

## **KATA PENGANTAR**

Segala puji syukur penulis panjatkan kepada Allah SWT atas segala limpahan karunia, rahmat dan hidayahNya sehingga kami sebagai penulis dapat menyelesaikan tugas project akhir dari mata kuliah Mikroprocessor dan Teknik Antarmuka 2 yang berjudul

### **“ KONTROL MOTOR DAN VISUALISASI KECEPATAN DENGAN MENGUNAKAN PROCESSING ”**

Project Akhir ini dikerjakan berdasarkan teori yang pernah didapatkan di mata kuliah Mikroprocessor dan Teknik Antarmuka 2 serta bimbingan dari dosen mata kuliah Mikroprocessor dan Teknik Antarmuka 2 tersebut yaitu Bapak Akuwan Saleh S.ST, MT. Serta pihak - pihak lain yang sangat membantu hingga project akhir ini dapat diselesaikan.

Project Akhir ini disusun sebagai salah satu syarat agar lulus Mikroprocessor dan Teknik Antarmuka 2 pada semester 5 ini di Politeknik Elektronika Negeri Surabaya. Kami sebagai penulis mengucapkan terima kasih kepada seluruh pihak yang telah membantu hingga penulisan buku project akhir ini selesai.

Penulis menyadari bahwa masih banyak kekurangan dalam perancangan dan pembuatan buku project akhir ini. Oleh karena itu, kritik dan saran senantiasa penulis harapkan dari para pembaca dan semoga buku ini dapat memberikan manfaat bagi para pembaca. Aamiin.

Surabaya, 27 Desember 2017

Penyusun,

## DAFTAR ISI

COVER.....	i
KATA PENGANTAR .....	ii
DAFTAR ISI .....	iii
BAB I PENDAHULUAN .....	1
I. Latar Belakang.....	1
II. Rumusan Masalah.....	2
III. Tujuan.....	2
IV. Manfaat.....	2
BAB II ISI .....	3
I. Dasar Teori .....	3
1.1 Modul Arduino .....	3
1.1.1 Pengertian .....	3
1.1.2 Spesifikasi .....	4
1.1.3 Konfigurasi Pin .....	4
1.1.4 Kegunaan .....	6
1.1.5 Kelebihan .....	7
1.2 Komunikasi Serial .....	8
1.3 Modul Bluetooth .....	9
1.3.1 Pengertian .....	9
1.3.2 Spesifikasi .....	10
1.3.3 Konfigurasi Pin .....	10
1.4 Motor DC .....	12
1.4.1 Pengertian .....	12
1.4.2 Bagian-bagian Motor DC .....	12
1.4.3 Prinsip Kerja .....	13
1.4.4 Pengaturan kecepatan dengan teknik PWM .....	14
1.5 Driver H-Bridge .....	17
1.5.1 Pengertian .....	17
1.5.2 Konfigurasi Pin .....	17

1.6	LED .....	19
1.6.1	Pengertian .....	19
1.6.2	Prinsip Kerja .....	20
1.6.3	Kegunaan .....	21
1.7	Resistor .....	21
1.7.1	Pengertian .....	21
1.7.2	Simbol Resistor .....	22
1.7.3	Kapasitas Daya Resistor .....	22
1.7.4	Nilai Toleransi Resistor .....	23
1.7.5	Jenis-jenis Resistor .....	23
1.8	Gear Box plus Roda .....	25
1.8.1	Pengertian .....	25
1.8.2	Prinsip Kerja .....	26
II.	Komponen dan Peralatan yang digunakan .....	28
1.	Komponen .....	28
1.1	Hardware .....	28
1.2	Software .....	28
2.	Peralatan .....	28
III.	Gambar Rangkaian .....	29
1.	Skema Rangkaian .....	29
2.	Desain Layout PCB .....	29
3.	Simulasi di Proteus .....	31
4.	Tampilan Hardware .....	31
IV.	Hasil Percobaan .....	33
V.	Program .....	38
1.	Arduino .....	38
2.	Processing .....	45
VI.	Analisa .....	55
BAB III.	PENUTUP .....	75
I.	Kesimpulan .....	75
II.	Saran .....	75
DAFTAR PUSTAKA	.....	76

# **BAB I**

## **PENDAHULUAN**

### **I. Latar Belakang**

Teknologi mikrokontroller saat ini telah berkembang pesat. Sehingga aplikasinya semakin luas, penggunaan mikrokontroller untuk mempermudah dan mempercepat penggunaan alat lain, mikrokontroller dapat dikembangkan untuk mengontrol objek yang berada diatas permukaan air maupun didalam air. Salah satu perkembangan aplikasi menggunakan mikrokontroller adalah pengontrolan jarak jauh, dimana computer maupun laptop digunakan sebagai pusat control dengan media kabel maupun nirkabel. Dalam penelitian ini, penulis merancang sebuah alat untuk mengendalikan kecepatan putaran motor DC dengan jarak jauh melalui Bluetooth.

Motor DC merupakan salah satu jenis aktuator yang cukup banyak digunakan dalam bidang industri. Seiring dengan kemajuan teknologi, permasalahan pada dunia industri tentang ketidakstabilan dari kecepatan motor DC sangatlah kompleks, sehingga ketika motor DC tersebut bekerja dalam suatu proses yang membutuhkan kecepatan yang konstan, maka sistem tersebut akan terganggu. Pertimbangan penggunaan kendali dalam dunia industri sangat penting, terutama pada pengaturan kecepatan motor DC. Suatu sistem kendali kecepatan motor DC yang baik harus mempunyai ketahanan terhadap gangguan dan mempunyai respon yang cepat dan akurat. Misalnya pada industri plastik. Pada proses penggulungan plastik, kecepatan penggulungan plastik harus disesuaikan dengan kecepatan mesin pengirim plastik dan juga disesuaikan dengan jari-jari gulungan, jika tidak hasil gulungan plastik tidak rapi atau kusut.

## **II. Rumusan Masalah**

Permasalahan yang akan diteliti pada proyek akhir ini adalah :

1. Bagaimana cara agar motor DC dapat berputar lebih cepat dari sebelumnya
2. Bagaimana mengimplementasikan processing untuk kontrol motor DC
3. Bagaimana membuat koneksi processing dengan Arduino agar tidak menggunakan kabel USB

## **III. Tujuan**

1. Memenuhi tugas project praktikum mikroprosessor dan teknik antarmuka 2.
2. Merancang sebuah alat berbasis mikrokontroler untuk mengkontrol dan memantau aktivitas motor DC dengan menggunakan Bluetooth sebagai komunikasi serial.
3. Dapat menghubungkan arduino dengan android dengan menggunakan Bluetooth HC-05.
4. Dapat memvisualisasikan aktivitas motor DC dengan menggunakan processing.

## **IV. Manfaat**

Manfaatnya adalah dapat diaplikasikan untuk efisiensi aktivitas motor DC pada kehidupan sehari-hari.

# BAB I

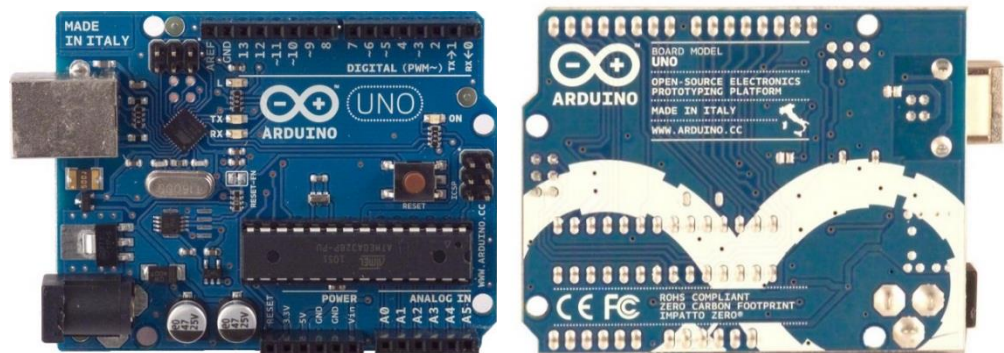
## ISI

### I. Dasar Teori

#### 1.1 Modul Arduino

##### 1.1.1 Pengertian

Arduino Uno adalah board berbasis mikrokontroler pada ATmega328. Board ini memiliki 14 digital input / output pin (dimana 6 pin dapat digunakan sebagai output PWM), 6 input analog, 16 MHz osilator kristal, koneksi USB, jack listrik tombol reset. Pin-pin ini berisi semua yang diperlukan untuk mendukung mikrokontroler, hanya terhubung ke komputer dengan kabel USB atau sumber tegangan bisa didapat dari adaptor AC-DC atau baterai untuk menggunakannya.



a. Tampak depan

b. Tampak belakang

Gambar 1. Modul Arduino Uno

Board Arduino Uno memiliki fitur-fitur baru sebagai berikut :

- Pinout 1.0: ditambah pin SDA dan SCL yang dekat dengan pin AREF dan dua pin baru lainnya yang diletakkan dekat dengan pin RESET, IOREF yang memungkinkan shield-shield untuk menyesuaikan tegangan yang disediakan dari board. Untuk ke depannya, shield akan dijadikan kompatibel/cocok dengan board yang menggunakan AVR yang beroperasi dengan tegangan 5V dan dengan Arduino Due yang beroperasi dengan tegangan 3.3V. Yang ke-dua ini merupakan sebuah pin yang tak terhubung, yang disediakan untuk tujuan kedepannya.
- Sirkuit RESET yang lebih kuat.

- Atmega 16U2 menggantikan 8U2.

### 1.1.2 Spesifikasi

#### Spesifikasi Arduino Uno

<b>Mikrokontroler</b>	<b>ATmega328</b>
<b>Tegangan pengoperasian</b>	<b>5V</b>
<b>Tegangan input yang disarankan</b>	<b>7-12V</b>
<b>Batas tegangan input</b>	<b>6-20V</b>
<b>Jumlah pin I/O digital</b>	<b>14 (6 di antaranya menyediakan keluaran PWM)</b>
<b>Jumlah pin input analog</b>	<b>6</b>
<b>Arus DC tiap pin I/O</b>	<b>40 Ma</b>
<b>Arus DC untuk pin 3.3V</b>	<b>50 mA</b>
<b>Memori Flash</b>	<b>32 KB (ATmega328), sekitar 0.5 KB digunakan oleh bootloader</b>
<b>SRAM</b>	<b>2 KB (ATmega328)</b>
<b>EEPROM</b>	<b>1 KB (ATmega328)</b>
<b>Clock Speed</b>	<b>16 MHz</b>

### 1.1.3 Konfigurasi Pin Arduino

#### Daya (Power)

Arduino UNO dapat disuplai melalui koneksi USB atau dengan sebuah power suplai eksternal. Sumber daya dipilih secara otomatis. Suplai eksternal (non-USB) dapat diperoleh dari sebuah adaptor AC ke DC atau battery. Adaptor dapat dihubungkan dengan mencolokkan sebuah center-positive plug yang panjangnya 2,1 mm ke power jack dari board. Kabel

lead dari sebuah battery dapat dimasukkan dalam header/kepala pin Ground (Gnd) dan pin Vin dari konektor POWER.

Board Arduino UNO dapat beroperasi pada sebuah suplai eksternal 6 sampai 20 Volt. Jika disuplai dengan yang lebih kecil dari 7 V, kiranya pin 5 Volt mungkin mensuplai kecil dari 5 Volt dan board Arduino UNO bisa menjadi tidak stabil. Jika menggunakan suplai yang lebih dari besar 12 Volt, voltage regulator bisa kelebihan panas dan membahayakan board Arduino UNO. Range yang direkomendasikan adalah 7 sampai 12 Volt.

Pin-pin dayanya adalah sebagai berikut:

- VIN. Tegangan input ke Arduino board ketika board sedang menggunakan sumber suplai eksternal (seperti 5 Volt dari koneksi USB atau sumber tenaga lainnya yang diatur). Kita dapat menyuplai tegangan melalui pin ini, atau jika penyuplaian tegangan melalui power jack, aksesnya melalui pin ini.
- 5V. Pin output ini merupakan tegangan 5 Volt yang diatur dari regulator pada board. Board dapat disuplai dengan salah satu suplai dari DC power jack (7-12V), USB connector (5V), atau pin VIN dari board (7-12). Penyuplaian tegangan melalui pin 5V atau 3,3V membypass regulator, dan dapat membahayakan board. Hal itu tidak dianjurkan.
- 3V3. Sebuah suplai 3,3 Volt dihasilkan oleh regulator pada board. Arus maksimum yang dapat dilalui adalah 50 mA.
- GND. Pin ground.

## **Memori**

ATmega328 mempunyai 32 KB (dengan 0,5 KB digunakan untuk bootloader). ATmega 328 juga mempunyai 2 KB SRAM dan 1 KB EEPROM (yang dapat dibaca dan ditulis (RW/read and written) dengan EEPROM library).



## Input dan Output

Setiap 14 pin digital pada Arduino Uno dapat digunakan sebagai input dan output, menggunakan fungsi `pinMode()`, `digitalWrite()`, dan `digitalRead()`. Fungsi-fungsi tersebut beroperasi di tegangan 5 Volt. Setiap pin dapat memberikan atau menerima suatu arus maksimum 40 mA dan mempunyai sebuah resistor pull-up (terputus secara default) 20-50 kOhm. Selain itu, beberapa pin mempunyai fungsi-fungsi special :

- Serial: 0 (RX) dan 1 (TX). Digunakan untuk menerima (RX) dan memancarkan (TX) serial data TTL (Transistor-Transistor Logic). Kedua pin ini dihubungkan ke pin-pin yang sesuai dari chip Serial Atmega8U2 USB-ke-TTL.
- External Interrupts: 2 dan 3. Pin-pin ini dapat dikonfigurasi untuk dipicu sebuah interrupt (gangguan) pada sebuah nilai rendah, suatu kenaikan atau penurunan yang besar, atau suatu perubahan nilai. Lihat fungsi `attachInterrupt()` untuk lebih jelasnya.
- PWM: 3, 5, 6, 9, 10, dan 11. Memberikan 8-bit PWM output dengan fungsi `analogWrite()`.
- SPI: 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK). Pin-pin ini mensupport komunikasi SPI menggunakan SPI library.
- LED: 13. Ada sebuah LED yang terpasang, terhubung ke pin digital 13. Ketika pin bernilai HIGH LED menyala, ketika pin bernilai LOW LED mati.

Arduino UNO mempunyai 6 input analog, diberi label A0 sampai A5, setiapnya memberikan 10 bit resolusi (contohnya 1024 nilai yang berbeda). Secara default, 6 input analog tersebut mengukur dari ground sampai tegangan 5 Volt, dengan itu mungkin untuk mengganti batas atas dari rangenya dengan menggunakan pin AREF dan fungsi `analogReference()`. Di sisi lain, beberapa pin mempunyai fungsi special :

- TWI: pin A4 atau SDA dan pin A5 atau SCL. Mensupport komunikasi TWI dengan menggunakan Wire library

Ada sepasang pin lainnya pada board:

- AREF. Referensi tegangan untuk input analog. Digunakan dengan `analogReference()`.
- Reset. Membawa saluran ini LOW untuk mereset mikrokontroler. Secara khusus, digunakan untuk menambahkan sebuah tombol reset untuk melindungi yang memblock sesuatu pada board.

Lihat juga pemetaan antara pin Arduino dengan port Atmega328. Pemetaan untuk Atmega8, 168, dan 328 adalah identik.

#### 1.1.4 Kegunaan

Arduino yang dikontrol penuh oleh mikrokontroler ATmega328, banyak hal yang bisa dilakukan itu semua tergantung kreatifitas anda. Arduino dapat disambungkan dan mengontrol led, beberapa led, bahkan banyak led, motor DC, relay, servo, modul dan sensor-sensor, serta banyak lagi komponen lainnya. Platform Arduino sudah sangat populer sekarang ini, sehingga tidak akan kesulitan untuk memperoleh informasi, tutorial dan berbagai eksperimen yang menarik yang tersedia banyak di internet.

Dengan Arduino, dunia hardware bisa bekerja sama dengan dunia software. Anda bisa mengontrol hardware dari software dimana hardware bisa memberikan data kepada software. Semuanya bisa dilakukan dengan relatif mudah, murah, dan menyenangkan.

#### 1.1.5 Kelebihan

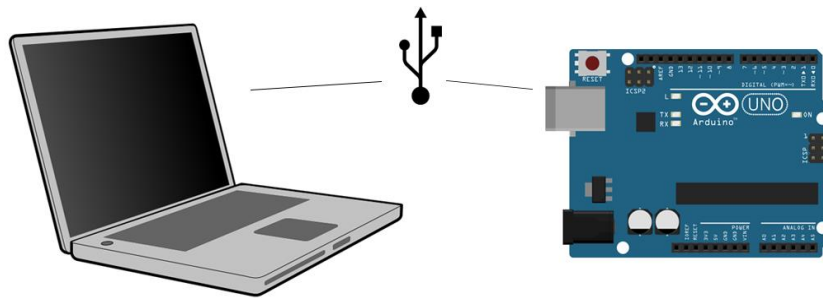
- Murah – Papan (perangkat keras) Arduino biasanya dijual relatif murah (antara 125ribu hingga 400ribuan rupiah saja) dibandingkan dengan platform mikrokontroler pro lainnya. Jika ingin lebih murah lagi, tentu bisa dibuat sendiri dan itu sangat mungkin sekali karena semua sumber daya untuk membuat sendiri Arduino tersedia lengkap di website Arduino bahkan di website-website komunitas Arduino lainnya. Tidak hanya cocok untuk Windows, namun juga cocok bekerja di Linux.

- Sederhana dan mudah pemrogramannya – Perlu diketahui bahwa lingkungan pemrograman di Arduino mudah digunakan untuk pemula, dan cukup fleksibel bagi mereka yang sudah tingkat lanjut. Untuk guru/dosen, Arduino berbasis pada lingkungan pemrograman Processing, sehingga jika mahasiswa atau murid-murid terbiasa menggunakan Processing tentu saja akan mudah menggunakan Arduino.
- Perangkat lunaknya Open Source – Perangkat lunak Arduino IDE dipublikasikan sebagai Open Source, tersedia bagi para pemrogram berpengalaman untuk pengembangan lebih lanjut. Bahasanya bisa dikembangkan lebih lanjut melalui pustaka-pustaka C++ yang berbasis pada Bahasa C untuk AVR.
- Perangkat kerasnya Open Source – Perangkat keras Arduino berbasis mikrokontroler ATMEGA8, ATMEGA168, ATMEGA328 dan ATMEGA1280 (yang terbaru ATMEGA2560). Dengan demikian siapa saja bisa membuatnya (dan kemudian bisa menjualnya) perangkat keras Arduino ini, apalagi bootloader tersedia langsung dari perangkat lunak Arduino IDE-nya. Bisa juga menggunakan breadboard untuk membuat perangkat Arduino beserta periferal-periferal lain yang dibutuhkan.

## 1.2 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 paralel, 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  $8 \leq n \leq 128$ .



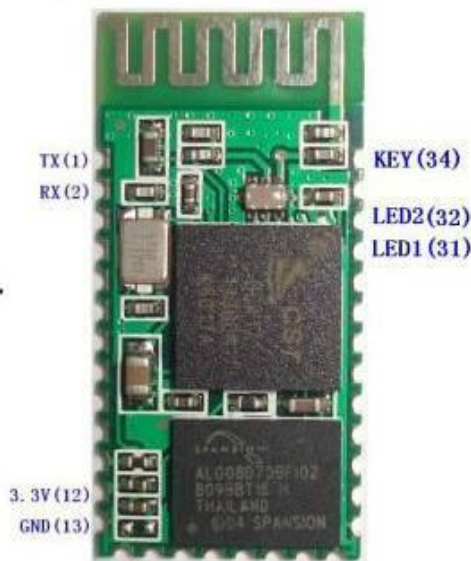
Gambar 2. Komunikasi Serial antara Arduino dengan PC

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 penggunaan 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.

## 1.3 Modul Bluetooth

### 1.3.1 Pengertian

HC-05 Adalah sebuah modul Bluetooth SPP (Serial Port Protocol) yang mudah digunakan untuk komunikasi serial wireless (nirkabel) yang mengkonversi port serial ke Bluetooth. HC-05 menggunakan modulasi bluetooth V2.0 + EDR (Enhanced Data Rate) 3 Mbps dengan memanfaatkan gelombang radio berfrekuensi 2,4 GHz. Modul ini dapat digunakan sebagai slave maupun master.



Gambar 3.1 Modul Bluetooth HC-05

HC-05 memiliki 2 mode konfigurasi, yaitu AT mode dan Communication mode. AT mode berfungsi untuk melakukan pengaturan konfigurasi dari HC-05. Sedangkan Communication mode berfungsi untuk melakukan komunikasi bluetooth dengan piranti lain. 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.

Dalam penggunaannya, HC-05 dapat beroperasi tanpa menggunakan driver khusus. Untuk berkomunikasi antar Bluetooth, minimal harus memenuhi dua kondisi berikut :

- Komunikasi harus antara master dan slave.
- Password harus benar (saat melakukan pairing).

Jarak sinyal dari HC-05 adalah 30 meter, dengan kondisi tanpa halangan.

#### 1.3.2 Spesifikasi

Hardware :

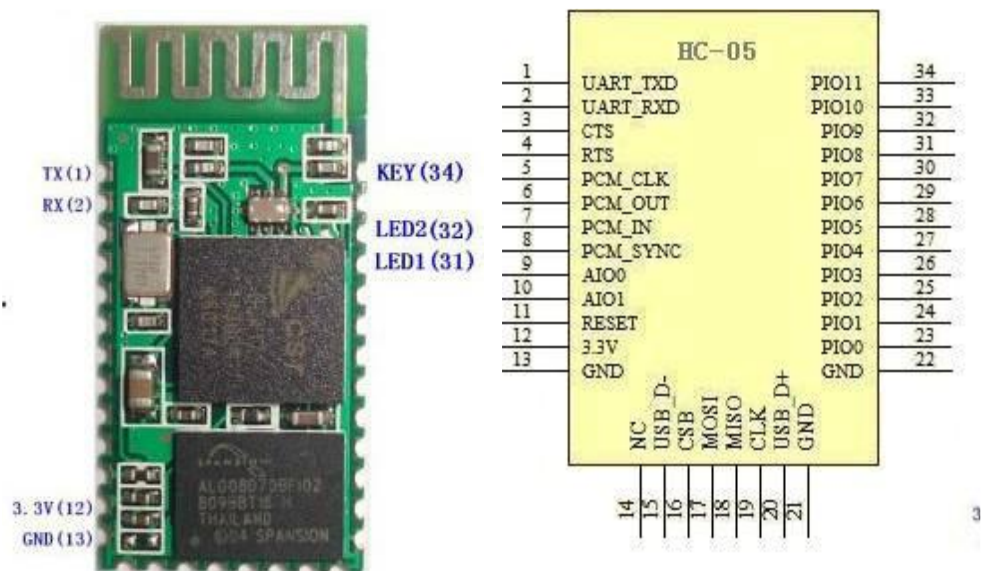
- Sensitivitas -80dBm (Typical)
- Daya transmit RF sampai dengan +4dBm.
- Operasi daya rendah 1,8V – 3,6V I/O.
- Kontrol PIO.
- Antarmuka UART dengan baudrate yang dapat diprogram.
- Dengan antena terintegrasi.

Software :

- Default baudrate 9600, Data bit : 8, Stop bit = 1, Parity : No Parity, Mendukung baudrate : 9600, 19200, 38400, 57600, 115200, 230400 dan 460800.
- Auto koneksi pada saat device dinyalakan (default).
- Auto reconnect pada menit ke 30 ketika hubungan putus karena range koneksi.

