BAB IV
PEMBUATAN ALAT

4.1 Pembuatan Perangkat Keras

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 kecepatan kendaraan menggunakan sensor photo beam berbasis ESP32 dapat dilihat pada tabel 4-1.

Tabel 4- 1 Alat yang digunakan

No	Nama Alat	Jumlah
1	Multimeter	1 Buah
2	Obeng Set	1 Buah
3	Gunting	1 Buah
4	Tang Potong	1 Buah
5	Tang kupas	1 Buah
6	Solder	1 Buah
7	Bor Tangan Set	1 Buah
8	Jangka Sorong	1 Buah
9	Gerinda	1 Set
10	Alat Tulis	1 Set

Pada Tabel 4-2 merupakan bahan yang digunakan untuk membangun rancang bangun sistem monitoring kecepatan kendaraan menggunakan sensor photo beam berbasis ESP32.

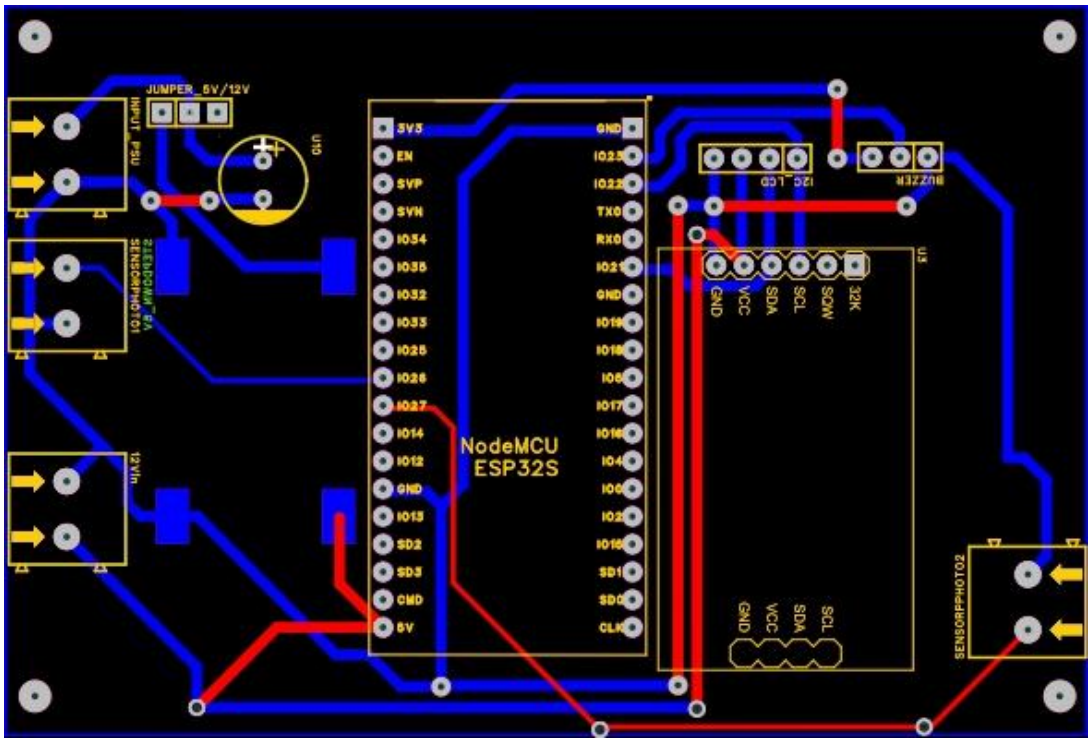
Tabel 4- 2 Bahan yang digunakan

No	Nama Bahan	Jumlah
1	ESP32	1 Buah
2	Sensor Photo Beam	1 Set
3	Panel Surya 20Wp	1 Buah
4	Baterai 12V 7Ah	1 Buah
5	SCC 12V 10A	1 Buah
6	Kabel PV	2 Meter
7	LCD 16x2 I2C	1 Buah
8	Buzzer	1 Buah
9	Modul Step-Down DC-DC	1 Buah
10	Power Supply 12V 3A	1 Buah
11	Kabel Jumper	Secukupnya
12	Kabel Power	3 Buah
13	Kapasitor 470uf	1 Buah
14	PCB	1 Buah

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 mengendalikan sistem, membaca, menggerakkan, dan mendeteksi keberhasilan pengeluaran yang diinginkan. 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. 1 Layout PCB

