MODUL BASIC ARDUINO

Bagian 1

Pengenalan Arduino

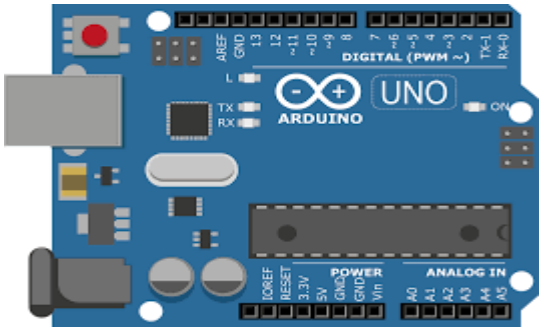
Handbook Basic Arduino ditujukan untuk anda yang masih pemula atau awam mengenai mikrokontroler dan arduino dimana anda dapat dengan mudah mempelajari dan mempraktekannya. Anda akan di ajarkan bagaimana prinsip kerja arduino serta cara menggunakan arduino serta software yang digunakan untuk membuat program arduino dengan cara yang sederhana sehingga dapat mudah di mengerti dan di ikuti, didalam handbook ini juga akan membahas mengenai sensor-sensor, konsep elektronik dan pembahasan program yang sederhana. Didalam handbook disiapkan proyek-proyek arduino sederhana, didalam setiap pembahasan akan di berikan flowchart, skematik rangkaian serta pembahasan program, serta materi-materi sederhana mengenai sensor-sensor yang digunakan didalam proyek tersebut, dengan tujuan anda akan dapat dengan mudah memahami dari setiap proses pembuatan proyek arduino tersebut.

1.1 Apa itu Arduino

Arduino adalah pengendali mikro *single-board* yang bersifat *open-source*, diturunkan dari *wiring platform* dirancang untuk memudahkan penggunaan elektronik dalam berbagai bidang. Hardwarenya memiliki prosesor **Atmel AVR** dan softwarenya memilik bahasa pemograman sendiri. Arduino juga merupakan hardware terbuka yang di tujukan kepada siapa saja yang ingin membuat purnarupa peralatan elektronik interaktif berdasarkan hardware dan software yang fleksibel dan mudah di gunakan.

Bahasa permograman yang dipakai Arduino merupakan bahasa pemograman sendiri yang memiliki kemiripan *syntax* dengan bahasa pemograman C. Karena sifatnya yang terbuka atau *open-source* maka siapa saja dapat dengan mudah mendownload skematik hardware arduino.

Dalam handbook ini kita akan menggunakan *board* Arduino Uno R3 seperti gambar 1.1. Board Arduino terdiri dari ATMega 328 sebagai mikrokontrolernya yang sdah terintegrasi dengan komponen pendukung lainya sehingga siap pakai, Arduino menggunakan *software IDE* yang berfungsi sebagai *software* pemograman.



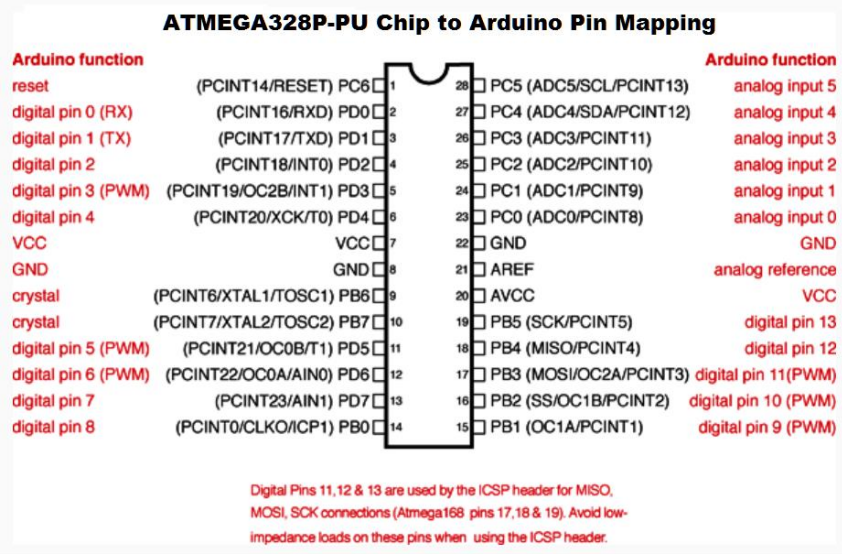
Gambar 1.1

Arduino Uno terdapat *pin-pin* yang mempunyai fungsi yang berbeda-beda, arduino uno memiliki 14 pin digital pin input/output, dimana 6 pin digunakan sebagai output PWM, 6 pin input analog, 16Mhz resonator keramik, koneksi USB, catudaya eksternal, header ICSP, tombol reset. Spesifikasi-spesifikasi dan pemetaan pin bisa di lihat pada tabel 1.1, dan gambar 1.2.

Tabel 1.1

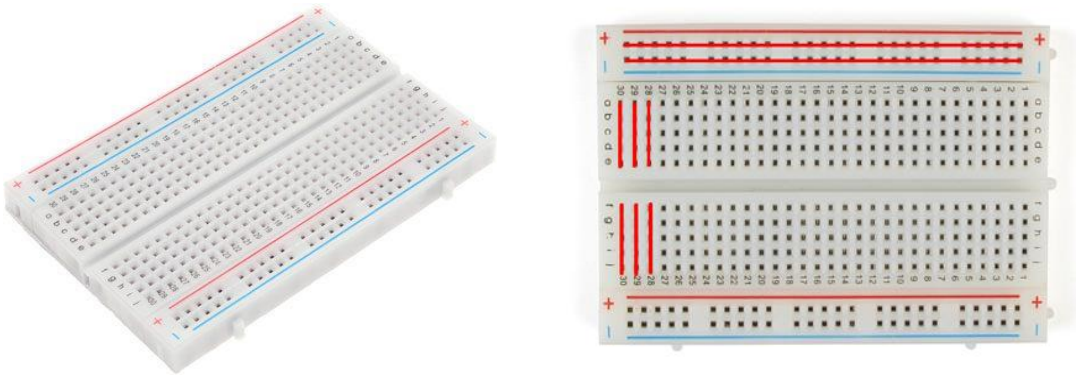
Spesifikasi Arduino Uno R3	Keterangan
Mikrokontroler	ATMega 328
Tegangan Operasi	5 volt
Input Voltage (disarankan)	7 – 12 volt
Output Voltage (batas max)	6 – 20 volt
Digital I/O Pin	14 (6 Pin sebagai Output PWM)
Analog Iput Pin	6 pin
Arus DC per pin I/O	40 mA
Arus DC unutk pin 3.3v	50 mA
Flash Memory	32 KB (ATMega 328) 0.5 KB (Bootloader)
SRAM	2 KB (ATMega 328)
EEPROM	1 KB (ATMega 328)
Clock Speed	16 MHz

