Kontrol dan Monitoring Aktivitas Motor DC Menggunakan Android Memanfaatkan Komunikasi Serial Bluetooth HC-05

Kelompok 3:

1. Ach.Chusnul Chikam (1210151032)
2. Aulia Rahma NF (1210151049)
3. Alvitariani Khairinia Kusuma (1210151054)
4. Diah Ayu Pitaloka (1210151057)
5. Nurul Hidayah (1210151059)

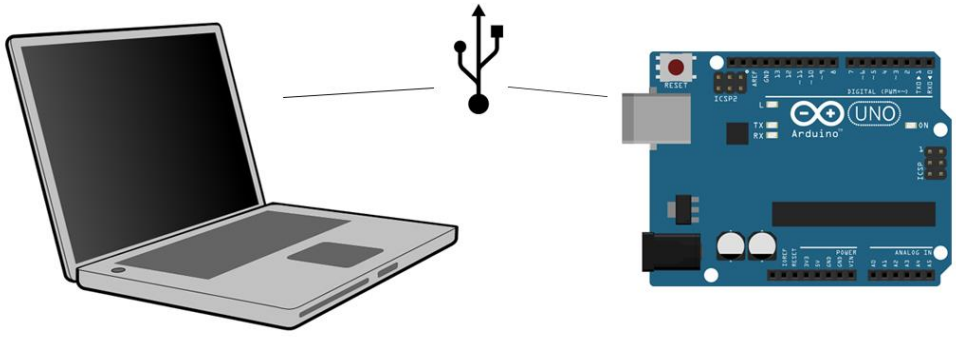
I. TUJUAN

- Dapat mengetahui cara kerja komunikasi serial
- Dapat menghubungkan arduino dengan android dengan menggunakan Bluetooth HC-05
- Dapat mengontrol aktivitas motor dc menggunakan Bluetooth HC-05 sebagai komunikasi serial
- Dapat memantau aktivitas motor dc dari android

II. DASAR TEORI

2.1 Komunikasi Serial

Komunikasi serial adalah komunikasi yang pengiriman datanya per-bit secara berurutan dan bergantian. Komunikasi ini mempunyai suatu kelebihan yaitu hanya membutuhkan satu jalur dan kabel yang sedikit dibandingkan dengan komunikasi paralel. Pada prinsipnya komunikasi serial merupakan komunikasi dimana pengiriman data dilakukan per bit sehingga lebih lambat dibandingkan komunikasi parallel, atau dengan kata lain komunikasi serial merupakan salah satu metode komunikasi data di mana hanya satu bit data yang dikirimkan melalui seuntai kabel pada suatu waktu tertentu. Pada dasarnya komunikasi serial adalah kasus khusus komunikasi paralel dengan nilai n = 1, atau dengan kata lain adalah suatu bentuk komunikasi paralel dengan jumlah kabel hanya satu dan hanya mengirimkan satu bit data secara simultan.Hal ini dapat disandingkan dengan komunikasi paralel yang sesungguhnya di mana n-bit data dikirimkan bersamaan, dengan nilai umumnya $8 \leq n \leq 128$.

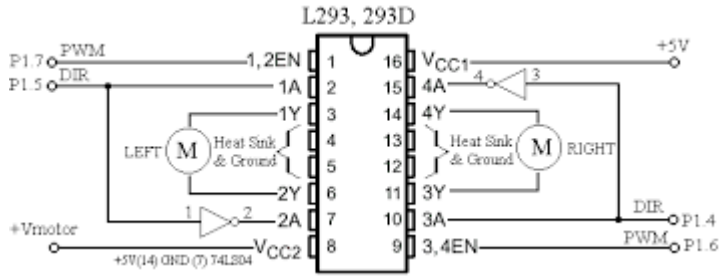


Komunikasi serial ada dua macam, *asynchronous serial* dan *synchronous serial*. *Synchronous serial* adalah komunikasi dimana hanya ada satu pihak (pengirim atau penerima) yang menghasilkan clock dan mengirimkan clock tersebut bersama-sama dengan data. Contoh pengunaan *synchronous serial* terdapat pada transmisi data keyboard. *Asynchronous serial* adalah komunikasi dimana kedua pihak (pengirim dan penerima) masing-masing

menghasilkan clock namun hanya data yang ditransmisikan, tanpa clock. Agar data yang dikirim sama dengan data yang diterima, maka kedua frekuensi clock harus sama dan harus terdapat sinkronisasi. Setelah adanya sinkronisasi, pengirim akan mengirimkan datanya sesuai dengan frekuensi clock pengirim dan penerima akan membaca data sesuai dengan frekuensi clock penerima. Contoh penggunaan asynchronous serial adalah pada Universal Asynchronous Receiver Transmitter (UART) yang digunakan pada serial port (COM) komputer.

Salah satu penerapan komunikasi serial adalah dengan Bluetooth. Pada project yang dibuat, kami menggunakan motor dc sebagai output komunikasi serial. Komponen yang digunakan diantarannya

2.2 Driver Motor DC (L293D)



Driver motor digunakanuntukmengontrolarahputarandankecepatan motor DC yang merupakanpenggerakutamadarirangkaianproyekakhirini. IC driver motor L293 yang didalamnya terdapat rangkaian *H-Bridge*akanmengontrolputaran motor sesuai data masukan digital yang berasaldariPLC Zelio SR2 B201 BD, danpada IC L293 inijugaterdapat pin untuk pengaturanaplikasi PWM *(Pulse Width Modulator)* yang akan mengatur kecepatan motor dc yang dikendalikannya. L293 memilikirangkaian dual H-Bridge, sehingga mampu mengendalikan dua buah motor DC sekaligus.

Karakteristikdari driver motor L293 adalah:

1. Tegangan operasi *supply* sampai dengan 36 Volt.
2. Total arus DC sampai dengan 1A.
3. Tegangan *logic* ”0” sampaidengan 1,5 Volt.
4. Memilikidua*Enable input*.

Fungsi dari tiap-tiap pin driver motor L293 adalah sebagai berikut:

1. *Output*1 dan Output 2 (pin 3 dan pin 6)

Pin inimerupakan *output* untuk bridge A.

2. Vs (pin 8)

   Merupakan pin supply tegangan untuk *output*.

3. Input 1 dan Input 2 (pin 2 dan pin 7)

   Pin inidigunakanuntukmengontrol bridge A.

4. *Enable 1*dan*Enable 2* (pin 1 dan pin 9)

   Pin iniberfungsi untuk mengaktifkan dan menonaktifkan bridge A dan bridge B.

5. Ground (pin 4, 5, 12, dan 13)

   Berfungsi sebagai *grounding* rangkaian driver.

6. Vss (pin 16)

   Pin ini berfungsi sebagai *supply logic*untuk driver.

7. Input 3 dan Input 4 (pin 10 dan 15)

   Berfungsi sebagai masukan pada bridge B.

8. Output 3 dan Output 4 (11 dan 14)

   Merupakan pin *output* untuk bridge B.

IC L293D adalah IC yang didesain khusus sebagai driver motor DC dan dapat dikendalikan dengan rangkaian mikrokontroller.Dalam 1 unit chip IC L293D terdiri dari 4 buah driver motor DC yang berdiri sendiri sendiri dengan kemampuan mengalirkan arus 1 Ampere tiap drivernya.Dalam IC 293 terdiri dari 16 pin ini dapat digunakan untuk 2 kendali motor H-Bridge (Bergerak Kiri & Kanan)

2.3 Motor DC

Motor DC adalah motor listrik yang memerlukan suplai tegangan arus searah pada kumparan medan untuk diubah menjadi energi gerak mekanik. Kumparan medan pada motor dc disebut stator (bagian yang tidak berputar) dan kumparan jangkar disebut rotor (bagian yang berputar). Motor arus searah, sebagaimana namanya, menggunakan arus langsung yang tidak langsung/direct-unidirectional. Motor DC memiliki 3 bagian atau komponen utama untuk dapat berputar sebagai berikut.

Bagian Atau Komponen Utama Motor DC

- **Kutub medan**. Motor DC sederhana memiliki dua kutub medan: kutub utara dan kutub selatan. Garis magnetik energi membesar melintasi ruang terbuka diantara kutub-kutub dari utara ke selatan. Untuk motor yang lebih besar atau lebih komplek terdapat satu atau lebih elektromagnet.

- **Current Elektromagnet atau Dinamo.** Dinamo yang berbentuk silinder, dihubungkan ke as penggerak untuk menggerakan beban. Untuk kasus motor DC yang kecil, dinamo berputar dalam medan magnet yang dibentuk oleh kutub-kutub, sampai kutub utara dan selatan magnet berganti lokasi.
- **Commutator**. Komponen ini terutama ditemukan dalam motor DC. Kegunaannya adalah untuk transmisi arus antara dinamo dan sumber daya.

Keuntungan utama motor DC adalah sebagai pengendali kecepatan, yang tidak mempengaruhi kualitas pasokan daya. Motor ini dapat dikendalikan dengan mengatur:

- Tegangan dinamo – meningkatkan tegangan dinamo akan meningkatkan kecepatan
- Arus medan – menurunkan arus medan akan meningkatkan kecepatan.

2.4 LED

Diode pancaran cahaya (bahasa Inggris: light-emitting diode; LED) adalah suatu semikonduktor yang memancarkan cahaya monokromatik yang tidak koheren ketika diberi tegangan maju. Gejala ini termasuk bentuk elektroluminesensi. Warna yang dihasilkan bergantung pada bahan semikonduktor yang dipakai, dan bisa juga ultraviolet dekat atau inframerah dekat.

2.5 ADC

ADC (*Analog To Digital Converter*) adalah perangkat elektronika yang berfungsi untuk mengubah sinyal analog (sinyal kontinyu) menjadi sinyal digital. Perangkat ADC (*Analog To Digital Convertion*) dapat berbentuk suatu modul atau rangkaian elektronika maupun suatu chip IC.

ADC memiliki 2 karakter prinsip yakni kecepatan sampling dan resolusi.

- **Kecepatan sampling suatu ADC** menyatakan "seberapa sering sinyal analog dikonversikan ke bentuk sinyal digital pada selang waktu tertentu". Kecepatan sampling biasanya dinyatakan dalam ***sample per second*** **(SPS)**.
- **Resolusi ADC** menentukan "ketelitian nilai hasil konversi ADC". Sebagai contoh: ADC 8 bit akan memiliki output 8 bit data digital, ini berarti sinyal input dapat dinyatakan dalam 255 (2n–1) nilai diskrit. ADC 12 bit memiliki 12 bit output data digital, ini berarti sinyal input dapat dinyatakan dalam 4096 nilai diskrit. Dari contoh diatas ADC 12 bit akan memberikan ketelitian nilai hasil konversi yang jauh lebih baik daripada ADC 8 bit.