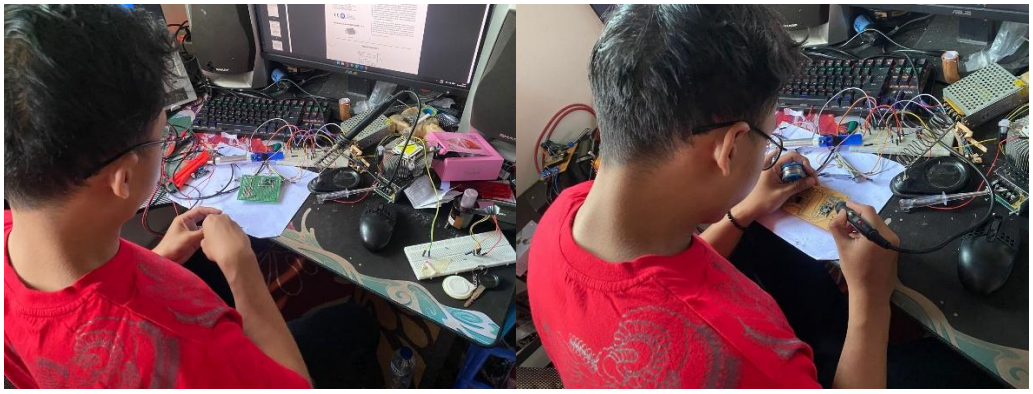
Langkah awal dimulai dengan membuat skematik rangkaian elektronik menggunakan perangkat lunak EasyEDA seperti ditunjukkan pada Gambar 4.1. Dalam skematik ini, seluruh komponen utama seperti mikrokontroler ESP32, sensor photo beam, modul catu daya (step-down dan step-up), LCD I2C, buzzer, serta konektor daya disusun secara sistematis agar hubungan antar komponen dapat dirancang dengan benar. Penyusunan skematik ini bertujuan untuk memastikan setiap jalur sinyal, jalur catu daya, dan koneksi input-output terhubung sesuai dengan kebutuhan sistem. Setelah skematik selesai dirancang, tahap selanjutnya adalah pembuatan layout PCB berdasarkan koneksi yang telah ditentukan. Proses ini meliputi penempatan komponen pada papan rangkaian, pengaturan jalur penghantar (routing), penyesuaian ukuran papan, serta pemberian label pada pin penting seperti jalur tegangan input, ground, koneksi sensor, serta pin komunikasi I2C untuk LCD. Dengan layout PCB yang tersusun rapi dan terorganisir, proses pembuatan dan perakitan rangkaian dapat dilakukan dengan lebih mudah, sekaligus meminimalkan kemungkinan kesalahan koneksi pada sistem yang dirancang.

2. Proses Pencetakan dan Penyolderan PCB

Setelah proses perancangan skematik dan layout selesai dilakukan menggunakan perangkat lunak EasyEDA, desain PCB dicetak menggunakan printer laser pada media transparan. Tahap selanjutnya adalah proses pemindahan jalur rangkaian dari hasil cetakan ke papan tembaga. Proses ini dilakukan dengan menempelkan lembar cetakan pada permukaan papan PCB, kemudian dipanaskan menggunakan setrika atau laminator agar toner menempel dan berpindah ke permukaan tembaga sesuai pola jalur yang dirancang.

Setelah proses transfer jalur berhasil, dilakukan proses pelarutan tembaga (*etching*) dengan merendam papan PCB dalam larutan ferric chloride ($FeCl_3$). Larutan ini akan mengikis bagian tembaga yang tidak terlindungi toner sehingga hanya menyisakan jalur konduktif sesuai dengan desain layout. Setelah proses etching selesai, papan PCB dibersihkan untuk menghilangkan sisa toner dan kotoran, kemudian dikeringkan sebelum tahap berikutnya.

Langkah selanjutnya adalah proses pelubangan (*drilling*) pada papan PCB menggunakan bor mini untuk membuat lubang pemasangan komponen elektronik seperti pin header, terminal konektor, dan modul pendukung. Setelah lubang selesai dibuat, komponen-komponen utama seperti mikrokontroler ESP32, konektor sensor photo beam, modul catu daya (step-down dan step-up), LCD I2C, serta buzzer dipasang pada PCB dan disolder agar terhubung secara permanen dengan jalur rangkaian. Selain itu, konektor sumber daya dipasang menggunakan terminal blok untuk memudahkan penyambungan adaptor atau sumber tegangan eksternal secara aman dan fleksibel. Konektor untuk sensor dan perangkat keluaran juga disediakan agar proses pemasangan dan perawatan sistem dapat dilakukan dengan mudah. Penempatan komponen yang rapi serta jalur rangkaian yang efisien memastikan sistem monitoring kecepatan kendaraan dapat bekerja dengan stabil dan meminimalkan risiko kesalahan koneksi selama pengoperasian.



Gambar 4. 2 Proses Pencetakan & Penyolderan PCB

3. Integrasi Sistem

Beberapa komponen pada sistem monitoring kecepatan kendaraan berbasis ESP32, seperti sensor photo beam (transmitter dan receiver), modul LCD I2C, buzzer, serta sumber catu daya, tidak disolder secara langsung ke papan PCB utama. Sebagai gantinya, komponen-komponen tersebut dihubungkan menggunakan kabel jumper, konektor terminal, atau header pin socket.

Tujuan penggunaan konektor ini adalah untuk memberikan fleksibilitas dalam penempatan komponen saat proses integrasi sistem ke dalam bentuk prototipe. Hal ini penting karena posisi sensor photo beam harus disesuaikan secara presisi agar dapat mendeteksi kendaraan yang melintas dengan akurat. Selain itu, LCD dan buzzer juga perlu ditempatkan pada posisi yang mudah dilihat dan didengar oleh pengguna.