Dibawah ini merupakan pemetaan pin-pin arduino yg menggunakan ATMega328 sebagai mikrokontrolernya.



Gambar 1.2 Pemetaan pin-pin arduino yang menggunakan ATMega 328p

Didalam handbook ini kita akan menggunakan *projectboard* atau *breadboard* sebagai alat bantu untuk menghubungkan kabel-kabel jumper dengan pin-pin arduino yg kita gunakan serta menghubungkan pin-pin sensor sehingga menjadi serangkaian proyek/alat elektronika tanpa harus menggunakan solder sebagai alat untuk menghubungkan antar pin.



Gambar 1.3 *Breadboad/Projectboard*

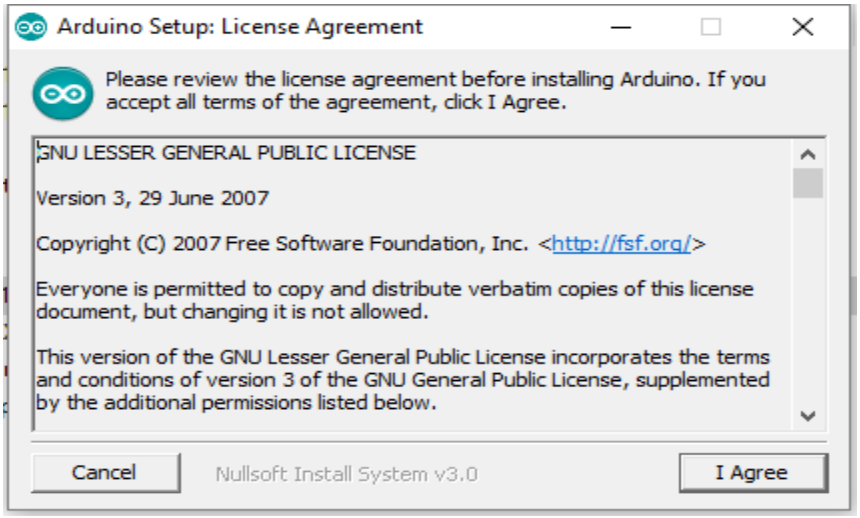
Gambar diatas merupakan contoh *breadboard/projectboard* dimana pada gambar dijelaskan letak posisi jalur yang terdiri dari jalur vertical dan horizontal, jalur horizontal terdapat 4 jalur yaitu 2 jalur bagian atas dan 2 jalur bagian bawah yang terdapat tanda beda pontensial (+) dan (-), jalur (+) tersambung sepanjang garis merah dan (-) yang tersambung sepanjang garis biru. Jalur vertikal yaitu jalur yang terhubung secara tegak lurus dimana terdapat huruf dan angka disetiap jalur yang menunjukkan tanda posisi pin 30 buah jalur atas dan 30 buah jalur bagian bawah, jalur vertikal ini berfungsi nanti sebagai tempat menghubungakan pin-pin antar sensor dan kontrollernya serta komponen pendukung lainnya.

1.2 Instalasi Arduino IDE

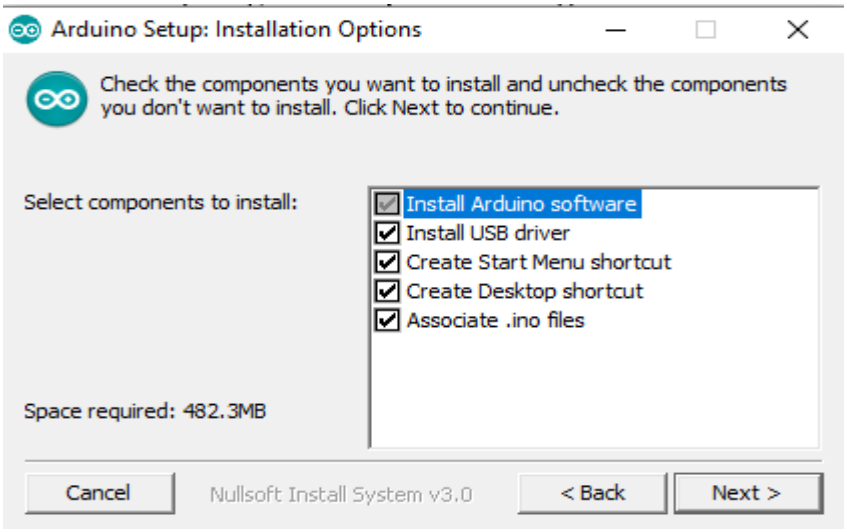
Arduino IDE merupakan software yang kita gunakan untuk memprogram arduino dan mengupload program tersebut kedalam modul arduino, dari banyaknya jenis dan tipe arduino software yang digunakan untuk memprogram arduino ialah Arduino IDE. Anda bisa mendownload Arduino IDE di website resmi arduino. Arduino IDE yang kita gunakan ialah versi terbaru 1.8.8 bisa anda cek didalam CD yang sudah kita incude-kan dengan handbook, tinggal meng-install software dengan mengikuti langkah di bawah ini:

1.2.1 Langkah Instalasi Arduino IDE

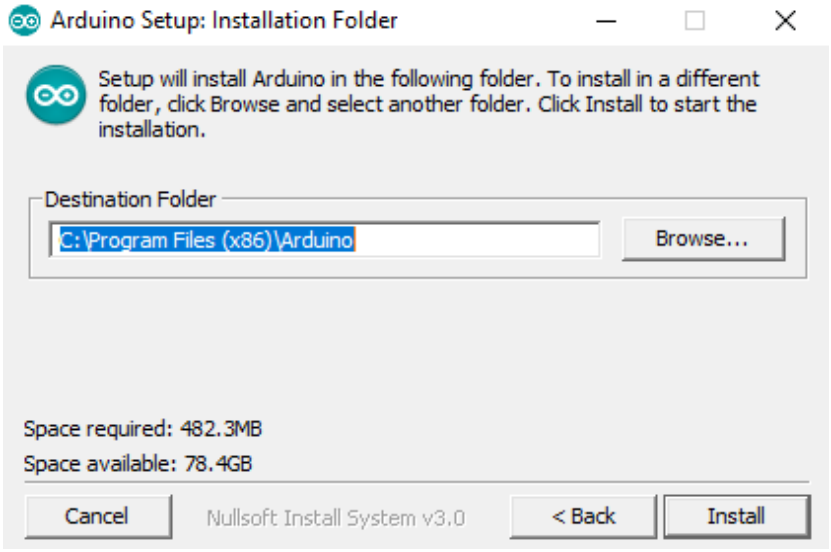
1. Buka CD instalasi yang sudah di includekan dengan handbook, klik file arduino-1.8.8 – windows.
2. Maka akan muncul seperti gambar ini, dan klik **I Agree**.



3. Kemudian centang semua kotak dan pilih **Next**.



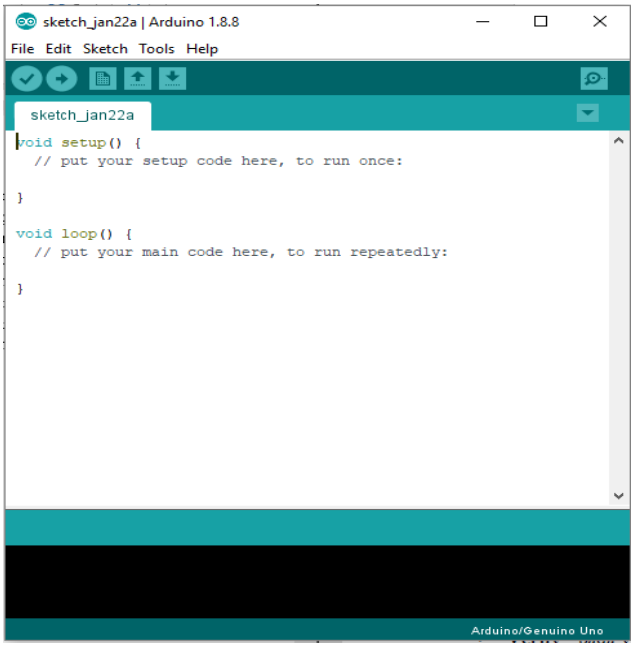
4. Klik **Instal** kemudian tunggu sampai proses instalasi selesai dan pilih **Close**.



1.2.2 Arduino IDE

Dalam pemograman *arduino board* kita membutuhkan aplikasi software yang digunakan untuk memprogram dan mengupload program ke dalam *arduino board*, software yang kita gunakan ialah Arduino IDE dimana setiap jenis dan tipe arduino semuanya menggunakan Arduino IDE sebagai software pemogramannya.

Aplikasi Arduino IDE ini berguna untuk memprogram, mengupload, mengedit serta membuat *sourcecode* arduino yang nanti dipergunakan untuk mengontrol sensor atau menggerakan *actuator*, motor servo dll didalam suatu proyek sederhana ataupun *big project* yang menggunakan arduino sebagai kontrolernya. Dibawah ini merupakan tampilan dari Arduino IDE 1.8.8 :



Gambar 1.4. *Interface* Arduino IDE

Gambar 1.7 merupakan *interface* dari arduino IDE yang nantinya kita gunakan untuk membuat, membuka program, upload program ke dalam hardware arduino uno, didalam software arduino IDE terdapat tools yang mempunyai fungsinya masing-masing, tools tersebut terdiri dari :

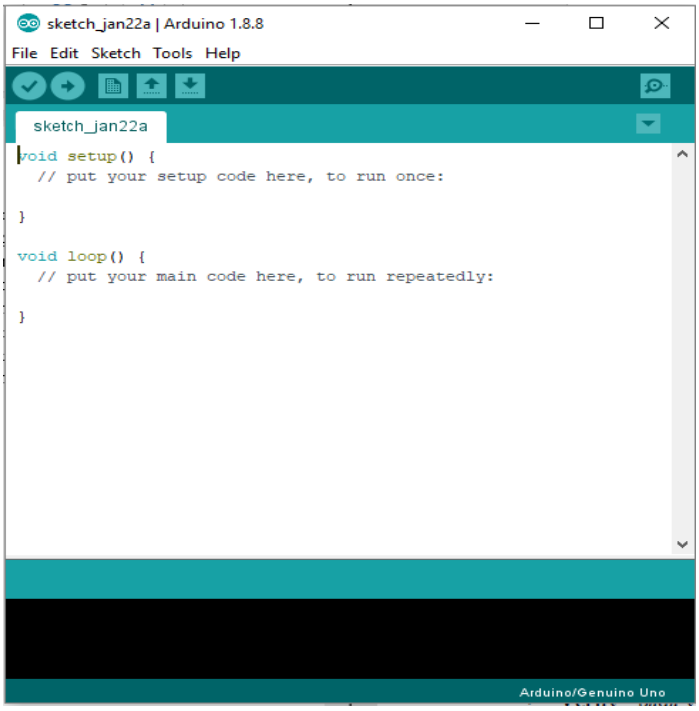
1. **Verify/Compile** merupakan tools yang berfungsi untuk mengecek file *skecth* dari program yang sudah di buat, sehingga nanti terlihat apakah ada eror atau kesalahan dalam penulisan *source code.*
2. **Upload** tool ini berfungsi untuk mengupload *skecth* program kedalam *board* arduino.
3. **New Skecth** berfungsi untuk membuat *skecth* baru.
4. **Open Skecth** berfungsi untuk membuka file *skectch* yang sudah pernah dibuat dengan file extensi **.ino**
5. **Save Skecth** tool yang berfungsi untuk menyimpan *skecth program.*

6. **Serial Monitor** tool ini berfungsi untuk membuka *interface* komunikasi serial dari program-program yang sudah dibuat.