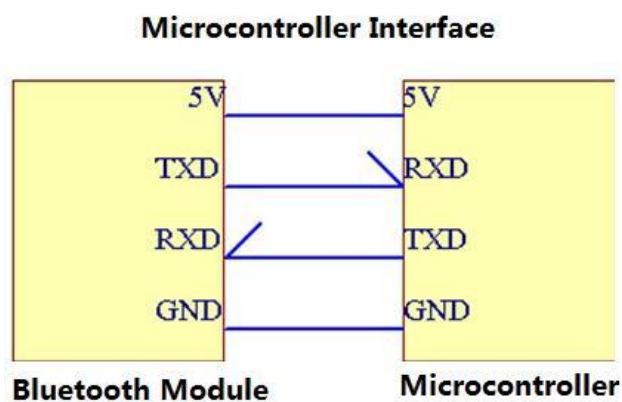
### 1.3.3 Konfigurasi Pin

Modul Bluetooth HC-05 dengan supply tegangan sebesar 3,3 V ke pin 12 modul Bluetooth sebagai VCC. Pin 1 pada modul Bluetooth sebagai transmitter. kemudian pin 2 pada Bluetooth sebagai receiver. Berikut ini adalah konfigurasi untuk modul Bluetooth HC-05 :



Gambar 3.2 Konfigurasi Modul Bluetooth HC-05

Berikut merupakan Bluetooth-to-Serial-Module HC-05 yang dapat dilihat pada gambar 2.3 dibawah ini:



Gambar 3.3 Bluetooth-to-Serial-Module HC-05

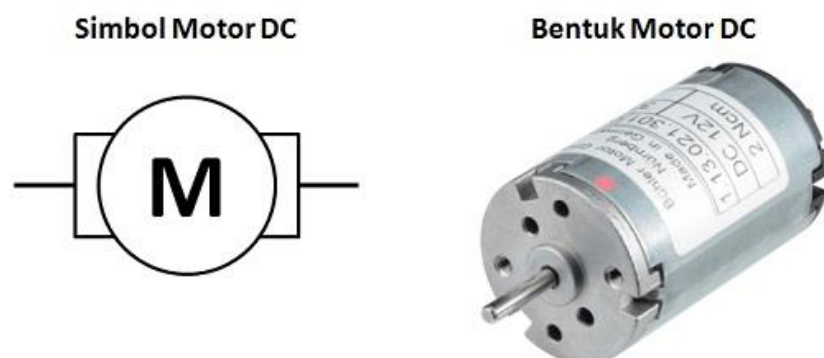
Konfigurasi pin modul Bluetooth HC-05 dapat dilihat pada table berikut ini :

No	Nomor Pin	Nama	Fungsi
1	Pin 1	Key	-
2	Pin 2	VCC	Sumber Tegangan 5 Volt
3	Pin 3	GND	Ground Tegangan
4	Pin 4	TXD	Mengirim Data
5	Pin 5	RXD	Menerima Data
6	Pin 6	STATE	-

## 1.4 Motor DC

### 1.4.1 Pengertian

Motor DC adalah jenis motor listrik yang bekerja menggunakan sumber tegangan DC. Motor DC atau motor arus searah sebagaimana namanya yaitu menggunakan arus langsung dan tidak langsung/direct-unidirectional.



Gambar 4.1 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 DC digunakan pada penggunaan khusus dimana diperlukan penyalan torque yang tinggi atau percepatan yang tetap untuk kisaran kecepatan yang luas.

#### 1.4.2 Bagian-bagian Motor DC

Motor DC memiliki 3 bagian atau komponen utama untuk dapat berputar sebagai berikut.:

##### **Kutub medan.**

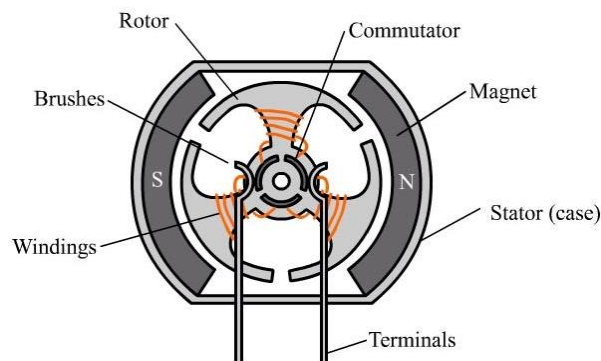
Motor DC sederhana memiliki dua kutub medan: kutub utara dan kutub selatan. Garis magnetik energi membesar melintasi ruang terbuka diantara kutub-kutub dari utara ke selatan. Untuk motor yang lebih besar atau lebih komplek terdapat satu atau lebih elektromagnet.

##### **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.

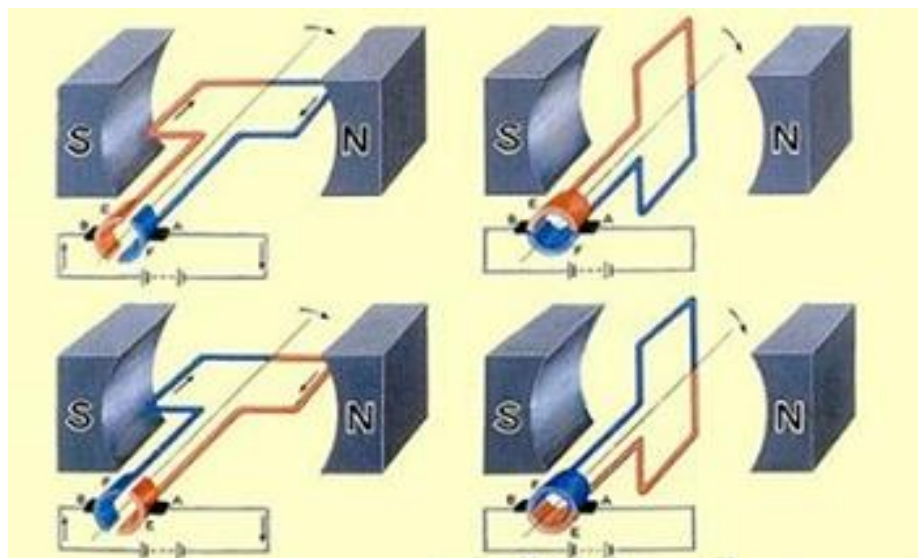


Gambar 4.2 Komponen Utama Motor DC

### 1.4.3 Prinsip Kerja Motor DC

Terdapat dua bagian utama pada sebuah Motor Listrik DC, yaitu Stator dan Rotor. Stator adalah bagian motor yang tidak berputar, bagian yang statis ini terdiri dari rangka dan kumparan medan. Sedangkan Rotor adalah bagian yang berputar, bagian Rotor ini terdiri dari kumparan Jangkar. Dua bagian utama ini dapat dibagi lagi menjadi beberapa komponen penting yaitu diantaranya adalah Yoke (kerangka magnet), Poles (kutub motor), Field winding (kumparan medan magnet), Armature Winding (Kumparan Jangkar), Commutator (Komutator) dan Brushes (kuas/sikat arang).

Pada prinsipnya motor listrik DC menggunakan fenomena elektromagnet untuk bergerak, ketika arus listrik diberikan ke kumparan, permukaan kumparan yang bersifat utara akan bergerak menghadap ke magnet yang berkutub selatan dan kumparan yang bersifat selatan akan bergerak menghadap ke utara magnet. Saat ini, karena kutub utara kumparan bertemu dengan kutub selatan magnet ataupun kutub selatan kumparan bertemu dengan kutub utara magnet maka akan terjadi saling tarik menarik yang menyebabkan pergerakan kumparan berhenti.



Gambar 4.3 Prinsip Kerja Motor DC

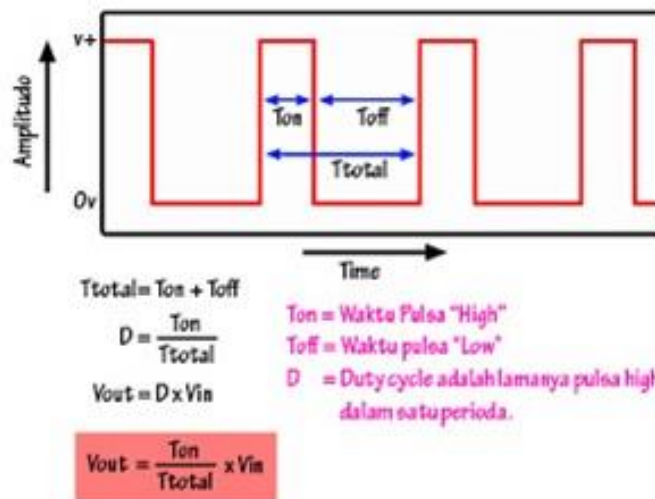
Untuk menggerakannya lagi, tepat pada saat kutub kumparan berhadapan dengan kutub magnet, arah arus pada kumparan dibalik. Dengan demikian, kutub utara kumparan akan berubah menjadi kutub selatan dan kutub selatannya akan berubah menjadi kutub utara. Pada saat perubahan kutub tersebut terjadi, kutub selatan kumparan akan berhadapan dengan kutub selatan magnet dan kutub utara kumparan akan berhadapan dengan kutub utara magnet. Karena kutubnya sama, maka akan terjadi tolak menolak sehingga kumparan bergerak memutar hingga utara kumparan berhadapan dengan selatan magnet dan selatan kumparan berhadapan dengan utara magnet. Pada saat ini, arus yang mengalir ke kumparan dibalik lagi dan kumparan akan berputar lagi karena adanya perubahan kutub. Siklus ini akan berulang-ulang hingga arus listrik pada kumparan diputuskan.

#### 1.4.4 Pengaturan kecepatan dengan teknik PWM

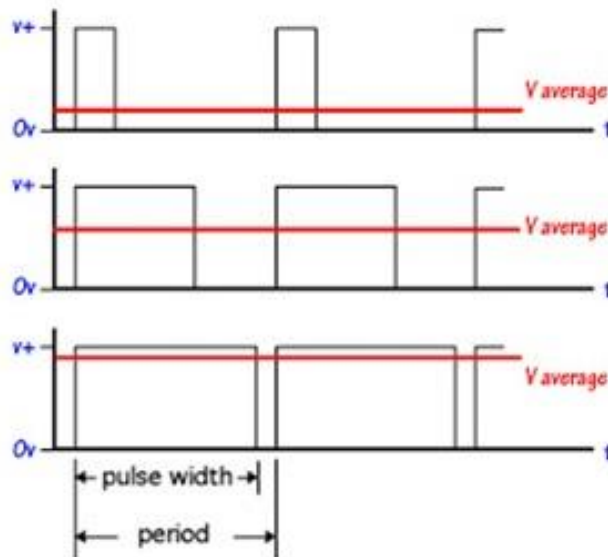
Cara pengaturan kecepatan yang digunakan adalah dengan menggunakan teknik PWM. Teknik PWM (Pulse Width Modulation) adalah salah satu teknik untuk mengatur kecepatan motor DC yang umum digunakan. Dengan menggunakan PWM kita dapat mengatur kecepatan yang diinginkan dengan mudah.

Teknik PWM untuk pengaturan kecepatan motor adalah pengaturan kecepatan motor dengan cara merubah-ubah besarnya duty cycle pulsa. Pulsa yang berubah ubah duty cycle-nya inilah yang menentukan kecepatan motor. Besarnya amplitudo dan frekuensi pulsa adalah tetap, sedangkan besarnya duty cycle berubah-ubah sesuai dengan kecepatan yang diinginkan, semakin besar duty cycle maka semakin cepat pula kecepatan motor, dan sebaliknya semakin kecil duty cycle maka semakin pelan pula kecepatan motor.

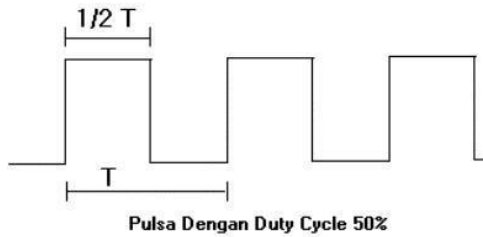
Lebar pulsa PWM berbanding lurus dengan amplitude sinyal asli yang belum termodulasi. Artinya, sinyal PWM memiliki frekuensi gelombang yang tetap namun duty cycle bervariasi antara 0% hingga 100%.



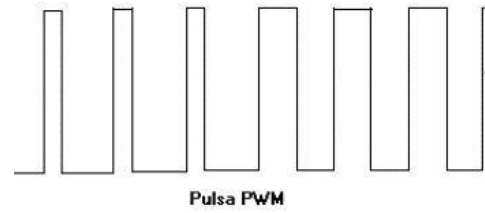
Dari persamaan diatas, diketahui bahwa perubahan duty cycle akan merubah tegangan output atau tegangan rata-rata seperti gambar dibawah ini.



Sebagai contoh bentuk pulsa yang dikirimkan adalah seperti pada gambar 4.4, pulsa kotak dengan duty cycle pulsa 50%. Sedangkan sebagai contoh bentuk pulsa PWM adalah seperti pada gambar 4.5.



Gambar 4.4



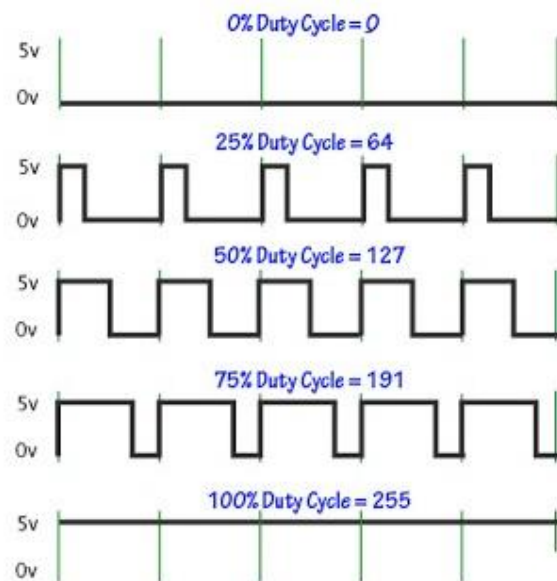
Gambar 4.5

Dari gambar 1, semakin besar duty cycle pulsa kotak maka semakin lama pula posisi logika high. Jika motor diatur agar berjalan ketika diberi logika high, maka jika memberi pulsa seperti pada gambar 1 diatas, maka motor akan berada pada kondisi “nyala-mati-nyala-mati” sesuai dengan bentuk pulsa tersebut. Semakin lama motor berada pada kondisi “nyala” maka semakin cepat pula kecepatan motor tersebut. Motor akan berputar dengan kecepatan maksimum jika mendapat pulsa dengan duty cycle 100%. Dengan kata lain motor mendapat logika high terus menerus.

Dengan mengatur besarnya duty cycle pulsa kotak yang dikirimkan, kita dapat mengatur banyaknya logika high yang diberikan pada motor, dengan kata lain mengatur lamanya waktu motor untuk berputar dalam satu periode pulsa. Jika lamanya waktu motor untuk berputar dalam satu periode pulsa ini berubah maka kecepatan putaran motor juga akan berubah, sesuai dengan duty cycle atau waktu motor untuk berputar dalam satu periode pulsa.

PWM merupakan salah satu teknik untuk mendapatkan sinyal analog dari sebuah piranti digital. Sebenarnya sinyal PWM dapat dibangkitkan dengan banyak cara, yaitu secara analog menggunakan IC op-amp atau secara digital. Secara analog setiap perubahan PWM-nya sangat halus, sedangkan secara digital setiap perubahan PWM dipengaruhi oleh resolusi PWM itu sendiri. Resolusi adalah jumlah variasi perubahan nilai dalam PWM tersebut. Misalkan suatu PWM memiliki resolusi 8 bit, berarti PWM ini memiliki variasi perubahan nilai sebanyak 256 variasi mulai dari

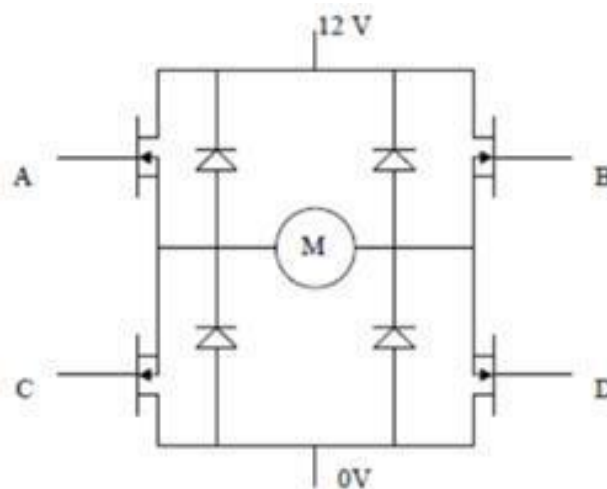
0 – 225 perubahan nilai yang mewakili duty cycle 0% – 100% dari keluaran PWM tersebut.



## 1.5 Driver H-Bridge

### 1.5.1 Pengertian

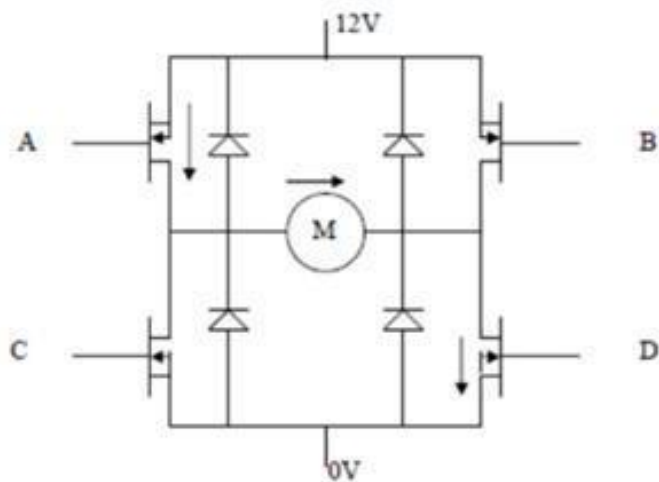
H-bridge adalah sebuah perangkat keras berupa rangkaian yang berfungsi untuk menggerakkan motor. Rangkaian ini diberi nama H-bridge karena bentuk rangkaiannya yang menyerupai huruf H seperti pada Gambar berikut :



Gambar 5.1 Driver H-Bridge

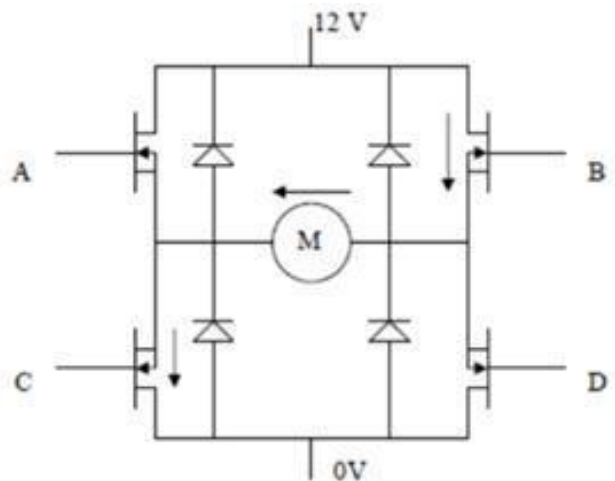
### 1.5.2 Konfigurasi Pin H-Bridge

Rangkaian ini terdiri dari dua buah MOSFET kanal P dan dua buah MOSFET kanal N. Prinsip kerja rangkaian ini adalah dengan mengatur mati-hidupnya ke empat MOSFET tersebut. Huruf M pada gambar adalah motor DC yang akan dikendalikan. Bagian atas rangkaian akan dihubungkan dengan sumber daya kutub positif, sedangkan bagian bawah rangkaian akan dihubungkan dengan sumber daya kutub negatif. Pada saat MOSFET A dan MOSFET D on sedangkan MOSFET B dan MOSFET C off, maka sisi kiri dari gambar motor akan terhubung dengan kutub positif dari catu daya, sedangkan sisi sebelah kanan motor akan terhubung dengan kutub negatif dari catu daya sehingga motor akan bergerak searah jarum jam dijelaskan pada gambar berikut.



Gambar 5.2 H-bridge konfigurasi MOSFET A&D on, B&C off

Sebaliknya, jika MOSFET B dan MOSFET C on sedangkan MOSFET A dan MOSFET D off, maka sisi kanan motor akan terhubung dengan kutub positif dari catu daya sedangkan sisi kiri motor akan terhubung dengan kutub negatif dari catu daya. Maka motor akan bergerak berlawanan arah jarum jam dijelaskan pada gambar berikut.



Gambar 5.3 H-bridge konfigurasi MOSFET A&D off, B&C on

Konfigurasi lainnya adalah apabila MOSFET A dan MOSFET B on sedangkan MOSFET C dan MOSFET D off. Konfigurasi ini akan menyebabkan sisi kiri dan kanan motor terhubung pada kutub yang sama yaitu kutub positif sehingga tidak ada perbedaan tegangan diantara dua buah polaritas motor, sehingga motor akan diam.

Konfigurasi seperti ini disebut dengan konfigurasi break. Begitu pula jika MOSFET C dan MOSFET D saklar on, sedangkan MOSFET A dan MOSFET C off, kedua polaritas motor akan terhubung pada kutub negatif dari catu daya. Maka tidak ada perbedaan tegangan pada kedua polaritas motor, dan motor akan diam. Konfigurasi yang harus dihindari adalah pada saat MOSFET A dan MOSFET C on secara bersamaan atau MOSFET B dan MOSFET D on secara bersamaan. Pada konfigurasi ini akan terjadi hubungan arus singkat antara kutub positif catu daya dengan kutub negatif catu daya.

#### Konfigurasi Pengujian H-bridge MOSFET

A	B	C	D	Aksi
1	0	0	1	Motor berputar searah jarum jam
0	1	1	0	Motor berputar berlawanan arah jarum jam
0	0	0	0	Bebas
0	0	1	1	Pengereman
1	1	0	0	Pengereman

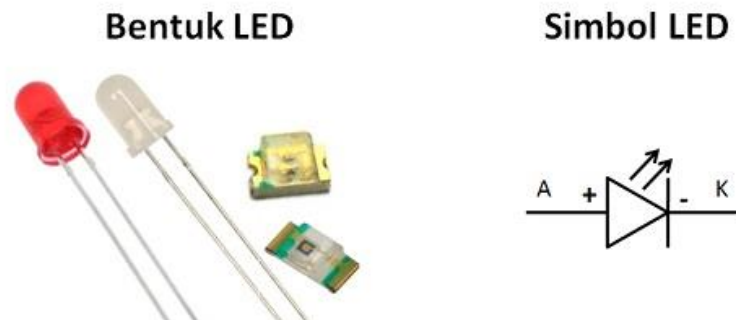


## 1.6 LED

### 1.6.1 Pengertian

Light Emitting Diode atau sering disingkat dengan LED adalah komponen elektronika yang dapat memancarkan cahaya monokromatik ketika diberikan tegangan maju. LED merupakan keluarga Dioda yang terbuat dari bahan semikonduktor. Warna-warna Cahaya yang dipancarkan oleh LED tergantung pada jenis bahan semikonduktor yang dipergunakannya. LED juga dapat memancarkan sinar inframerah yang tidak tampak oleh mata seperti yang sering kita jumpai pada Remote Control TV ataupun Remote Control perangkat elektronik lainnya.

Bentuk LED mirip dengan sebuah bohlam (bola lampu) yang kecil dan dapat dipasangkan dengan mudah ke dalam berbagai perangkat elektronika. Berbeda dengan Lampu Pijar, LED tidak memerlukan pembakaran filamen sehingga tidak menimbulkan panas dalam menghasilkan cahaya. Oleh karena itu, saat ini LED (Light Emitting Diode) yang bentuknya kecil telah banyak digunakan sebagai lampu penerang dalam LCD TV yang mengganti lampu tube.