Selain meningkatkan fleksibilitas penempatan, penggunaan konektor juga memudahkan proses perawatan dan penggantian komponen. Apabila terjadi kerusakan pada sensor, modul tampilan, atau perangkat lainnya, komponen dapat dilepas dan diganti tanpa perlu melakukan penyolderan ulang pada PCB. Hal ini sangat menguntungkan pada tahap pengembangan prototipe yang memungkinkan adanya perubahan desain atau penyesuaian sistem. Gambar 4.3 menunjukkan komponen yang telah terhubung dengan papan rangkaian utama.

Gambar 4. 3 Integrasi Sistem

4.1.3 Pembuatan Perangkat Mekanik

Desain komponen dilakukan secara menyeluruh menggunakan perangkat lunak SolidWorks. Sistem mekanik pada alat ini difokuskan pada penempatan dan kestabilan sensor photo beam yang digunakan untuk mendeteksi kendaraan yang melintas, sehingga data kecepatan dan jumlah kendaraan dapat diperoleh secara akurat.

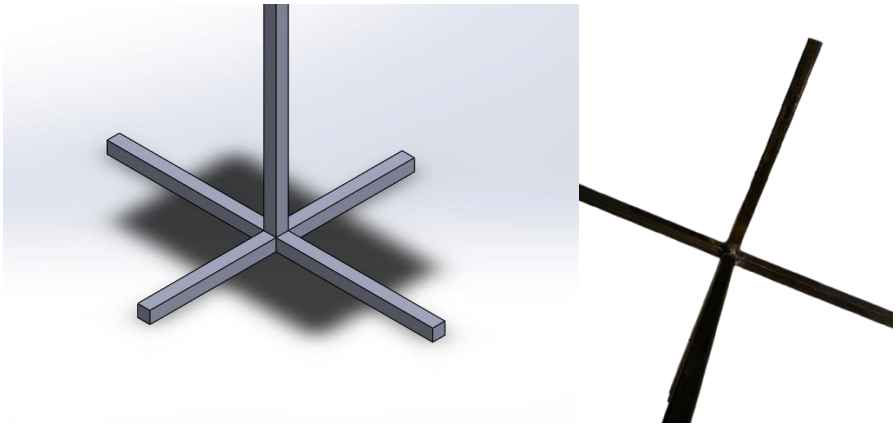
Selain itu, rancangan casing juga mencakup ruang untuk penempatan ESP32, power supply, serta rangkaian pendukung lainnya. Casing menggunakan box panel yang dirancang untuk melindungi komponen dari debu, air, dan panas, serta dilengkapi ventilasi untuk menjaga kestabilan suhu di dalamnya.

Secara keseluruhan, sistem ini terdiri dari beberapa komponen mekanik utama yang dibuat menggunakan material baja (steel) agar memiliki kekuatan dan ketahanan yang baik terhadap kondisi lapangan. Komponen-komponen tersebut meliputi:

1. Penyangga bawah (Gambar 4.4), berfungsi sebagai fondasi alat yang menopang seluruh struktur agar tetap stabil saat digunakan.
2. Tiang penyangga sensor (Gambar 4.5), digunakan sebagai dudukan untuk menempatkan sensor photo beam pada posisi yang telah ditentukan.
3. Bracket sensor Transmitter dan Receiver (Gambar 4.6), berfungsi untuk menempatkan bagian pemancar sensor secara presisi agar sejajar.

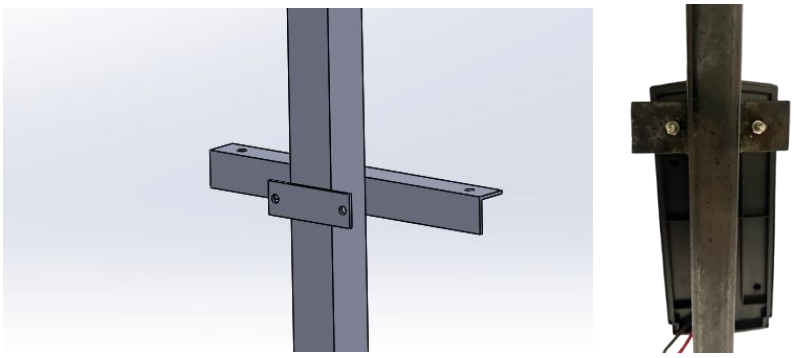
4. Box panel (Gambar 4.7), sebagai tempat penyimpanan komponen elektronik seperti ESP32, Power supply, dan rangkaian lainnya.
5. Tutup Atas (Gambar 4.8), berfungsi untuk melindungi sensor dari hujan dan panas langsung tanpa menganggu kinerja pembacaan.

Setiap komponen dirancang agar dapat dirakit menggunakan metode pengelasan maupun baut, sesuai dengan kebutuhan kekuatan dan kemudahan perawatan. Desain modular ini memudahkan proses instalasi di lapangan, perawatan, serta memungkinkan pengembangan sistem di masa mendatang.