1.3 Pemograman Arduino

Arduino merupakan *board* mikrokontroler yang dapat kita gunakan untuk mengontrol sensor-sensor ataupun aktuator yang dapat mempermudah pekerjaan kita, misalnya alat display suhu ruangan yang meggunakan sensor suhu sebagai pendeteksi suhu di dalam suatu ruangan dan di tampilkan pada sebuah display sehingga kita dapat mengetahui berapa suhu yang didalam ruangan tersebut, untuk membuat karya tersebut kita menggunakan *arduino board,* sensor-sensor serta *source code* yang kita buat menggunakan arduino IDE.

Sebelum menggunakan arduino IDE sebagai sarana membuat *source code* program kita terlebih dahulu harus mengetahui apa saja perintah dan bagaimana cara membuat *source code* yang dapat kita gunakan untuk mengontrol suatu alat, dibawah ini merupakan tampilan awal saat kita membuat software arduino IDE.



Gambar 1.5 Tampilan awal Arduino IDE

Diatas merupakan tampilan awal dari arduino IDE, terdapat perintah **void setup()** dan **void loop(),** yang akan muncul setiap kali kita memulai untuk membuat *skecth program* / *source code* program. Arduino IDE ini siap kita gunakan tanpa harus ada tambahan settingan supaya bisa digunakan untuk membuat *source code.*

```
void setup() {
// semua kode yang ada di disini akan di
baca sekali oleh arduino
}
void loop() {
  //  semua kode yang ada disini akan di baca
berulang kali (terus menerus) oleh arduino
}
```

Void Setup( ) merupakan perintah program yang berarti semua kode program yang ada di dalamnya akan dibaca hanya 1 kali oleh Arduino. Biasanya berisi berupa kode perintah untuk menentukan fungsi dari sebuah pin.

Void loop( ) semua kode yang berada di dalamanya akan di baca setelah **void setup( )** dan akan di baca secara terus menerus oleh arduino.

Kurung kurawal { } merupakan titik awal dan titik akhir suatu pemograman

```
void loop()
{
... isi program / statement
... isi program / statement
}
```

Titik koma “ ; “ merupakan pemisah antar instruksi satu dengan yang lainnya, beberapa statement boleh ditulis dalam 1 baris dipisahkan dengan tanda titik koma “ ; “.

```
void setup (){
pinMode(13, OUTPUT);
}

void loop (){
digitalWrite(13, HIGH);
}
```

pinMode( ) digunakan untuk melakukan konfigurasi secara spesifik fungsi dari sebuah *pin,* apakah digunakan sebagai *input* atau berfungsi sebagai *output.* Seperti contoh program dibawah ini

```
pinMode(0, INPUT); //konfigurasi pin 0 arduino sebagai pin
                     input
pinMode(13, OUTPUT); //konfigurasi pin 13 arduino sebagai pin
```

digitalRead( ) digunakan untuk membaca nilai pin digital secara spesifik, apakah bernilai HIGH atau LOW, terlihat seperti contoh penggunaan *function* **digitalRead( )** dibawahi ini :

```
digitalRead(0); //membaca nilai digital dari pin 0 arduino
```

digitalWrite( ) digunakan untuk membaca nilai digital, selain itu *function* ini juga digunakan untuk menuliskan atau memberikan nilai pada suatu pin digital secara spesifik, *function* **digitalWrite( )** memberikan nilai pin digital yang spesifik bernilai HIGH atau LOW

delay( ) merupakan perintah untuk memberikan jeda waktu sebelum melanjutkan baris perintah program selanjutnya dengan satuan milisecond dengan kata lain **delay(1000);** berarti memberkan jeda waktu selama 1000 *milisecond* atau selama 1 detik, seperti contoh *function* dibawah ini :

```
digitalWrite(13, HIGH); //memberikan nilai digital HIGH pada
                          pin 13 arduino
delay(1000);//memberikan jeda waktu selama 1000ms sebelum
              melanjutkan perintah selanjutnya
digitalWrite(13, LOW); //memberikan nilai digital LOW pada pin
                         13 arduino
```

analogRead( ) digunakan untuk membaca nilai analog melalui pin analog. *Board* Arduino Uno memiliki 6 ch analog, dengan resolusi 10 bit analog to digital converter, dengan resolusi sebesar 10 bit memungkinkan pemetaan tegangan antara 0 volt sampai 5 volt dalam nilai *interger* dari 0 sampai 1023. Sehingga resolusi pembacaan nilai analog ialah 5 volt dibagi 1024 unit, atau sekita 4,9mV per unit. Untuk membaca suatu input analog membutuhkan sekitar 100milisecond, contoh cara penulisannya ialah sebagai beriikut :

```
analogRead(A0); //membaca nilai analog dari pin A0 arduino
```

Serial.print( ) fungsi ini digunakan untuk menuliskan suatu kalimat ke Serial Monitor, tetapi tidak mengirimkan data apapun melainkan hanya berupa tulisan/text visual pada pengguna, cara penulisan perintah serial print ialah sebagai berikut :

```
Serial.print(); // perintah serial interface atau untuk
                   komunikasi serial
```

Serial.begi(bautrate) pengguna dapat melakukan komunikasi serial antara arduino dengan PC ataupun laptop. Dengan menggunakan serial monitor kita bisa mengirimkan data ke

arduino dengan cara mengetikkanya pada textbox serial monitor. Variable baudrate merupakan rasio modulasi dan harus di cocokan dengan baudrate hardware yang di komunikasikan. Seperti terlihat pada contoh di bawah ini:

```
Serial.begin(9600); // komunikasi berjalan pada baudrate 9600
```