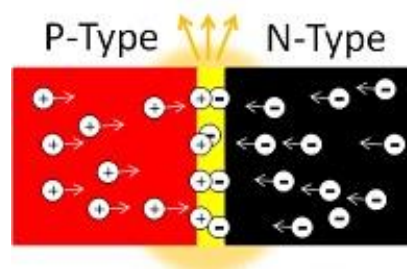
Gambar 6.1 Bentuk dan Simbol dari LED

### 1.6.2 Prinsip Kerja

Seperti dikatakan sebelumnya, LED merupakan keluarga dari Dioda yang terbuat dari Semikonduktor. Cara kerjanya pun hampir sama dengan Dioda yang memiliki dua kutub yaitu kutub Positif (P) dan Kutub

Negatif (N). LED hanya akan memancarkan cahaya apabila dialiri tegangan maju (bias forward) dari Anoda menuju ke Katoda.

LED terdiri dari sebuah chip semikonduktor yang di doping sehingga menciptakan junction P dan N. Yang dimaksud dengan proses doping dalam semikonduktor adalah proses untuk menambahkan ketidakmurnian (impurity) pada semikonduktor yang murni sehingga menghasilkan karakteristik kelistrikan yang diinginkan. Ketika LED dialiri tegangan maju atau bias forward yaitu dari Anoda (P) menuju ke Katoda (K), Kelebihan Elektron pada N-Type material akan berpindah ke wilayah yang kelebihan Hole (lubang) yaitu wilayah yang bermuatan positif (P-Type material). Saat Elektron berjumpa dengan Hole akan melepaskan photon dan memancarkan cahaya monokromatik (satu warna).









Gambar 6.2 Cara Kerja LED

LED atau Light Emitting Diode yang memancarkan cahaya ketika dialiri tegangan maju ini juga dapat digolongkan sebagai Transduser yang dapat mengubah Energi Listrik menjadi Energi Cahaya.

### 1.6.3 Kegunaan

Teknologi LED memiliki berbagai kelebihan seperti tidak menimbulkan panas, tahan lama, tidak mengandung bahan berbahaya seperti merkuri, dan hemat listrik serta bentuknya yang kecil ini semakin populer dalam bidang teknologi pencahayaan. Berbagai produk yang memerlukan cahaya pun mengadopsi teknologi Light Emitting Diode (LED) ini. Berikut ini beberapa pengaplikasiannya LED dalam kehidupan sehari-hari.

🔦 Lampu Penerangan Rumah.

-  Lampu Penerangan Jalan.
-  Papan Iklan (Advertising).
-  Backlight LCD (TV, Display Handphone, Monitor).
-  Lampu Dekorasi Interior maupun Exterior.
-  Lampu Indikator.
-  Pemancar Infra Merah pada Remote Control (TV, AC, AV Player).

## 1.7 Resistor

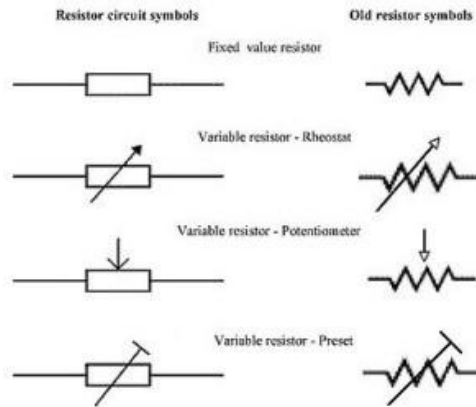
### 1.7.1 Pengertian

Resistor adalah komponen elektronika yang berfungsi untuk menghambat atau membatasi aliran listrik yang mengalir dalam suatu rangkaian elektronika. Sebagaimana fungsi resistor yang sesuai namanya bersifat resistif dan termasuk salah satu komponen elektronika dalam kategori komponen pasif. Satuan atau nilai resistansi suatu resistor di sebut Ohm dan dilambangkan dengan simbol Omega ( $\Omega$ ).

Sesuai hukum Ohm bahwa resistansi berbanding terbalik dengan jumlah arus yang mengalir melaluinya. Selain nilai resistansinya (Ohm) resistor juga memiliki nilai yang lain seperti nilai toleransi dan kapasitas daya yang mampu dilewatkannya. Semua nilai yang berkaitan dengan resistor tersebut penting untuk diketahui dalam perancangan suatu rangkaian elektronika oleh karena itu pabrikan resistor selalu mencantumkan dalam kemasan resistor tersebut.

### 1.7.2 Simbol Resistor

Berikut adalah simbol resistor dalam bentuk gambar yang sering digunakan dalam suatu desain rangkaian elektronika.



Gambar 7.1 Simbol Resistor

Resistor dalam suatu teori dan penulisan formula yang berhubungan dengan resistor disimbolkan dengan huruf “R”. Kemudian pada desain skema elektronika resistor tetap disimbolkan dengan huruf “R”, resistor variabel disimbolkan dengan huruf “VR” dan untuk resistor jenis potensiometer ada yang disimbolkan dengan huruf “VR” dan “POT”.

### 1.7.3 Kapasitas Daya Resistor

Kapasitas daya pada resistor merupakan nilai daya maksimum yang mampu dilewatkan oleh resistor tersebut. Nilai kapasitas daya resistor ini dapat dikenali dari ukuran fisik resistor dan tulisan kapasitas daya dalam satuan Watt untuk resistor dengan kemasan fisik besar. Menentukan kapasitas daya resistor ini penting dilakukan untuk menghindari resistor rusak karena terjadi kelebihan daya yang mengalir sehingga resistor terbakar dan sebagai bentuk efisiensi biaya dan tempat dalam pembuatan rangkaian elektronika.

### 1.7.4 Nilai Toleransi Resistor

Toleransi resistor merupakan perubahan nilai resistansi dari nilai yang tercantum pada badan resistor yang masih diperbolehkan dan dinyatakan resistor dalam kondisi baik. Toleransi resistor merupakan salah satu perubahan karakteristik resistor yang terjadi akibat operasional resistor tersebut.

Nilai toleransi resistor ini ada beberapa macam yaitu resistor dengan toleransi kesalahan 1% (resistor 1%), resistor dengan toleransi kesalahan 2% (resistor 2%), resistor dengan toleransi kesalahan 5% (resistor 5%) dan resistor dengan toleransi 10% (resistor 10%). Nilai toleransi resistor selalu dicantumkan di kemasan resistor dengan kode warna maupun kode huruf. Sebagai contoh resistor dengan toleransi 5% maka dituliskan dengan kode warna pada cincin ke 4 warna emas atau dengan kode huruf J pada resistor dengan fisik kemasan besar. Resistor yang banyak dijual dipasaran pada umumnya resistor 5% dan resistor 1%.

#### 1.7.5 Jenis-jenis Resistor

Berdasarkan jenis dan bahan yang digunakan untuk membuat resistor dibedakan menjadi resistor kawat, resistor arang dan resistor oksida logam atau resistor metal film.

##### 1. Resistor Kawat

Resistor kawat atau wirewound resistor merupakan resistor yang dibuat dengan bahan kawat yang dililitkan. Sehingga nilai resistansi resistor ditentukan dari panjangnya kawat yang dililitkan. Resistor jenis ini pada umumnya dibuat dengan kapasitas daya yang besar.



Gambar 7.2 Resistor Kawat

##### 2. Resistor Arang

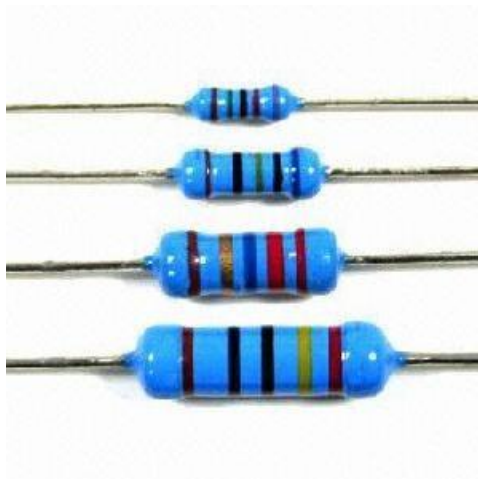
Resistor arang atau resistor karbon merupakan resistor yang dibuat dengan bahan utama batang arang atau karbon. Resistor karbon ini merupakan resistor yang banyak digunakan dan banyak diperjual belikan. Dipasaran resistor jenis ini dapat kita jumpai dengan kapasitas daya 1/16 Watt, 1/8 Watt, 1/4 Watt, 1/2 Watt, 1 Watt, 2 Watt dan 3 Watt.



Gambar 7.3 Resistor Arang

### 3. Resistor Oksida Logam (Metal Film Resistor)

Resistor oksida logam atau lebih dikenal dengan nama resistor metal film merupakan resistor yang dibuat dengan bahan utama oksida logam yang memiliki karakteristik lebih baik. Resistor metal film ini dapat ditemui dengan nilai toleransi 1% dan 2%. Bentuk fisik resistor metal film ini mirip dengan resistor karbon hanya beda warna dan jumlah cincin warna yang digunakan dalam penilaian resistor tersebut. Sama seperti resistor karbon, resistor metal film ini juga diproduksi dalam beberapa kapasitas daya yaitu 1/8 Watt, 1/4 Watt, 1/2 Watt. Resistor metal film ini banyak digunakan untuk keperluan pengukuran, perangkat industri dan perangkat militer.



Gambar 7.4 Resistor Oksida Logam

## 1.8 Gearbox plus roda

### 1.8.1 Pengertian

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



Gambar 8.1 Motor DC, Gearbox plus Roda

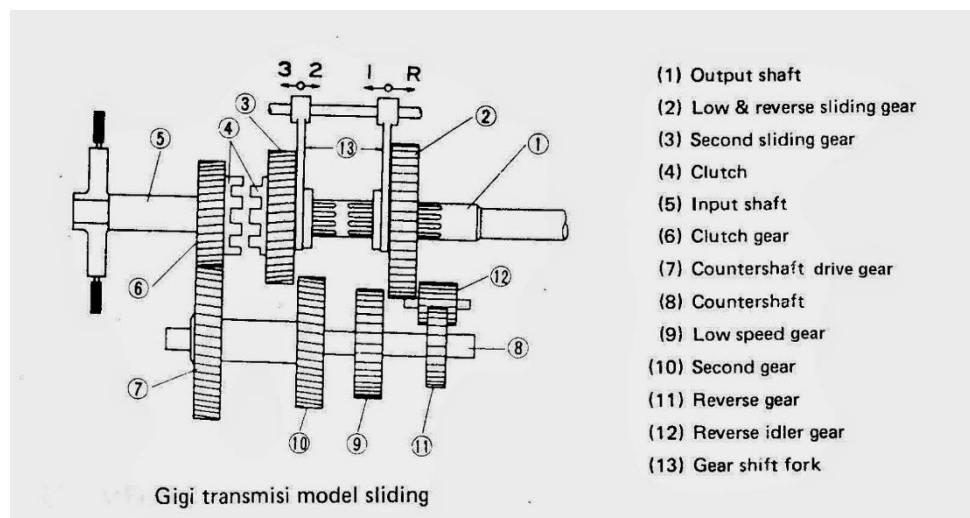
Gearbox atau transmisi adalah salah satu komponen utama motor yang disebut sebagai sistem pemindah tenaga, transmisi berfungsi untuk memindahkan dan mengubah tenaga dari motor yang berputar, yang digunakan untuk memutar spindel mesin maupun melakukan gerakan feeding. Transmisi juga berfungsi untuk mengatur 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.

### 1.8.2 Prinsip Kerja

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.



Gambar 8.2 Bagian-bagian Gearbox

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 berfungsi untuk meneruskan putaran dari clutch kopling ke mainshaft (poros utama), sehingga putaran bisa diteruskan ke gear-gear. Input shaft juga sebagai poros dukungan 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).  
Main shaft yang berfungsi sebagai tempat kedudukan 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.
- Clutch housing.  
Clutch housing adalah rumah dari clutch kopling yang berfungsi sebagai pelindung clutch kopling, clutch housing juga berfungsi sebagai tempat kedudukan 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 transmit

## II. Komponen dan Peralatan yang digunakan

### 1. Komponen

Hardware :

✓ Modul Arduino	1 buah
✓ Modul Bluetooth	1 buah
✓ Motor DC	2 buah
✓ Driver H-Bridge	1 buah
✓ LED	4 buah
✓ Resistor 220 $\Omega$	4 buah
✓ Baterai 12 Volt	1 buah
✓ Gear Box	2 buah
✓ Roda	2 buah

Software :

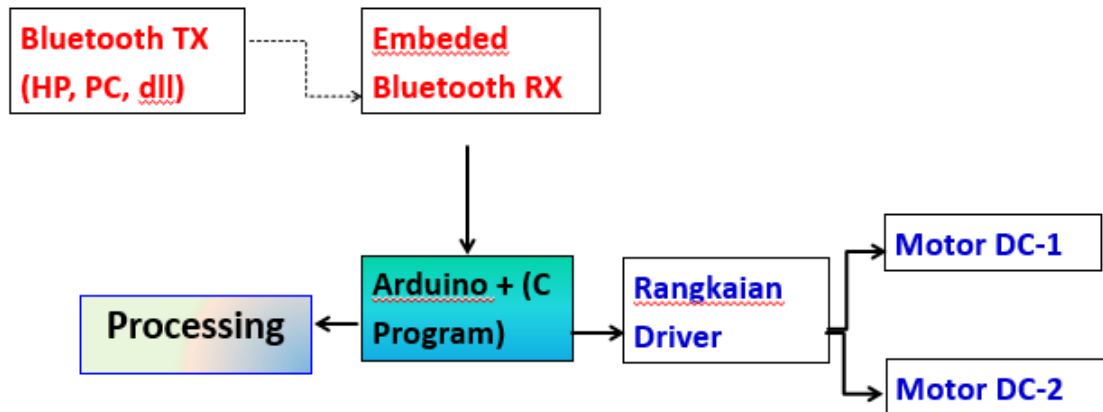
- ✓ Software Arduino
- ✓ Software Processing

### 2. Peralatan

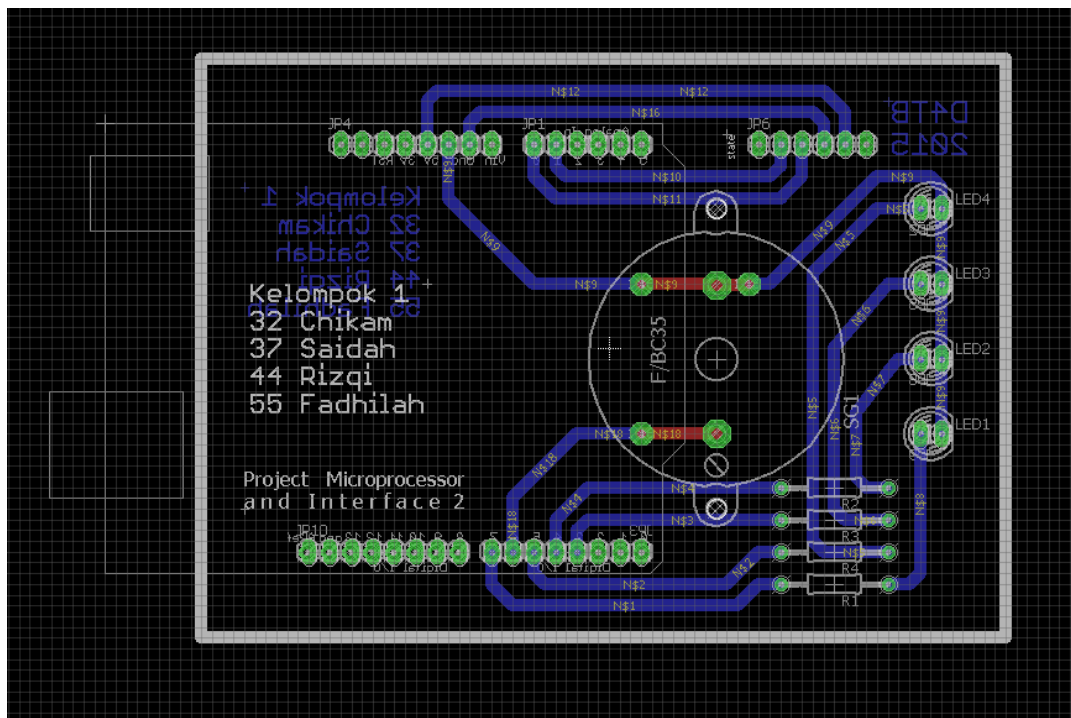
- PCB Layout
- Kabel Jumper
- Timah
- Solder