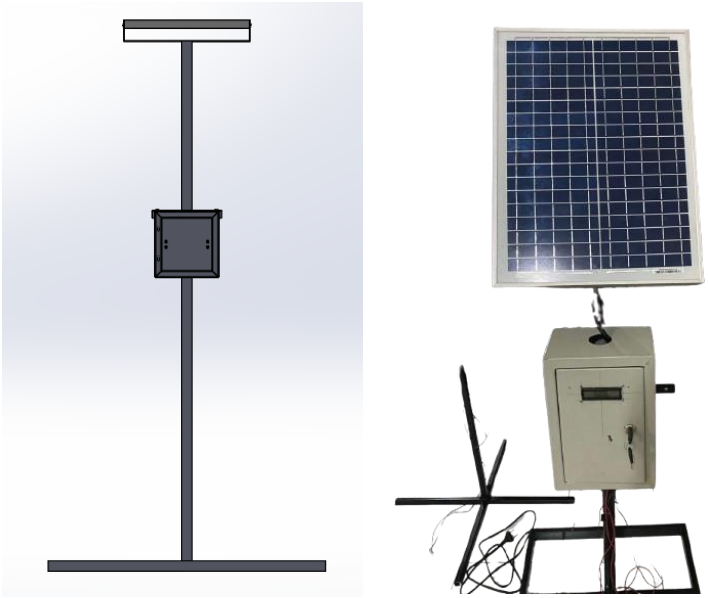


Gambar 4. 4 Penyangga Bawah

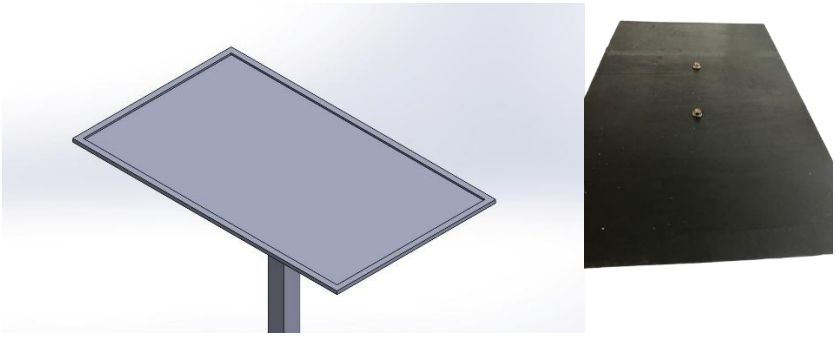
Gambar 4. 5 Penyangga Utama



Gambar 4. 6 Bracket sensor transmitter dan receiver



Gambar 4. 7 Box Panel



Gambar 4. 8 Tutup Atas

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 mengatur kerja seluruh komponen seperti sensor photo beam, ESP32, LCD I2C, dan buzzer. Struktur kode disusun secara modular untuk memudahkan pengembangan, pengujian, dan perawatan sistem.

Pembagian program yang dikembangkan meliputi beberapa bagian, yaitu:

1. Program untuk inisialisasi pin dan konfigurasi awal ESP32.
2. Program untuk membaca sinyal dari sensor photo beam, transmitter dan receiver.
3. Program untuk mendeteksi waktu saat kendaraan melewati sensor pertama dan sensor kedua
4. Program untuk menghitung kecepatan kendaraan berdasarkan selisih waktu tempuh antar sensor dan jarak yang telah ditentukan.
5. Program untuk menampilkan hasil perhitungan kecepatan pada LCD I2C.
6. Program untuk mengirim dan menyimpan data hasil pengukuran ke Microsoft Excel
7. Program untuk mengaktifkan buzzer sebagai indicator saat kendaraan terdeteksi.
8. Program looping untuk melakukan pembacaan data secara berulang secara real-time.

4.2.1 Inisialisasi pin dan konfigurasi awal ESP32

ESP32 merupakan mikrokontroler utama yang digunakan sebagai pusat pengendali pada sistem monitoring kecepatan dan jumlah kendaraan. Pada tahap awal program dilakukan proses inisialisasi pin dan konfigurasi awal ESP32 agar seluruh komponen pada sistem dapat bekerja dengan baik sesuai fungsi yang telah dirancang.

Pada proses inisialisasi, program mendefinisikan pin-pin GPIO yang digunakan untuk menghubungkan sensor photo beam, LCD I2C, buzzer, serta komponen pendukung lainnya. Pin input digunakan untuk menerima sinyal deteksi dari sensor photo beam, sedangkan pin output digunakan untuk mengendalikan

perangkat keluaran seperti buzzer dan tampilan LCD. Selain itu, dilakukan juga konfigurasi komunikasi serial sebagai media monitoring dan debugging sistem selama proses pengujian berlangsung.