Dalam membuat kode sebuah program kita membutuhkan sisipan keterangan dari *source code* yang kita buat, dengan tujuan mempermudah mengetahui posisi perintah-perintah program tertentu yang dianggap penting, dan juga untuk menjelaskan mengenai fungsi dan pin-pin atau *syntax* program yang kita gunakan. Adapun cara penulisan keterangan tersebut ialah dengan cara menggunakan *double slash* // , atau dengan cara untuk pembuka keterangan menggunakan /* dan penutup keterangan */ . Biasanya warna tulisan untuk keterangan itu berwarna abu-abu, dengan terdapat perbedaan warna mempermudah untuk melihat yang mana keterangan dan yang mana bukan.

```
pinMode(0, INPUT); //ini merupakan keterangan atau komentar
```

```
pinMode(0, INPUT);    /* ini komentar atau keterangan
                         ini masih keterangan */
```

Demikian penjelasan singkat mengenai pengenalan arduino dari penjelasan arduino *board*, penginstalasian software Arduino IDE serta penjelasan singkat mengenai perintah-perintah yang digunakan didalam arduino IDE.

Bagian II

Perintah – Perintah Dalam Pemograman Arduino

Dalam penggunaan arduino IDE dan *board* arduino kita juga wajib mengetahui bagaimana cara memprogram arduino *board* tersebut agar dapat digunakan untuk menciptakan suatu karya teknologi yang bs bermafaat bagi kita atau pun orang lain. Dalam bahasa pemograman tersebut menggunakan perintah-perintah yang bisa kita gunakan sesuai dengan fungsinya untuk membuat suatu statement/keadaan sesuai dengan yang kita inginkan, tentunya dalam lingkup bahasa pemograman arduino. Bahasa pemograman arduino tidak terlalu berbeda jauh dengan bahasa pemograman C begitupun perintah-perintah yang digunakan.

2.1 Tipe Data

Tipe data ialah kelompok data berdasarkan jenis-jenis tertentu. Dalam bahasa pemograman, terdapat beberapa tiper data, setiap bahsa pemograman memiliki tipe datanya masing-masing. Beberapa jenis tipe data bisa dilihat tabel di bawah ini :

Type	Ukuran dalam Bits	Range
char	8	- 127 s/d 127
unsigned char	8	0 s/d 255
signed char	8	-127 s/d 127
Int	16	-32,767 s/d 32,767
Unsigned int	16	0 s/d 65,535
Signed int	16	-31,767 s/d 32,767
Short int	16	-31,767 s/d 32,767
Unsigned short int	16	0 s/d 65,535
Signed short int	16	-31,767 s/d 32,767
Long int	32	-2,147,483,647 s/d 2,147,483,647
Long long int	64	$-(2^{63} -1)$ s/d $2^{63} - 1$

Signed long int	32	-2,147,483,647 s/d 2,147,483,647
Unsigned long int	32	0 s/d 4,294,967,295
Unsigned long long int	64	$2^{64} - 1$
Float	32	1E-37 s/d 1E+37
Double	64	1E-37 s/d 1E+37
Long double	80	1E-37 s/d 1E+37

Integer (int) ialah jenis tipe data angka berupa bilangan bulat (bukan pecahan ataupun desimal) contoh : 20, 15, 1, 2 dst.

Float (float) ialah jenis tipe data angka berupa bilangan pecahan atau bilangan desimal. Contoh 15,5 17,9 18,4 dst.

Char (char) tipe data berupa karakter tunggal / *single character* dimana semua karater yang termasuk kedalam tipe data ialah seluruh karakter yang ada pada tabel ASCII (American Standard Code for Information Interchage). Contoh: H, i, 4, $, ^, !. dst.

Boolean (bool) ialah jenis tipe data yang hanya terdiri dari dua nilai saja, yaitu TRUE (benar) dan FALSE (salah) atau bisa dalam bentuk angka 1 untuk TRUE dan angka 0 untuk FALSE.

String (string) adalah tipe data berupa kumpulan dari 1 karakter atau lebih yang biasanya tergabung menjadi suatu kata, frasa, maupun kalimat. Contoh : ”Pemograman”, “Hello World”, “Belajar Basic Arduino”. String bisa berupa frasa, kata ataupun kalimat, kata string biasanya di apit tanda petik untuk menandakan suatu keasatuan.

2.2 Operasi Aritmatika dan Operator Relasi Logika

2.2.1 Operator Aritmatika

Selain tipe data pemograman arduino juga menggunakan operasi aritmatika sebagai operator tambahan yang mempunyai kemampuan atau arti tertentu. Seperti terlihat pada tabel Operator Aritmatika dibawah:

Operator	Keterangan
=	Operator assignment, untuk memberi nilai variable
+	Operator penambahan
-	Operator pengurangan
++	Operator Increment
__	Operator Decrement
*	Operator pengalian
/	Operator pembagian Jika dalam penggunaan integer (int) maka hasil dari pembagian merupakan bilangan bulat. Jika dalam penggunaan float/double maka hasil dari pembagian merupakan bilang desimal.
%	Operator modulus (sisa pembagian)
==	Sama dengan
!=	Tidak sama dengan
<	Lebih kecil
>	Lebih besar
<=	Lebih kecil atau sama dengan
>=	Lebih besar atau sama dengan
<>	Tidak sama dengan

2.2.2. Operator Logika

Operator logika adalah operator yang digunakan untuk membandingkan 2 kondisi logika, yaitu logika benar (TRUE) dan logika salah (FALSE), operator logika sering digunakan untuk kondisi IF, jenis operan dalam operator logika ini ialah jenis tipe variable dengan tipe boolean. Operator yang sering dipakai didalam pemorgraman ialah sbb :

Operator	Keterangan
&&	Operasi Logika AND
| |	Operasi Logika OR
!	Operasi Logika NOT

2.3 Flow Control

Dalam *flow control* atau kontrol aliran ini kita dapat dimungkinkan untuk memanipulasi aliran jalannya program yang kita tulis didalam *skecth* arduino IDE, seperti menyeleksi, memilih, mengulangi, atau melompati suatu pernyataan. Dengan adanya *flow control* memungkinkan kita untuk membuat program yang lebih *fleksible* dan *intractive* untuk kebutuhan pengguna.

