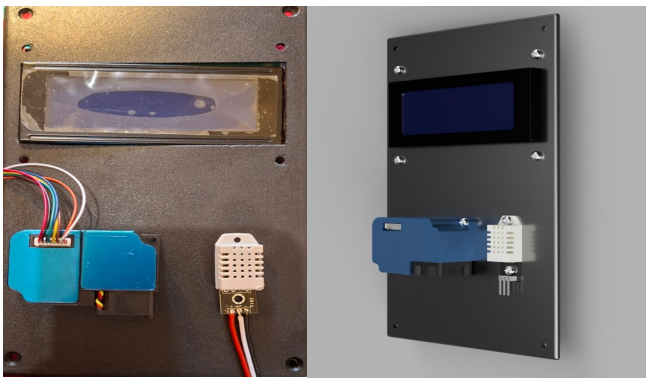
Gambar 4. 8 Panel LCD 20x4 12C



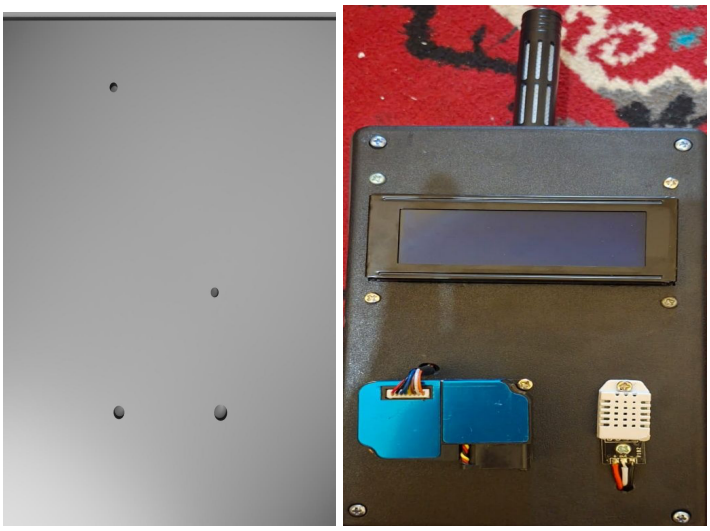
Gambar 4. 9 Ventilasi Udara



Gambar 4. 10 Lubang Konektor Catu Daya



Gambar 4. 11 Tutup Box



Gambar 4. 12 Dudukan Sensor

4.2 Pembuatan Program Pada Arduino IDE

Pemrograman mikrokontroler ESP32 dilakukan menggunakan perangkat lunak Arduino IDE dengan bahasa pemrograman C++. Program ini merupakan bagian utama dari sistem karena mengendalikan seluruh proses kerja alat, mulai dari pembacaan data sensor, pengolahan data, penampilan hasil pengukuran pada LCD, hingga pengiriman data ke Google Spreadsheet melalui jaringan WiFi. Struktur program disusun secara modular agar memudahkan proses pengembangan, pengujian, dan pemeliharaan sistem. Pembagian program yang dikembangkan meliputi beberapa bagian, yaitu:

1. Program untuk inisialisasi ESP32 beserta konfigurasi awal seluruh perangkat, seperti LCD I2C, sensor DHT22, sensor MH-Z16, sensor ZH03B, koneksi WiFi, dan sinkronisasi waktu (NTP).
2. Program untuk membaca data sensor DHT22, yang digunakan untuk memperoleh nilai suhu dan kelembapan udara dalam ruangan.
3. Program untuk membaca data sensor MH-Z16, yang digunakan untuk mengukur konsentrasi karbon dioksida (CO_2) dalam satuan ppm.
4. Program untuk membaca data sensor ZH03B, yang digunakan untuk mengukur konsentrasi partikulat udara, yaitu PM1.0, PM2.5, dan PM10.