Tahap konfigurasi awal ini menjadi dasar penting agar seluruh perangkat keras dapat saling terhubung dan bekerja secara sinkron dalam mendeteksi kecepatan serta menghitung jumlah kendaraan yang melintas. Program inisialisasi pin dan konfigurasi awal ESP32 yang digunakan akan dijelaskan pada kode program 4.1.

```
#include <Arduino.h>
#include <WiFi.h>
#include <RTClib.h>
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
#include <ESP_Google_Sheet_Client.h>
//inisiasi pin
int sens = 26;
int sens1 = 27;
int buzz = 23;
```

Kode Program 4. 1 Program Inisialisasi Pin dan Library ESP32

Fungsi setup() pada ESP32 digunakan untuk mengatur mode pin sebagai input dan output sesuai dengan kebutuhan sistem monitoring kecepatan dan jumlah kendaraan. Pada bagian ini, pin sensor photo beam diatur sebagai INPUT_PULLUP, sedangkan pin buzzer diatur sebagai OUTPUT. Selain itu, program juga mengatur kondisi awal buzzer menggunakan perintah digitalWrite() seperti yang dijelaskan pada kode program 4.2.

```
void setup() {
WRITE_PERI_REG(RTC_CNTL_BROWN_OUT_REG, 0);
//set pinmode
pinMode(2, OUTPUT);
pinMode(sens, INPUT_PULLUP);
pinMode(sens1, INPUT_PULLUP);
```

```
pinMode(buzz, OUTPUT);
digitalWrite(buzz, HIGH);
```

Kode Program 4. 2 Pin Arduino

Aktivasi sistem dilakukan melalui pembacaan logika HIGH/LOW pada sensor photo beam yang terhubung ke pin input ESP32. Sensor akan mendeteksi kendaraan yang melintas berdasarkan terputusnya sinar inframerah antara transmitter dan receiver. Data hasil pembacaan sensor kemudian diproses oleh ESP32 untuk menghitung kecepatan serta jumlah kendaraan yang terdeteksi. Proses pengolahan data dilakukan dengan menghitung selisih waktu antara sensor pertama dan sensor kedua, sehingga sistem dapat menentukan kecepatan kendaraan secara otomatis dengan tingkat pembacaan yang lebih akurat.

4.2.2 Pembacaan Sinyal Sensor Photo Beam

Sensor photo beam digunakan sebagai komponen utama untuk mendeteksi kendaraan yang melintas pada area pengujian. Sistem bekerja dengan membaca perubahan logika sinyal dari sensor ketika cahaya inframerah antara transmitter dan receiver terhalang oleh kendaraan. Proses pembacaan sinyal dilakukan oleh ESP32 melalui pin input yang telah dikonfigurasi sebelumnya.

Pada program, ESP32 akan terus memantau kondisi sensor untuk mengetahui waktu kendaraan melewati sensor pertama dan sensor kedua. Data waktu tersebut kemudian digunakan sebagai dasar perhitungan kecepatan kendaraan serta pencatatan jumlah kendaraan yang melintas. Pembacaan sinyal sensor dilakukan melalui fungsi digitalRead() pada masing-masing pin sensor yang telah dikonfigurasi sebelumnya sebagai input. ESP32 akan membaca kondisi sensor pertama dan sensor kedua secara terus menerus pada program utama. Program pembacaan sinyal sensor photo beam yang digunakan akan dijelaskan pada kode program 4.3

```
input = digitalRead(sens);
input1 = digitalRead(sens1);
```

Kode Program 4. 3 Program Pembacaan Sensor Photo Beam

Hasil pembacaan sensor kemudian digunakan untuk menentukan kondisi deteksi kendaraan. Ketika sensor pertama mendeteksi kendaraan lebih dahulu,

sistem akan menganggap kendaraan bergerak masuk. Sebaliknya, jika sensor kedua mendeteksi terlebih dahulu maka kendaraan dianggap bergerak keluar. Pada tahap ini sistem juga mulai menghitung waktu menggunakan fungsi millis() sebagai dasar perhitungan kecepatan kendaraan. Program pembacaan deteksi arah kendaraan yang digunakan akan di jelaskan pada kode program 4.4

```
//Pembacaan Sensor
//Sensor 1 pertama detect
if(input == LOW && LastInA == HIGH && Direction == 0){
detect = MASUK;
Cons = 2;
Direction = 1;
}
//Sensor 2 pertama detect
if(input1 == LOW && LastInB == HIGH && Direction == 0){
detect = KELUAR;
Cons = 3;
Direction = 2;
}
```

Kode Program 4. 4 Program Deteksi Arah Kendaraan

Setelah kedua sensor berhasil mendeteksi kendaraan, sistem akan menghitung selisih waktu antar pembacaan sensor untuk memperoleh data kecepatan kendaraan. Selain itu, urutan pembacaan sensor juga digunakan untuk menentukan arah pergerakan kendaraan secara otomatis. Program validasi pembacaan sensor yang digunakan akan di jelaskan pada kode program 4.5.

```
//Sensor 2 detect kedua
if(input1 == LOW && Direction == 1 && LastInB == HIGH){
detect = CALCm;
Cons = 1;
Direction = 0;
}
//Sensor 1 detect kedua
```

```
if(input == LOW && Direction == 2 && LastInA == HIGH){
detect = CALCk;
Cons = 1;
Direction = 0;
}
```

Kode Program 4. 5 Program Validasi Pembacaan Sensor

Nilai waktu hasil pembacaan sensor kemudian digunakan untuk menghitung kecepatan kendaraan berdasarkan jarak antar sensor yang telah ditentukan sebelumnya. Data hasil pembacaan selanjutnya akan ditampilkan pada LCD dan dikirim menuju Google *Spreadsheet* sebagai media monitoring berbasis IoT.