Berikut beberapa pernyataan yang dipakai didalam pemograman arduino :

1. ***If***

   Penyataan *if* adalah salah satu pernyataan penyeleksian yang mencari kebenaran dari *conditional expression* yang disebutkan, *conditional expression* harus berupa bilangan *boolean* atau operasi yang menghasilkan bilangan *boolean* dan menyatakan benar (TRUE) atau salah (FALSE) atas ekspresi tersebut.

```
if(Kondisi){
      Pernyataan ;
}
```

```
if(SensorLDR > 600){
      digitalWrite (13, HIGH) ;
}
```

2. ***If – else***

   Pernyataan *if – else* berfungsi untuk melakukan pengetesan atau pengujian kondisi sehingga apabila sebuah kondisi telah terpenuh maka program akan menjalakan pernyataan tersebut, apabila tidak memenuhi kondisi maka program akan melewati pernyataan tersebut.

```
if(Kondisi){
      Pernyataan 1 ;
}
else {
      Pernyataan 2 ;
}
```

```
if(SensorLDR > 600){
      digitalWrite (13, HIGH) ;
}
Else {
      digitalWrite (13, LOW) ;
}
```

3. ***If – else majemuk***

Pernyataan ini memiliki beberapa pengecualian, sehingga apabila dalam kondisi1 program akan menjalankan pernyataan 1, jika tidak cek kondisi 2 jika dalam kondisi 2 jalankan pernyataan 2 jika tidak cek kondisi 3 jika dalam kondisi 3 maka jalankan pernyataan 3, jika tidak ada yang memenuhi kondisi – kondisi tersebut maka program akan menjalan pernyataan 4.

```
if(Kondisi1){
      Pernyataan 1 ;
}
else if(Kondisi2) {
      Pernyataan 2 ;
}
else if(Kondisi3) {
      Pernyataan 3 ;
}
else {
      Pernyataan 4 ;
}
```

```
if(SensorLDR > 600){
      digitalWrite (13, HIGH) ;
}
else if(SensorLDR > 700) {
      digitalWrite (12, HIGH) ;
}
else if(SensorLDR > 800) {
      digitalWrite (11, HIGH) ;
}
else {
      digitalWrite (13, LOW);
      digitalWrite (12, LOW);
      digitalWrite (11, LOW); }
```

4. **For**

   Digunakan apabila membutuhkan perintah pengulangan kode program beberapa kali, namun untuk melakukanya dibutuhkan sebuah *counter*, semisal *counter up* (i++), ataupun *counter down* (i--).

5. **Switch**

```
for(int i = 0 ; i < #pengulangan ; i++){
      Pernyataan ;
}
```

```
for(int i = 0 ; i <3; i++){
      pinMode(ledPin1 [i], OUTPUT) ;
}
```

Digunakan untuk menguji suatu nilai pada variabel dengan konstanta-konstanta tertentu, konstanta tersebut diawali dengan *case* dan di akhiri dengan *break*, jika variabel sama dengan konstanta 1 maka jalankan pernyataan 1, jika variabel sam dengan konstanta 2 maka jalankan pernyataan 2 dan begitu seterusnya, ketika variabel tidak memenuhi konstata manapun maka jalankan default.

```
switch(var){
      case 1 :
            Pernyataan 1
      break;
      case 2 ;
            Pernyataan 2
      break;
      default; }
```

```
switch(pininput){
      case 1 :
            digitalWrite(pinoutput1, HIGH);
      break;
      case 2 ;
            digitalWrite(pinoutput2, HIGH);
      break;
      default;
}
```

6. **While**

Berfungsi ketika ingin menjalankan instruksi yang menjadi syarat atau kondisi secara terus menerus , sampai dengan kondisi tersebut menjadi salah (FALSE).

```
while(kondisi)
{
      pernyataan ;
}
```

```
while(var < 100)
{
var++ ; //pernyataan pengulangan
          hingga 200 kali
}
```

7. **Do – While**

Pernyataan pengulangan *do-while* hampir sama dengan pernyataan *while*, bedanya pada pernyataan *while* kondisi di uji terlebih dahulu dan apabila kondisi bernilai TRUE maka pernyataan akan di eksekusi, namun pada pernyataan pengulangan *do-while* terjadi sebaliknya, pernyataan yang di eksekusi dahulu baru setelah itu kondisinya di uji, apabila kondisi terebut benar atau bernilai TRUE maka pernyataan tersebut akan diulang, jika kondisi tersebut salah, maka program akan keluar dari blok *do-while.*

```
Do {pernyataan utama;}

While (kondisi)
```

```
Do {
      delay (100);
            X = bacasensor();
      }
While (x < 200);
```

8. **Break**

Break digunakan untuk keluar dari *do while* dan *while*, untuk melewati kondisi normal.

```
for (x =0 ; x<255; x++){
      analogWrite (PWMpin, x);
      sensor=analogRead (sensor);
if (sensor > threshold){
      x =0;
      break;
      }
      delay (50);
      }
```

Bagian 3

Pengertian Ultrasonic Sensor

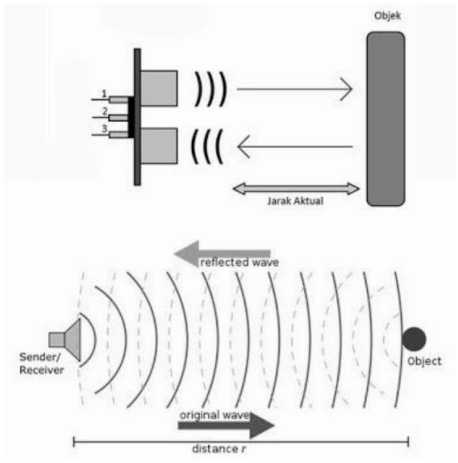
Sensor Ultrasonik adalah alat elektronika yang kemampuannya bisa mengubah dari energi listrik menjadi energi mekanik dalam bentuk gelombang suara ultrasonik. Sensor ini terdiri dari rangkaian pemancar Ultrasonik yang dinamakan transmitter dan penerima ultrasonik yang disebut receiver. Alat ini digunakan untuk mengukur gelombang ultrasonik. Gelombang ultrasonik adalah gelombang mekanik yang memiliki cirri-ciri longitudinal dan biasanya memiliki frekuensi di atas 20 Khz. Gelombong Utrasonik dapat merambat melalui zat padat, cair maupun gas. Gelombang Ultrasonik adalah gelombang rambatan energi dan momentum mekanik sehingga merambat melalui ketiga elemen tersebut sebagai interaksi dengan molekul dan sifat enersia medium yang dilaluinya.