5. Program untuk mengolah dan menampilkan data hasil pengukuran pada LCD I2C, meliputi informasi suhu, kelembapan, konsentrasi CO_2, partikulat, tanggal, waktu, dan status pengiriman data.
6. Program untuk menghubungkan ESP32 ke jaringan WiFi menggunakan WiFiManager sehingga alat dapat terhubung ke internet sebelum melakukan pengiriman data.
7. Program untuk mengirim dan menyimpan data hasil pengukuran ke Google Spreadsheet melalui Google Apps Script secara berkala sebagai media pencatatan data monitoring.
8. Program utama (looping) untuk melakukan pembacaan sensor, pembaruan tampilan LCD, serta pengiriman data secara otomatis dan berulang selama sistem beroperasi.

4.2.1 Inisialisasi Pin dan Konfigurasi awal ESP32

Bagian ini berisi pemanggilan pustaka (*library*) dan inisialisasi pin yang diperlukan untuk membangun sistem monitoring kualitas udara berbasis ESP32. Pustaka Wire.h dan LiquidCrystal_I2C.h digunakan untuk komunikasi I2C dengan LCD, sedangkan HardwareSerial.h mendukung komunikasi UART pada sensor. Pembacaan data sensor dilakukan menggunakan pustaka SD_ZH03B.h untuk sensor partikulat ZH03B, MHZ16_uart.h untuk sensor CO_2 MH-Z16, dan DHT.h untuk sensor suhu dan kelembapan DHT22. Sementara itu, pustaka WiFi.h, WiFiManager.h, HTTPClient.h, dan WiFiClientSecure.h digunakan untuk menghubungkan ESP32 ke jaringan WiFi, mengelola konfigurasi koneksi, serta mengirim data ke Google Spreadsheet melalui protokol HTTPS. Pustaka time.h digunakan untuk sinkronisasi waktu dengan server NTP sehingga setiap data memiliki informasi waktu yang akurat. Setelah seluruh pustaka dipanggil, dilakukan inisialisasi pin perangkat keras menggunakan direktif #define, yaitu GPIO 4 untuk sensor DHT22, GPIO 17 dan GPIO 16 sebagai jalur UART sensor ZH03B, serta GPIO 35 dan GPIO 32 sebagai jalur UART sensor MH-Z16.

```
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
#include <HardwareSerial.h>
#include <SD_ZH03B.h>
#include <MHZ16_uart.h>
#include <WiFi.h>
#include <WiFiManager.h>
#include <time.h>
#include <DHT.h>
#include <HTTPClient.h>
#include <WiFiClientSecure.h>

#define DHTPIN 4
#define DHTTYPE DHT22
#define ZH_RX_PIN 17
#define ZH_TX_PIN 16
```

Gambar 4. 13 Program Inisialisasi Pin dan Konfigurasi awal ESP32

4.2.2 Konfigurasi Sinkronisasi Waktu (NTP)

Bagian ini berisi konfigurasi sinkronisasi waktu menggunakan Network Time Protocol (NTP) agar ESP32 memiliki referensi waktu yang akurat untuk pencatatan (*timestamp*) setiap data hasil pemantauan. Server NTP yang digunakan adalah pool.ntp.org**,** sedangkan parameter gmtOffset_sec diatur sebesar 25.200 detik untuk menyesuaikan zona waktu Waktu Indonesia Barat (GMT+7). Nilai daylightOffset_sec ditetapkan 0 karena Indonesia tidak menerapkan Daylight Saving Time (DST)**.** Dengan konfigurasi tersebut, waktu pada ESP32 akan tersinkronisasi secara otomatis setelah perangkat berhasil terhubung ke jaringan WiFi, sehingga setiap data yang dikirim ke Google Spreadsheet memiliki informasi waktu yang akurat.

```
// Konfigurasi Waktu NTP
const char* ntpServer          = "pool.ntp.org";
const long  gmtOffset_sec      = 25200; // GMT+7 Indonesia
const int   daylightOffset_sec = 0;
```

Gambar 4. 14 Program Konfigurasi Sinkronisasi Waktu (NTP)