4.2.3 Deteksi Waktu Kendaraan Melintas Sensor

Deteksi waktu kendaraan melintas sensor dilakukan untuk mengetahui selisih waktu kendaraan saat melewati sensor pertama dan sensor kedua. Sistem memanfaatkan fungsi millis() pada ESP32 untuk menghitung waktu secara real-time selama proses deteksi berlangsung. Nilai waktu ini nantinya digunakan sebagai dasar dalam proses perhitungan kecepatan kendaraan.

Ketika sensor pertama mendeteksi kendaraan, sistem akan mulai menyimpan waktu awal ke dalam variabel Timer. Proses ini dilakukan untuk menandai awal kendaraan melintas pada area pengukuran. Program penyimpanan waktu awal deteksi kendaraan yang digunakan akan di jelaskan pada kode program 4.6.

```
case STAND:
seconds = Waktu/1000;
kecepatan = (2.5 / (seconds) *3.6);
lcd.clear();
lcd.setCursor(5,0);
lcd.print("STAND   ");
lcd.setCursor(0,1);
lcd.print("Speed : ");
lcd.print(kecepatan);
lcd.print("KM/H");
```

```
        Kirim = true;
        break;
        case MASUK:
        lcd.clear();
        lcd.setCursor(5,0);
        lcd.print("Detect A ");
        Timer = millis();
        Serial.print("Timer : ");
        break;
        case KELUAR:
        lcd.clear();
        lcd.setCursor(5,0);
        lcd.print("Detect B ");
        Timer = millis();
        break;
```

Kode Program 4. 6 Program Penyimpanan Waktu Awal Deteksi Kendaraan

Setelah kendaraan melewati sensor kedua, sistem akan menghitung selisih waktu menggunakan fungsi millis() untuk memperoleh waktu tempuh kendaraan antar sensor. Nilai tersebut kemudian disimpan pada variabel Waktu sebagai parameter utama dalam perhitungan kecepatan kendaraan. Program perhitungan waktu kendaraan melintas sensor yang digunakan akan di jelaskan pada kode program 4.7.

```
    case CALCm:
        lcd.setCursor(5,0);
        lcd.print("Masuk  ");
        Waktu = (millis()-Timer);
        Serial.print("Waktu : ");
        Arah = MASUKK;
        break;
    case CALCk:
        lcd.setCursor(5,0);
```

```
lcd.print("Keluar  ");
Waktu = (millis()-Timer);
Serial.print("Waktu : ");
Arah = KELUARR;
break;
```

Kode Program 4. 7 Program Perhitungan Waktu Kendaraan Melintas Sensor

Nilai waktu yang diperoleh menunjukkan lama kendaraan bergerak dari sensor pertama menuju sensor kedua. Semakin kecil nilai waktu yang dihasilkan, maka semakin tinggi kecepatan kendaraan yang melintas pada area pengukuran.

4.2.4 Perhitungan Kecepatan Kendaraan

Perhitungan kecepatan kendaraan dilakukan berdasarkan selisih waktu kendaraan saat melewati sensor photo beam pertama dan sensor kedua. Sistem memanfaatkan data waktu hasil pembacaan sensor untuk menghitung kecepatan kendaraan secara otomatis menggunakan ESP32.

Pada program, nilai kecepatan dihitung dengan membandingkan jarak antar sensor terhadap waktu tempuh kendaraan. Jarak antar sensor pada sistem ditetapkan sebesar 5 meter, sedangkan waktu diperoleh dari hasil pengurangan waktu deteksi sensor pertama dan sensor kedua menggunakan fungsi millis(). Hasil perhitungan kemudian dikonversi ke dalam satuan kilometer per jam (km/h). Program perhitungan kecepatan kendaraan yang digunakan akan di jelaskan pada kode program 4.8.

```
seconds = Waktu/1000;
kecepatan = (2.5 / (seconds) *3.6);
```

Kode Program 4. 8 Program Perhitungan Kecepatan Kendaraan

Hasil perhitungan kecepatan selanjutnya ditampilkan pada LCD sebagai media monitoring secara real-time. Selain itu, data kecepatan juga akan dikirim ke Google *Spreadsheet* untuk proses penyimpanan dan monitoring berbasis Internet of Things (IoT). Program tampilan kecepatan kendaraan pada LCD yang digunakan akan di jelaskan pada kode program 4.9.

```
lcd.setCursor(0,1);
lcd.print("Speed : ");
```

```
lcd.print(kecepatan);
lcd.print("KM/H");
```

Kode Program 4. 9 Program Tampilan Kecepatan Kendaraan pada LCD

Nilai kecepatan yang diperoleh akan terus diperbarui setiap kali kendaraan berhasil terdeteksi melewati kedua sensor photo beam. Dengan metode ini, sistem dapat melakukan monitoring kecepatan kendaraan secara otomatis dan real-time.