Prinsip kerja: **Prinsip kerja ADC** adalah mengkonversi sinyal analog ke dalam bentuk besaran yang merupakan rasio perbandingan sinyal input dan tegangan referensi.
Sebagai contoh, bila tegangan referensi 5 volt, tegangan input 3 volt, rasio input terhadap referensi adalah 60%. Jadi, jika menggunakan ADC 8 bit dengan skala maksimum 255, akan didapatkan sinyal digital sebesar 60% x 255 = 153 (bentuk decimal) atau 10011001 (bentuk biner). signal = (sample/max_value) * reference_voltage

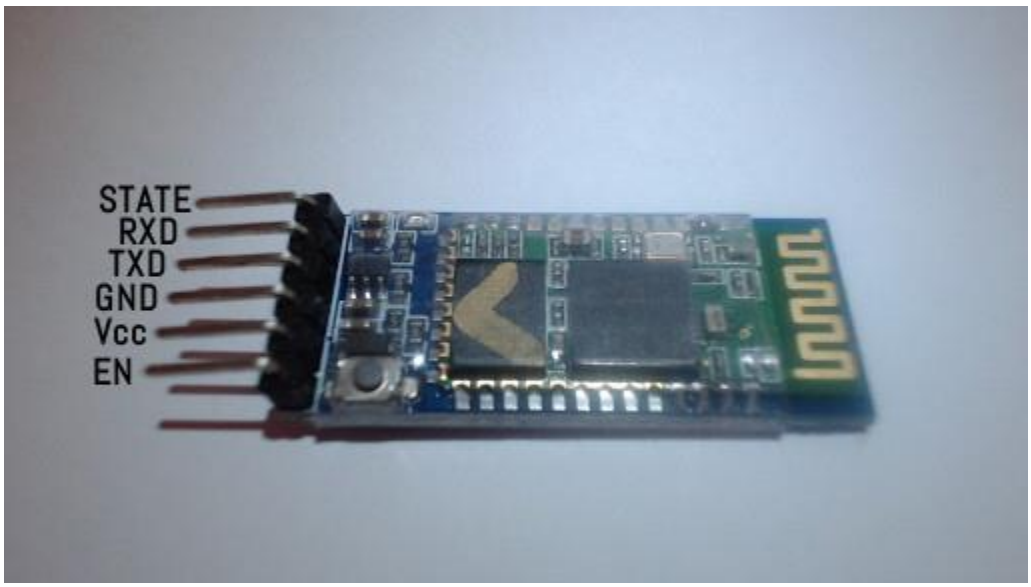
= (153/255) * 5

= 3 Volts

ADC pada mikrokontroler hanya mengubah nilai tegangan analog yang masuk menjadi suatu nilai ADC yang memiliki range tertentu. Dan untuk project yang kami gunakan ini fungsi dari ADC tidak hanya untuk pengkonverisan dari nilai tegangan analog semata, namun juga sebagai penggerak kecepatan dari motor DC.

2.6 Bluetooth HC-05

Bluetooth Module HC-05 merupakan module komunikasi nirkabel pada frekuensi 2.4GHz dengan pilihan koneksi bisa sebagai slave, ataupun sebagai master. Sangat mudah digunakan dengan mikrokontroler untuk membuat aplikasi wireless. Interface yang digunakan adalah serial RXD, TXD, VCC dan GND. Built in LED sebagai indikator koneksi bluetooth.



Keterangan pinout di atas adalah sebagai berikut:

- **EN** fungsinya untuk mengaktifkan mode *AT Command Setup* pada modul HC-05. Jika pin ini ditekan sambil ditahan sebelum memberikan tegangan ke modul HC-05, maka modul akan mengaktifkan mode *AT Command Setup*. Secara default, modul HC-05 aktif dalam mode *Data*.
- **Vcc** adalah pin yang berfungsi sebagai input tegangan. Hubungkan pin ini dengan sumber tegangan 5V.
- **GND** adalah pin yang berfungsi sebagai *ground*. Hubungkan pin ini dengan *ground* pada sumber tegangan.
- **TX** adalah pin yang berfungsi untuk mengirimkan data dari modul ke perangkat lain (mikrokontroler). Tegangan sinyal pada pin ini adalah 3.3V sehingga dapat langsung

dihubungkan dengan pin RX pada arduino karena tegangan sinyal 3.3V dianggap sebagai sinyal bernilai **HIGH** pada arduino.

- **RX** adalah pin yang berfungsi untuk menerima data yang dikirim ke modul HC-05. Tegangan sinyal pada pin sama dengan tegangan sinyal pada pin TX, yaitu 3.3V. Untuk keamanan, sebaiknya gunakan pembagi tegangan jika menghubungkan pin ini dengan arduino yang bekerja pada tegangan 5V. Pembagi tegangan tersebut menggunakan 2 buah resistor. Resistor yang digunakan sebagai pembagi tegangan pada tutorial ini adalah 1K ohm dan 2K ohm. Untuk lebih jelasnya, dapat dilihat pada bagian implementasi koneksi antara modul HC-05 dan arduino UNO.
- **STATE** adalah pin yang berfungsi untuk memberikan informasi apakah modul terhubung atau tidak dengan perangkat lain.

Seperti dijelaskan di atas, modul HC-05 memiliki dua mode kerja yaitu mode *AT Command* dan mode *Data*. Modul HC-05 menggunakan mode *Data* secara default. Berikut ini adalah keterangan untuk kedua mode tersebut:

- **AT Command**. Pada mode ini, modul HC-05 akan menerima instruksi berupa perintah *AT Command*. Mode ini dapat digunakan untuk mengatur konfigurasi modul HC-05. Perintah *AT Command* yang dikirimkan ke modul HC-05 menggunakan huruf kapital dan diakhiri dengan karakter CRLF (*\r\n* atau *0x0d 0x0a* dalam heksadesimal).
- **Data**. Pada mode ini, modul HC-05 dapat terhubung dengan perangkat bluetooth lain dan mengirimkan serta menerima data melalui pin TX dan RX. Konfigurasi koneksi serial pada mode ini menggunakan baudrate: 9600 bps, data: 8 bit, stop bits: 1 bit, parity: None, handshake: None. Adapun password default untuk terhubung dengan modul HC-05 pada mode *Data* adalah 0000 atau 1234.

Penerapan Bluetooth pada project:

- ✓ Sebagai pengirim dan penerima data
- ✓ Untuk menampilkan data

2.7 Gearbox Plus Roda

Gearbox merupakan suatu alat khusus yang diperlukan untuk menyesuaikan daya atau torsi (momen/daya) dari motor yang berputar, dangan gearbox juga adalah alat pengubah daya dari motor yang berputar menjadi tenaga yang lebih besar

Gearbox atau transmisi adalah salah satu komponen utama motor yang disebut sebagai sistem pemindah tenaga, transmisi berfungsi untuk memindahkan dan mengubah tenaga dari motor yang berputar, yang digunakan untuk memutar spindel mesin maupun melakukan gerakan feeding. Transmisi juga berfungsi untuk mengatur kecepatan gerak dan torsi serta berbalik putaran, sehingga dapat bergerak maju dan mundur.

Transmisi manual atau lebih dikenal dengan sebutan gearbox, mempunyai beberapa fungsi antara lain :

1. Merubah momen puntir yang akan diteruskan ke spindel mesin.

2. Menyediakan rasio gigi yang sesuai dengan beban mesin.

3. Menghasilkan putaran mesin tanpa selip

Prinsip Kerja Gearbox

Putaran dari motor diteruskan ke input shaft (poros input) melalui hubungan antara clutch/ kopling, kemudian putaran diteruskan ke main shaft (poros utama), torsi/ momen yang ada di mainshaft diteruskan ke spindel mesin, karena adanya perbedaan rasio dan bentuk dari gigi-gigi tersebut sehingga rpm atau putaran spindel yang di keluarkan berbeda, tergantung dari rpm yang di inginkan. Berikut penjelasan beberapa part yang terdapat dalam gearbox.

- Input shaft (poros input)

Input shaft adalah komponen yang menerima momen output dari unit kopling, poros input juga befungsi untuk meneruskan putaran dari clutch kopling ke mainshaft (poros utama), sehingga putaran bisa di teruskan ke gear-gear. Input shaft juga sebagai poros dudukan bearing dan piston ring, selain itu berfungsi juga sebagai saluran oli untuk melumasi bagian dari pada inputshaft tersebut.

- Gear shift housing (rumah lever pemindah rpm)

Gear shift housing adalah housing dari pada lever pemindah gigi yang berfungsi untuk mengatur ketepatan perpindahan gigi, apabila gigi sudah dipindahkan maka lever akan terkunci sehingga lever tidak bisa berpindah sendiri pada saat spindel sedang berputar.

- Main shaft (poros utama)

Mainshaft yang berfungsi sebagai tempat dudukan gear, sinchromest, bearing dan komponen-komponen lainnya. Main shaft juga berfungsi sebagai poros penerus putaran dari input

shaft sehingga putaran dapat di teruskan ke spindel, main shaft juga berfungsi sebagai saluran tempat jalannya oli.

- Planetary gear section (unit gigi planetari)

Planetary adalah alat pengubah rpm di suatu range tertentu dimana rpm dapat di ubah sesuai dengan kebutuhan proses pengerjaan dan dapat pula mengubah arah putaran spindel.

- Oil pump assy (pompa oli)

Oil pump berfungsi untuk memompa dan memindahkan oli dari transmisi case (rumah transmisi) menuju ke sistem untuk dilakukan pelumasan terhadap komponen-komponen yang ada di dalam transmisi secara menyeluruh.

- Clucth housing

Clutch housing adalah rumah dari clucth kopling yang berfungsi sebagai pelindung clutch kopling, clutch housing juga berfungsi sebagai tempat dudukan dari pada oil pump dan input shaft.

- Transmisi gear/ roda gigi transmisi

Transmisi gear atau roda gigi transmisi berfungsi untuk mengubah input dari motor menjadi output gaya torsi yang meninggalkan transmisi sesuai dengan kebutuhan mesin.

- Bearing

Bearing berfungsi untuk menjaga kerenggangan dari pada shaft (poros), agar pada saat unit mulai bekerja komponen yang ada di dalam transmisi tidak terjadi kejutan, sehingga transmisi bisa bekerja dengan smooth (halus).

- Piston ring (ring penyekat oli).

Piston ring berfungsi sebagai penyekat agar tidak terjadi kebocoran pada sistem pelumasan, piston ring juga berfungsi sebagai pengencang input shaft agar input shaft tidak rengang pada saat unit berjalan.

- Sun gear (gigi matahari)

Sun gear berfungsi untuk meneruskan putaran ke planetary gear section. Sun gear berhubungan langsung dengan gear yang ada pada unit planetary yang berfungsi sebagai penerus putaran, momen dari transmisi.