4.2.3 Konfigurasi Google Spreadsheet

Bagian ini berisi pendefinisian alamat tujuan (*endpoint*) untuk proses pengiriman data hasil monitoring ke Google Spreadsheet. Alamat tersebut disimpan dalam variabel GOOGLE_SCRIPT_URL, yang berisi URL hasil *deployment* Google Apps Script. URL ini berfungsi sebagai penghubung antara ESP32 dan Google Spreadsheet sehingga data hasil pembacaan sensor dapat dikirim melalui protokol HTTPS dan secara otomatis disimpan pada lembar kerja sebagai media pencatatan (*data logging*) secara jarak jauh.

```
24  // Konfigurasi Google Spreadsheet
25  String GOOGLE_SCRIPT_URL = "https://script.google.com/macros/s/AKfycbzAnP8u7LcZkRKqTVBaHRKaB9RvXtLiuXm8_tbfCZMxpkpa0CynYWrQ4hr2S-MPsIZS/exec";
```

Gambar 4. 15 Program Konfigurasi Google Spreadsheet

4.2.4 Konfigurasi Interval Pengiriman Data

Bagian ini mengatur mekanisme pengiriman data secara berkala menggunakan fungsi millis() sehingga proses pengiriman tidak menghambat pembacaan sensor. Variabel waktuKirimTerakhir menyimpan waktu pengiriman terakhir, sedangkan jedaKirimData ditetapkan sebesar 10 detik sebagai jeda minimum antar pengiriman. Interval aktual antar data yang tersimpan lebih panjang dari nilai tersebut, dengan rata-rata sekitar 15 detik, karena waktu pembacaan ketiga sensor dan proses pengiriman HTTPS menambah durasi setiap siklus program. Metode ini memungkinkan ESP32 menjalankan pembacaan sensor dan pembaruan tampilan LCD tanpa fungsi delay() yang bersifat menghambat.

```
25  unsigned long waktuKirimTerakhir = 0;
26  // Interval pengiriman diatur 16 detik, namun realisasinya berkisar 15–18 detik
27  // akibat waktu proses pembacaan sensor dan pengiriman data HTTP ke Google Spreadsheet
28  const unsigned long jedaKirimData = 16000;
29  // '–' = standby, 'V' = sukses, 'X' = gagal
30  char statusKirim = '–';
31
32  unsigned long waktuStatusKirimMuncul = 0;
33  const unsigned long durasiTampilStatusKirim = 16000; // V tampil 1 detik
```

Gambar 4. 16 Program Konfigurasi Interval Pengiriman Data

4.2.5 Konfigurasi Status Pengiriman Data

Bagian ini digunakan untuk menampilkan status proses pengiriman data pada LCD. Variabel statusKirim berfungsi sebagai indikator dengan tiga kondisi, yaitu **'-'** sebagai status siaga, ‘V’ sebagai tanda pengiriman berhasil, dan ‘X’ sebagai tanda pengiriman gagal. Selain itu, sistem mengatur durasi tampilan status berhasil selama satu detik sebelum kembali ke kondisi siaga sehingga informasi yang ditampilkan pada LCD selalu sesuai dengan kondisi pengiriman data terkini.

```
30  // '-' = standby, 'V' = sukses, 'X' = gagal
31  char statusKirim = '-';
32
33  unsigned long waktuStatusKirimMuncul = 0;
34  const unsigned long durasiTampilStatusKirim = 1000; // V tampil 1 detik
```