### III. Gambar Rangkaian

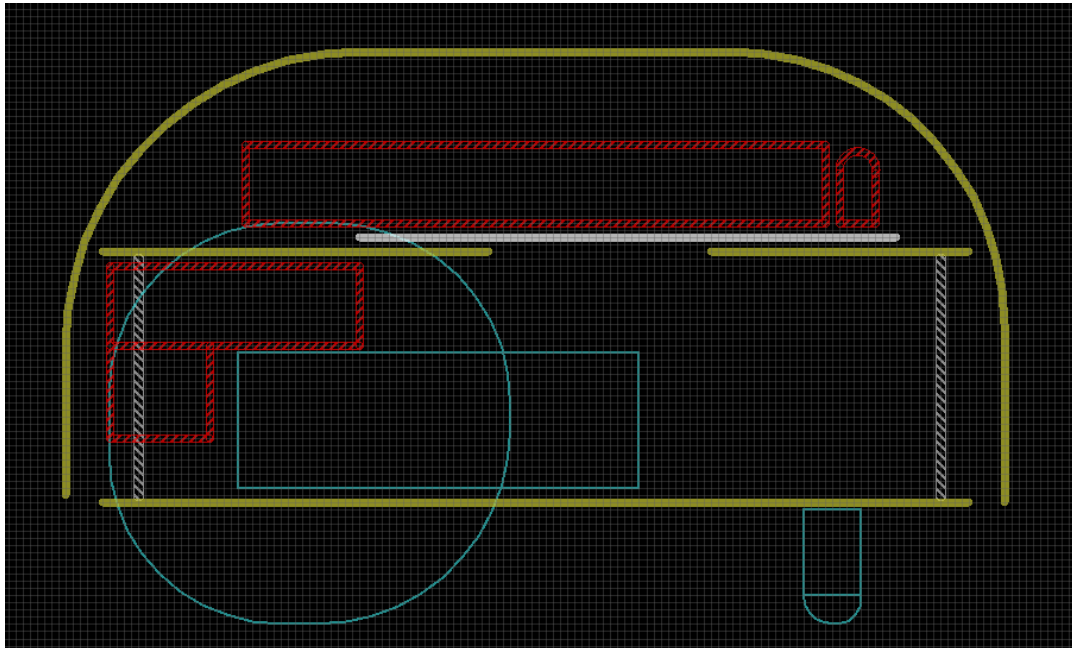
Skema Rangkaian



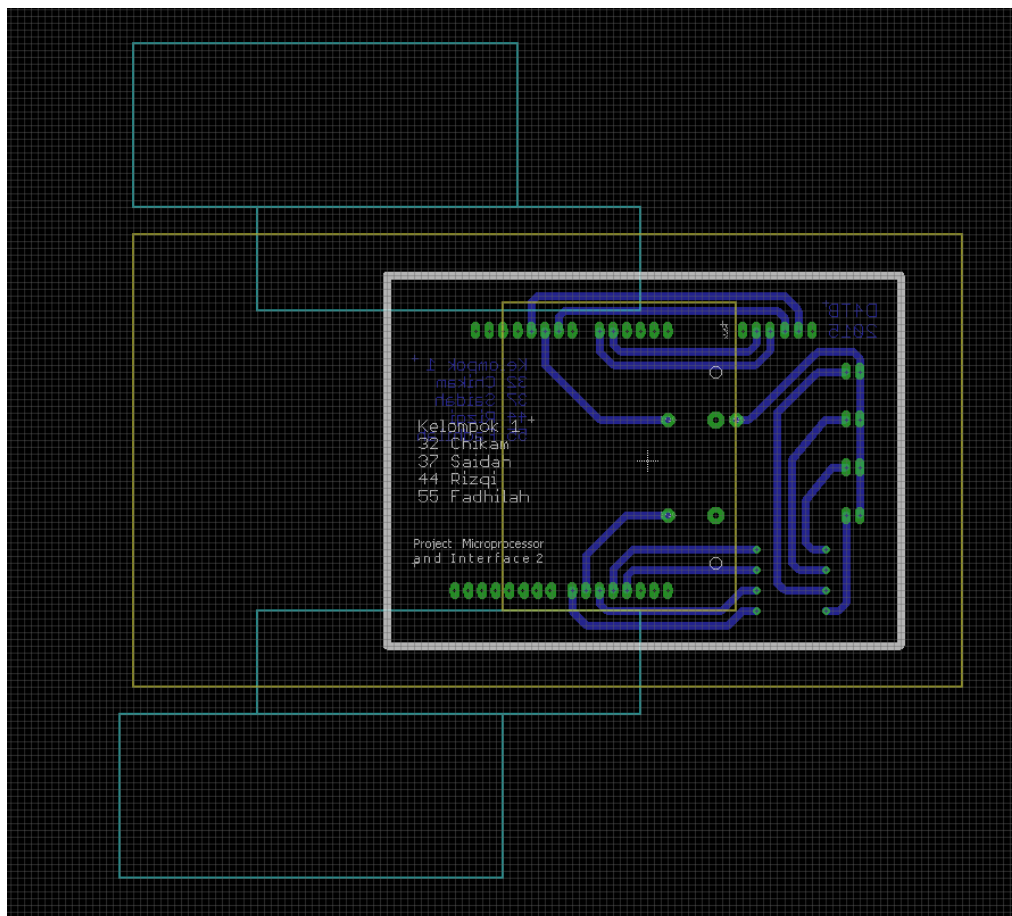
Desain Layout PCB



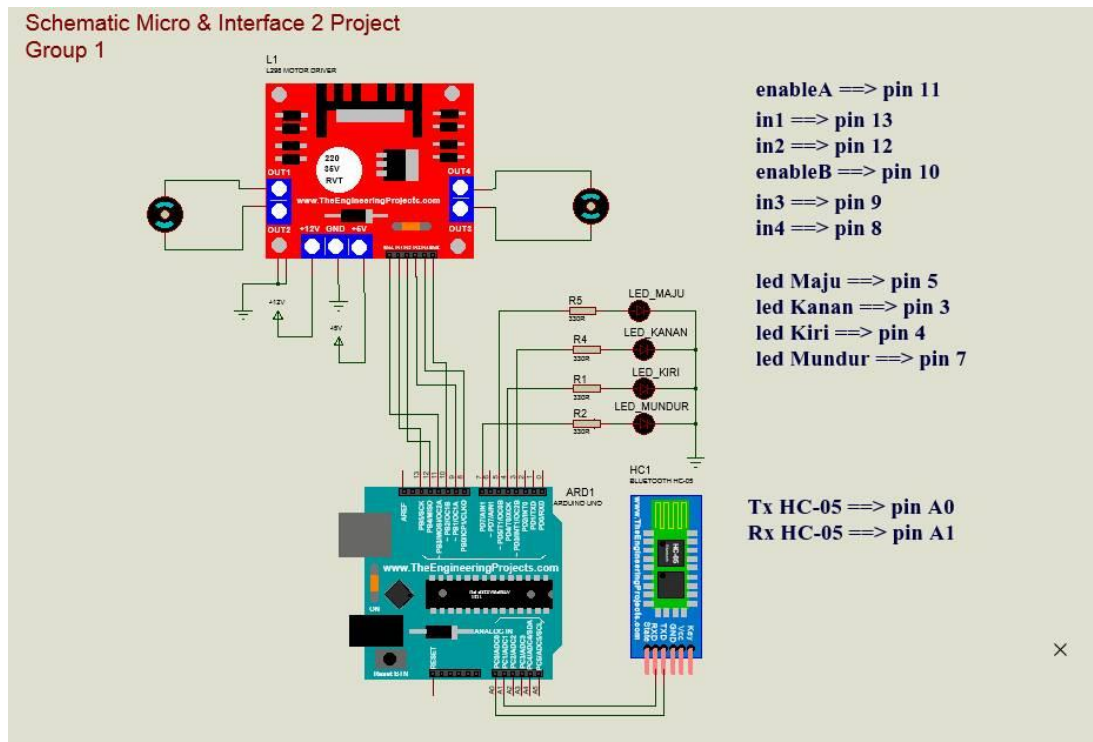
## Rancangan mekanik tampak samping



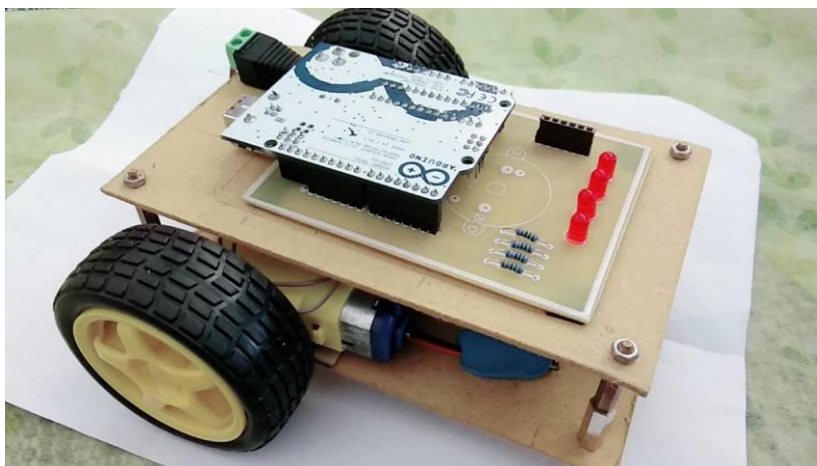
## Rancangan mekanik tampak atas

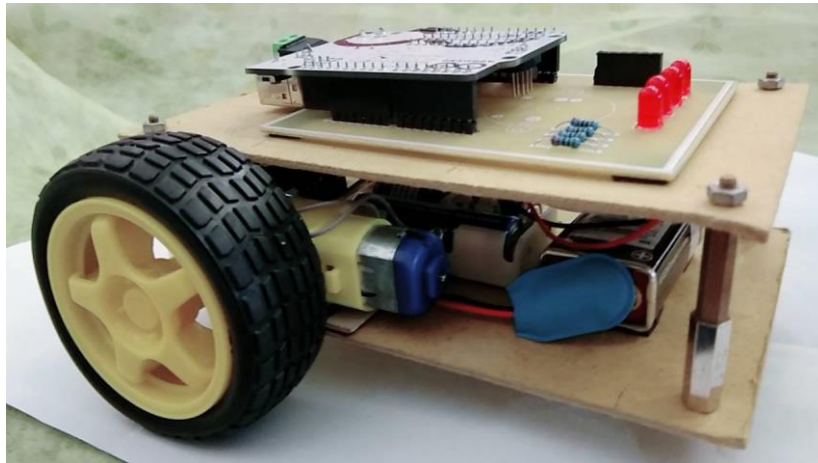


## Simulasi di Proteus



## Tampilan Hardware Tanpa Penutup





Tampilan Hardware dengan Menggunakan Penutup



#### IV. Hasil Percobaan

Setelah melakukan proses awal perancangan, perakitan dan pada proses pengujian alat ini sudah berfungsi sesuai dengan cara kerja yang diinginkan yakni Motor DC akan berputar dengan kecepatan yang sesuai dengan input yang diberikan yaitu berdasarkan penekanan tombol.

Tabel data yang didapat untuk kondisi motor DC saat bergerak :

in1	in2	in3	in4	Aksi
1	0	0	1	Bergerak mundur
0	1	1	0	Bergerak maju
0	0	0	0	Berhenti
0	1	0	0	Bergerak maju ke kanan
0	0	1	0	Bergerak maju ke kiri

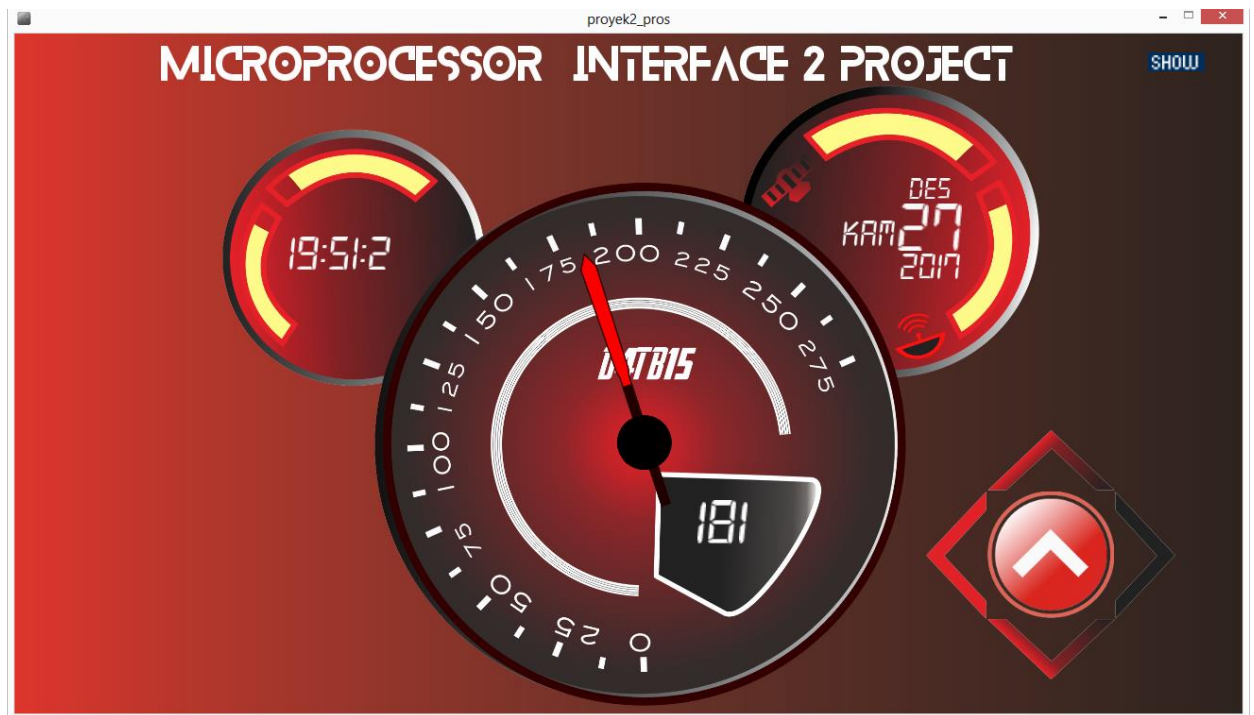
Tabel data yang didapat untuk kondisi LED ketika motor DC bergerak :

LED 1	LED 2	LED 3	LED 4	Aksi
0	0	0	0	Bergerak mundur
1	0	0	0	Bergerak maju
0	0	0	0	Berhenti
0	0	1	0	Bergerak maju ke kanan
0	1	0	0	Bergerak maju ke kiri

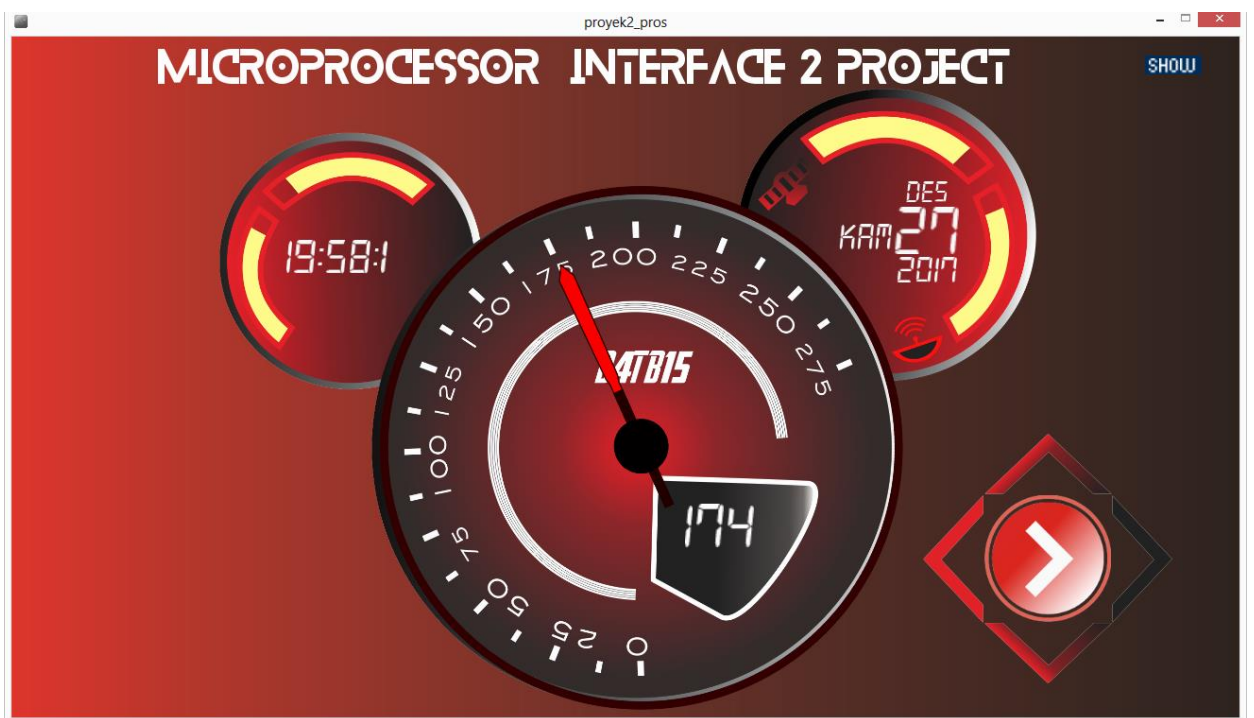


## #Tampilan Processing

Ketika motor bergerak maju dengan kecepatan 181

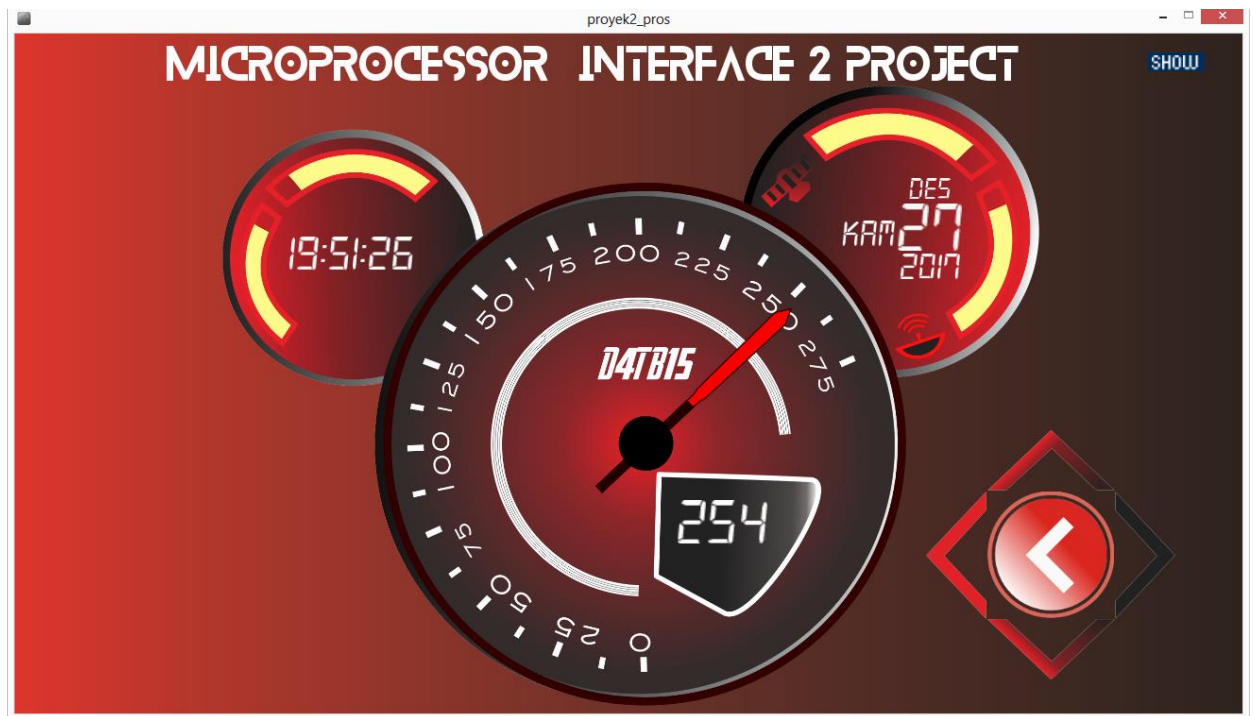


Ketika motor bergerak ke kanan dengan kecepatan 174

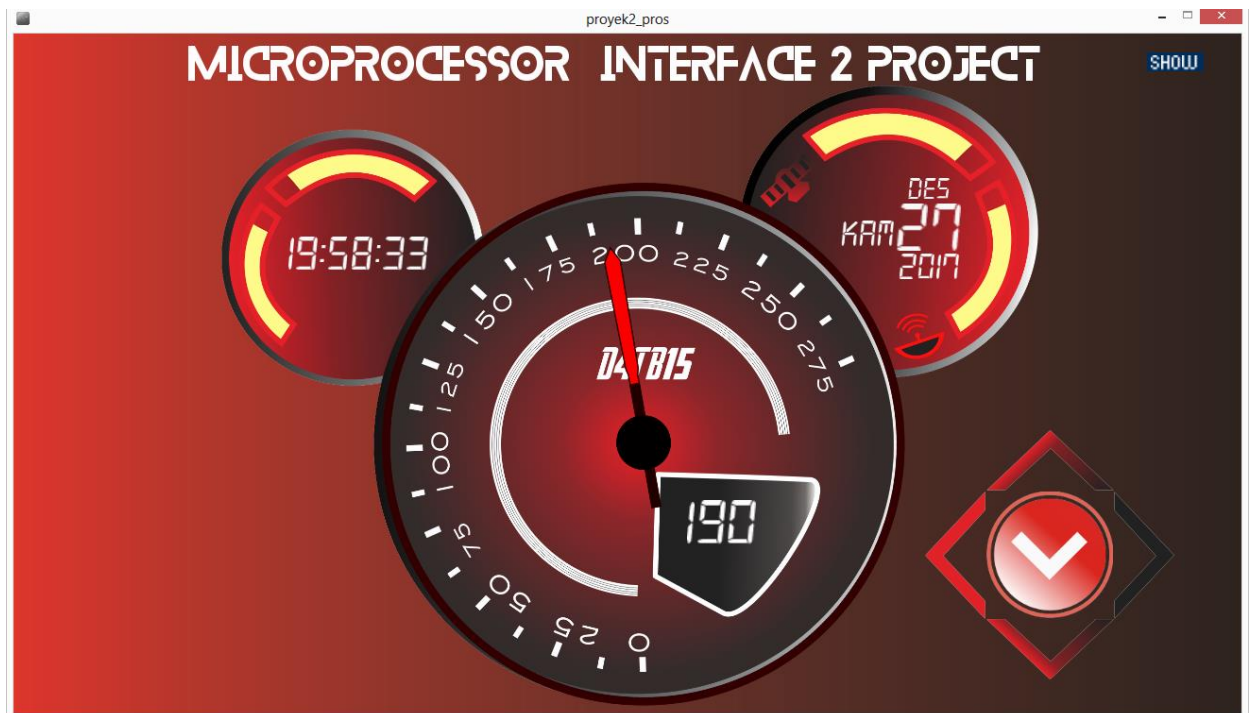




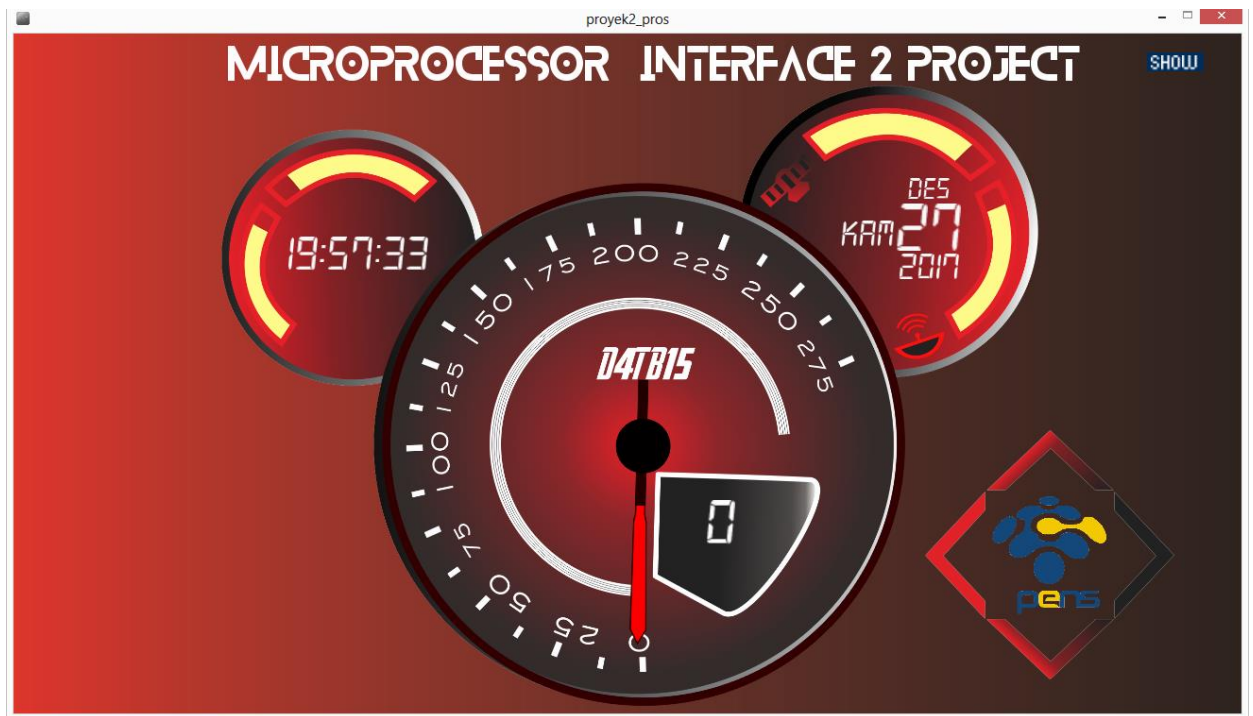
Ketika motor bergerak ke kiri dengan kecepatan 254



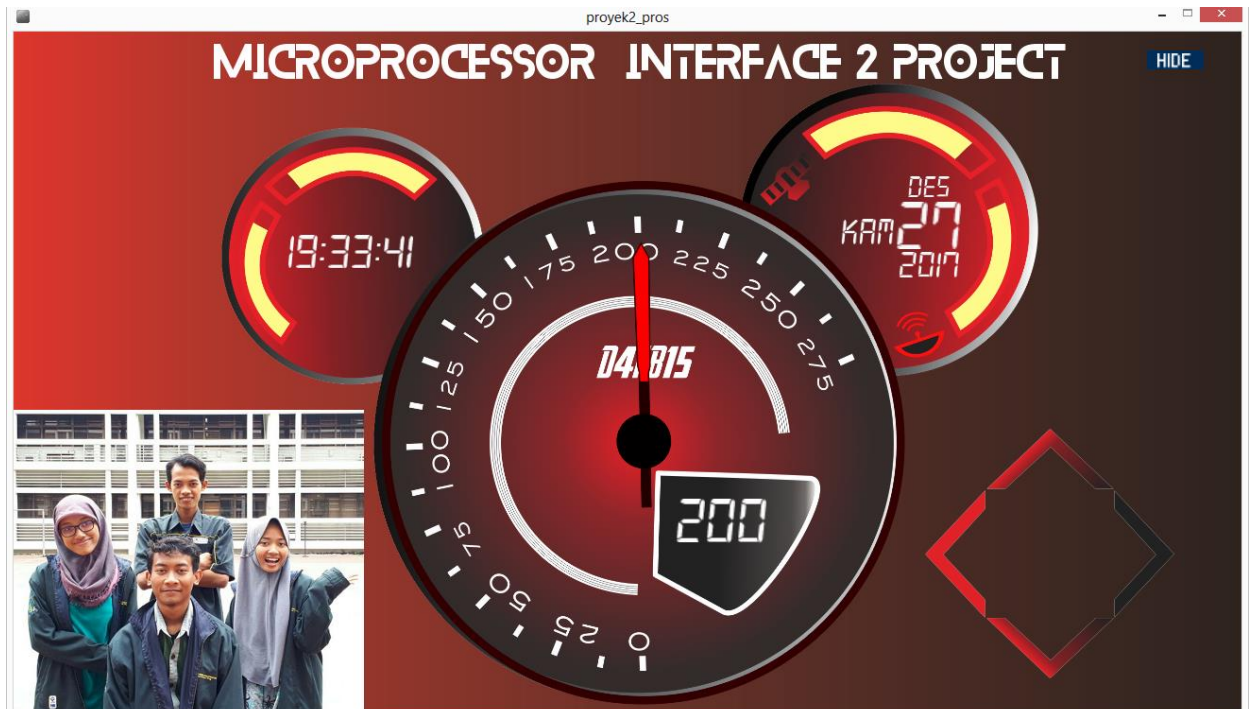
Ketika motor bergerak ke belakang / mundur dengan kecepatan 190