III. KOMPONEN DAN PERALATAN

3.1 Komponen

3.1.1 Mikrokontroler arduino uno

3.1.2 Motor dc

3.1.3 LED

3.1.4 IC LS392D

3.1.5 Gear Box

3.1.6 Roda bebas

3.1.7 Roda

3.2 Alat

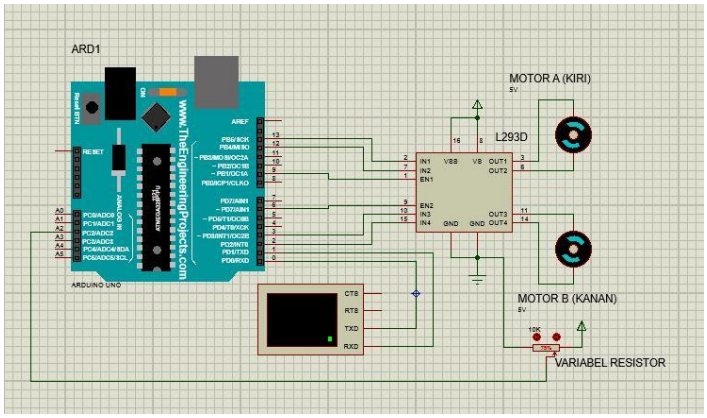
3.2.1 Kabel Jumper

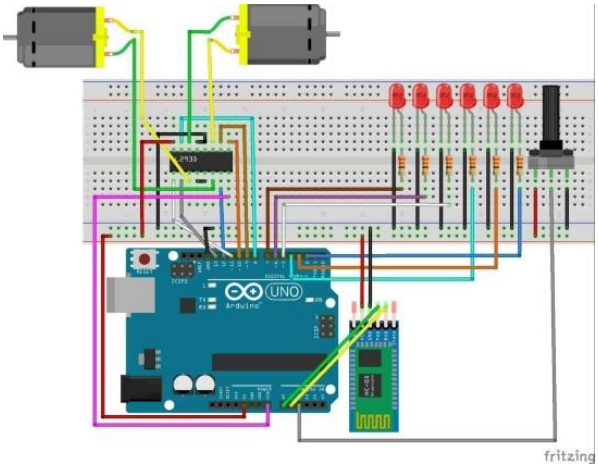
3.2.2 PCB layout

3.2.3 Solder dan Timah

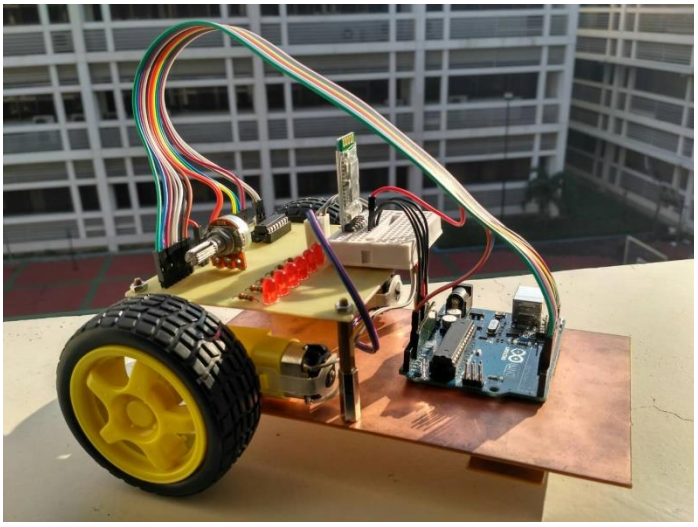
IV.GAMBAR RANGKAIAN

4.1 Simulasi Dan Gambar Rangkaian

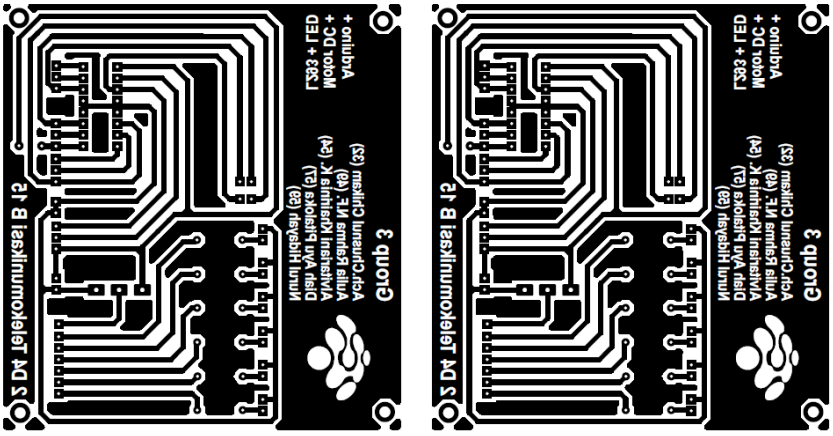




4.2 Hardware Project



4.4 Desain Layout PCB

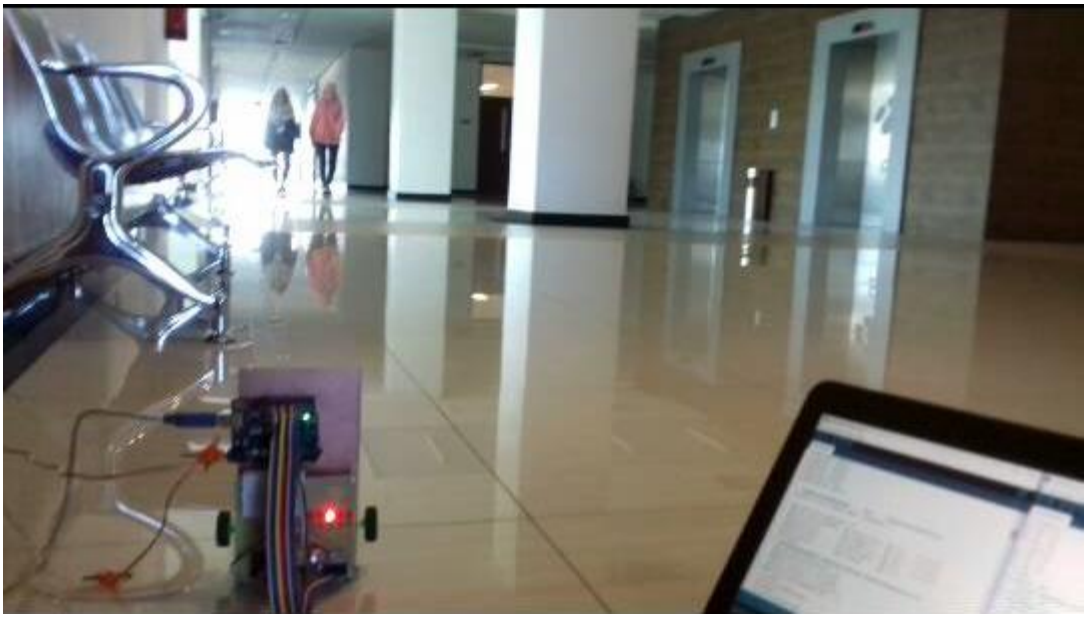


V. PENGAMBILAN DATA

Pada project ini, kami mengambil data mengenai Bluetooth serial. Dimana kita dapat berkomunikasi serial melalui bluetooth untuk mengontrol aktivitas motor DC. Sehingga motor dc dapat berputar ke kanan, ke kiri, dan berhenti.

Dalam komunikasi serialnya, terdapat beberapa pilihan untuk mengatur motor DC tersebut diantaranya:

1. Motor DC berjalan maju
2. Motor DC berjalan mundur
3. Motor kanan ON, kiri Off (belok kiri)
4. Motor kiri ON, kanan Off (belok kanan)
5. Motor kiri dan kanan Off, Animasi Led



Selain itu, data yang berhasil kami dapatkan adalah jarak yang mampu ditempuh oleh Bluetooth tersebut. Dengan menggunakan Bluetooth, kami dapat mengontrol motor dc dengan jarak yang lumayan jauh yaitu sekitar +- 69 meter.Namun pada jarak yang jauh tersebut, komunikasi serial semakin melambat dibandingkan pada saat mengirim di jarak yang dekat

Kemudian juga ditampilkan kecepatannya apabila kami memutar potensiometer kearah kanan, maka motor dc akan berputar semakin cepat. Apabila diputar kearah kiri maka motor dc akan berputar semakin lambat. Cara untuk mengkonversikan nilai tegangan ke kecepatannya adalah :

Saat kecepatan = 200

Maka tegangannya : $\frac{x}{9} \times 254 = 200$

$254\ x = 1800$

$x = 7.08$ V

Saat kecepatan = 150

Maka tegangannya : $\frac{x}{9} \times 254 = 150$

$254\ x = 1350$

$x = 5.315$ V

Saat kecepatan = 100

Maka tegangannya : $\frac{x}{9} \times 254 = 100$

$254\ x = 900$

$x = 3.54$ V

Saat kecepatan = 50

Maka tegangannya : $\frac{x}{9} \times 254 = 50$

$254\ x = 450$

$x = 1.771$ V

Dari pengamatan yang telah kami lakukan, ketika kita mengatur ADC dibawah 100, roda tidak bergerak namun motor dc dapat berputar.

Berdasarkan perhitungan delay animasi LED pada project kami yaitu selama 56.9 detik. Namun, secara realnya, project kami menghasilkan delay animasi LED selama 1 menit. Hal tersebut mungkin dikarenakan beberapa factor. Seperti jauh atau dekatnya jarak saat berkomunikasi.