Gambar 4. 17 Program Konfigurasi Status Pengiriman Data

4.2.6 Inisialisasi Objek Perangkat Keras

Bagian ini berisi proses pembentukan objek dari setiap perangkat yang digunakan pada sistem. Objek LCD diinisialisasi menggunakan alamat I2C 0x27 dengan ukuran tampilan 20×4. Selanjutnya dilakukan inisialisasi objek sensor DHT22, sensor ZH03B, dan sensor MH-Z16 sesuai dengan pustaka yang digunakan. Tahap ini bertujuan agar setiap perangkat keras dapat dikenali dan dikendalikan oleh ESP32 sesuai dengan konfigurasi yang telah ditentukan.

```
37  // Konfigurasi LCD 20x4
38  LiquidCrystal_I2C lcd(0x27, 20, 4);
39
40  // Konfigurasi Sensor DHT22
41  DHT dht(DHTPIN, DHTTYPE);
42
43  // Konfigurasi Sensor Debu ZH03B
44  HardwareSerial ZHSerial(1);
45  SD_ZH03B ZH03B(ZHSerial, SD_ZH03B::SENSOR_ZH03B);
46
47  // Konfigurasi Sensor CO2 MH-Z16
48  #define MHZ_RX_PIN 35
49  #define MHZ_TX_PIN 32
50  MHZ16_uart mhz16;
```

```
52  // Fungsi ketika ESP membuat portal WiFi
53  void configModeCallback(WiFiManager *myWiFiManager) {
54    lcd.clear();
55
56    lcd.setCursor(0, 0);
57    lcd.print("Setting WiFi");
58
59    lcd.setCursor(0, 1);
60    lcd.print("SSID:ESP32-Monitor");
61
62    lcd.setCursor(0, 2);
63    lcd.print("Pass:12345678");
64
65    lcd.setCursor(0, 3);
66    lcd.print("IP:192.168.4.1");
67
68    Serial.println("================================");
69    Serial.println("Mode konfigurasi WiFi aktif");
70    Serial.println("Hubungkan HP/laptop ke WiFi:");
71    Serial.println("SSID: ESP32-Monitoring");
72    Serial.println("Password: 12345678");
73    Serial.println("Buka browser: 192.168.4.1");
74    Serial.println("================================");
75  }
```

Gambar 4. 18 Program Inisialisasi Objek Perangkat Keras

4.2.7 Fungsi Callback Mode Konfigurasi (configModeCallback)

Fungsi configModeCallback merupakan callback yang digunakan sebagai fitur antarmuka pengguna dalam mengelola konektivitas jaringan. Fungsi ini akan dipanggil secara otomatis oleh sistem ketika WiFiManager gagal menemukan atau terhubung ke jaringan WiFi yang tersimpan, sehingga perangkat beralih ke mode Access Point (AP).

Pada fungsi ini, tampilan LCD dibersihkan menggunakan lcd.clear(), kemudian diikuti dengan pengaturan posisi kursor dan penampilan teks menggunakan lcd.setCursor() dan lcd.print(). Informasi yang ditampilkan mencakup panduan konfigurasi jaringan, SSID default “ESP32-Monitor”, kata sandi, serta alamat IP lokal “192.168.4.1” yang dapat diakses melalui browser. Informasi yang sama juga dikirimkan ke Serial Monitor menggunakan Serial.println() untuk keperluan pemantauan dan debugging.