Ada beberapa penjelasan mengenai gelombang ultrasonik. Sifat dari gelombang ultrasonik yang melalui medium menyebabkan getaran partikel dengan medium aplitudo sama dengan arah rambat longitudinal sehingga menghasilkan partikel medium yang membentuk suatu rapatan atau biasa disebut Strain dan tegangan yang biasa disebut Strees. Proses lanjut yang menyebabkan terjadinya rapatan dan regangan di dalam medium disebabkan oleh getaran partikel secara periodic selama gelombang ultrasonik lainnya. Gelombang ultrasonik merambat melalui udara dengan kecepatan 344 meter per detik, mengenai obyek dan memantul kembali ke sensor ultrasonik. Seperti yang telah umum diketahui, gelombang ultrasonik hanya bisa didengar oleh makhluk tertentu seperti kelelawar dan ikan paus. Kelelawar menggunakan gelombang ultrasonik untuk berburu di malam hari sementara paus menggunakanya untuk berenang di kedalaman laut yang gelap.

Cara Kerja Ultrasonic Sensor

Pada sensor ultrasonik, gelombang ultrasonik dibangkitkan melalui sebuah alat yang disebut dengan piezoelektrik dengan frekuensi tertentu. Piezoelektrik ini akan menghasilkan

gelombang ultrasonik (umumnya berfrekuensi 40kHz) ketika sebuah osilator diterapkan pada benda tersebut. Secara umum, alat ini akan menembakkan gelombang ultrasonik menuju suatu area atau suatu target. Setelah gelombang menyentuh permukaan target, maka target akan memantulkan kembali gelombang tersebut. Gelombang pantulan dari target akan ditangkap oleh sensor, kemudian sensor menghitung selisih antara waktu pengiriman gelombang dan waktu gelombang pantul diterima.



Gambar cara kerja sensor ultrasonik dengan transmitter dan receiver (atas), sensor ultrasonik dengan single sensor yang berfungsi sebagai transmitter dan receiver sekaligus Secara detail, cara kerja sensor ultrasonik adalah sebagai berikut:

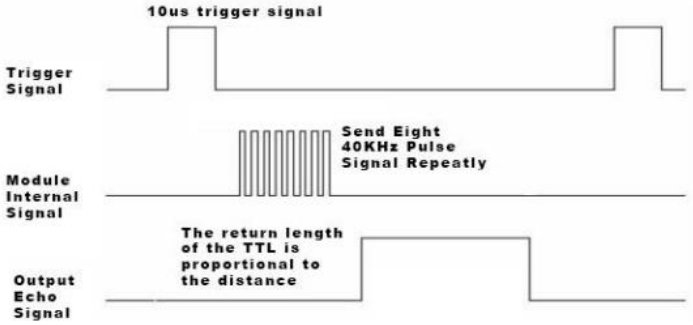
1. Sinyal dipancarkan oleh pemancar ultrasonik dengan frekuensi tertentu dan dengan durasi waktu tertentu. Sinyal tersebut berfrekuensi diatas 20kHz. Untuk mengukur jarak benda (sensor jarak), frekuensi yang umum digunakan adalah 40kHz.
2. Sinyal yang dipancarkan akan merambat sebagai gelombang bunyi dengan kecepatan sekitar 340 m/s. Ketika menumbuk suatu benda, maka sinyal tersebut akan dipantulkan oleh benda tersebut.
3. Setelah gelombang pantulan sampai di alat penerima, maka sinyal tersebut akan diproses untuk menghitung jarak benda tersebut. Jarak benda dihitung berdasarkan rumus : $S = 340.t/2$ dimana S merupakan jarak antara sensor ultrasonik dengan benda (bidang pantul), dan t adalah selisih antara waktu pemancaran gelombang oleh transmitter dan waktu ketika gelombang pantul diterima receiver.

HC-SR04

Sensor ini merupakan sensor ultrasonik siap pakai, satu alat yang berfungsi sebagai pengirim, penerima, dan pengontrol gelombang ultrasonik. Alat ini bisa digunakan untuk mengukur jarak benda dari 2cm - 4m dengan akurasi 3mm. Alat ini memiliki 4 pin, pin Vcc, Gnd, Trigger, dan Echo. Pin Vcc untuk listrik positif dan Gnd untuk ground-nya. Pin Trigger untuk trigger keluarnya sinyal dari sensor dan pin Echo untuk menangkap sinyal pantul dari benda.

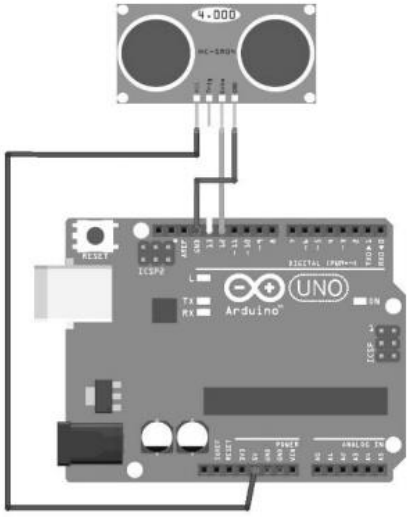
Cara menggunakan alat ini yaitu: ketika kita memberikan tegangan positif pada pin Trigger selama 10uS, maka sensor akan mengirimkan 8 step sinyal ultrasonik dengan frekuensi 40kHz. Selanjutnya, sinyal akan diterima pada pin Echo. Untuk mengukur jarak benda yang memantulkan sinyal tersebut, maka selisih waktu ketika mengirim dan menerima sinyal digunakan untuk menentukan jarak benda tersebut. Rumus untuk menghitungnya sudah saya sampaikan di atas.