VII. LISTING PROGRAM

```
#include <SoftwareSerial.h>
  // motor A
int enA = 11;
int in1 = 13;
int in2 = 12;
  // motor B
int enB = 10;
int in3 = 9;
int in4 = 8;
  // adc
int potPin = A2;
  // variabel pembacaan nilai adc
int potValue = 0;
int motorSpeed = 0;
  // variabel pembacaan input keyboard
int data;
  // led
int led[]={7,6,5,4,3,2};

SoftwareSerial mySerial(A0, A1); // RX, TX

void setup() {
  // Open serial communications and wait for port to open
  // set the data rate for the SoftwareSerial port
  mySerial.begin(9600);
  mySerial.println("Bluetooth Communication Is Ready");
  for(int i=0; i<6; i++){
```

```
  pinMode(led[i], OUTPUT);
 }
 pinMode(enA, OUTPUT);
 pinMode(enB, OUTPUT);
 pinMode(in1, OUTPUT);
 pinMode(in2, OUTPUT);
 pinMode(in3, OUTPUT);
 pinMode(in4, OUTPUT);
 pinMode(potPin, INPUT);

// ########################     INTRO     ########################
  mySerial.println("PROJECT MIKRO & ANTARMUKA 1");

  mySerial.println("\nKelompok 3");
  mySerial.println("Anggota:  Ach.Chusnul Chikam  NRP 32");    delay(400);
  mySerial.println("         Aulia Rahmah N.F.   NRP 49");    delay(400);
  mySerial.println("         Alvita Khairinia K. NRP 54");    delay(400);
  mySerial.println("         Diah Ayu Pitaloka   NRP 57");    delay(400);
  mySerial.println("         Nurul Hidayah       NRP 59");
  delay(400);
// ##################### PENGECEKAN KOMPONEN ########################
   mySerial.print("Inisialisasi...");
   mySerial.print("1..");  delay(100);
   mySerial.print("2..");  delay(100);
   mySerial.println("3..");  delay(100);
  mySerial.println("Pengecekan LED.....");
  delay(600);
  digitalWrite(led[0],HIGH);
```

```
delay(400);
digitalWrite(led[0],LOW);
digitalWrite(led[1],HIGH);
delay(400);
digitalWrite(led[1],LOW);
digitalWrite(led[2],HIGH);
delay(400);
digitalWrite(led[2],LOW);
digitalWrite(led[3],HIGH);
delay(400);
digitalWrite(led[3],LOW);
digitalWrite(led[4],HIGH);
delay(400);
digitalWrite(led[4],LOW);
digitalWrite(led[5],HIGH);
delay(400);
digitalWrite(led[5],LOW);
delay(400);
mySerial.println(" Pengecekan LED Selesai...");
delay(600);
mySerial.println("Selanjutnya..");
mySerial.println(" Pengecekan Motor DC.....");
delay(600);
digitalWrite(in1, LOW);
digitalWrite(in2, HIGH);
digitalWrite(in3, HIGH);
digitalWrite(in4, LOW);
// kecepatan 250
```

```
    analogWrite(enA, 250);
    analogWrite(enB, 250);
    delay(600);
    digitalWrite(in1, HIGH);
    digitalWrite(in2, LOW);
    digitalWrite(in3, LOW);
    digitalWrite(in4, HIGH);
    // kecepatan 250
    analogWrite(enA, 250);
    analogWrite(enB, 250);
    delay(600);
    // Motor DC mati
    digitalWrite(in1, LOW);
    digitalWrite(in2, LOW);
    digitalWrite(in3, LOW);
    digitalWrite(in4, LOW);
    mySerial.println("");
    delay(200);
    mySerial.println(" UNTUK MENGETES KOMUNIKASI SERIAL ANTARA ARDUINO
DENGAN BLUETOOTH,");
    mySerial.println(" BERI MASUKAN!!!");
}
//  ##################### FUNGSI INDIKATOR LED ######################
void ledMaju(){
  digitalWrite(led[0], LOW);
  digitalWrite(led[1], HIGH);
  digitalWrite(led[2], LOW);
  digitalWrite(led[3], LOW);
  digitalWrite(led[4], LOW);
```

```
  digitalWrite(led[5], LOW);
}
void ledMundur(){
  digitalWrite(led[0], LOW);
  digitalWrite(led[1], LOW);
  digitalWrite(led[2], HIGH);
  digitalWrite(led[3], LOW);
  digitalWrite(led[4], LOW);
  digitalWrite(led[5], LOW);
}
void ledKanan(){
  digitalWrite(led[0], LOW);
  digitalWrite(led[1], LOW);
  digitalWrite(led[2], LOW);
  digitalWrite(led[3], HIGH);
  digitalWrite(led[4], LOW);
  digitalWrite(led[5], LOW);
}
void ledKiri(){
  digitalWrite(led[0], LOW);
  digitalWrite(led[1], LOW);
  digitalWrite(led[2], LOW);
  digitalWrite(led[3], LOW);
  digitalWrite(led[4], HIGH);
  digitalWrite(led[5], LOW);
}
void ledMati(){
  while(mySerial.available()==0){
```

```
    animasiLed();
  }
}
//  ##########################################################################
//  ###################### FUNGSI AKTIVITAS MOTOR DC ######################

void maju(){
  while(mySerial.available()==0){
    setKecepatan();
    mySerial.println("\n  Motor Berjalan MAJU ");
    mySerial.print("  dengan Kecepatan \t");
    mySerial.println(motorSpeed);
    // Menyalakan motor A & B(maju)
    digitalWrite(in1, HIGH);
    digitalWrite(in2, LOW);
    digitalWrite(in3, LOW);
    digitalWrite(in4, HIGH);
    // set kecepatan
    analogWrite(enA, motorSpeed);
    analogWrite(enB, motorSpeed);
    delay(led[1]);
    ledMaju();
  }
}

void mundur(){
  while(mySerial.available()==0){
    setKecepatan();
```

```
    mySerial.println("\n  Motor Berjalan MUNDUR ");
    mySerial.print("  dengan Kecepatan \t");
    mySerial.println(motorSpeed);
    // Menyalakan motor A & B (mundur)
    digitalWrite(in1, LOW);
    digitalWrite(in2, HIGH);
    digitalWrite(in3, HIGH);
    digitalWrite(in4, LOW);
    // set kecepatan
    analogWrite(enA, motorSpeed);
    analogWrite(enB, motorSpeed);
    delay(led[1]);
    ledMundur();
  }
}

void kiri(){
  while(mySerial.available()==0){
    setKecepatan();
    mySerial.println("\n  Motor KANAN ON, Motor Kiri off");
    mySerial.print("  dengan Kecepatan \t");
    mySerial.println(motorSpeed);
    mySerial.println("");
    // Mematikan motor A
    digitalWrite(in1, HIGH);
    digitalWrite(in2, LOW);
    // Menyalakan motor B
    digitalWrite(in3, LOW);
```

```
    digitalWrite(in4, LOW);
    // set speed motor dc
    analogWrite(enB, motorSpeed);
    delay(led[1]);
    ledKiri();
  }
}

void kanan(){
  while(mySerial.available()==0){
    setKecepatan();
    mySerial.println("\n  Motor KIRI ON, Motor Kanan off");
    mySerial.print("  dengan Kecepatan \t");
    mySerial.println(motorSpeed);
    mySerial.println("");
    // Menyalakan motor A
    digitalWrite(in1, LOW);
    digitalWrite(in2, LOW);
    // Mematikan motor B
    digitalWrite(in3, LOW);
    digitalWrite(in4, HIGH);
    // set speed motor dc
    analogWrite(enA, motorSpeed);
    delay(led[1]);
    ledKanan();
  }
}
```

```
void hop(){
   mySerial.println("\n  MOTOR KANAN DAN KIRI OFF");
   mySerial.println("");
  // Mematikan A & motor B
  digitalWrite(in1, LOW);
  digitalWrite(in2, LOW);
  digitalWrite(in3, LOW);
  digitalWrite(in4, LOW);
  ledMati();
}
//  ##########################################################################
//  ################### FUNGSI UNTUK SETTING KECEPATAN ###################

void setKecepatan(){
  //  membaca nilai adc
  potValue = analogRead(potPin);
  // pembacaan adc menjadi 0 - 255 untuk dijadikan parameter nilai kecepatan
  motorSpeed = map(potValue, 0, 1023, 0, 255);
  delay(200);
}
//  ##########################################################################


void loop(){
  //  Ketika port serial mempunyai masukan
  while(mySerial.available()!=0){
    data = mySerial.read();
      switch(data){
```

```
case '1':
  mySerial.println("\n  Anda memasukkan angka 1");
  maju();
break;
case '2':
  mySerial.println("\n  Anda memasukkan angka 2");
  mundur();
break;
case '3':
  mySerial.println("\n  Anda memasukkan angka 3");
  kanan();
break;
case '4':
  mySerial.println("\n  Anda memasukkan angka 4");
  kiri();
break;
case '5':
  mySerial.println("\n  Anda memasukkan angka 5");
  hop();
break;
default:
  mySerial.println("\nMasukan Salah!!!");
  mySerial.println("Pilih: 1 ==> Kedua Motor Berputar Maju");
  mySerial.println("       2 ==> Kedua Motor Berputar Mundur");
  mySerial.println("       3 ==> Mengaktifkan Motor Kanan saja (Belok Kiri)");
  mySerial.println("       4 ==> Mengaktifkan Motor Kiri saja (Belok Kanan)");
  mySerial.println("       5 ==> Kedua Motor Berhenti");
break;
```

```
    }
   }
}

//  ################### FUNGSI UNTUK ANIMASI LED ###################
void animasiLed(){
  int t = 350;
  int plus = 150;
  int tt = 350;
  int tam = 100;

  for(int k=0; k<2; k++){
    for(int i=0; i<6; i++){
      digitalWrite(led[i], HIGH);
      delay(t);
    }
    for(int i=5; i>=0; i--){
      digitalWrite(led[i], LOW);
      delay(tt);
    }
    t = t + plus;
    tt = tt + tam;
   }
    for(int i=0; i<6;){
        digitalWrite(led[i], HIGH);
        delay(500);
        digitalWrite(led[i], LOW);
        delay(100);
```

```
    i+=2;
}
for(int i=1; i<6;){
    digitalWrite(led[i], HIGH);
    delay(500);
    digitalWrite(led[i], LOW);
    delay(100);
    i+=2;
}
for(int i=0; i<6;){
  digitalWrite(led[i], HIGH);
  delay(500);
  digitalWrite(led[i], LOW);
  delay(100);
  i+=2;
}

for(int c=0; c<3; c++){
  for(int i=0; i<3; i++){
    digitalWrite(led[i], HIGH);
    digitalWrite(led[5-i], HIGH);
    delay(500);
    digitalWrite(led[i], LOW);
    digitalWrite(led[5-i], LOW);
    delay(100);
  }
}
```

```
for(int i=0; i<3; i++){
  digitalWrite(led[i], HIGH);
  digitalWrite(led[5-i], HIGH);
  delay(500);
}

for(int i=0; i<6; i++){
  digitalWrite(led[i], LOW);
  delay(300);
}
for(int i=0; i<6; i++){
  digitalWrite(led[i], HIGH);
}
delay(500);
for(int i=0; i<6; i++){
  digitalWrite(led[i], LOW);
  delay(200);
}
for(int i=0; i<6; i++){
  digitalWrite(led[i], HIGH);
}
delay(500);
for(int i=0; i<6; i++){
  digitalWrite(led[i], LOW);
  delay(100);
}
for(int i=0; i<6; i++){
  digitalWrite(led[i], HIGH);
```

```
}
delay(500);

////////////////////////////////
for(int i=0; i<6; i++){
  digitalWrite(led[i], HIGH);
  delay(500);
  digitalWrite(led[i], LOW);
  delay(100);
}

for(int i=0; i<6; i++){
  digitalWrite(led[i], HIGH);
  digitalWrite(led[i+1], HIGH);
  delay(500);
  digitalWrite(led[i], LOW);
  digitalWrite(led[i+1], LOW);
  delay(100);
}

for(int i=0; i<6; i++){
  digitalWrite(led[i], HIGH);
  digitalWrite(led[i+1], HIGH);
  digitalWrite(led[i+2], HIGH);
  delay(500);
  digitalWrite(led[i], LOW);
  digitalWrite(led[i+1], LOW);
  digitalWrite(led[i+2], LOW);
```

```
  delay(100);
}

for(int i=0; i<6; i++){
  digitalWrite(led[i], HIGH);
  digitalWrite(led[i+1], HIGH);
  digitalWrite(led[i+2], HIGH);
  digitalWrite(led[i+3], HIGH);
  delay(500);
  digitalWrite(led[i], LOW);
  digitalWrite(led[i+1], LOW);
  digitalWrite(led[i+2], LOW);
  digitalWrite(led[i+3], LOW);
  delay(100);

for(int i=0; i<6; i++){
  digitalWrite(led[i], HIGH);
  digitalWrite(led[i+1], HIGH);
  digitalWrite(led[i+2], HIGH);
  digitalWrite(led[i+3], HIGH);
  digitalWrite(led[i+4], HIGH);
  delay(600);
  digitalWrite(led[i], LOW);
  digitalWrite(led[i+1], LOW);
  digitalWrite(led[i+2], LOW);
  digitalWrite(led[i+3], LOW);
  digitalWrite(led[i+4], LOW);
  delay(150);
```

```
    }

    for(int i=0; i<6; i++){
      digitalWrite(led[i], HIGH);
      digitalWrite(led[i+1], HIGH);
      digitalWrite(led[i+2], HIGH);
      digitalWrite(led[i+3], HIGH);
      digitalWrite(led[i+4], HIGH);
      digitalWrite(led[i+5], HIGH);
      delay(600);
      digitalWrite(led[i], LOW);
      digitalWrite(led[i+1], LOW);
      digitalWrite(led[i+2], LOW);
      digitalWrite(led[i+3], LOW);
      digitalWrite(led[i+4], LOW);
      digitalWrite(led[i+5], LOW);
      delay(150);
    }

    t = 0;
    tt= 0;
}
```