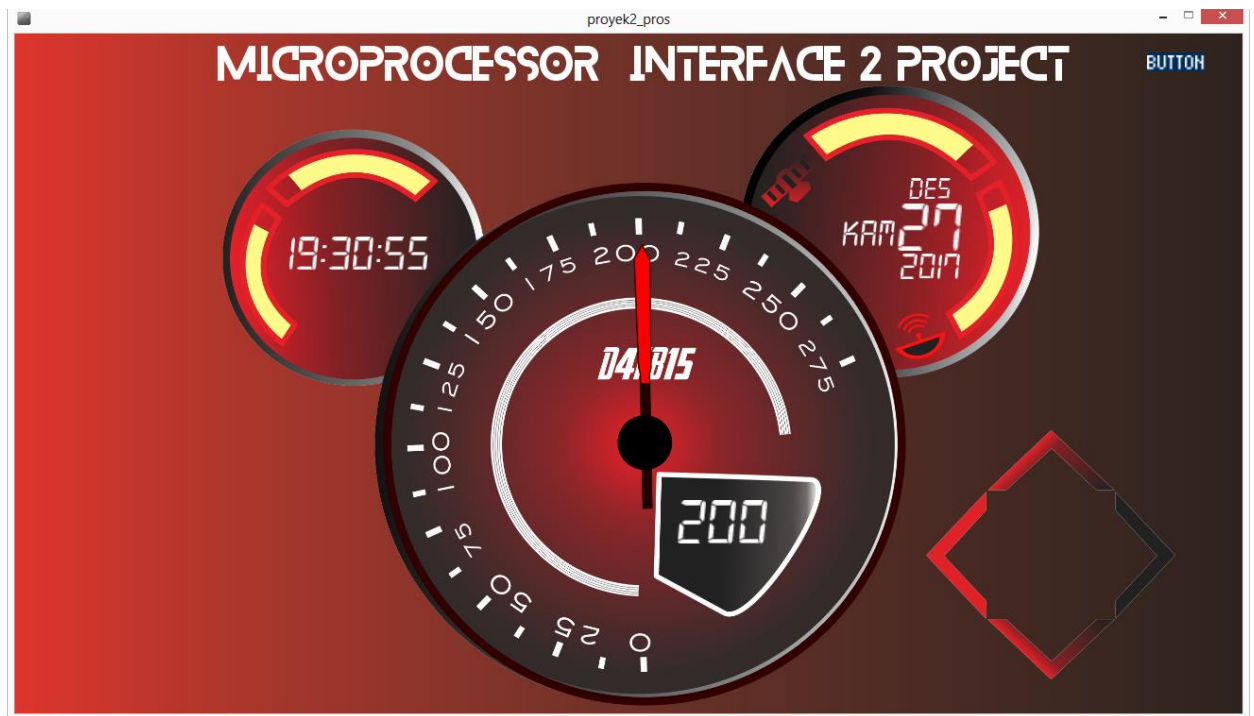
Ketika motor berhenti



Ketika tombol show ditekan / diklik, maka akan muncul



Ketika tombol hide ditekan / diklik, maka akan muncul



## V. Listing Program

### Arduino

```
#include <SoftwareSerial.h>

// motor A
int enA = 11;
int in1 = 13;
int in2 = 12;

// motor B
int enB = 10;
int in1 = 9;
int in2 = 8;

int motorSpeed = 200;

//led
int led[]={5,3,4,7};
int data = 0;
int val;

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);      // Untuk Serial Bluetooth
    // Serial.begin(9600);      // Untuk Serial Biasa
    for(int i=0; i<4; i++)
    {
        pinMode(led[i], OUTPUT);
    }

    pinMode(enA, OUTPUT);
    pinMode(enB, OUTPUT);
    pinMode(en1, OUTPUT);
    pinMode(en2, OUTPUT);
    pinMode(en3, OUTPUT);
    pinMode(en4, OUTPUT);
}
```

```
// ##### FUNGSI INDIKATOR LED #####
```

```
void ledMaju()
{
    digitalWrite(led[0], 1);
    digitalWrite(led[1], 0);
    digitalWrite(led[2], 0);
    digitalWrite(led[3], 0);
}
```

```
void ledKanan()
{
    digitalWrite(led[0], 0);
    digitalWrite(led[1], 0);
    digitalWrite(led[2], 1);
    digitalWrite(led[3], 0);
}
```

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

```
void ledMundur()
{
    digitalWrite(led[0], 0);
    digitalWrite(led[1], 0);
    digitalWrite(led[2], 0);
    digitalWrite(led[3], 1);
}
```

```
void ledMati()
{
    digitalWrite(led[0], 0);
    digitalWrite(led[1], 0);
    digitalWrite(led[2], 0);
    digitalWrite(led[3], 0);
}
```

```
// #####
// ##### FUNGSI AKTIVITAS MOTOR DC #####
```

```

void maju()
{
    // Menyalakan motor A & B(maju)
    digitalWrite(in1, LOW);
    digitalWrite(in2, HIGH);
    digitalWrite(in3, HIGH);
    digitalWrite(in4, LOW);

    // set kecepatan motor
    analogWrite(enA, motorSpeed);
    analogWrite(enB, motorSpeed);
    ledMaju();
}

```

```

void kanan()
{
    // Menyalakan motor A
    digitalWrite(in1, LOW);
    digitalWrite(in2, HIGH);

    // Mematikan motor B
    digitalWrite(in3, LOW);
    digitalWrite(in4, LOW);

    // Set kecepatan motor
    analogWrite(enB, motorSpeed);
    ledKiri();
}

```

```

void kiri()
{
    // Mematikan motor A
    digitalWrite(in1, LOW);
    digitalWrite(in2, LOW);

    // Menyalakan motor B
    digitalWrite(in3, HIGH);
    digitalWrite(in4, LOW);
}

```

```

        // Set kecepatan motor
        analogWrite(enA, motorSpeed);
        ledKanan();
    }

```

```

void mundur()
{
    // Menyalakan motor A & B (mundur)
    digitalWrite(in1, HIGH);
    digitalWrite(in2, LOW);
    digitalWrite(in3, LOW);
    digitalWrite(in4, HIGH);

    // set kecepatan motor
    analogWrite(enA, motorSpeed);
    analogWrite(enB, motorSpeed);
    ledMundur();
}

```

```

void berhenti()
{
    // Mematikan motor A & motor B
    digitalWrite(in1, LOW);
    digitalWrite(in2, LOW);
    digitalWrite(in3, LOW);
    digitalWrite(in4, LOW);
    ledMati();
}

```

```

void loop()
{
    // kita harus memperhatikan dari mana data yang dikirimkan
    // menggunakan "Serial" jika dari Arduino biasa
    // menggunakan "mySerial" jika dari bluetooth
    if (mySerial.available() > 0)
    {
        int data = mySerial.read();    // Membaca data yang dikirim dari processing
        if (data == 'W')                // MAJU

```

```

    {
        motorSpeed += 1;
        maju();
        val = 1;
    }

    if (data == 'D')                                // KANAN
    {
        motorSpeed += 1;
        kanan();
        val = 2;
    }

    if (data == 'A')                                // KIRI
    {
        motorSpeed += 1;
        kiri();
        val = 3;
    }

    if (data == 'S')                                // MUNDUR
    {
        motorSpeed += 1;
        mundur();
        val = 4;
    }

    if (data == 'X')                                //STOP
    {
        motorSpeed = 0;
        berhenti();
    }

    // ##### UNTUK MENGURANGI KECEPATAN MOTOR
    #####
    if (data == 'Z')                                //Mengurangi Nilai Kecepatan
    {
        if(val == 1)
        {
            motorSpeed -= 1;
            maju();
        }
    }

```



```

        if(val == 2)
        {
            motorSpeed -= 1;
            kanan();
        }

        if(val == 3)
        {
            motorSpeed -= 1;
            kiri();
        }
        if(val == 4)
        {
            motorSpeed -= 1;
            mundur();
        }
    }

    if (data == 'Q')
    {
        motorSpeed -= 1;
    }
}

```

## **Processing**

```
import processing.serial.*; //Import Serial library
import controlP5.*;
Serial myPort;
ControlP5 cp5;
controlP5.Button b;
bouncyWordJudul judul_;
PFont fontJudul;

// INISIALISASI UNTUK SPEEDOMETER
PImage jarum, background, speedo, judul;
PImage UP_, RIGHT_, LEFT_, DOWN_, setUp, setRIGHT, setLEFT, setDOWN, pens;
int rec_x=355, rec_y=640, rec_height=10, z1=200;

// INISIALISASI LAIN
int w, indikator;
int buttonValue = 1;
float angle;
boolean isOpen;

// INISIALISASI UNTUK WAKTU
String[] dayName =
{
    "SEN", "SEL", "RAB", "KAM", "JUM", "SAB", "MIN"
};
String[] monthName =
{
    "Jan", "Feb", "Mar", "Apr", "Mei", "Jun",
    "Jul", "Agu", "Sep", "Okt", "Nov", "Des"
};

int hari = day();
int bulan = month();
int tahun = year();

void setup()
{
    myPort = new Serial(this, "COM2", 9600);
    size(1300, 720);
    stroke(255);
    smooth();

    //***** MEMBUKA SEMUA DATA FOTO DARI FILE *****/
    background = loadImage("background.png");
    jarum = loadImage("JARUM.png");
    speedo = loadImage("speedo.png");
    UP_ = loadImage("_atas.png");
```

```

RIGHT_ = loadImage("_kanan.png");
LEFT_ = loadImage("_kiri.png");
DOWN_ = loadImage("_bawah.png");
setUP = loadImage("up.png");
setRIGHT = loadImage("right.png");
setLEFT = loadImage("left.png");
setDOWN = loadImage("down.png");
pens = loadImage("pens.png");

// BUTTON UNTUK OPEN/HIDE PHOTO
cp5 = new ControlP5(this);
cp5.addButton("button")
    .setPosition(width-100, 20)
    .setSize(60, 20)
    ;

b = cp5.addButton("buttonValue")
    .setPosition(0, 420)
    .setImages(loadImage("foto.png"), loadImage("foto.png"),
loadImage("foto.png"))
    ;

cp5.getController("button")
    .getCaptionLabel()
    .setSize(20)
    ;

b.getCaptionLabel()
    .toUpperCase(false)
    ;

// LOAD FONT UNTUK JUDUL
fontJudul = loadFont("Blanka-Regular-46.vlw");
textFont(fontJudul);
judul_ = new bouncyWordJudul("", -3, 300, 50);
}

void draw()
{
    imageMode(CORNER);
    image(background, 0, 0, width, height);
    judul_.draw();
    image(loadImage("up_.png"), 1030, 420, 135, 64);
    image(loadImage("right_.png"), 1165, 485, 64, 135);
    image(loadImage("left_.png"), 965, 485, 64, 135);
    image(loadImage("down_.png"), 1030, 620, 135, 64);
    image(speedo, 220, 55, 865, 655);
}

```

```

// CLOCK FUNCTION
int s = second();
int m = minute();
int h = hour();
fill(250, 250, 250);
textFont(loadFont("Digital-7Italic-48.vlw"));
textSize(52);
textAlign(LEFT);
text(h + ":" + m + ":" + s, 290, 250);
textAlign(CENTER);
textSize(40);
text(dayName[4], 905, 225);           // Pengaturan Hari Manual
textSize(32);
text(monthName[bulan-1], 970, 175);  // Cetak Bulan
textSize(40);
text(tahun, 970, 260);               // Cetak Tahun
textSize(74);
text(hari, 970, 230);                // Cetak Hari

// TO SHOW/HIDE PHOTO
float x = b.x(b.getPosition());
x += ((isOpen==true ? 0:-380) - x) * 0.5;
;
float y = b.y(b.getPosition());
b.setPosition(x, y);

// INDIKATOR ARAH MOTOR
if (indikator==1)
{
    ellipse(1100, 550,100,100);
    image(UP_, 1030, 485, 134, 134);
}

if (indikator==2)
{
    ellipse(1100, 550,100,100);
    image(RIGHT_, 1030, 485, 134, 134);
}

if (indikator==3)
{
    ellipse(1100, 550,100,100);
    image(LEFT_, 1030, 485, 134, 134);
}

if (indikator==4)
{
    ellipse(1100, 550,100,100);
    image(DOWN_, 1030, 485, 134, 134);
}

```

```

    }

    if (indikator==5)
    {
        ellipse(1100, 550,100,100);
        image(pens, 1030, 485, 134, 134);
    }

    speedo();
}

//***** KONTROLLER ARAH DAN KECEPATAN MOTOR
*****//
void keyPressed()
{
    imageMode(CORNER);
    if (key == CODED)
    {
        if (keyCode == UP)
        {
            image(setUP, 1030, 420, 135, 64);
            println("MAJU");
            indikator=1;
            if (z1>=0 && z1<=254)
            {
                z1++;
                myPort.write('W');
            } else if (z1==255) {
                z1 = 254;
                myPort.write('Q');
            }
        }

        if (keyCode == RIGHT)
        {
            image(setRIGHT, 1165, 485, 64, 135);
            println("KANAN");
            indikator=2;
            if (z1>=0 && z1<=254)
            {
                z1++;
                myPort.write('D');
            } else if (z1==255) {
                z1 = 254;
                myPort.write('Q');
            }
        }

        if (keyCode == LEFT)
        {

```

```

        image(setLEFT, 965, 485, 64, 135);
        println("KIRI");
        indikator=3;
        if (z1>=0 && z1<=254)
        {
            z1++;
            myPort.write('A');
        } else if (z1==255) {
            z1 = 254;
            myPort.write('Q');
        }
    }

    if (keyCode == DOWN)
    {
        image(setDOWN, 1030, 620, 135, 64);
        println("MUNDUR");
        indikator=4;
        if (z1>=0 && z1<=254)
        {
            z1++;
            myPort.write('S');
        } else if (z1==255) {
            z1 = 254;
            myPort.write('Q');
        }
    }
}

if (key == 'z')
{
    println("REM.....");
    if (z1>0 && z1<=256)
    {
        z1--;
        myPort.write('Z');
    }

    if (0==z1)
    {
        z1 = 0;
    }
}

if (key == 'x')
{
    println("STOP");
    indikator=5;
    z1 = 0;
    myPort.write('X');
}

```

```

    }
}

//***** MENGATUR PERPUTARAN JARUM SPEEDO
*****//
void speedo()
{
    imageMode(CENTER);
    textAlign(CENTER);
    pushMatrix();
    translate(667, 435);
    angle = map(z1, 0, 255, 39.5, 79);
    rotate(angle*0.1);
    tint(255, 0, 0);
    image(jarum, 0, 0, 555, 555);
    noTint();
    popMatrix();
    textSize(68);
    fill(255, 255, 255);
    text(z1, 750, 540);
}

// Show Hide Photo
public void button(float theValue)
{
    isOpen = !isOpen;
    cp5.getController("button").setCaptionLabel((isOpen==true) ? "hide":"show");
}

// d = Hari dalam sebulan (Senin ..... Minggu)
// m = Bulan (January = 1 : December = 12)
// y = 4 digit Tahun
// Returns 0 = Minggu .. 6 = Sabtu

public int date(int d, int m, int y)
{
    if (m < 3)
    {
        m += 12;
        y--;
    }
    return (d + int((m+1)*2.6) + y + int(y/4) + 6*int(y/100) + int(y/400) + 6) % 7;
}

// RUNNING TEXT JUDUL
class bouncyWordJudul
{
    float px, py, vx, vy, ypos;
    bouncyWordJudul(String theWord, float ivx, float ipx, float ipy)
    {

```

```

        py=ipy;
        vy=0;
        vx=ivx;
        px=ipx;
        frameRate(300);
        smooth();
    }

    void draw()
    {
        px+=vx;
        py+=vy;
        if (px<570)
        {
            px=570;
            vx=-vx;
        }

        if (px>700)
        {
            px=700;
            vx=-vx;
        }

        textFont(fontJudul);
        text("Microprocessor & Interface 2 Project", px, py);
    }
}

```



## **VI. Analisa**

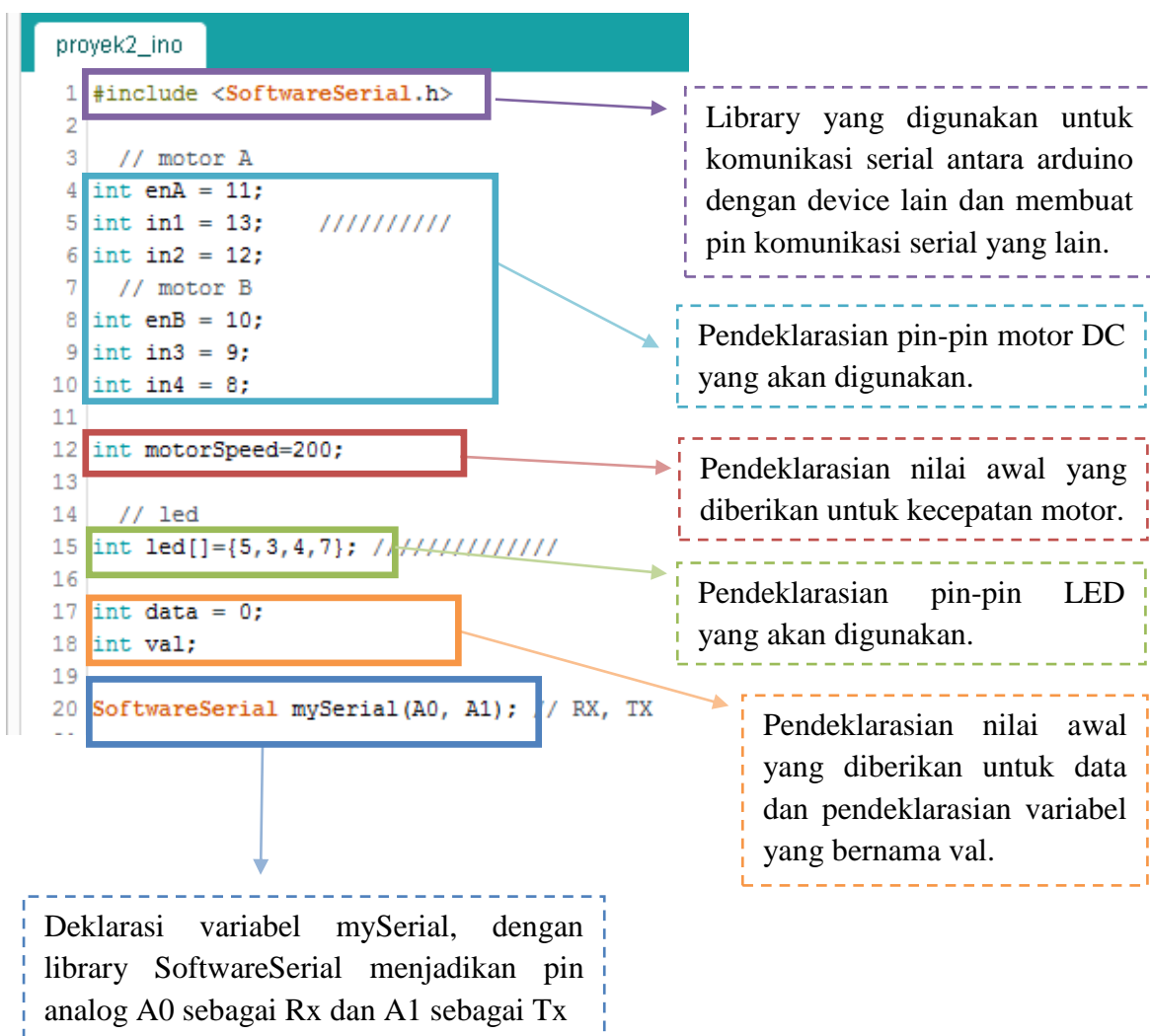
Pada tugas project yang telah kami buat ini, kami membuat sebuah rangkaian integrasi untuk mengatur kecepatan motor DC dengan menggunakan Smartphone dan memvisualisasikan kecepatan motor tersebut menggunakan processing. Komponen yang kami gunakan tersebut terdiri dari Modul Arduino, Motor DC, IC L293N, Modul Bluetooth, Buzzer, LED dan Gearbox plus Roda.

Untuk hardwarenya, modul bluetooth yang kami gunakan tersebut yaitu HC-05 berfungsi sebagai pengganti komunikasi serial sehingga kita tidak perlu menggunakan kabel untuk me-run program Arduino yang kita buat. Oleh karena itu, kita harus menyandingkan bluetooth HC-05 dengan bluetooth yang ada di komputer atau laptop yang sedang kita gunakan lebih dulu. Kemudian untuk penggerak agar Motor DC yang kami gunakan tersebut dapat berjalan, kami gunakan sebuah driver atau IC L298N. Pada prinsipnya project yang kami buat saat ini hampir sama seperti project tahun lalu mengenai control Motor DC. Hanya saja pada project kali ini, kita mengganti drivernya agar motor DC dapat bergerak lebih cepat dibandingkan sebelumnya. Driver motor L298N yang kami gunakan ini pada prinsipnya sama seperti L293D. Cara pengontrolannya pun sama. Perbedaannya mendasar hanya terletak pada karakteristik elektroniknya, yaitu kemampuan L298N dalam melewatkan arus untuk motor DC lebih besar yaitu sebesar 3A. Hanya saja pada penggunaan IC ini sebagai driver motor, tidak semudah menggunakan IC L293D. Agar mampu bekerja dengan baik maka diperlukan beberapa komponen dioda pendukung. Akan tetapi kami menggunakan IC L298N yang sudah berbentuk modul sehingga penggunaannya lebih praktis. Selain itu modul L298N tersebut juga sudah dilengkapi dengan regulator DC sebesar 5 Volt yang mampu mensuplai arus sebesar kurang lebih 1 Ampere.

Sedangkan untuk hardware lainnya yaitu LED hanya digunakan sebagai indikator pergerakan motor. Fungsi resistor yang kami pasang pada rangkaian dimana melibatkan LED tersebut adalah sebagai pembagi tegangan dan pembatas arus. Lalu untuk gearbox dan rodanya berfungsi menyesuaikan daya atau torsi dari motor yang berputar sehingga motor akan berputar menjadi tenaga yang lebih besar.

Pada project kami kali ini, kami tidak hanya berfokus pada program Arduino seperti sebelumnya. Kami juga menggunakan software processing agar membuat tampilan atau bentuk visualisasi pengaturan kecepatan Motor DC menjadi lebih menarik dan tidak membosankan lagi.

Kami membuat program Arduino nya terlebih dahulu untuk mengecek perputaran Motor DC, baru setelah itu kami buat program processingnya sebagai bentuk visualisasi dari perputaran Motor DC tersebut. Pada program arduinonya, kami buat sedemikian rupa programnya agar motor DC yang kami rangkai dapat berputar sesuai dengan yang kami inginkan. Pertama kita harus deklarasikan terlebih dahulu variable-variabel yang akan kita gunakan sebelum masuk ke fungsi loop() dan fungsi setup() di Arduino, agar nanti kita hanya perlu memanggil variabelnya, tanpa harus menuliskan lagi panjang lebar tentang berapa besarnya nilai variabel yang kita gunakan atau pada pin ke berapa variabel tersebut terpasang.



Sebelumnya, kita harus mengimport atau menambahkan library SoftwareSerial untuk komunikasi serial antara arduino dengan device lain. Agar library SoftwareSerial tersebut dapat digunakan, maka kita perlu mencakup file header SoftwareSerial.h terlebih

dahulu. Dengan adanya library `SoftwareSerial`, kita bisa membuat pin komunikasi serial yang lain tidak hanya satu melainkan bisa banyak port. Kami gunakan pin analog A0 sebagai RX dan pin analog A1 sebagai TX. Penggunaan pin analog memiliki fitur untuk dapat mengubah sinyal analog yang masuk menjadi nilai digital yang mudah diukur. Pin digital hanya dapat mengenali sinyal 0 volt sebagai nilai LOW dan 5 volt sebagai nilai HIGH. Sedangkan Pin analog dapat mengenali sinyal pada rentang nilai voltase tersebut. Hal ini sangat berguna misalkan ketika kita hendak mengukur sesuatu dari sensor dan menggunakan nilai masukan tersebut untuk keperluan lain. Nama variabel `mySerial` yang kita gunakan tersebut dapat diganti dengan nama lain, hanya perlu diingat bahwa nama variabel tersebut harus sama setiap kali memanggil function terkait library `SoftwareSerial`.