4.2.5 Penampilan Data Kecepatan pada LCD I2C

LCD I2C digunakan sebagai media tampilan untuk menampilkan informasi hasil pembacaan sistem secara real-time. Informasi yang ditampilkan meliputi status deteksi kendaraan, arah kendaraan, serta nilai kecepatan kendaraan yang telah dihitung oleh ESP32. Penggunaan LCD I2C bertujuan agar pengguna dapat memantau kondisi sistem secara langsung tanpa perlu menggunakan perangkat tambahan.

Pada tahap awal program, LCD diinisialisasi menggunakan library LiquidCrystal_I2C.h. Sistem kemudian mengaktifkan backlight dan menampilkan tampilan awal sebagai indikator bahwa sistem telah berhasil dijalankan. Program inisialisasi LCD I2C yang digunakan akan di jelaskan pada kode program 4.10.

```
//inisiasi lcd
lcd.init();
lcd.backlight();
lcd.setCursor(2,0);
lcd.print("DETEKSI SPEED");
lcd.setCursor(4,1);
lcd.print("KENDARAAN");
```

Kode Program 4. 10 Program Inisialisasi LCD I2C

Setelah proses deteksi kendaraan selesai dilakukan, nilai kecepatan kendaraan akan ditampilkan pada LCD secara otomatis. Sistem juga menampilkan status kendaraan berdasarkan hasil pembacaan sensor photo beam. Program penampilan data kecepatan pada LCD yang digunakan akan di jelaskan pada kode program 4.11

```
lcd.setCursor(0,1);
lcd.print("Speed : ");
lcd.print(kecepatan);
lcd.print("KM/H");
```

Kode Program 4. 11 Program Penampilan Data Kecepatan pada LCD

Selain menampilkan nilai kecepatan, LCD juga digunakan untuk menampilkan kondisi deteksi kendaraan seperti kendaraan masuk, keluar, maupun status standby sistem. Tampilan tersebut akan diperbarui secara real-time sesuai hasil pembacaan sensor yang diterima oleh ESP32.

4.2.6 Pengiriman dan Penyimpanan Data ke *Spreadsheet*

Sistem monitoring yang dirancang dilengkapi dengan fitur Internet of Things (IoT) yang memungkinkan data hasil pembacaan kendaraan dapat dikirim dan disimpan secara otomatis ke Google *Spreadsheet* melalui koneksi internet. Fitur ini bertujuan untuk memudahkan proses monitoring dan penyimpanan data kendaraan secara real-time.

Pada tahap awal, ESP32 akan terhubung ke jaringan WiFi menggunakan SSID dan password yang telah ditentukan pada program. Setelah koneksi berhasil dilakukan, sistem akan melakukan inisialisasi Google *Spreadsheet* menggunakan library ESP_Google_Sheet_Client.h. Program koneksi WiFi ESP32 yang digunakan akan di jelaskan pada kode program 4.12

```
//set wifi
const char* ssid     = "Adit";
const char* password = "adittatoan";
int terhubung = LOW;
```

Kode Program 4. 12 Program Koneksi WiFi ESP32

Setelah koneksi internet berhasil, sistem melakukan autentikasi akun Google *Spreadsheet* menggunakan PROJECT_ID, CLIENT_EMAIL, dan PRIVATE_KEY. Tahap ini diperlukan agar ESP32 dapat mengakses dan mengirim data menuju *spreadsheet* yang telah ditentukan. Program inisialisasi google spreadsheet yang digunakan akan di jelaskan pada kode program 4.13.

```
//spread Sheet
GSheet.printf("ESP Google Sheet
Clientv%s\n\n",ESP_GOOGLE_SHEET_CLIENT_VERSION);
GSheet.setPrerefreshSeconds(10 * 60);
GSheet.begin(CLIENT_EMAIL, PROJECT_ID, PRIVATE_KEY);
```

Kode Program 4. 13 Program Inisialisasi Google Spreadsheet

Data yang dikirim ke Google *Spreadsheet* meliputi waktu pembacaan kendaraan, arah kendaraan, dan nilai kecepatan kendaraan. Proses pengiriman data dilakukan menggunakan fungsi append() sehingga setiap data baru akan tersimpan pada baris berikutnya secara otomatis. Program pengiriman data ke google spreadsheet yang digunakan akan di jelaskan pada kode program 4.14

```
valueRange.add("majorDimension", "ROWS");
valueRange.set("values/[0]/[0]", "Date & Time"); // row 1/ column 1
valueRange.set("values/[0]/[1]", "Arah"); // row 2/ column 1
valueRange.set("values/[0]/[2]", "Kecepatan"); // row 3/ column 1
valueRange.set("values/[0]/[3]", "Selisih Waktu");
bool success = GSheet.values.append(&response /* returned response */, spreadsheetId /* spreadsheet Id to update */, "Sheet1!A1" /* range to update */, &valueRange /* data to update */);
```

Kode Program 4. 14 Program Pengiriman Data ke Google Spreadsheet

Data hasil monitoring yang tersimpan pada Google *Spreadsheet* dapat digunakan sebagai media pencatatan dan analisis lalu lintas kendaraan secara digital dan real-time.