VIII. ANALISA

8.1 Analisa program

```
#include <SoftwareSerial.h>
  // motor A
int enA = 11;
int in1 = 13;
int in2 = 12;
  // motor B
int enB = 10;
int in3 = 9;
int in4 = 8;
  // adc
int potPin = A0;
  // variabel pembacaan nilai adc
int potValue = 0;
int motorSpeed = 0;
  // variabel pembacaan input keyboard
int data;
  // led
int led[]={7,6,5,4,3,2};

SoftwareSerial mySerial(A1, A2); // RX, TX
```

Menggunakan library SoftwareSerial untuk komunikasi serial antara arduino dengan device lain dan membuat pin komunikasi serial yang lain

Pendeklarasian pin arduino yang digunakan,dan telah disesuaikan dengan interface nya

Deklarasi variabel mySerial, dengan library SoftwareSerial menjadikan pin analog A1 sebagai Rx dan A2 sebagai Tx

```
void setup() {
  // Open serial communications and wait for port to open

  // set the data rate for the SoftwareSerial port
  mySerial.begin(9600);
  mySerial.println("Bluetooth Communication Is Ready");
  for(int i=0; i<6; i++){
    pinMode(led[i], OUTPUT);
  }
  pinMode(enA, OUTPUT);
  pinMode(enB, OUTPUT);
  pinMode(in1, OUTPUT);
  pinMode(in2, OUTPUT);
  pinMode(in3, OUTPUT);
  pinMode(in4, OUTPUT);
  pinMode(potPin, INPUT);

  //  ########################      INTRO      ########################
    mySerial.println("PROJECT MIKRO & ANTARMUKA 1");

    mySerial.println("\nKelompok 3");
    mySerial.println("Anggota:  Ach.Chusnul Chikam  NRP 32");    delay(400);
    mySerial.println("         Aulia Rahmah N.F.   NRP 49");    delay(400);
    mySerial.println("         Alvita Khairinia K. NRP 54");    delay(400);
    mySerial.println("         Diah Ayu Pitaloka   NRP 57");    delay(400);
    mySerial.println("         Nurul Hidayah       NRP 59");
    delay(400);
```

Baud Rate untuk komunikasi serial diset 9600, telah disesuaikan dengan baud rate bluetooth HC-05 yang dipakai, kemudian mencetak kalimat yang tertampil

Setup 6 led, pin” ic l293d sebagai Output dan potensio sebagai Input

mySerial mencatak nama” kelompok dan mengirimkan ke perangkat android

```
//  ##################### PENGECEKAN KOMPONEN ########################
    mySerial.print("Inisialisasi...");
    mySerial.print("1..");  delay(100);
    mySerial.print("2..");  delay(100);
    mySerial.println("3..");  delay(100);
   mySerial.println("Pengecekan LED.....");
   delay(600);
   digitalWrite(led[0],HIGH);
   delay(400);
   digitalWrite(led[0],LOW);
   digitalWrite(led[1],HIGH);
   delay(400);
   digitalWrite(led[1],LOW);
   digitalWrite(led[2],HIGH);
   delay(400);
   digitalWrite(led[2],LOW);
   digitalWrite(led[3],HIGH);
   delay(400);
   digitalWrite(led[3],LOW);
   digitalWrite(led[4],HIGH);
   delay(400);
   digitalWrite(led[4],LOW);
   digitalWrite(led[5],HIGH);
   delay(400);
   digitalWrite(led[5],LOW);
   delay(400);
   mySerial.println(" Pengecekan LED Selesai...");
   delay(600);
```

Pengecekan led otomatis yang dilakukan oleh program, dimulai dari led 1 sampai led 6. Dengan pengecekan ini kita akan tahu apakah komponen masih berfungsi dengan baik atau tidak sebelum masuk pada proses selanjutnya. Setelah dicek akan muncul pemberitahuan dilayar android "Pengecekan LED Selesai"

```
  mySerial.println(" Pengecekan Motor DC.....");
  delay(600);
  digitalWrite(in1, LOW);
  digitalWrite(in2, HIGH);
  digitalWrite(in3, HIGH);
  digitalWrite(in4, LOW);
  // kecepatan 250
  analogWrite(enA, 250);
  analogWrite(enB, 250);
  delay(600);
  digitalWrite(in1, HIGH);
  digitalWrite(in2, LOW);
  digitalWrite(in3, LOW);
  digitalWrite(in4, HIGH);
  // kecepatan 250
  analogWrite(enA, 250);
  analogWrite(enB, 250);
  delay(600);
  // Motor DC mati
  digitalWrite(in1, LOW);
  digitalWrite(in2, LOW);
  digitalWrite(in3, LOW);
  digitalWrite(in4, LOW);
  mySerial.println("");
  delay(200);
  mySerial.println(" UNTUK MENGETES KOMUNIKASI SERIAL ANTARA ARDUINO DENGAN BLUETOOTH,");
  mySerial.println(" BERI MASUKAN!!!");
}
```

Setelah pengecekan led, selanjutnya adalah pengecekan motor DC dengan kecepatan penuh(255) berjalan maju 600ms lalu berjalan mundur selama 600ms kemudian mati . Dengan pengecekan ini kita akan tahu apakah komponen masih berfungsi dengan baik atau tidak sebelum masuk pada proses selanjutnya.

Mencetak perintah agar user memberikan input lewat android, disini juga akan terlihat apakah bluetooth sudah tersambung dengan arduino atau tidak. Setelah diberi input jika ada respon dari arduino yang tampil dilayar android berarti sudah tersambung. Arduino dan android sudah berkomunikasi lewat bluetooth.


```
//  ##################### FUNGSI INDIKATOR LED ######################
void ledMaju(){
  digitalWrite(led[0], LOW);
  digitalWrite(led[1], HIGH);
  digitalWrite(led[2], LOW);
  digitalWrite(led[3], LOW);
  digitalWrite(led[4], LOW);
  digitalWrite(led[5], LOW);
}

void ledMundur(){
  digitalWrite(led[0], LOW);
  digitalWrite(led[1], LOW);
  digitalWrite(led[2], HIGH);
  digitalWrite(led[3], LOW);
  digitalWrite(led[4], LOW);
  digitalWrite(led[5], LOW);
}

void ledKanan(){
  digitalWrite(led[0], LOW);
  digitalWrite(led[1], LOW);
  digitalWrite(led[2], LOW);
  digitalWrite(led[3], HIGH);
  digitalWrite(led[4], LOW);
  digitalWrite(led[5], LOW);
}

void ledKiri(){
  digitalWrite(led[0], LOW);
  digitalWrite(led[1], LOW);
  digitalWrite(led[2], LOW);
  digitalWrite(led[3], LOW);
  digitalWrite(led[4], HIGH);
  digitalWrite(led[5], LOW);
}
```

Fungsi" indikator led, gunanya sebagai tanda apa kondisi motor saat ini. Hal tersebut dapat dilihat dari posisi led mana yang nyala.

Jika,

Maju ➔ led kedua nyala

Mundur ➔ led ketiga nyala

Kanan ➔ led keempat nyala

Kiri ➔ led kelima nyala

```
void ledMati(){
  while(mySerial.available()==0){
    animasiLed();
  }
}
```

Untuk led mati ini bukan led nya yang mati tetapi motor DC nya, led nya akan nyala sesuai animasi yang telah dibuat

Pengecekan kondisi jika tak ada data yang dimasukkan berarti memenuhi kondisi dan masuk ke proses selanjutnya

```
void maju(){
  while(mySerial.available()==0){
    setKecepatan();
    mySerial.println("\n  Motor Berjalan MAJU ");
    mySerial.print("  dengan Kecepatan \t");
    mySerial.println(motorSpeed);
    // Menyalakan motor A & B(maju)
    digitalWrite(in1, HIGH);
    digitalWrite(in2, LOW);
    digitalWrite(in3, LOW);
    digitalWrite(in4, HIGH);
    // set kecepatan
    analogWrite(enA, motorSpeed);
    analogWrite(enB, motorSpeed);
    delay(led[1]);
    ledMaju();
  }
}
```

Fungsi maju(), saat tak ada input yang dimasukkan akan lanjut ke baris program selanjutnya dan memanggil fungsi kecepatan untuk pembacaan nilai kecepatan. Lalu mencetak keterangan aktivitas motor ke android. Motor A berputar searah jarum jam, dan motor B berputar berlawanan jarum jam (MAJU). Indikator led

```
void mundur()
  while(mySerial.available()==0){
    setKecepatan();
    mySerial.println("\n  Motor Berjalan MUNDUR ");
    mySerial.print("  dengan Kecepatan \t");
    mySerial.println(motorSpeed);
    // Menyalakan motor A & B (mundur)
    digitalWrite(in1, LOW);
    digitalWrite(in2, HIGH);
    digitalWrite(in3, HIGH);
    digitalWrite(in4, LOW);
    // set kecepatan
    analogWrite(enA, motorSpeed);
    analogWrite(enB, motorSpeed);
    delay(led[1]);
    ledMundur();
  }
}
```