Sebelum masuk pada program utama, kita harus melakukan inisialisasi komunikasi serial baik secara hardware (`Serial.begin()`) maupun software (`mySerial.begin()`) dengan deklarasi serta baud rate yang digunakan yakni 9600. Penggunaan baud rate tersebut digunakan untuk mengindikasikan seberapa cepat data dikirim melalui komunikasi serial. Semakin besar nilai baud rate yang digunakan maka semakin tinggi kecepatan transfer. Nilai baud rate dapat diatur dengan menggunakan standar kecepatan yang disediakan, diantaranya 1.200, 2.400, 4.800, 9600, 19.200, 38.400, 57.600, dan 115.200 bps. Salah satu kecepatan yang paling umum digunakan adalah 9.600 bps. Baud Rate untuk komunikasi serial yang telah kita setting sebesar 9600 tersebut harus disesuaikan dengan baud rate bluetooth HC-05 yang dipakai. Hal ini disebabkan agar bisa saling berkomunikasi antara Arduino dengan Bluetooth HC-05. Nilai tersebut merupakan nilai yang mana kecepatan komunikasi bukanlah suatu hal yang kritis untuk dipertimbangkan. Karena komunikasi yang melibatkan sinyal elektrik dan proses sinkronisasi data sangat rentan dengan error dan derau, maka disarankan untuk tidak melebihi kecepatan 115.200 bps untuk komunikasi pada Arduino.

```

22 void setup() {
23   // Open serial communications and wait for port to open
24   // set the data rate for the SoftwareSerial port
25   // mySerial.begin(9600);           // Untuk Serial Bluetooth
26   Serial.begin(9600);               // Untuk Serial Biasa
27   for(int i=0; i<4; i++){
28     pinMode(led[i], OUTPUT);
29   }
30   pinMode(enA, OUTPUT);
31   pinMode(enB, OUTPUT);
32   pinMode(in1, OUTPUT);
33   pinMode(in2, OUTPUT);
34   pinMode(in3, OUTPUT);
35   pinMode(in4, OUTPUT);
36 }

```

Deklarasi pin-pin LED sebagai output.

Deklarasi pin-pin motor sebagai output.

Syntax `pinMode()` yang kita gunakan tersebut adalah untuk mengkonfigurasi pin tertentu agar berperilaku sebagai input atau output. LED dan motor DC yang kami gunakan tersebut digunakan sebagai output, sedangkan inputnya sendiri berasal dari keyboard laptop kita masing-masing.

Fungsi yang digunakan pada Arduino tidak hanya dua fungsi yaitu fungsi `setup()` dan fungsi `loop()`. Melainkan kita bisa membuat fungsi-fungsi lainnya seperti fungsi untuk indikator nyala LED, dan fungsi kontrol aktivitas motor. Pembuatan fungsi-fungsi tersebut bertujuan agar penulisan syntax program di fungsi `loop()` tidak terlalu banyak. Sehingga kita hanya perlu memanggil nama fungsinya, maka otomatis program-program yang ada di dalam fungsi tersebut akan diproses.

Pada fungsi indikator LED didalamnya terdiri dari 5 buah fungsi yaitu fungsi `ledMaju()`, fungsi `ledKanan()`, fungsi `ledKiri()`, fungsi `ledMundur()`, dan fungsi `ledMati()`. Masing-masing kelima fungsi tersebut digunakan sebagai indikator LED untuk mengetahui tanda apa yang ditunjukkan pada kondisi atau aktivitas motor saat ini. Hal tersebut dapat dilihat dari posisi led mana saja yang menyala.

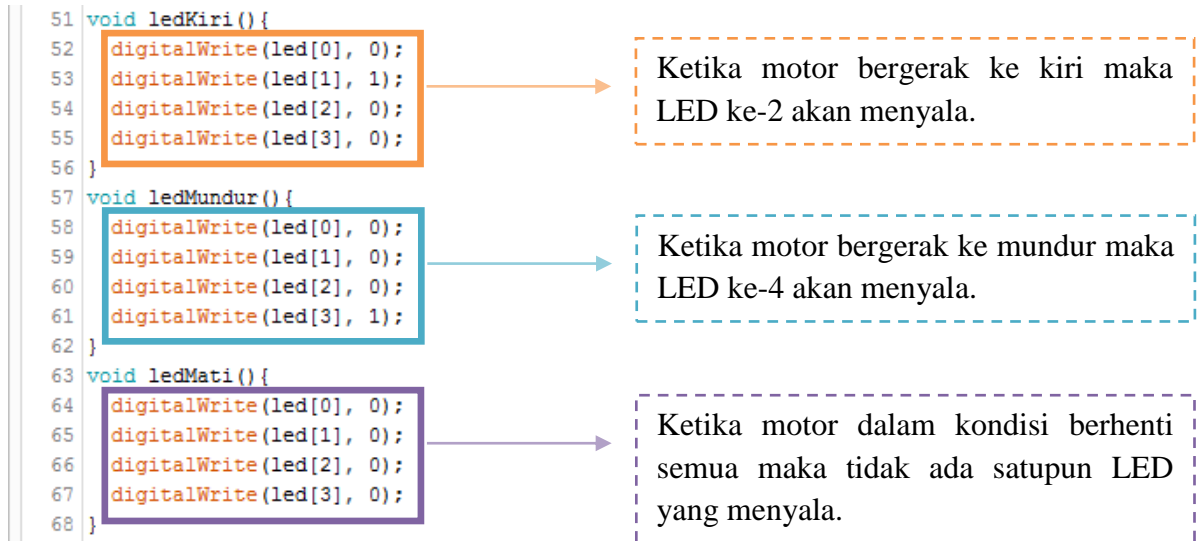
```

38 // ##### FUNGSI INDIKATOR LED #####
39 void ledMaju(){
40   digitalWrite(led[0], 1);
41   digitalWrite(led[1], 0);
42   digitalWrite(led[2], 0);
43   digitalWrite(led[3], 0);
44 }
45 void ledKanan(){
46   digitalWrite(led[0], 0);
47   digitalWrite(led[1], 0);
48   digitalWrite(led[2], 1);
49   digitalWrite(led[3], 0);
50 }

```

Ketika motor bergerak maju maka LED ke-1 akan menyala.

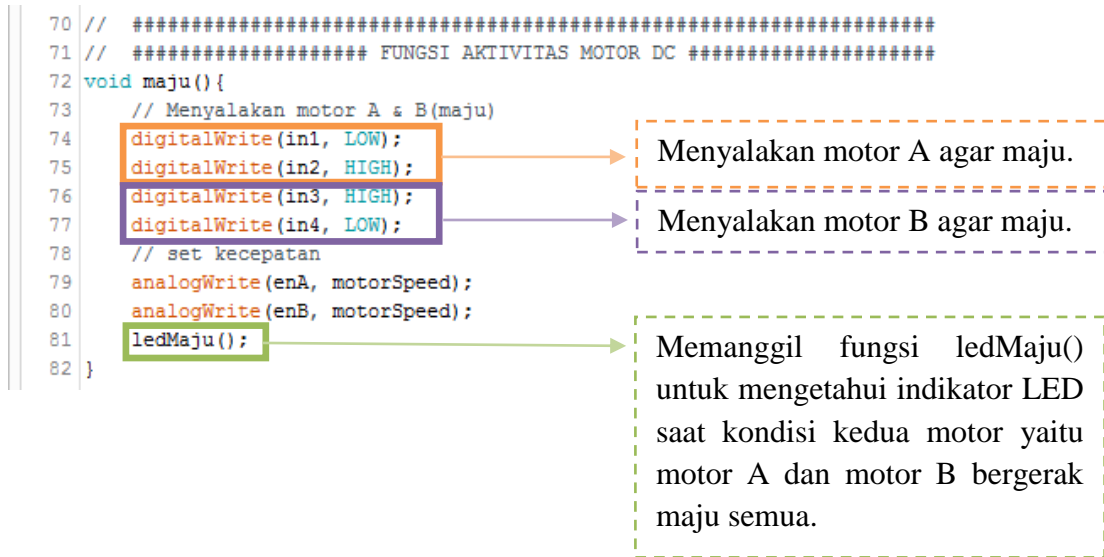
Ketika motor bergerak ke kanan maka LED ke-3 akan menyala.



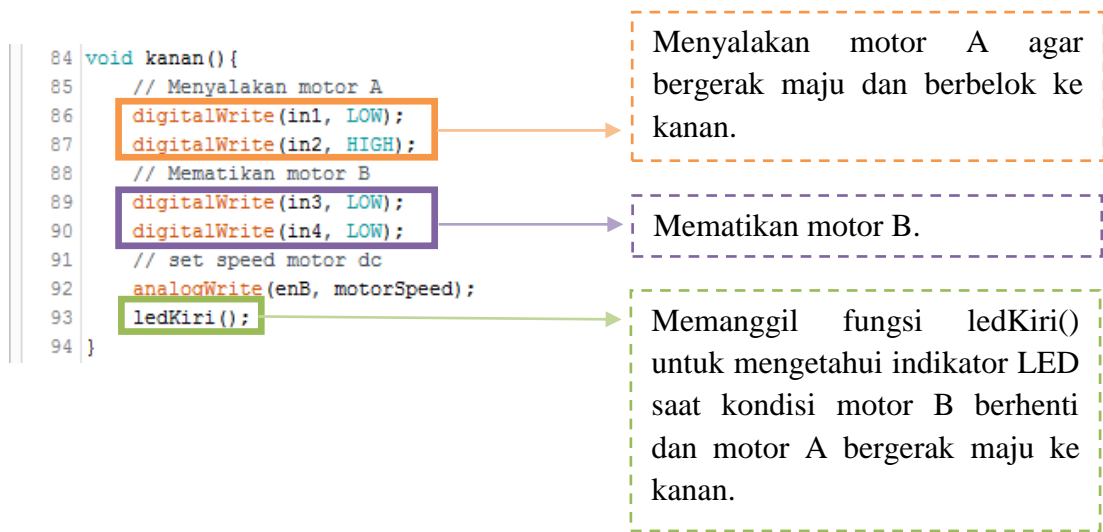
Penulisan LED yang kita gunakan tersebut menggunakan matriks array sehingga dapat mempermudah kita dalam pengaturan data maupun penelusurannya (peng-index-an). Di dalam array, penulisan index nya selalu diawali dengan angka 0. Karena kami menggunakan 4 buah LED maka penulisan matriks array untuk LED yaitu led[0] yang berarti LED ke-1, led[1] yang berarti LED ke-2, led[2] yang berarti LED ke-3, dan led[3] yang berarti LED ke-4.

Dari LED ke-1 sampai ke-4 tersebut pada masing-masing fungsi menggunakan digitalWrite(). Untuk mengecek atau mengetes apakah program dapat bekerja dengan baik atau tidak, kita dapat memasang LED tambahan dan di programnya dapat menggunakan syntax analogWrite(). Fungsi digitalWrite() adalah fungsi yang digunakan untuk menuliskan nilai secara digital pada suatu pin. Nilai yang diberikan pada digitalWrite() bisa berupa angka “1” untuk HIGH atau angka “0” untuk LOW. Sedangkan pada analogWrite(), nilai yang terbaca seringkali berupa analog (memiliki banyak nilai, misal dari 0 – 1 terdapat nilai 1.01, 1.02, dst.), bukan digital lagi (memiliki 2 nilai yaitu 0 dan 1). LED akan menyala bergantung pada seberapa besar nilai yang tersimpan pada motorSpeed. LED tambahan tersebut mengatur intensitas atau seberapa terang tersebut menyala.

Setelah itu, fungsi selanjutnya adalah fungsi untuk aktivitas motor DC. Pada fungsi untuk aktivitas motor DC tersebut didalamnya terdiri dari 5 buah fungsi yaitu fungsi maju(), fungsi kanan(), fungsi kiri(), fungsi mundur(), dan fungsi berhenti().

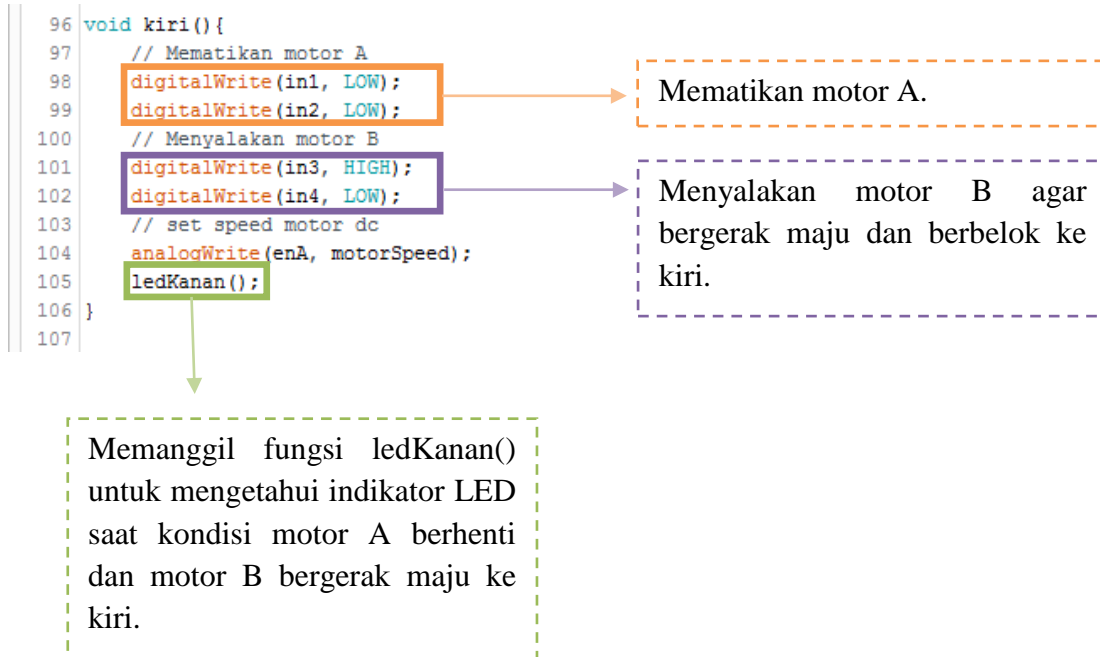


Fungsi `maju()` pada fungsi aktivitas motor diatas tersebut digunakan untuk menyalakan motor A dan motor B agar kedua motor tersebut bisa bergerak maju. Di setiap motor yang kita gunakan pada rangkaian baik itu motor A maupun motor B mempunyai 1 buah Pin EN dan 2 buah Pin In dimana pin EN atau pin enable berfungsi mengijinkan driver menerima perintah untuk menggerakan motor DC, sedangkan pin IN adalah pin input sinyal kendali motor DC. Apabila kita menginginkan kedua motor dapat bergerak maju, maka kita harus melihat terlebih dahulu tabel konfigurasi dari driver motor L298N. Pada tabel konfigurasi tersebut, dapat diketahui bahwa motor akan bergerak maju apabila pada motor A pin in1 diberi logika “0” dan pin in2 diberi logika “1”, sedangkan pada motor B pin in3 diberi logika “1”, dan pin in4 diberi logika “0”. Setelah kita beri nilai logika pada pin IN di masing-masing motor, kita atur atau setting kecepatan masing-masing motor dengan menggunakan `analogWrite()`. Nilai kecepatan masing-masing motor telah ditetapkan sebelumnya dengan mengambil nilai dari variabel `motorSpeed` yaitu sebesar 200. Nilai variabel `motorSpeed` sebesar 200 tersebut merupakan nilai default yang diberikan. Untuk mengetahui kondisi pergerakan motor, kita dapat menggunakan indikator LED dengan memanggil fungsi `ledMaju()`. Pemanggilan fungsi tersebut sebagai indikator LED tergantung dari kondisi apa yang diminta. Fungsi `ledMaju()` tersebut yang kita tuliskan sebelumnya di sebelum fungsi aktivitas motor DC, tidak akan terproses selama fungsi `ledMaju()` tersebut belum terpanggil meskipun penulisan syntaxnya diletakkan di awal.



Fungsi `kanan()` pada fungsi aktivitas motor diatas tersebut merupakan fungsi ketika motor berbelok ke kanan. Ketika motor berjalan maju dan berbelok ke kanan, maka motor B akan berhenti sedangkan motor A akan tetap bergerak. Dalam hal ini motor A berada di sisi sebelah kiri mobil sedangkan motor B berada di sisi sebelah kanan mobil. Karena motor B berhenti maka pin `in3` maupun pin `in4` diberi logika “0” semua. Untuk motor A, pin `in1` dan pin `in2` diberi logika masih sama seperti sebelumnya ketika kondisi maju yaitu pin `in1` diberi logika “0”, dan pin `in4` diberi logika “1”.

Setelah itu untuk setting kecepatan motor, kita hanya perlu mensetting kecepatan motor A yang sedang bergerak dengan menggunakan `analogWrite()`. Nilai kecepatan motor A tersebut masih sama seperti sebelumnya yang diambil dari variabel `motorSpeed` yaitu sebesar 200. Untuk mengetahui kondisi pergerakan motor A tersebut, kita dapat menggunakan indikator LED dengan memanggil fungsi `ledKiri()`. Alasan kami menggunakan dan memanggil fungsi `ledKiri()` tersebut sebagai indikator saat motor berbelok ke kanan adalah karena yang bergerak adalah motor yang berada di sisi kiri mobil bukan sisi kanannya. Sehingga kita tidak menggunakan fungsi `ledKanan()` sebagai indikator pergerakan motor meskipun motor berbelok ke kanan.

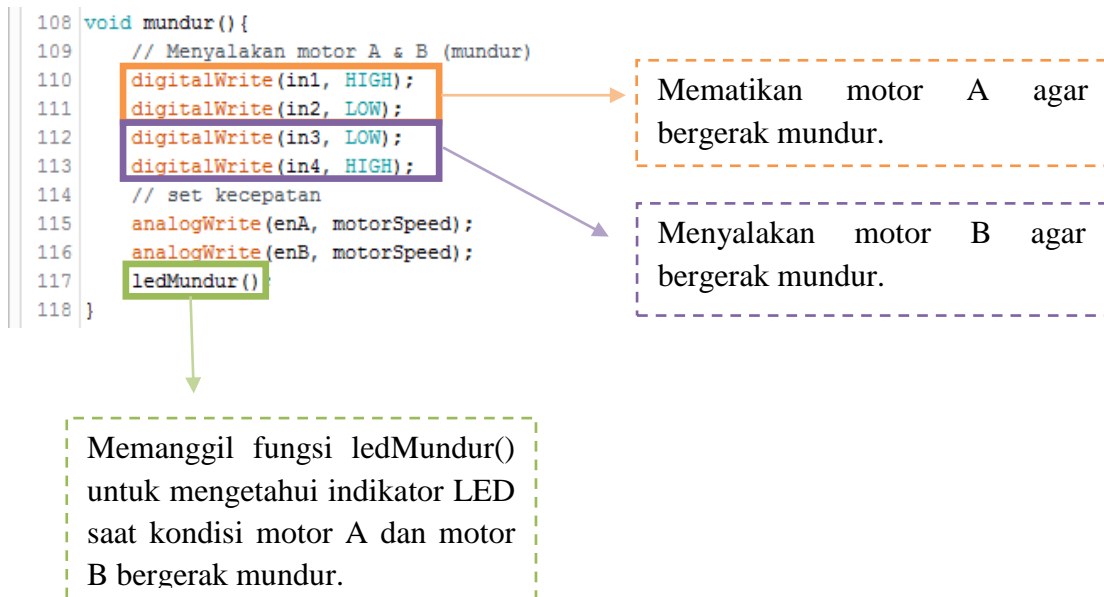


Fungsi `kiri()` pada fungsi aktivitas motor diatas tersebut merupakan fungsi ketika motor berbelok ke kiri. Hal tersebut merupakan kebalikan dari fungsi sebelumnya yaitu saat motor berbelok ke kanan. Ketika motor berjalan maju dan berbelok ke kiri, maka motor A akan berhenti sedangkan motor B akan tetap bergerak. Dalam hal ini motor A berada di sisi sebelah kiri mobil sedangkan motor B berada di sisi sebelah kanan mobil. Karena motor A berhenti maka pin `in3` maupun pin `in4` diberi logika “0” semua. Untuk motor B, pin `in1` dan pin `in2` diberi logika masih sama seperti sebelumnya ketika kondisi maju yaitu pin `in1` diberi logika “0”, dan pin `in4` diberi logika “1”.

Setelah itu untuk setting kecepatan motor, kita hanya perlu mensetting kecepatan motor B yang sedang bergerak dengan menggunakan `analogWrite()`. Untuk motor A kita tidak perlu mensetting kecepatannya karena motor A sudah dalam kondisi diam atau berhenti. Nilai kecepatan motor B tersebut masih sama seperti sebelumnya yang diambil dari variabel `motorSpeed` yaitu sebesar 200. Untuk mengetahui kondisi pergerakan motor B tersebut, kita dapat menggunakan indikator LED dengan memanggil fungsi `ledKanan()`. Alasan kami menggunakan dan memanggil fungsi `ledKanan()` tersebut sebagai indikator saat motor berbelok ke kiri masih sama seperti alasan sebelumnya yaitu karena yang bergerak adalah motor yang berada di sisi kanan mobil bukan sisi kirinya. Sehingga kita tidak menggunakan

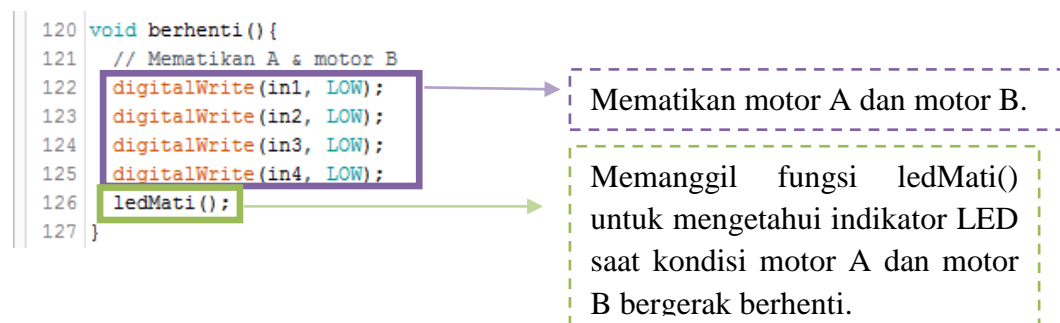


fungsi ledKiri() sebagai indikator pergerakan motor meskipun motornya tersebut berbelok ke kiri.



Fungsi mundur() pada fungsi aktivitas motor diatas tersebut digunakan untuk menyalakan motor A dan motor B agar kedua motor tersebut bisa bergerak mundur. Pada tabel konfigurasi, motor akan bergerak mundur apabila pada motor A pin in1 diberi logika “1” dan pin in2 diberi logika “0”, sedangkan pada motor B pin in3 diberi logika “0”, dan pin in4 diberi logika “1”. Hal tersebut kebalikan dari nilai logika saat kedua motor bergerak maju.

Setelah kita beri nilai logika pada pin IN di masing-masing motor, kita atur atau setting kecepatan masing-masing motor dengan menggunakan analogWrite(). Nilai kecepatan masing-masing motor telah ditetapkan sebelumnya dengan mengambil nilai dari variabel motorSpeed yaitu sebesar 200. Untuk mengetahui kondisi pergerakan motor, kita dapat menggunakan indikator LED dengan memanggil fungsi ledMundur().