Dengan mekanisme ini, sistem dapat memberikan instruksi konfigurasi jaringan secara interaktif dan informatif, sehingga pengguna dapat melakukan pengaturan ulang WiFi secara nirkabel tanpa perlu mengubah kode program secara langsung.

```
// Fungsi koneksi WiFi dengan konfigurasi ulang setiap alat menyala
void setupWiFi() {
  WiFi.mode(WIFI_STA);
  WiFiManager wm;
  // Callback untuk menampilkan info di LCD saat mode setting WiFi
  wm.setAPCallback(configModeCallback);

  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("Reset WiFi lama");
  lcd.setCursor(0, 1);
  lcd.print("Setting ulang...");
  Serial.println("Menghapus WiFi lama...");

  wm.resetSettings();
  delay(1000);

  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("Buka WiFi ESP");
  lcd.setCursor(0, 1);
  lcd.print("ESP32-Monitoring");
  lcd.setCursor(0, 2);
  lcd.print("Pass:12345678");
  lcd.setCursor(0, 3);
  lcd.print("IP:192.168.4.1");

  bool berhasil = wm.startConfigPortal("ESP32-Monitoring", "12345678");

  if (!berhasil) {
    lcd.clear();
    lcd.setCursor(0, 0);
    lcd.print("Setting WiFi");
    lcd.setCursor(0, 1);
    lcd.print("Gagal");

    Serial.println("Konfigurasi WiFi gagal. ESP restart...");
    delay(3000);
    ESP.restart();
  }

  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("WiFi Terhubung");
  lcd.setCursor(0, 1);
  lcd.print(WiFi.SSID());
  lcd.setCursor(0, 2);
  lcd.print("IP:");
  lcd.setCursor(3, 2);
  lcd.print(WiFi.localIP());

  Serial.println("--------------------------------");
  Serial.println("WiFi berhasil terhubung!");
  Serial.print("SSID: ");
  Serial.println(WiFi.SSID());
  Serial.print("IP ESP32: ");
  Serial.println(WiFi.localIP());
  Serial.println("--------------------------------");

  delay(3000);
  lcd.clear();
}
```

Gambar 4. 19 Program Fungsi Callback Mode Konfigurasi (configModeCallback)

4.2.8 Konfigurasi koneksi WiFi (setupWiFi)

Fungsi setupWiFi() digunakan untuk mengatur dan mengelola konektivitas jaringan nirkabel pada mikrokontroler menggunakan pustaka WiFiManager. Pada tahap awal, perangkat diatur dalam mode station (WIFI_STA) dan disertai callback untuk memperbarui tampilan saat berpindah ke mode Access Point (AP). Sistem juga dikonfigurasi untuk menghapus kredensial WiFi yang tersimpan sebelumnya melalui perintah wm.resetSettings(), sehingga perangkat selalu melakukan konfigurasi ulang saat dinyalakan.

Selanjutnya, wm.startConfigPortal() dijalankan untuk membuat jaringan mandiri dengan SSID “ESP32-Monitoring” dan kata sandi “12345678” guna memungkinkan pengguna memasukkan kredensial baru melalui captive portal, yang juga ditampilkan melalui instruksi pada LCD. Jika proses koneksi gagal, sistem akan menampilkan pesan error dan melakukan restart otomatis melalui ESP.restart(). Sebaliknya, jika koneksi berhasil, LCD dan Serial Monitor menampilkan status terhubung beserta SSID dan alamat IP, kemudian sistem melanjutkan ke proses utama.

```
140  void setup() {
141    Serial.begin(115200);
142
143    lcd.init();
144    lcd.backlight();
145    lcd.noCursor();
146    lcd.noBlink();
147
148    setupWiFi();
149
150    configTime(gmtOffset_sec, daylightOffset_sec, ntpServer);
151
152    dht.begin();
153
154    ZHSerial.begin(9600, SERIAL_8N1, ZH_RX_PIN, ZH_TX_PIN);
155    delay(100);
156    ZH03B.setMode(SD_ZH03B::QA_MODE);
157
158    mhz16.begin(MHZ_RX_PIN, MHZ_TX_PIN, 2);
159
160    lcd.clear();
161  }
```

Gambar 4. 20 Program Konfigurasi koneksi WiFi (setupWiFi)