Fungsi mundur(), saat tak ada input yang dimasukkan akan lanjut ke baris program selanjutnya dan memanggil fungsi kecepatan untuk pembacaan nilai kecepatan. Lalu mencetak keterangan aktivitas motor ke android. Motor A berputar berlawanan jarum jam, dan motor B berputar searah jarum jam (MUNDUR). Indikator

```
void kanan(){
  while(mySerial.available()==0){
    setKecepatan();
    mySerial.println("\n  Motor KIRI ON, Motor Kanan off");
    mySerial.print("  dengan Kecepatan \t");
    mySerial.println(motorSpeed);
    mySerial.println("");
    // Menyalakan motor A
    digitalWrite(in1, LOW);
    digitalWrite(in2, LOW);
    // Mematikan motor B
    digitalWrite(in3, LOW);
    digitalWrite(in4, HIGH);
    // set speed motor dc
    analogWrite(enA, motorSpeed);
    delay(led[1]);
    ledKanan();
  }
```

Fungsi kanan(), saat tak ada input yang dimasukkan akan lanjut ke baris program selanjutnya dan memanggil fungsi kecepatan untuk pembacaan nilai kecepatan. Lalu mencetak keterangan aktivitas motor ke android. Motor A berputar searah jarum jam, dan motor B off (Belok Kiri). Indikator led untuk kondisi kanan dinyalakan.

```
void kiri(){
  while(mySerial.available()==0){
    setKecepatan();
    mySerial.println("\n  Motor KANAN ON, Motor Kiri off"
    mySerial.print("  dengan Kecepatan \t");
    mySerial.println(motorSpeed);
    mySerial.println("");
    // Mematikan motor A
    digitalWrite(in1, HIGH);
    digitalWrite(in2, LOW);
    // Menyalakan motor B
    digitalWrite(in3, LOW);
    digitalWrite(in4, LOW);
    // set speed motor dc
    analogWrite(enB, motorSpeed);
    delay(led[1]);
    ledKiri();
  }
```

Fungsi kiri(), saat tak ada input yang dimasukkan akan lanjut ke baris program selanjutnya dan memanggil fungsi kecepatan untuk pembacaan nilai kecepatan. Lalu mencetak keterangan aktivitas motor ke android. Motor A off, dan motor B berputar berlawanan jarum jam (Belok Kanan). Indikator led untuk kondisi kiri dinyalakan.

```
void hop(){
   mySerial.println("\n  MOTOR KANAN DAN KIRI OFF");
   mySerial.println("");
  // Mematikan A & motor B
  digitalWrite(in1, LOW);
  digitalWrite(in2, LOW);
  digitalWrite(in3, LOW);
  digitalWrite(in4, LOW);
  ledMati();
}
```

Fungsi hop(), mencetak keterangan aktivitas motor ke android bahwa Motor A dan Motor B off (Berhenti). Indikator led untuk kondisi mati dinyalakan. Saat kedua motor mati animasiLed()

```
void setKecepatan(){
  //  membaca nilai adc
  potValue = analogRead(potPin);
  // pembacaan adc menjadi 0 - 255 untuk dijadikan parameter nilai kecepatan
  motorSpeed = map(potValue, 0, 1023, 0, 255);
  delay(200);
}
```

Poses yang pertama kali dilakukan saat ada input yang masuk yitu pembacaan nilai adc dari potensio, nilai dikonversi dari nilai 0-1023 menjadi 0-255.

Pengondisian tentang input yang dikirim dari android, Saat input dimasukkan artinya memenuhi syarat != 0 maka akan lanjut ke proses berikutnya yaitu pembacaan nilai variable “data”

```
void loop(){
  //  Ketika port serial mempunyai masukan
  while(mySerial.available()!=0){
    data = mySerial.read();
      switch(data){
        case '1':
          mySerial.println("\n  Anda memasukkan angka 1");
          maju();
        break;
        case '2':
          mySerial.println("\n  Anda memasukkan angka 2");
          mundur();
        break;
        case '3':
          mySerial.println("\n  Anda memasukkan angka 3");
          kanan();
        break;
        case '4':
          mySerial.println("\n  Anda memasukkan angka 4");
          kiri();
        break;
        case '5':
          mySerial.println("\n  Anda memasukkan angka 5");
          hop();
        break;
```

Jika input yang terbaca:

1 ➔ Maju *

2 ➔ Mundur *

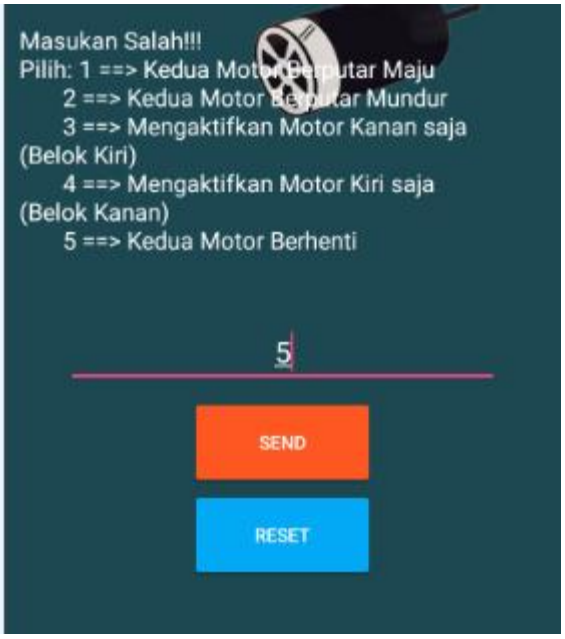
3 ➔ Kanan *

4➔ Kiri *

5 ➔ Berhenti *

```
        default:
          mySerial.println("\nMasukan Salah!!!");
          mySerial.println("Pilih: 1 ==> Kedua Motor Berputar Maju");
          mySerial.println("       2 ==> Kedua Motor Berputar Mundur");
          mySerial.println("       3 ==> Mengaktifkan Motor Kanan saja (Belok Kiri)");
          mySerial.println("       4 ==> Mengaktifkan Motor Kiri saja (Belok Kanan)");
          mySerial.println("       5 ==> Kedua Motor Berhenti");
        break;
      }
    }
}
```

Jika input selain 1,2,3,4,5 maka akan ada peringatan bahwa input salah dan diberi petunjuk untuk input dan output nya. Program juga akan tetap berjalan dengan perintah sebelumnya. Parilaku input yang salah ini tidak mempengaruhi aktivitas motor DC




```
//  ################## FUNGSI UNTUK ANIMASI LED ###################
void animasiLed(){
  int t = 350;
  int plus = 150;
  int tt = 350;
  int tam = 100;

  for(int k=0; k<2; k++){
    for(int i=0; i<6; i++){
      digitalWrite(led[i], HIGH);
      delay(t);
    }
    for(int i=5; i>=0; i--){
      digitalWrite(led[i], LOW);
      delay(tt);
    }
    t = t + plus;
    tt = tt + tam;
    }
```

Led nyala dari led 1 ke led 6, pertama nyala dengan waktu 350ms lalu mati dari led 6 ke led 1 selama 300ms.

Di looping kedua led nyala dengan delay, 350+150=500 ms. lalu mati dengan delay 300+100=400 ms

```
for(int i=0; i<6;){
    digitalWrite(led[i], HIGH);
    delay(500);
    digitalWrite(led[i], LOW);
    delay(100);
    i+=2;
}
for(int i=1; i<6;){
    digitalWrite(led[i], HIGH);
    delay(500);
    digitalWrite(led[i], LOW);
    delay(100);
    i+=2;
}
for(int i=0; i<6;){
  digitalWrite(led[i], HIGH);
  delay(500);
  digitalWrite(led[i], LOW);
  delay(100);
  i+=2;
}
```

Nyala led nya dimulai dari led 1,lalu led (1+2), led (3+2) sampai tidak memenuhi kondisi looping lagi. kemudian di for kedua, nyala led dimulai led 2, lalu led (2+2), led (4+2) sampai tidak memenuhi kondisi looping lagi. Di for ketiga sama seperti looping pertama

Disini dilakukan looping 3x saat memenuhi kondisi perulangan. Led nyala mulai dari led 1 sama led 6, led 2 sama led 5, led 3 sama led 4. Lalu led mati mulai dari led 1 sama led 6, led 2 sama led 5, led 3 sama led 4.

```
for(int c=0; c<3; c++){
  for(int i=0; i<3; i++){
    digitalWrite(led[i], HIGH);
    digitalWrite(led[5-i], HIGH);
    delay(500);
    digitalWrite(led[i], LOW);
    digitalWrite(led[5-i], LOW);
    delay(100);
  }
}
```

```
for(int i=0; i<3; i++){
  digitalWrite(led[i], HIGH);
  digitalWrite(led[5-i], HIGH);
  delay(500);
}
```

```
for(int i=0; i<6; i++){
  digitalWrite(led[i], LOW);
  delay(300);
}
for(int i=0; i<6; i++){
  digitalWrite(led[i], HIGH);
}
delay(500);
```

Led mati dari led 1 ke led 6 dengan delay 300ms, lalu nyala dari led 1 ke led 6 dengan delay 500ms.

```
for(int i=0; i<6; i++){
  digitalWrite(led[i], LOW);
  delay(200);
}
for(int i=0; i<6; i++){
  digitalWrite(led[i], HIGH);
}
delay(500);
```

Led mati dari led 1 ke led 6 dengan delay 300ms, lalu nyala dari led 1 ke led 6 dengan delay 500ms.

```
for(int i=0; i<6; i++){
  digitalWrite(led[i], LOW);
  delay(100);
}

for(int i=0; i<6; i++){
  digitalWrite(led[i], HIGH);
}
delay(500);
```

Led mati dari led 1 ke led 6 dengan delay 300ms, lalu nyala dari led 1 ke led 6 dengan delay 500ms.

```
for(int i=0; i<6; i++){
  digitalWrite(led[i], HIGH);
  delay(500);
  digitalWrite(led[i], LOW);
  delay(100);
}
```

Led nyala satu , lalu berjalan dengan urutan led 1, led 2, led3, dst sampai hilang dengan delay perpindahan 100ms

```
for(int i=0; i<6; i++){
  digitalWrite(led[i], HIGH);
  digitalWrite(led[i+1], HIGH);
  delay(500);
  digitalWrite(led[i], LOW);
  digitalWrite(led[i+1], LOW);
  delay(100);
}
```