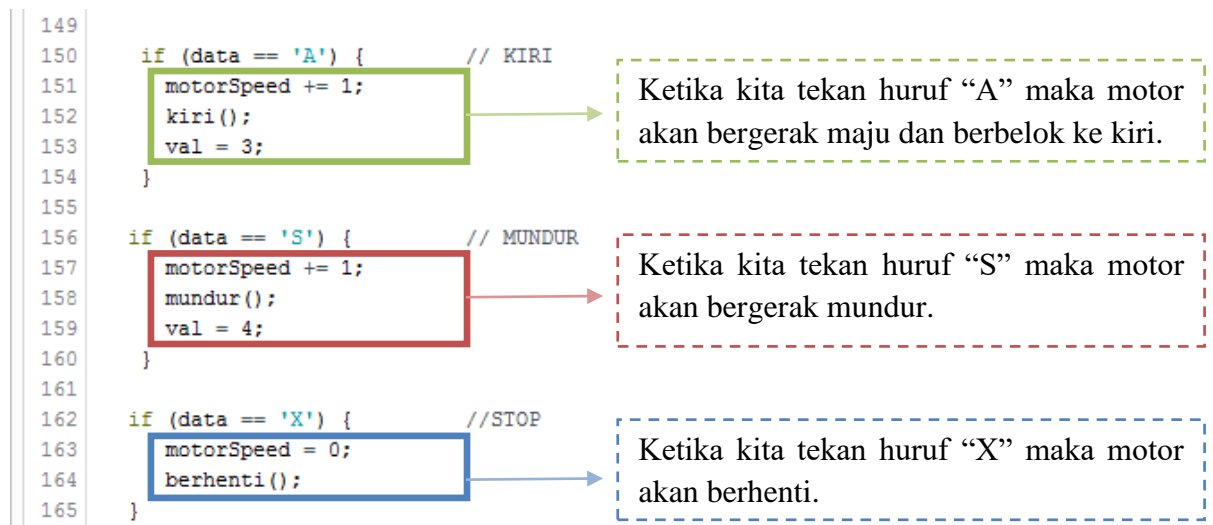
Fungsi berhenti() pada fungsi aktivitas motor diatas tersebut merupakan fungsi terakhir pada aktivitas motor DC yang digunakan untuk mematikan motor A dan motor B agar kedua motor tersebut dalam kondisi diam. Karena semua motor yaitu motor A dan motor B dalam kondisi berhenti maka semua pin IN kita beri logika bernilai “0” atau “LOW”. Pada kondisi ini kita tidak perlu mensetting kecepatan motor DC. Untuk mengetahui kondisi motor tersebut bahwa kedua motor benar-benar berhenti maka kita dapat menggunakan indikator LED dengan memanggil fungsi ledMati().

Selanjutnya kita ke fungsi loop() dimana fungsi loop() tersebut merupakan program utama yang akan dilihat apakah komunikasi serial dengan dua perangkat tersebut dapat dijalankan. Pada fungsi loop() tersebut dilakukan pengkondisian tentang input yang dikirim. Sebelumnya kita harus perhatikan terlebih dahulu data atau input yang dikirimkan tersebut berasal dari mana. Apabila inputnya berasal dari Arduino maka kita gunakan Serial.available(), sedangkan apabila inputnya berasal dari bluetooth maka kita gunakan mySerial.available(). Saat input yang kita masukkan tersebut memenuhi syarat yaitu lebih besar dari 0 maka akan lanjut ke proses berikutnya. Dimana proses selanjutnya yaitu pembacaan nilai variabel “data”. Saat diberi input misal “1” maka program akan membaca dan memproses nya yaitu masuk case pertama. Ada 5 buah kondisi pembacaan nilai dari keyboard ketika memasukkan nilai variabel “data” untuk menambah kecepatan motor bergerak yaitu berupa huruf “W”, “D”, “A”, “S” dan “X”.

```
130 void loop()
131 {
132   // PERHATIKAN DATA YANG DIKIRIM DARI MANA?
133   // PAKAI "Serial" JIKA DARI ARDUINO BIASA
134   // PAKAI "mySerial" JIKA DARI BLUETOOTH
135   if (Serial.available() > 0)
136   {
137     int data = Serial.read(); // MEMBACA DATA YANG DIKIRIM, DALAM HAL INI DARI PROCESSING
138     if (data == 'W') { // MAJU
139       motorSpeed += 1;
140       maju();
141       val = 1;
142     }
143
144     if (data == 'D') { // KANAN
145       motorSpeed += 1;
146       kanan();
147       val = 2;
148     }
149   }
```

Ketika kita tekan huruf “W” maka motor akan bergerak maju.

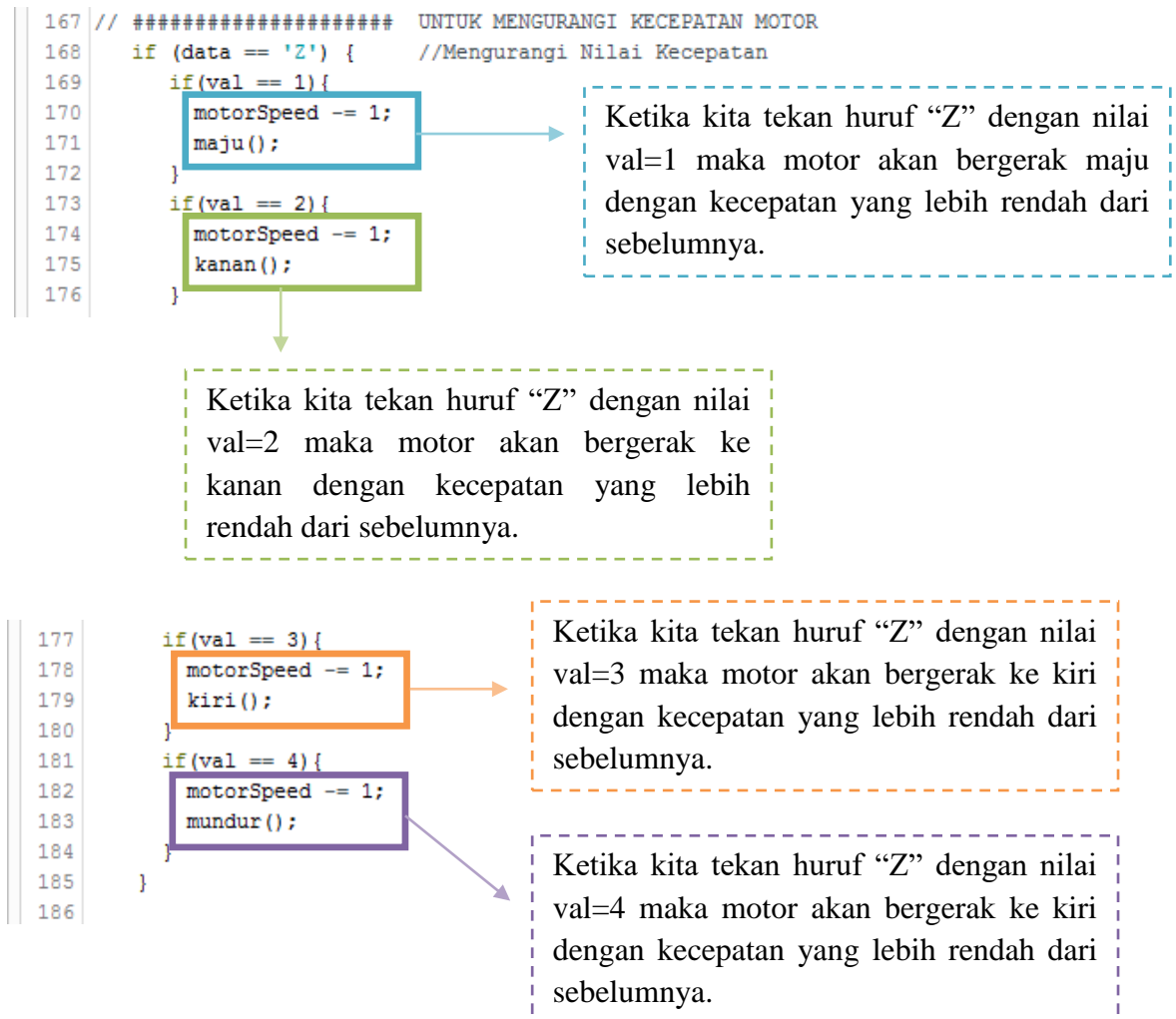
Ketika kita tekan huruf “D” maka motor akan bergerak maju dan berbelok ke kanan.



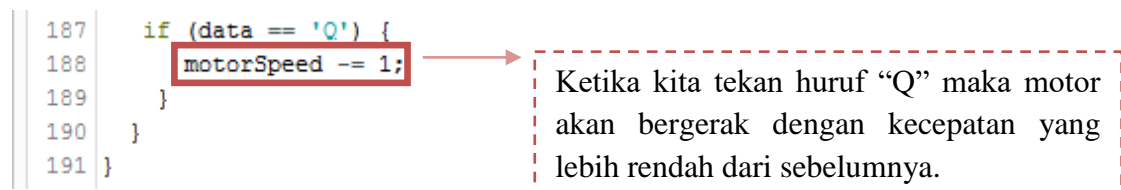
Ketika kita tekan huruf "W", kecepatan motor akan bertambah 1 dari kecepatan motor sebelumnya. Variabel "val" yang sebelumnya bernilai 0 akan berubah menjadi bernilai 1. Perubahan nilai variabel "val" menjadi bernilai 1 tersebut khusus untuk kondisi ketika kita menekan huruf "W" di keyboard laptop kita masing-masing. Kemudian ke syntax program selanjutnya dimana fungsi maju() akan terpanggil sehingga otomatis motor akan bergerak maju. Ketika kita tekan huruf "D", kecepatan motor akan bertambah 1 dari kecepatan motor sebelumnya. Karena sebelumnya variabel "val" sudah bernilai 1, maka nilai variabel "val" yang baru untuk kondisi ini akan berubah menjadi bernilai 2. Kemudian ke syntax program selanjutnya dimana fungsi kanan() akan terpanggil sehingga otomatis motor akan bergerak maju dan berbelok ke kanan. Ketika kita tekan huruf "A", kecepatan motor akan bertambah 1 dari kecepatan motor sebelumnya. Karena sebelumnya variabel "val" sudah bernilai 2, maka nilai variabel "val" yang baru untuk kondisi ini akan berubah menjadi bernilai 3. Kemudian ke syntax program selanjutnya dimana fungsi kiri() akan terpanggil sehingga otomatis motor akan bergerak maju dan berbelok ke kiri.

Ketika kita tekan huruf "S", kecepatan motor akan bertambah 1 dari kecepatan motor sebelumnya. Karena sebelumnya variabel "val" sudah bernilai 3, maka nilai variabel "val" yang baru untuk kondisi ini akan berubah menjadi bernilai 4. Kemudian ke syntax program selanjutnya dimana fungsi mundur() akan terpanggil sehingga otomatis motor akan bergerak mundur. Ketika kita tekan huruf "X", kecepatan motor akan 0. Hal ini disebabkan pada kondisi ini motor dalam keadaan berhenti meskipun nilai variabel "val" sebelumnya bernilai 4. Proses tersebut akan terjadi berulang

sampai terjadi pembacaan input lagi. Untuk mengurangi nilai kecepatan motor saat bergerak, kita gunakan nilai variabel “data” yaitu berupa huruf “Z” dan “Q”.

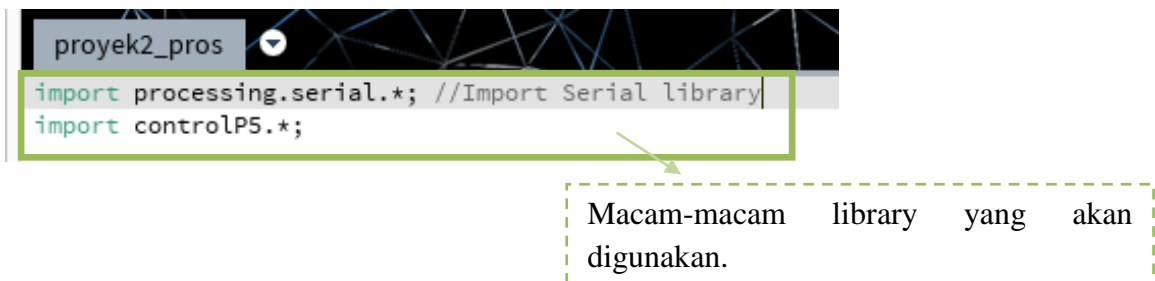


Ketika kita tekan huruf “Z”, maka ada 4 buah case yang terjadi yaitu ketika val bernilai 1, 2, 3 dan 4. Masing-masing case tersebut sama seperti sebelumnya yaitu akan memanggil fungsi-fungsinya sesuai penulisan pemanggilan fungsi di case yang digunakan. Namun perbedaannya yaitu terjadi perubahan nilai kecepatan motor DC dimana nilai kecepatannya akan dikurangi 1 dari nilai sebelumnya. Motor akan bergerak sesuai nilai kecepatan dari variabel “val” yang diberikan.

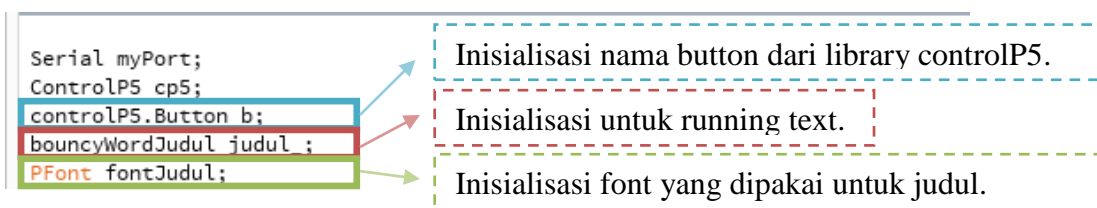


Ketika kita tekan huruf “Q”, maka kecepatan motor DC akan berkurang 1. Saat penekanan huruf di keyboard kita masing-masing, kita harus memperhatikan besar kecilnya huruf yang kita tekan tersebut. Jika kita menekan huruf yang salah maka otomatis program tidak akan diproses karena tidak dapat terdeteksi meskipun huruf yang kita tekan sama yaitu misalkan huruf “W” dengan menggunakan capital dan huruf “w” biasa. Kesalahan penekanan tersebut akan berpengaruh dan fatal sekali meskipun hanya kesalahan kecil.

Pada program Arduino ini, sebenarnya proses yang terjadi dimulai dari fungsi loop() dimana di dalam fungsi loop() tersebut berisi beberapa fungsi. Fungsi-fungsi tersebut akan terpanggil satu per satu sesuai urutan kita memanggilnya. Setelah kita buat program Arduino, kita buat program processingnya. Kami membuat tampilan speedometer untuk memvisualisasikan kecepatan motor yang kami buat. Pada program processingnya kita harus memasukkan terlebih dahulu library-library yang akan digunakan nantinya. Hal tersebut bertujuan untuk mempermudah kita untuk mengakses syntax-syntax yang belum tersedia di processing. Dengan adanya library tersebut, processing dapat memungkinkan digunakan untuk berbagai aplikasi.



Library processing.serial merupakan library yang digunakan untuk komunikasi serial. Sedangkan library controlP5 ini banyak tersedia GUI seperti Sliders, Buttons, Toggles, Knobs, Textfields, RadioButtons, Checkboxes yang sangat mudah kita tambahkan di processing sketch. Setelah kita import library, kita juga deklarasi variabel-variabel yang akan kita gunakan.



```
// INISIALISASI UNTUK SPEEDOMETER
PImage jarum, background, speedo, judul;
PImage UP_, RIGHT_, LEFT_, DOWN_, setUp, setRIGHT, setLEFT, setDOWN, pens;
int rec_x=355, rec_y=640, rec_height=10, z1=200;
```

Inisialisasi gambar-gambar yang ada di speedometer.

```
// INISIALISASI LAIN
int w, indikator;
int buttonValue = 1;
float angle;
boolean isOpen;
```

Mendeklarasikan variabel-variabel lain yang akan digunakan.

Inisialisasi untuk waktu terutama hari yang akan digunakan.

```
// INISIALISASI UNTUK WAKTU
String[] dayName =
{
  "SEN", "SEL", "RAB", "KAM", "JUM", "SAB", "MIN"
};

String[] monthName =
{
  "Jan", "Feb", "Mar", "Apr", "Mei", "Jun",
  "Jul", "Agu", "Sep", "Okt", "Nov", "Des"
};
```

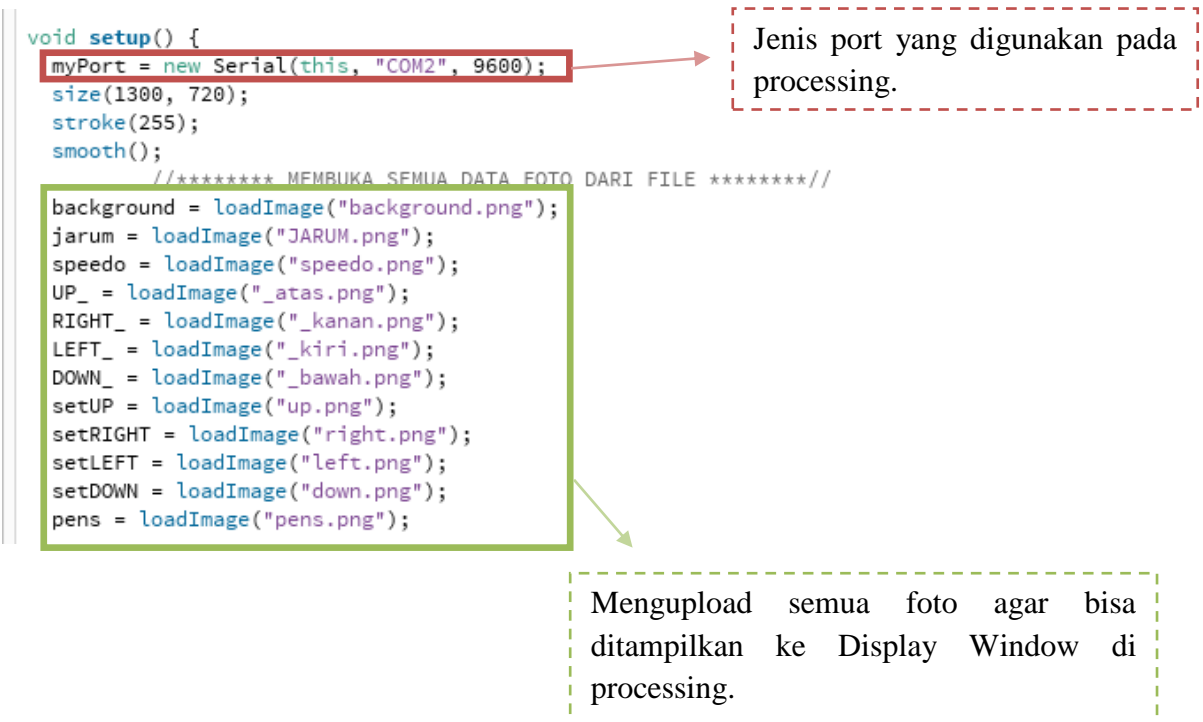
Inisialisasi untuk waktu terutama nama-nama bulan yang akan digunakan.

```
int hari = day();
int bulan = month();
int tahun = year();
```

Mendeklarasikan semua variabel yang digunakan untuk keterangan waktu.

dan  
void draw(). Void setup() adalah fungsi yang akan dieksekusi hanya sekali oleh program. Pada umumnya, skrip yang dimasukkan di dalam void setup() adalah fungsi size(), mengatur warna background, mengatur warna garis, atau memuat gambar maupun huruf. Jika berkomunikasi secara serial dengan peralatan lain di luar komputer, kita dapat mengatur kecepatan serial di dalam fungsi ini. Sedangkan void draw() adalah fungsi yang akan dieksekusi berkali-kali oleh program, dan baru akan berhenti ketika Display Window kita close atau tombol Stop ditekan. Program akan dieksekusi step per step yaitu dari skrip paling atas, melewati void setup() dan selanjutnya menuju void draw(). Di dalam void draw(), program akan dijalankan berulang-ulang. Satu perjalanan dalam mengeksekusi program di dalam void draw

dikenal dengan istilah frame. Secara default, Processing akan mengeksekusi 60 frame perdetik. Namun, kecepatan tersebut dapat kita diatur.



Untuk fungsi `size()` kita harus tentukan ukuran panjang kali lebar tampilan Display Window, jika tidak maka Display Window tersebut otomatis akan berukuran 100 x 100 piksel. Pada bagian ini, kita akan membuat sebuah layar dengan ukuran 1300 x 720 piksel. Fungsi `smooth()` yang kita gunakan tersebut berfungsi untuk menampilkan gambar yang halus (anti aliasing). Kemudian saat hendak mengupload semua foto tersebut kita gunakan perintah `loadImage()`. Foto yang akan kita upload harus berada di dalam satu folder yang sama dengan program processing. Hal tersebut disebabkan karena apabila file fotonya tidak ditambahkan ke sketsa processing yang kita buat maka file fotonya tidak dapat diakses dan dibaca. Selain itu format foto yang kita masukkan harus benar. Kita buat tombol atau button yang akan digunakan sebagai perintah untuk menampilkan atau menyembunyikan foto. Untuk jenis port yang digunakan di processing harus berbeda dengan jenis port yang digunakan di Arduino. Misal di Arduino menggunakan port COM1 maka di processing kita harus menggunakan port COM2 dan begitu juga sebaliknya. Hal tersebut disebabkan agar kedua program dapat di run. Jika tidak maka tidak bisa di run program keduanya.

```
// BUTTON UNTUK OPEN/HIDE PHOTO
cp5 = new ControlP5(this);
cp5.addButton("button");
    .setPosition(width-100, 20)
    .setSize(60, 20)
    ;
```

Membuat tombol dengan nama "button".

Tombol yang kita buat tersebut berukuran 60 x 20 piksel dengan posisi (width-100) x 20 piksel. Nilai width yang dituliskan tersebut merupakan lebar jendela tampilan yang otomatis sudah menjadi default dalam satuan piksel. Nilai default dari lebar jendela tampilan tersebut menjadi 100 jika ukuran tidak digunakan dalam program. Kita harus teliti dalam menuliskan programnya di processing karena jika terjadi kesalahan, misalnya tidak menuliskan tanda semikolon (titik koma) di akhir skrip, maka akan muncul pesan error pada Message Area,

```
b = cp5.addButton("buttonValue")
    .setPosition(0, 400)
    .setImages(loadImage("foto.png"), loadImage("foto.png"), loadImage("foto.png")) // FOTO BARENG
    // .setImages(loadImage("fot.png"), loadImage("fot.png"), loadImage("fot.png")) // FOTOKU
    ;
cp5.getController("button")
    .getCaptionLabel()
    .setSize(20)
    ;
b.getCaptionLabel()
    .toUpperCase(false)
    ;
```

Menampilkan foto yang di akan di show dan hide.

Syntax program diatas tersebut berfungsi untuk menampilkan sebuah foto yang diwujudkan dalam bentuk button. Apabila button yang sebelumnya kita tekan maka button yang berisi foto-foto akan muncul. Dan apabila buttonnya kita tekan sekali lagi maka otomatis foto-fotonya akan hilang atau tersembunyi. Foto-foto tersebut merupakan foto-foto tambahan sebagai animasi yang berisi foto-foto anggota kelompok kami.

Kita bisa meload font untuk sebuah judul di Display Windows processing. Font yang akan kita load tersebut harus berada di dalam satu folder yang sama. Hal tersebut sama seperti ketika kita sedang mengupload gambar atau foto. Kita harus mendownload terlebih dahulu jenis font yang kita gunakan tersebut karena jenis fontnya belum tersedia di tools processing. Animasi font yang kita upload tersebut bertujuan agar kita tidak terlalu bosan menggunakan font-font yang hanya tersedia di processing selama ini.



```
// LOAD FONT UNTUK JUDUL
fontJudul = loadFont("Blanka-Regular-46.vlw");
textFont(fontJudul);
judul_ = new bouncyWordJudul("", -3, 300, 50);
}
```

Jenis font yang akan digunakan khusus untuk mengupload font pada judul.

Posisi dari jenis font yang kita gunakan telah disesuaikan dengan kondisi real.