4.2.9 Program Pembacaan Sensor

Bagian ini menjelaskan proses pembacaan data dari seluruh sensor yang dilakukan pada awal fungsi loop(). ESP32 membaca nilai suhu dan kelembapan dari sensor DHT22, konsentrasi gas karbon dioksida dari sensor MH-Z16, serta konsentrasi partikulat dari sensor ZH03B. Untuk mengoptimalkan tampilan LCD, program menggunakan fungsi millis() sehingga nilai PM1.0, PM2.5, dan PM10 ditampilkan secara bergantian setiap satu detik. Metode ini memungkinkan seluruh parameter kualitas udara dapat dipantau tanpa mengurangi keterbacaan informasi pada LCD.

```
163  void loop() {
164
165  // Baca Sensor
166    float t = dht.readTemperature();
167    float h = dht.readHumidity();
168    int co2 = mhz16.getPPM();
169    bool isZHValid = ZH03B.readData();
170
171    static unsigned long waktuGantiDebu = 0;
172    static int statusDebu = 0; // 0 = PM1.0, 1 = PM2.5, 2 = PM10
173
174    if (millis() - waktuGantiDebu >= 1000) {
175      statusDebu++;
176
177      if (statusDebu > 2) {
178        statusDebu = 0;
179      }
180
181      waktuGantiDebu = millis();
182    }
183
184    int pm25 = isZHValid ? ZH03B.getPM2_5() : 0;
185
186    // 2. Tampilkan ke LCD
187    struct tm timeinfo;
188
189    lcd.setCursor(0, 0);
190
191    if (getLocalTime(&timeinfo)) {
192      char timeStr[21];
193
194      sprintf(timeStr, "%02d/%02d/%04d %02d:%02d:%02d",
195              timeinfo.tm_mday,
196              timeinfo.tm_mon + 1,
197              timeinfo.tm_year + 1900,
198              timeinfo.tm_hour,
199              timeinfo.tm_min,
200              timeinfo.tm_sec);
201
202      lcd.print(timeStr);
203    } else {
204      lcd.print("Sinkronisasi waktu");
205    }
```

Gambar 4. 21 Program Pembacaan Sensor

4.2.10 Program Tampilan LCD

Bagian ini menjelaskan proses penyajian hasil pengukuran pada LCD 20×4. Sistem terlebih dahulu mengambil informasi tanggal dan waktu hasil sinkronisasi NTP, kemudian menampilkan data suhu, kelembapan, konsentrasi CO_2, dan partikulat secara berkala. Selain itu, program melakukan validasi terhadap hasil pembacaan sensor. Jika data yang diterima tidak valid, LCD akan menampilkan pesan "Error" sebagai indikator adanya gangguan pada sensor atau komunikasi. Program juga menampilkan status pengiriman data ke Google Spreadsheet sehingga pengguna dapat mengetahui kondisi sistem secara langsung.

```
206
207  // Status V hanya muncul sekali/sesaat
208  if (statusKirim == 'V' && millis() - waktuStatusKirimMuncul >= durasiTampilStatusKirim) {
209    statusKirim = '-';
210  }
211
212  lcd.setCursor(19, 1);
213  lcd.print(statusKirim);
214
215  lcd.setCursor(0, 1);
216
217  if (isnan(t) || isnan(h)) {
218    lcd.print("T/H : Error      ");
219  } else {
220    char dhtStr[20];
221    sprintf(dhtStr, "T:%.1fC H:%.0f%%   ", t, h);
222    lcd.print(dhtStr);
223  }
224
225  lcd.setCursor(0, 2);
226  lcd.print("CO2 : ");
227
228  if (co2 > 0) {
229    lcd.print(co2);
230    lcd.print(" ppm      ");
231  } else {
232    lcd.print("Error          ");
233  }
234
235  lcd.setCursor(0, 3);
236  lcd.print("Debu: ");
```

```
238    if (isZHValid) {
239      if (statusDebu == 0) {
240        lcd.print(ZH03B.getPM1_0());
241        lcd.print(" ug/m3 1.0 ");
242      } else if (statusDebu == 1) {
243        lcd.print(ZH03B.getPM2_5());
244        lcd.print(" ug/m3 2.5 ");
245      } else {
246        lcd.print(ZH03B.getPM10_0());
247        lcd.print(" ug/m3 10  ");
248      }
249    } else {
250      lcd.print("Error        ");
251    }
```