Led nyala dua , lalu berjalan dengan urutan led 1 dan led2, led 2 dan led3, dst sampai hilang dengan delay perpindahan 100ms

```
for(int i=0; i<6; i++){
  digitalWrite(led[i], HIGH);
  digitalWrite(led[i+1], HIGH);
  digitalWrite(led[i+2], HIGH);
  delay(500);
  digitalWrite(led[i], LOW);
  digitalWrite(led[i+1], LOW);
  digitalWrite(led[i+2], LOW);
  delay(100);
}
```

Led nyala tiga , lalu berjalan dengan urutan dari led 1 ke led 6 dst sampai hilang dengan delay perpindahan 100ms

```
for(int i=0; i<6; i++){
  digitalWrite(led[i], HIGH);
  digitalWrite(led[i+1], HIGH);
  digitalWrite(led[i+2], HIGH);
  digitalWrite(led[i+3], HIGH);
  delay(500);
  digitalWrite(led[i], LOW);
  digitalWrite(led[i+1], LOW);
  digitalWrite(led[i+2], LOW);
  digitalWrite(led[i+3], LOW);
  delay(100);
}
```

Led nyala empat , lalu berjalan dengan urutan dari led 1 ke led 6 dst sampai hilang dengan delay perpindahan 100ms

```
for(int i=0; i<6; i++){
  digitalWrite(led[i], HIGH);
  digitalWrite(led[i+1], HIGH);
  digitalWrite(led[i+2], HIGH);
  digitalWrite(led[i+3], HIGH);
  digitalWrite(led[i+4], HIGH);
  delay(600);
  digitalWrite(led[i], LOW);
  digitalWrite(led[i+1], LOW);
  digitalWrite(led[i+2], LOW);
  digitalWrite(led[i+3], LOW);
  digitalWrite(led[i+4], LOW);
  delay(150);
}
```

Led nyala empat , lalu berjalan dengan urutan dari led 1 ke led 6 dst sampai hilang dengan delay perpindahan 100ms

```
for(int i=0; i<6; i++){
  digitalWrite(led[i], HIGH);
  digitalWrite(led[i+1], HIGH);
  digitalWrite(led[i+2], HIGH);
  digitalWrite(led[i+3], HIGH);
  digitalWrite(led[i+4], HIGH);
  digitalWrite(led[i+5], HIGH);
  delay(600);
  digitalWrite(led[i], LOW);
  digitalWrite(led[i+1], LOW);
  digitalWrite(led[i+2], LOW);
  digitalWrite(led[i+3], LOW);
  digitalWrite(led[i+4], LOW);
  digitalWrite(led[i+5], LOW);
  delay(150);
}

t = 0;
tt= 0;
}
```

Led nyala empat , lalu berjalan dengan urutan dari led 1 ke led 6 dst sampai hilang dengan delay perpindahan 100ms

Pengembalian nilai variabel t (untuk delay nyala pada looping animasi led pertama) menjadi 0. Pengembalian nilai variabel tt (untuk delay mati pada looping animasi led pertama) menjadi 0.

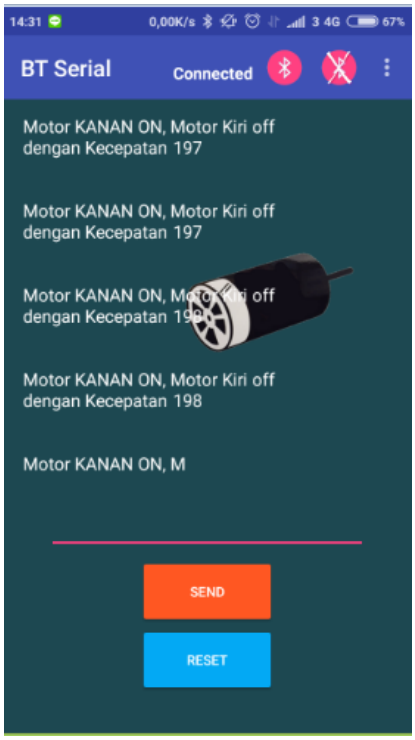
Langkah awal untuk melakukan komunikasi antara android dengan arduino ada menyandingkan bluetooth HC-05 dengan bluetooth diandroid terlebih dulu. Setelah bluetooth telah disandingkan(paired) kita tinggal membuka aplikasi Bluetooth Serial dan mengkoneksikannya. Untuk kondisi konek tidaknya dapat dilihat di bagian toolbar aplikasi.

Pada hardware nya kita menambahkan library softwareserial sebagai pengganti pin 0 dan pin 1 komunikasi UART pada arduino. Alasannya karena dengan memakai lbrary tambahan ini pin Tx dan Rx dapat diatur lebih bebas.

Penjelasan lanjutan untuk program difungsi loop(), Jadi program akan melakukan pembacaan terhadap input yang dimasukkan dari android. Saat diberi input misal “1” maka program akan membaca dan memproses nya yaitu masuk case pertama. Motor akan bergerak maju sesuai dengan apa yang ada pada fungi yang dipanggil yaitu fungsi maju(). Keterangan bahwa mototr berjalan maju dan dengan kecepatan berapa dapat terlihat di layar android.

Proses ini akan diulang ulang sampai terjadi pembacan input lagi. Misalkan lagi diberi input “W” maka akan ada keterangan bahwa input salah dan diberi petunjuk untuk nilai masukan yang dimaksudkan. Program tidak mengeksekusi input yang salah program masih berjalan dengan perintah sebelumnya. Proses komunikasi antara arduino dengan android juga selalu terjadi karena arduino selalu mencetak arah putaran dan nilai kecepaan per 200ms sekali di layar android melaui bluetooth. Saat arduino mencetak keterangan kondisi motor berarti dalam hal tersebut arduino telah mengirimkan data ke android. Kemudian saat dari android mengirimkan sebuah input maka arduino harus menerima data dari android dan memprosesnya sesuai program yang diupload. Dengan kondisi yang seperti ini berarti komunikasi antara arduino dan android lewat bluetooth ini dapat disebut sebagai komunikasi dua arah/duplek.

Untuk kecepatan motor DS sendiri diatur dari pembacaan nilai adc dari potensiometer dengan perintah “mapped” nilai adc dari 0-1023 di konversi ke 0-255. Hal ini karena pwm hanya mampu membaca nilai sebesar 255 dalam kondisi maksimum. Dalam project ini paling maksimal hanya sampai di nilai 253/252 untuk nilai minimum juga tidak sampai 0 karena motor sudah tidak mampu berputar saat nilai speednya mulai dari <110.



Untuk kondisi saat motor dalam keadaan mati, atau berhenti akan ada animasi led yang dijalankan sesuai dengan fungsi animasiLed(). Saat animasi led sedang diproses, dan ditengah-tengah proses kita memberi input, input tidak langsung dibaca melainkan menjalankan baris demi baris dari fungsi animasiLed sampai baris terkhir baru dibaca input yang telah dimasukkan tadi.

8.2 Analisa Bluetooth

Aplikasi Bluetooth Serial

Aplikasi ini terdiri dari 4 class, yaitu class MainActivity, Bluetooth, SubActivity, dan About. Kelas yang akan saya jelaskan disini hanya kelas Bluetooth karena kelas inilah yang akan mengirimkan dan menangkap data yang masuk dari Arduino.

#Mendeteksi Bluetooth Device

Sebelum aplikasi dapat berkomunikasi melalui bluetooth, kita harus memverifikasi apakah bluetooth tersedia pada pada perangkat dan jika tersedia maka bluetooth akan diaktifkan. Berikut merupakan kode yang akan melakukan hal tersebut:

```
btAdapter = BluetoothAdapter.getDefaultAdapter();
pairedDevices = new ArrayList<>();
filter = new IntentFilter(BluetoothDevice.ACTION_FOUND);
```

Kode diatas menggunakan kelas BluetoothAdapter, kelas ini dibutuhkan jika kita akan membuat aplikasi yang berhubungan dengan bluetooth. Untuk mendapatkannya kita bisa memanggil method *getDefaultAdapter()* dari kelas *BluetoothAdapter* yang akan mempresentasikan apakah device memiliki Bluetooth Adapter (fitur perangkat bluetooth).

#MengaktifkanBluetooth

Selanjutnya kita perlu memastikan bluetooth sedang hidup atau tidak dengan memanggil method *isEnabled()*, jika method ini mengembalikan nilai *false* berarti bluetooth belum aktif. Untuk mengaktifkan bluetooth, panggil *startActivityForResult()* dengan *ACTION_REQUEST_ENABLE* action Intent. Berikut merupakan kode yang akan melakukan hal diatas:

```
i if (btAdapter == null) {

    Toast.makeText(getApplicationContext(), "No bluetooth detected",
Toast.LENGTH_SHORT).show();
    finish();
  } else {
    if (!btAdapter.isEnabled()) {
      turnOnBT();
    }
    getPairedDevices();
    startDiscovery();
  }
}
private void startDiscovery() {
  // TODO Auto-generated method stub
  btAdapter.cancelDiscovery();
  btAdapter.startDiscovery();
}
private void turnOnBT() {
  Intent intent = new Intent(BluetoothAdapter.ACTION_REQUEST_ENABLE);
  startActivityForResult(intent, 3);
}
```

Kode diatas akan menampilkan dialog ke user untuk mengaktifkan bluetooth.

#Mencari Bluetooth Device

Menggunakan *BluetoothAdapter* kita bisa mencari remote Bluetooth devices baik melalui device discovery atau dengan mengambil dari list paired (bonded) devices yaitu bluetooth device yang sudah dipasangkan dengan handphone. Untuk memanggil dari list paired device kita dapat menggunakan method *getBoundedDevices()*, method ini akan mengembalikan kumpulan-kumpulan bluetooth device yang sudah terpasang, kelas yang dipakai adalah *BluetoothDevice*, berikut potongan sintaknya:

```
public void onReceive(Context context, Intent intent) {
    String action = intent.getAction();
    if (BluetoothDevice.ACTION_FOUND.equals(action)) {
        BluetoothDevice device =
intent.getParcelableExtra(BluetoothDevice.EXTRA_DEVICE);
        devices.add(device);
        String s = "";
        for (int a = 0; a < pairedDevices.size(); a++) {
            if (device.getName().equals(pairedDevices.get(a))) {
                //append
                s = "(Paired)";
                break;
            }
        }
```

```
        listAdapter.add(device.getName() + " " + s + " " + "\n" + device.getAddress());
    } else if (BluetoothAdapter.ACTION_DISCOVERY_STARTED.equals(action)) {
    } else if (BluetoothAdapter.ACTION_DISCOVERY_FINISHED.equals(action)) {
    } else if (BluetoothAdapter.ACTION_STATE_CHANGED.equals(action)) {
        if (btAdapter.getState() == btAdapter.STATE_OFF) {
            turnOnBT();
        }
    }
}
}
```

Kode diatas diperlukan dari kelas *BluetoothDevice* untuk mengambil *mac address* yang akan digunakan untuk membuat koneksi antar device. Bluetooth device yang tersedia diatas dimasukan ke *ArrayList* atau *ArrayAdapter* yang selanjutnya akan ditampilkan ke user untuk dipilih.