Selanjutnya, program utama akan dituliskan pada void draw(). Pada void draw(), fungsi akan dieksekusi berkali-kali oleh program.

```
void draw() {
  imageMode(CORNER);
  image(background, 0, 0, width, height);
  judul_.draw();

  image(loadImage("up_.png"), 1030, 420, 135, 64);
  image(loadImage("right_.png"), 1165, 485, 64, 135);
  image(loadImage("left_.png"), 965, 485, 64, 135);
  image(loadImage("down_.png"), 1030, 620, 135, 64);
  image(speedo, 220, 55, 865, 655);
}
```

Meload gambar background dengan posisi corner dan fungsi dari running text dijalankan.

Jenis font yang akan digunakan khusus untuk mengupload font pada keterangan waktu.

Mengatur warna yang digunakan untuk mengisi bentuk sesuai RGB.

Meload foto untuk tampilan arah dari speedometer dengan posisi yang telah disesuaikan dengan kondisi real di tampilan processingnya.

```
// CLOCK FUNCTION
int s = second();
int m = minute();
int h = hour();
fill(250, 250, 250);
textFont(loadFont("Digital-7Italic-48.vlw"));
textSize(52);
textAlign(LEFT);
text(h + ":" + m + ":" + s, 290, 250);
textAlign(CENTER);
textSize(40);
text(dayName[4], 905, 225); // Pengaturan Hari Manual
textSize(32);
text(monthName[bulan-1], 970, 175); // CETAK BULAN
textSize(40);
text(tahun, 970, 260); // CETAK TAHUN
textSize(74);
text(hari, 970, 230); // CETAK HARI
```

Mencetak jam.

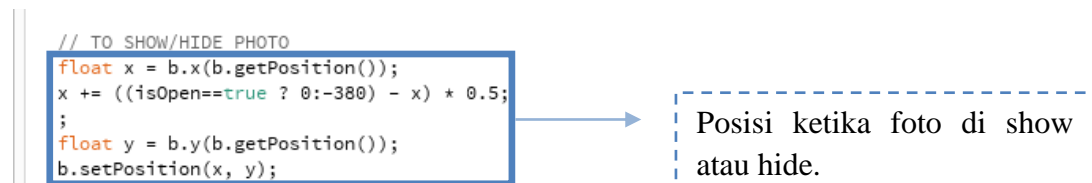
Pengaturan hari.

Mencetak bulan.

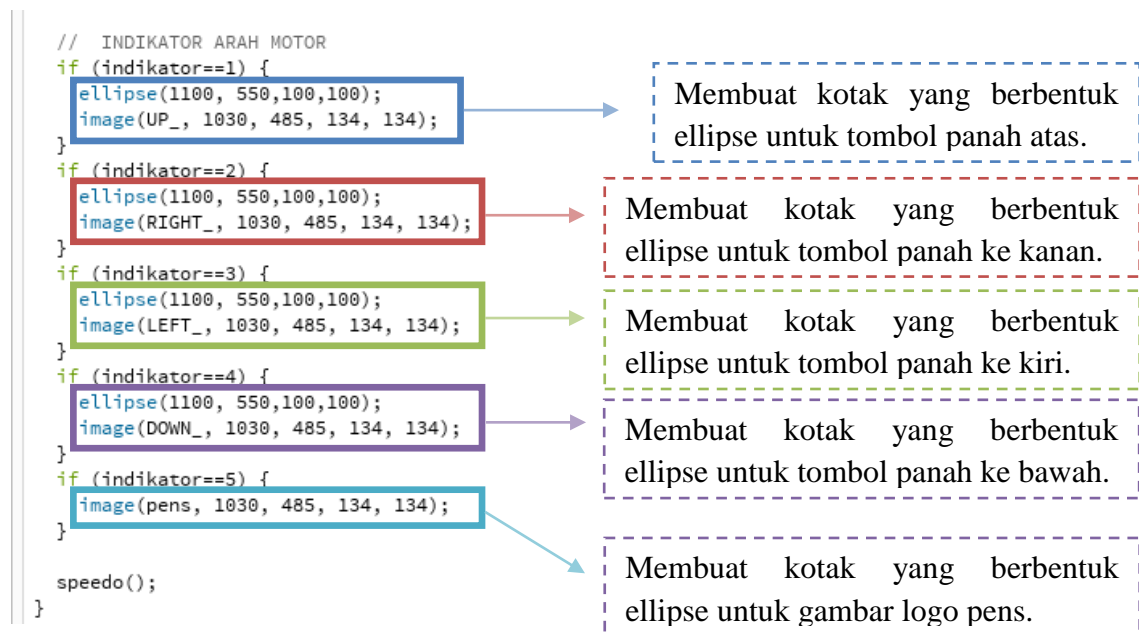
Mencetak tahun.

Mencetak hari.

Untuk pengaturan hari kita atur secara manual, sedangkan untuk keterangan waktu lainnya yang berupa jam, tanggal dan bulan tersebut ikut dalam system OS Windows yang kita gunakan masing-masing. Warna yang digunakan untuk keterangan waktu tersebut sesuai warna RGB (Red, Green, Blue). Karena nilai RGB yang digunakan tersebut memiliki perpaduan nilai maksimal yaitu masing-masing 255 maka warna yang dihasilkan adalah warna putih.



Yang dimaksudkan oleh syntax diatas merupakan foto animasi anggota kelompok kami. Ketika foto tersebut tersembunyi atau di hidden maka posisinya terletak sejauh -380 dari koordinat sumbu x.



Fungsi indikator diatas tersebut digunakan untuk memberikan informasi posisi arah putaran motor, dimana :

- Jika maju ➡ gambar panah atas
- Jika kanan ➡ gambar panah kanan
- Jika kiri ➡ gambar panah kiri
- Jika mundur ➡ gambar panah mundur
- Jika stop ➡ gambar pens

Sebelumnya, gambar-gambar yang digunakan tersebut harus kita deklarasikan semua di awal agar kita mudah untuk mengaksesnya. Masing-masing indikator diatas tersebut telah ditentukan ukurannya yang berupa bentuk ellipse beserta posisi gambarnya. Pada bentuk ellipse, kita harus mengetahui apa saja batas-batas yang digunakan semisal seperti koordinat sumbu x, koordinat sumbu y, lebar ellipse secara default, beserta panjang ellipse secara default.

Kita disini tidak hanya menggunakan 2 buah fungsi utama yang selama ini ada di processing, melainkan kami juga menggunakan fungsi-fungsi yang lainnya. Untuk mengontrol arah dan kecepatan motor, kami gunakan fungsi `keyPressed()`. Fungsi `keyPressed()` tersebut dipanggil sekali setiap kali tombol ditekan dimana tombol yang ditekan tersebut akan disimpan dalam variabel kunci. Untuk kunci non-ASCII, kita gunakan variabel `keyCode`. Apabila kunci yang kita gunakan tersebut termasuk dalam spesifikasi ASCII (BACKSPACE, TAB, ENTER, RETURN, ESC, dan DELETE) maka tidak memerlukan pemeriksaan untuk melihat apakah kuncinya dikodekan atau tidak. Hal ini disebabkan karena mengenai bagaimana cara sistem operasi menangani pengulangan kunci, menekan tombol dapat menyebabkan beberapa panggilan ke `keyPressed()`. Tingkat pengulangan diatur oleh sistem operasi, dan dapat dikonfigurasi secara berbeda-beda pada setiap komputer.

```
//***** KONTROL ARAH DAN KECEPATAN MOTOR *****//
```

```
void keyPressed() {
  imageMode(CORNER);
  if (key == CODED) {
    if (keyCode == UP) {
      image(setUP, 1030, 420, 135, 64);
      println("MAJU");
      indikator=1;
      if (z1>=0 && z1<=254) {
        z1++;
        myPort.write('W');
      } else if (z1==255) {
        z1 = 254;
        myPort.write('Q');
      }
    }
  }
}
```

jika yang ditekan tombol arah **atas** maka akan mengirimkan data tipe char "W".

```
if (keyCode == RIGHT) {
  image(setRIGHT, 1165, 485, 64, 135);
  println("KANAN");
  indikator=2;
  if (z1>=0 && z1<=254) {
    z1++;
    myPort.write('D');
  } else if (z1==255) {
    z1 = 254;
    myPort.write('Q');
  }
}
```

jika yang ditekan tombol arah **kanan** maka akan mengirimkan data tipe char "D".

```
if (keyCode == LEFT) {
  image(setLEFT, 965, 485, 64, 135);
  println("KIRI");
  indikator=3;
  if (z1>=0 && z1<=254) {
    z1++;
    myPort.write('A');
  } else if (z1==255) {
    z1 = 254;
    myPort.write('Q');
  }
}
```

jika yang ditekan tombol arah **kiri** maka akan mengirimkan data tipe char "A".

```
if (keyCode == DOWN) {
  image(setDOWN, 1030, 620, 135, 64);
  println("MUNDUR");
  indikator=4;
  if (z1>=0 && z1<=254) {
    z1++;
    myPort.write('S');
  } else if (z1==255) {
    z1 = 254;
    myPort.write('Q');
  }
}
```

jika yang ditekan tombol arah **bawah** maka akan mengirimkan data tipe char "S".

} Kondisi penekan tombol diatas tersebut sesuai dengan kondisi yang ada di program Arduino, sehingga antara program di processing dan program di Arduino saling berkesinambungan. Untuk kondisi jika nilai z1=255, maka akan diubah menjadi 254. Dengan demikian nilai z1 tidak akan melebihi 255 ini disesuaikan dengan nilai PWM yang dibaca motor DC dan akan mengirimkan data dengan tipe char "Q". Ketika data yang dikirimkan tersebut berupa tipe char "Q" maka kecepatan motor DC bukannya bertambah satu melainkan akan berkurang satu. Sedangkan apabila nilai z1 bukan 255 atau bahkan jauh dibawah angka 255, maka kecepatan

motor DC akan naik satu per satu sesuai dengan penekanan tombol panah yang diberikan baik itu tombol up, down, right, dan left di keyboard laptop kita masing-masing.

Selain itu kita juga bisa menekan tombol lain selain tombol panah untuk mengontrol arah dan kecepatan motor. Tombol tersebut adalah tombol “z” dan tombol “x”. Karena kedua tombol tersebut merupakan bilangan ASCII, maka kita tidak perlu menggunakan variabel keyCode dimana kita cukup menggunakan variable key.

The diagram illustrates the logic for handling key presses 'z' and 'x'. It consists of a code block on the left and two explanatory text boxes on the right, connected by arrows.

```
if (key == 'z') {  
    println("REM....");  
    if (z1>0 && z1<=256) {  
        z1--;  
        myPort.write('Z');  
    }  
    if (0==z1) {  
        z1 = 0;  
    }  
}  
  
if (key == 'x') {  
    println("STOP");  
    indikator=5;  
    z1 = 0;  
    myPort.write('X');  
}
```

Diagram illustrating the logic for key presses:

- Blue box: jika yang ditekan **tombol z** maka akan mengirimkan data tipe char “Z”.
- Red box: jika yang ditekan **tombol x** maka akan mengirimkan data tipe char “X”.

Ketika kita menekan tombol “z” maka data yang dikirimkan tersebut akan berupa tipe char “Z” dimana kecepatan motor akan berkurang satu per satu sesuai penekanan tombol yang diberikan. Apabila kecepatan motor nilainya sudah sama dengan 0 maka otomatis kecepatan motor tersebut juga akan bernilai 0. Ketika kita menekan tombol “x” maka data yang dikirimkan tersebut akan berupa tipe char “X”. Tipe data char “X” tersebut berarti kecepatan motornya bernilai 0 dan motor akan berhenti seketika itu juga.

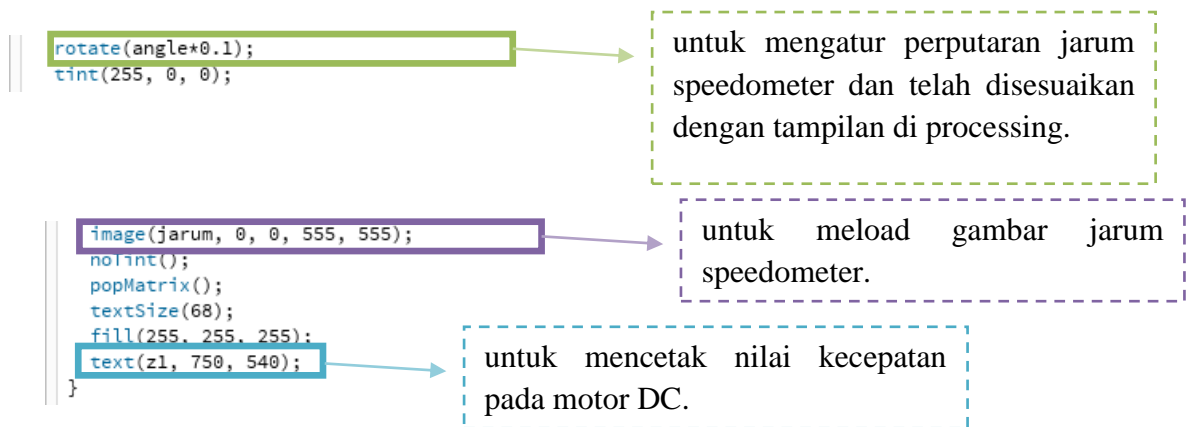
Selanjutnya kami mencoba mengatur perputaran jarum speedomernya. Pada bagian ini kami atur sedemikian rupa bagaimana cara jarum yang terpasang di speedometer dapat berjalan sesuai dengan kecepatan motor yang dimiliki. Kita harus mengatur tentang batas sudut perputaran jarum jam untuk berputar.

The diagram shows a code snippet for setting the speedometer needle's rotation. A specific line of code is highlighted with an orange box, and an arrow points from it to an explanatory text box.

```
//***** MENGATUR PERPUTARAN JARUM SPEEDO *****//  
void speedo() {  
    imageMode(CENTER);  
    textAlign(CENTER);  
    pushMatrix();  
    translate(667, 435);  
    angle = map(z1, 0, 255, 39.5, 79);  
}
```

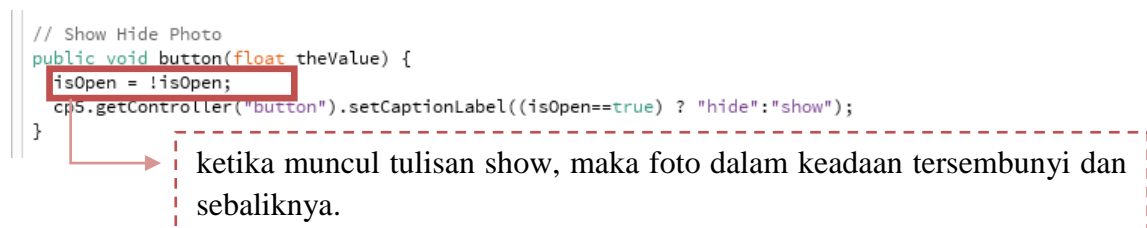
Diagram illustrating the logic for setting the speedometer needle rotation:

- Orange box: untuk mengatur batas sudut perputaran jarum jam.



Sebelumnya kita harus mengupload terlebih dahulu gambar jarum dengan posisi yang disesuaikan agar tertata tepat dengan speedometer di tampilan processingnya. Lalu kita atur batas sudut perputaran jarum jam dengan menggunakan syntax `map()`. Syntax `map()` tersebut digunakan untuk memetakan dari satu rentang ke rentang lainnya. Nilai masuk yang akan dikonversi adalah sebesar `z1` dimana nilai `z1` tersebut akan tergantung dari kecepatan motornya. Batas bawah dari rentang nilai yang digunakan tersebut adalah 0, sedangkan batas atasnya adalah 255. Untuk batas bawah kisaran target nilai yang digunakan adalah 39.5 dan batas atasnya adalah 79. Nilai-nilai yang digunakan tersebut telah disesuaikan dengan kondisi real pada tampilan processingnya. Apabila posisinya tidak pas maka kita harus mengatur nilainya kembali agar presisi.

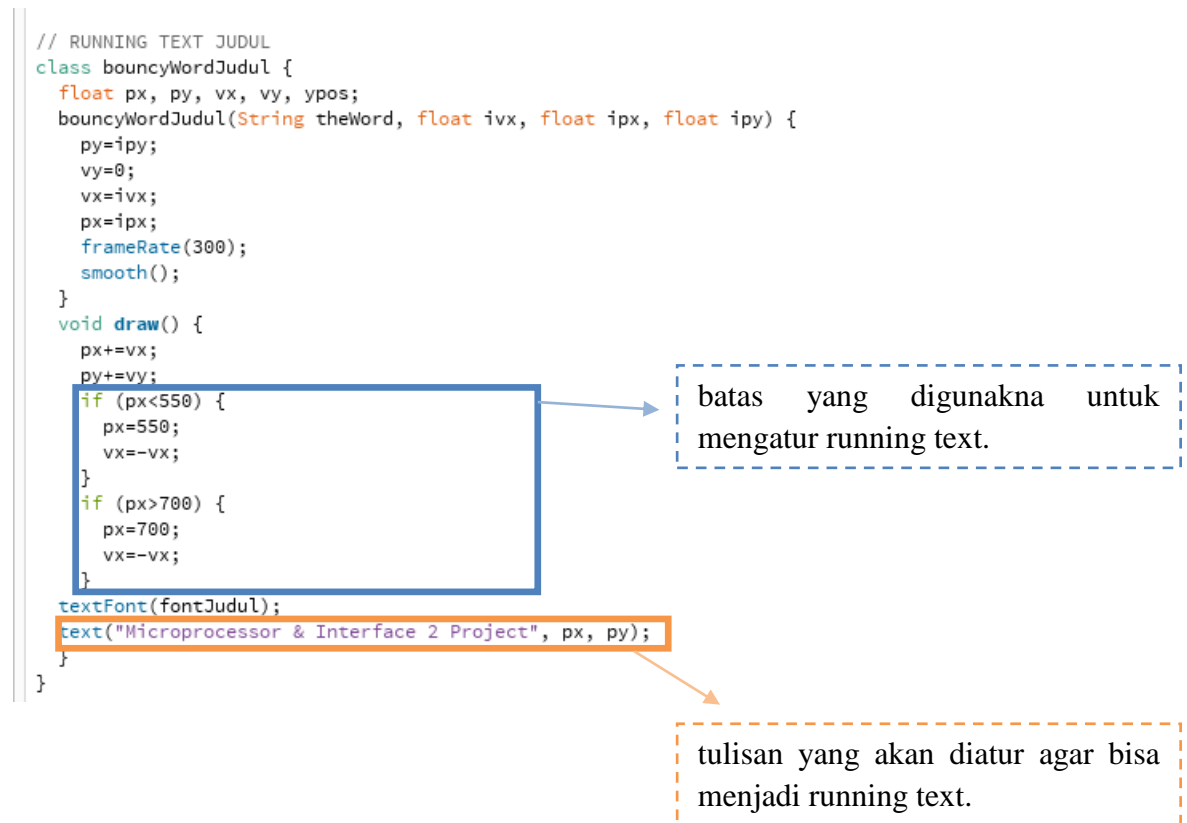
Untuk menampilkan tulisan `show` atau `hide` pada tampilan processing saat memunculkan gambar atau foto kelompok kami, kami gunakan syntax seperti gambar dibawah ini :



Syntax diatas tersebut akan melakukan pembacaan untuk button ketika diklik, apakah `show` atau `hide`. Jika pada pojok kanan tampilan processing terdapat button dengan tulisan `show`, maka itu berarti foto berada dalam posisi tersembunyi sehingga kita harus mengklik button tersebut agar fotonya muncul. Saat foto muncul, button berubah menjadi tulisan `hide`. Agar foto tersebut tersembunyi maka kita harus

mengklik sekali lagi button tersebut sehingga tulisan buttonnya berubah menjadi show.

Selain itu kita juga bisa untuk mengatur batas running text pada judul yang kita gunakan. Tampilan running text yang kami gunakan tersebut bertujuan agar ada sedikit kreatifitas dan tampilannya tidak monoton itu-itu saja sehingga kami berinisiatif untuk membuat tulisan yang bisa di running.



Untuk mengatur batas running text yaitu diposisi koordinat `x=550` sampai `x=700`. Posisi tersebut telah disesuaikan pada kondisi real di tampilan processing. Kita bisa mengatur posisi tersebut apabila ada sedikit tidak atau kurang pas dengan tampilan processingnya. Jenis tulisan yang digunakan pada running text diatas tersebut adalah sesuai seperti variabel `fontJudul` dengan font huruf `Blanka-Regular-46`.

## **BAB III**

### **PENUTUP**

#### **I. Kesimpulan**

Dari hasil project yang telah kami lakukan dapat disimpulkan bahwa :

- ✓ Kontrol pergerakan motor dilakukan dengan menggunakan processing dengan menggunakan beberapa tombol yang telah disetting sebelumnya.
- ✓ Kita juga tetap memerlukan input dari luar yaitu baterai 12 volt untuk menambah tegangan dan memaksimalkan putaran dari motor DC.
- ✓ Semakin besar input dari luar yang diberikan maka motor DC juga akan berputar semakin cepat dan begitu juga sebaliknya.
- ✓ Input yang diberikan dari luar tersebut besarnya harus sesuai dengan tegangan yang diminta oleh driver motor DC yang kita gunakan agar motor dapat berputar.
- ✓ Proses pengiriman data dari Arduino ke processing bisa dilakukan dengan cara berkomunikasi serial melalui Bluetooth.
- ✓ Dengan menggunakan Bluetooth, pengiriman data dari arduino ke processing dapat dilakukan tanpa harus menggunakan kabel usb.
- ✓ Monitoring pergerakan motor pada project yang kami buat tersebut diimplementasikan dengan menampilkan kondisi motor dc serta kecepatan dilayar menggunakan gambar speedometer.
- ✓ Dengan adanya processing, kita bisa memvisualisasikan bagaimana control dari pergerakan motor sehingga tampilan jadi lebih menarik dan tidak membosankan.
- ✓ Selain itu dengan menggunakan processing, kita dapat mengasah kreatifitas kita untuk melakukan sesuatu yang baru sehingga tidak harus selalu berpaku dengan modul-modul yang diberikan selama ini.

#### **II. Saran**

Beberapa saran untuk pengembangan project akhir ini diantaranya adalah :

1. Tampilan processingnya diupayakan agar lebih menarik lagi dengan berbagai macam animasi.



2. Diupayakan ada beberapa tambahan komponen yang digunakan misalnya seperti penggunaan sensor ultrasonic sehingga apabila terhalang oleh suatu benda, motor akan secara otomatis berbelok tanpa harus kita atur atau arahkan.

## DAFTAR PUSTAKA

- ✚ <http://belajar-dasar-pemrograman.blogspot.com/2013/03/arduino-uno.html>
- ✚ <http://robotic-electric.blogspot.co.id/2012/11/pulse-width-modulation-pwm.html>
- ✚ <https://ariefeeiggeennblog.wordpress.com/2014/02/07/pengertian-fungsi-dan-kegunaan-arduino/>
- ✚ <http://zoniaelektro.net/motor-dc/>
- ✚ <https://fahmizaleeits.wordpress.com/tag/h-bridge-adalah/>
- ✚ <http://zoniaelektro.net/motor-dc/>
- ✚ <https://splashtronic.wordpress.com/2012/05/13/hc-05-bluetooth-to-serial-module/>
- ✚ [http://rndc.or.id/wiki/index.php?title=Komunikasi Menggunakan Modul Bluetooth HC-05](http://rndc.or.id/wiki/index.php?title=Komunikasi_Menggunakan_Modul_Bluetooth_HC-05)
- ✚ <http://teknikelektronika.com/pengertian-piezoelectric-buzzer-cara-kerja-buzzer/>
- ✚ <https://indraharja.wordpress.com/2012/01/07/pengertian-buzzer/>
- ✚ <http://zoniaelektro.net/resistor-karakteristik-nilai-dan-fungsinya/>