4.2.7 Pengaktifan Buzzer sebagai Indikator

Buzzer digunakan sebagai indikator suara pada sistem untuk memberikan tanda ketika kendaraan terdeteksi oleh sensor photo beam. Penggunaan buzzer bertujuan untuk memberikan notifikasi secara langsung bahwa sistem berhasil mendeteksi kendaraan yang melintas pada area pengukuran.

Pada tahap awal program, pin buzzer dikonfigurasi sebagai output menggunakan fungsi pinMode(). Selain itu, sistem juga mengatur kondisi awal

buzzer dalam keadaan tidak aktif menggunakan perintah digitalWrite(). Program pada inisialisasi buzzer yang digunakan akan dijelaskan pada kode program 4.15

```
pinMode(buzz, OUTPUT);
digitalWrite(buzz, HIGH);
```

Kode Program 4. 15 Program Inisialisasi Buzzer

Buzzer akan aktif ketika salah satu sensor photo beam mendeteksi kendaraan yang melintas. Sistem menggunakan logika LOW dan HIGH untuk mengontrol kondisi buzzer melalui ESP32. Saat kendaraan terdeteksi, buzzer akan menyala selama waktu tertentu berdasarkan nilai timer yang telah ditentukan pada program. Program pada pengaktifan buzzer yang digunakan akan dijelaskan pada kode program 4.16

```
if (input == LOW  && LastInA == HIGH  && Active == false) {
digitalWrite(buzz, LOW);
Start = millis();
Active = true;
Serial.println("Buzzer On");
}
if (input1 == LOW  && LastInB == HIGH  && Active == false) {
digitalWrite(buzz, LOW);
Start = millis();
Active = true;
Serial.println("Buzzer On");
}
```

Kode Program 4. 16 Program Pengaktifan Buzzer

Setelah durasi buzzer selesai, sistem akan mematikan buzzer secara otomatis menggunakan fungsi millis() sebagai pengatur waktu. Dengan metode ini, buzzer hanya aktif sesaat ketika kendaraan berhasil terdeteksi oleh sensor. Program pada penonaktifan buzzer yang digunakan akan dijelaskan pada kode program 4.17

```
if (Active == true && (millis() - Start) >= Durasi) {
digitalWrite(buzz, HIGH);
Active = false;
```

```
Serial.println("Buzzer Off");
```

Kode Program 4. 17 Program Penonaktifan Buzzer

4.2.8 Sistem Looping untuk Monitoring Real-Time

Sistem monitoring kecepatan kendaraan bekerja secara terus menerus menggunakan fungsi loop() pada ESP32. Fungsi ini berperan sebagai proses utama yang menjalankan seluruh pembacaan sensor, perhitungan kecepatan, pengaktifan buzzer, penampilan data pada LCD, serta pengiriman data ke Google *Spreadsheet* secara real-time. Dengan adanya sistem looping, ESP32 dapat melakukan monitoring kendaraan secara berulang tanpa perlu menjalankan ulang program secara manual.

Pada program, fungsi loop() akan membaca kondisi sensor photo beam secara terus menerus menggunakan fungsi digitalRead(). Data hasil pembacaan kemudian diproses untuk menentukan arah kendaraan, menghitung waktu tempuh, dan memperoleh nilai kecepatan kendaraan. Program pada looping pembacaan sensor yang digunakan akan dijelaskan pada kode program 4.18

```
int BUZZER;
int input;
int input1;
input = digitalRead(sens);
input1 = digitalRead(sens1);
```

Kode Program 4. 18 Program Looping Pembacaan Sensor

Selain melakukan pembacaan sensor, fungsi loop() juga digunakan untuk menjalankan proses monitoring sistem secara keseluruhan, seperti pengaktifan buzzer, pembaruan tampilan LCD, dan pengiriman data ke Google *Spreadsheet* ketika kendaraan berhasil terdeteksi. Program pada looping pengiriman data yang digunakan akan dijelaskan pada kode program 4.19

```
if(Kirim && GSheet.ready()){
SpreadSheet();
Kirim = false;
}
```

Kode Program 4. 19 Program Looping Pengiriman Data

Dengan metode looping ini, sistem dapat bekerja secara otomatis dan real-time dalam melakukan monitoring kendaraan yang melintas pada area pengukuran. Seluruh proses akan terus berjalan selama ESP32 mendapatkan catu daya dan sistem tetap aktif.