#Menghubungkan Bluetooth Devic

Untuk memulai koneksi dengan remote device (device yang menangani server socket) kita harus mendapatkan objek *BluetoothDevice* yang merepresentasikan remote device(part #mencari bluetooth device). Kita harus menggunakan *BluetoothDevice* untuk memperoleh *BluetoothSocket* dan memulai koneksi. Berikut merupakan urutan yang perlu dilakukan:

1. Menggunakan kelas *BluetoothDevice*, dapatkan *BluetoothSocket* dengan memanggil *createRfcommSocketToServiceRecord(UUID)*.
2. Memulai koneksi dengan memanggil method *connect()*.

Contoh sintak:

```
private class ConnectThread extends Thread {
    private final BluetoothSocket mmSocket;
    private final BluetoothDevice mmDevice;
    public ConnectThread(BluetoothDevice device) {
        // Use a temporary object that is later assigned to mmSocket,
```

```
    // because mmSocket is final
    BluetoothSocket tmp = null;
    mmDevice = device;
    // Get a BluetoothSocket to connect with the given BluetoothDevice
    try {
        // MY_UUID is the app's UUID string, also used by the server code
        tmp = device.createRfcommSocketToServiceRecord(MY_UUID);
    } catch (IOException e) {
    }
 mmSocket = tmp;
}
public void run() {
 // Cancel discovery because it will slow down the connection
    btAdapter.cancelDiscovery();
    try {
        // Connect the device through the socket. This will block
        // until it succeeds or throws an exception
        mmSocket.connect();
        //connectedThread = new ConnectedThread(mmSocket);
    } catch (IOException connectException) {
        // Unable to connect; close the socket and get out
        try {
            mmSocket.close();
        } catch (IOException closeException) {
        }
        return;
    }
```

Perhatikan method *cancelDiscovery()* diatas dipanggil sbelum koneksi dibuat, ini sebaiknya dilakukan untuk menghentikan pencarian bluetooth device tanpa mengecek apakah sedang melakukan pencarian atau tidak. tetapi jika kita ingin mengeceknya terlebih dahulu kita bisa menggunakan method *isDiscovering()*.Dan karena method *connect()* akan mengeblok main activity thread sampai koneksi berhasil atau gagal, maka kita perlu

membuatnya didalam thread pula untuk menghindari crash nya aplikasi ketika sedang mencoba menghubungkan device.

#Memanage koneksi yang sudah terbentuk

Ketika koneksi sudah berhasil terbentuk, setiap device akan memiliki *BluetoothSocket* yang sudah terhubung dan sampai disini kita bisa saling bertukar data. Berikut tahapan menggunakan *BluetoothSocket* untuk saling bertukar data:

1. Buat *InputStream* dan *OutputStream* yang akan menghandle data transmisi melewati socket yaitu *getInputStream()* dan *getOutputStream().*
2. Membaca dan mengirimkan data ke streams dengan *read(byte[])* dan *write(byte[]).*

Potongan sintak:

```
static class ConnectedThread extends Thread {
    private final BluetoothSocket mmSocket;
    private final InputStream mmInStream;
    private final OutputStream mmOutStream;
    public ConnectedThread(BluetoothSocket socket) {
        mmSocket = socket;
        InputStream tmpIn = null;
        OutputStream tmpOut = null;
        // Get the input and output streams, using temp objects because
        // member streams are final
        try {
            tmpIn = socket.getInputStream();
            tmpOut = socket.getOutputStream();
        } catch (IOException e) {
        }
        mmInStream = tmpIn;
        mmOutStream = tmpOut;
    }
    StringBuffer sbb = new StringBuffer();
```

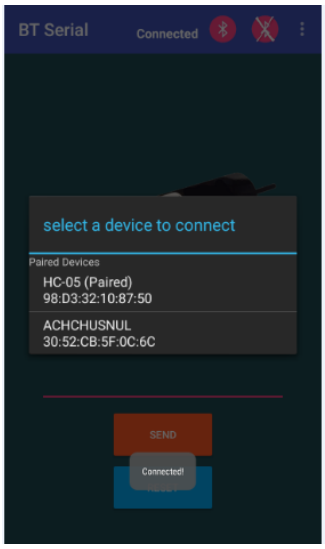
```
        public void run() {
            byte[] buffer;  // buffer store for the stream
            int bytes; // bytes returned from read()
            // Keep listening to the InputStream until an exception occurs
            while (true) {
                try {
                    try {
                        sleep(30);
                    } catch (InterruptedException e) {
                        // TODO Auto-generated catch block
                        e.printStackTrace();
                    }
                    buffer = new byte[1024];
                    // Read from the InputStream
                    bytes = mmInStream.read(buffer);
                    // Send the obtained bytes to the UI activity
                    mHandler.obtainMessage(MESSAGE_READ, bytes, -1, buffer).sendToTarget();
                } catch (IOException e) {
                    break;
                }
            }
        }
        /* Call this from the main activity to send data to the remote device */
        public void write(String income) {
            try {
                mmOutStream.write(income.getBytes());

                for (int i = 0; i < income.getBytes().length; i++)

        Log.v("outStream" + Integer.toString(i),
                        Character.toString((char)
    (Integer.parseInt(Byte.toString(income.getBytes()[i])))));
                try {
                    Thread.sleep(20);
```

```
            } catch (InterruptedException e) {
                // TODO Auto-generated catch block
                e.printStackTrace();
            }
        } catch (IOException e) {
        }
    }
    /* Call this from the main activity to shutdown the connection */
    public void cancel() {
        try {
            mmSocket.close();
        } catch (IOException e) {
        }
    }
  }

}
```



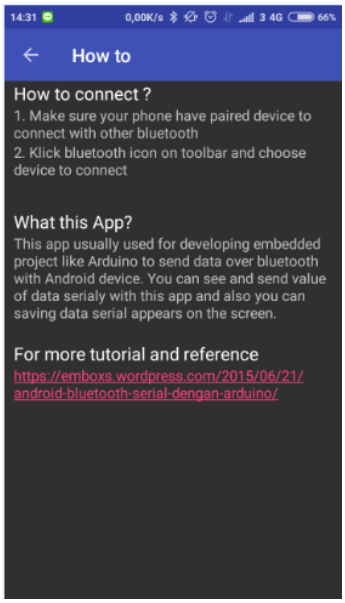
Thread diatas akan menunggu data yang datang melewati *InputStream*. Ketika *read(byte[])* mendapatkan data dari streams, maka selanjutnya data akan dikirimkan ke *MainActivity* dengan *Handler* dari kelas parent, hal ini akan terus berjalan selama ada data yang masuk. Untuk mengirim data keluar yaitu caranya dengan memanggil method *write()* pada

thread diatas dari kelas *MainActivity* dan mengirimkan bytes untuk dikirimkan. Method ini memanggil *write(byte[])* untuk mengirimkannya ke remote device. Sedangkan method *cancel()* dari thread diatas berfungsi untuk mengahiri koneksi *Bluetooth Socket.*

Pada project yang kami buat, terdapat kendala dimana ketika berkomunikasi serial menggunakan Bluetooth dijalankan, maka ketika kabel usb tidak dipasang dalam hardware tersebut, akan terjadi ketidaksesuaian program dan implementasi yang diinginkan (program terus looping dan tidak mau berhenti). Namun, ketika kabel usb dipasang, maka program dapat berjalan sesuai dengan apa yang diharapkan. Kondisi tersebut terjadi karena tegangan baterai yang kurang mencukupi untuk menjalankan project kami. Sehingga membutuhkan kabel usb untuk dipasang ke sumber dengan tegangan lumayan besar. Seperti tegangan yang disimpan pada power bank, laptop, dll.

Untuk Bluetooth,pada pengapliaksianya apabila sudah dikoneksikan ke suatu perangkat, maka perangkat lain tidak dapat tersambung dengan Bluetooth secara bersamaan.

Tampilan dari aplikasi Bluetooth serial :


14:31
0,00K/s
3 4G
66%
How to
How to connect ?
1. Make sure your phone have paired device to connect with other bluetooth
2. Klick bluetooth icon on toolbar and choose device to connect
What this App?
This app usually used for developing embedded project like Arduino to send data over bluetooth with Android device. You can see and send value of data serialy with this app and also you can saving data serial appears on the screen.
For more tutorial and reference
https://emboxs.wordpress.com/2015/06/21/android-bluetooth-serial-dengan-arduino/


14:31
0,00K/s
3 4G
66%
BT Serial
Connected
SEND
RESET


14:31
2,09K/s
3 4G
66%
BT Serial
Not Connected
Kelompok 3 Mikro dan Interface
Copyrigt by
©Ach.Chusnul Chikam, 2017
SEND
RESET

IX. KESIMPULAN

Berdasarkan project yang telah kami buat, kami dapat menarik kesimpulan bahwa:

- Untuk berkomunikasi serial dapat diaplikasikan dengan proses pengiriman melalui Bluetooth. Dimana dapat dibuat aplikasinya melalui Handphone yang mana programnya disesuaikan dengan kebutuhan dan dapat dibuat pada android studio.
- Dengan menggunakan Bluetooth, pengiriman data dari arduino ke android atau sebaliknya dapat dilakukan dengan jarak yang lebih jauh dibandingkan dengan menggunakan usb yaitu sekitar +-69m.
- Pemanfaatan adc untuk mengatur kecepatan dari motor dc dirasa lebih bebas karena dengan memutar potensiometer kita dapat mengatur kecepatan dari nilai adc yang dihasilkan
- Monitoring dari project diimplementasikan dengan menampilkan kondisi motor dc serta kecepatan dilayar android
- Kontrol kondisi motor dc dilakukan oleh android dengan memasukkan data yang telah disesuaikan dengan arduino dan dikirimkan melalui komunikasi serial Bluetooth hc-05
- Input dari luar yaitu baterai 9 volt diperlukan untuk menambah tegangan dan memaksimalkan putaran dari motor dc sehingga motor dapat berjalan lebih kencang, Selain itu juga lebih fleksibel karena tidak perlu selalu tersambung dengan kabel USB

X. REFERENSI

https://electrocontrol.wordpress.com/2011/05/25/driver-motor-dc-menggunakan-ic-l293d/

http://elektronika-dasar.web.id/wp-content/uploads/2012/06/Konstruksi-Pin-Driver-Motor-DC-IC-L293D.jpg

https://greetea17.wordpress.com/2013/11/03/komunikasi-serial/

http://teknisisampah.blogspot.co.id/2012/05/gearbox.html