Gambar 4. 22 Program Tampilan LCD

4.2.11 Program Pengiriman Data ke Google Spreadsheet

Bagian ini menjelaskan mekanisme pengiriman data hasil pemantauan ke Google Spreadsheet melalui jaringan internet. Proses pengiriman dilakukan setiap 10 detik menggunakan metode HTTPS**.** Data suhu, kelembapan, konsentrasi CO_2, dan PM2.5 dikirim dalam bentuk query parameter menuju URL Google Apps Script. Setelah server memberikan respons, sistem akan memperbarui indikator status pengiriman pada LCD. Jika koneksi WiFi terputus, ESP32 secara otomatis menjalankan fungsi WiFi.reconnect() untuk menghubungkan kembali perangkat ke jaringan.

```
253   // 3. Kirim Data ke Google Spreadsheet
254   if (millis() - waktuKirimTerakhir >= jedaKirimData) {
255
256     if (WiFi.status() == WL_CONNECTED) {
257       Serial.println("Mengirim data ke Spreadsheet...");
258
259       WiFiClientSecure client;
260       client.setInsecure();
261       HTTPClient http;
262       String urlLengkap = GOOGLE_SCRIPT_URL +
263                           "?co2=" + String(co2) +
264                           "&temp=" + String(t) +
265                           "&hum=" + String(h) +
266                           "&dust=" + String(pm25);
267
268       http.begin(client, urlLengkap);
269       http.setFollowRedirects(HTTPC_FORCE_FOLLOW_REDIRECTS);
270
271       int httpCode = http.GET();
272       if (httpCode == 200) {
273         Serial.println("Berhasil terkirim!");
274
275         statusKirim = 'V';
276         waktuStatusKirimMuncul = millis();
277
278       } else {
279         Serial.print("Gagal mengirim. Error: ");
280         Serial.println(http.errorToString(httpCode).c_str());
281         statusKirim = 'X';
282       }
283
284       http.end();
285
286     } else {
287       Serial.println("WiFi terputus. Mencoba reconnect...");
288       statusKirim = 'X';
289       WiFi.reconnect();
290     }
291
292     waktuKirimTerakhir = millis();
293   }
294
295   delay(10);
296 }
```

Gambar 4. 23 Program Pengiriman Data ke Google Spreadsheet

4.2.12 Program Utama (loop)

Bagian ini merupakan inti dari keseluruhan program karena dijalankan secara terus-menerus selama ESP32 beroperasi. Pada setiap siklus, sistem melakukan pembacaan seluruh sensor, memperbarui tampilan LCD, memeriksa status koneksi WiFi, serta mengirimkan data ke Google Spreadsheet sesuai interval yang telah ditentukan. Penggunaan metode non-blocking berbasis fungsi millis() memungkinkan seluruh proses tersebut berjalan secara bersamaan tanpa menghambat kinerja mikrokontroler. Dengan demikian, sistem mampu melakukan monitoring kualitas udara secara *real-time*, menampilkan hasil pengukuran secara langsung, dan menyimpan data secara otomatis ke media penyimpanan berbasis cloud.

```
void loop() {

  // Membaca data dari seluruh sensor
  float t = dht.readTemperature();
  float h = dht.readHumidity();
  int co2 = mhz16.getPPM();
  bool isZHValid = ZH03B.readData();

  // Menampilkan hasil pembacaan pada LCD
  ...

  // Mengirim data ke Google Spreadsheet setiap 10 detik
  if (millis() - waktuKirimTerakhir >= jedaKirimData) {

    if (WiFi.status() == WL_CONNECTED) {

      String urlLengkap = GOOGLE_SCRIPT_URL +
                          "?co2=" + String(co2) +
                          "&temp=" + String(t) +
                          "&hum=" + String(h) +
                          "&dust=" + String(pm25);

      http.begin(client, urlLengkap);
      int httpCode = http.GET();

      http.end();
    }

    waktuKirimTerakhir = millis();
  }

  delay(10);
}void setup() {
  // put your setup code here, to run once:

}

void loop() {
```

Gambar 4. 24 Program Utama (loop)