Berikut adalah visualisasi dari sinyal yang dikirimkan oleh sensor HC-SR04:



Visualisasi Sinyal Sensor Ultrasonik Sensor

Rangkaian Ultrasonic Sensor



Ilustrasi Rangkaian

Setelah meyelesaikan rangkaian pada sensor ultrasonik dan Arduino. Selanjutnya kita akan membuat program untuk mengambil data jarak yang terbaca oleh sensor ultrasonik. Langkah-langkah pembuatan program adalah sebagai berikut:

Buka Software Arduino

Buatlah variabel yang mewakli pin sensor yang telah terpasang pada pin output dan input logic Arduino. Tentukan pula fungsi masing-masing pinnya.

```
#define trigger 13
#define echo 12

void setup() {
  // put your setup code here, to run once:

  Serial.begin(9600);
  pinMode(trigger, OUTPUT);
  pinMode(echo, INPUT);

}
```

Pada void loop(), masukkan perintah yang akan dikerjakan Arduino untuk mengambil data jarak yang terbaca oleh sensor sesuai dengan cara kerja dan karakteristik sinyal yang dihasilkan oleh sensor.

```
void loop() {
  // put your main code here, to run repeatedly:

  long duration, distance, pulse;
  digitalWrite(trigger, LOW);
  delayMicroseconds(2);
  digitalWrite(trigger, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigger, LOW);

  duration = pulseIn(echo, HIGH);
  distance = (duration/58);

  Serial.print(distance);
  Serial.println(" cm");
  delay(100);

}
```

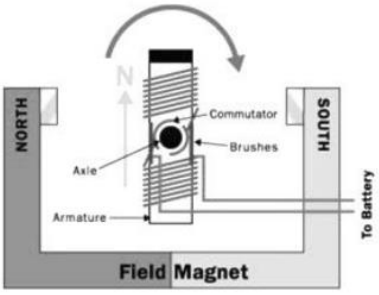
Kemudian data akan ditampilkan pada fitur Arduino yaitu serial monitor dengan cara klik icon pada software Arduino tersebut.

Pengertian Motor DC

Motor DC adalah piranti elektronik yang mengubah energi listrik menjadi energi mekanik berupa gerak rotasi. Pada motor DC terdapat jangkar dengan satu atau lebih kumparan terpisah. Tiap kumparan berujung pada cincin belah (komutator). Dengan adanya insulator antara komutator, cincin belah dapat berperan sebagai saklar kutub ganda (double pole, double throw switch). Motor DC bekerja berdasarkan prinsip gaya Lorentz, yang menyatakan ketika sebuah konduktor beraliran arus diletakkan dalam medan magnet, maka sebuah gaya (yang dikenal dengan gaya Lorentz) akan tercipta secara ortogonal diantara arah medan magnet dan arah aliran arus.

Motor DC yang digunakan pada robot beroda umumnya adalah motor DC dengan magnet permanen. Motor DC jenis ini memiliki dua buah magnet permanen sehingga timbul medan magnet di antara kedua magnet tersebut. Di dalam medan magnet inilah jangkar/rotor berputar. Jangkar yang terletak di tengah motor memiliki jumlah kutub yang ganjil dan pada setiap kutubnya terdapat lilitan. Lilitan ini terhubung ke area kontak yang disebut komutator.

Sikat (brushes) yang terhubung ke kutub positif dan negatif motor memberikan daya ke lilitan sedemikian rupa sehingga kutub yang satu akan ditolak oleh magnet permanen yang berada di dekatnya, sedangkan lilitan lain akan ditarik ke magnet permanen yang lain sehingga menyebabkan jangkar berputar. Ketika jangkar berputar, komutator mengubah lilitan yang mendapat pengaruh polaritas medan magnet sehingga jangkar akan terus berputar selama kutub positif dan negatif motor diberi daya.



Driver Motor H-Bridge

Driver motor yang kita pakai menggunakan konfigurasi jembatan H (H-Bridge),yang akan mengendalikan motor ke dua arah, searah jarum jam (clockwise) dan berlawanan arah jarum jam counter clockwise). Secara konsep rangkaian ini terdiri dari 4 saklar yang tersusun sedemikian rupa sehingga memungkinkan motor dapat teraliri arus dengan arah yang berkebalikan. Pemberian

polaritas tegangan pada terminal motor akan mempengaruhi arah arus yang melewati motor,dengan demikian motor akan berputar sesuai dengan arah arusnya.

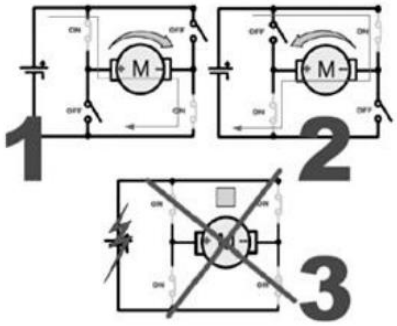
Pada rangkaian driver motor ini, saklar-saklar tersebut digantikan oleh transistor yang dikerjakan pada daerah saturasi dan cut-off (Switch). Bagaimana cara mengatur arah putar motor DC? Untuk mengatur arah putar motor DC cara yang paling mudah adalah membalik polaritas tegangan yang kita berikan pada terminal motor tersebut.

Bagaimana H-Bridge bekerja?

1. Ketika S1 dan S4 tertutup (diagonal) dan lainnya terbuka maka arus akan mengalur dari batery ke kutub positif motor kemudian keluar ke kutub negatif motor,makamotor akan berputar kearah kanan.

2. Ketika S2 dan S3 tertutup (diagonal) dan lainnya terbuka,maka arus akan mengalir sebaliknya,motor juga akan berputar kearah sebaliknya.

3. Jika semua saklar tertutup, maka motor akan berhenti, dan jika ini diteruskan maka akan menyebabkan rangkaian menjadi”short circuit“.



Mengontrol Arah Putaran Motor DC dengan Arduino

Seperti telah dijelaskan sebelumnya, untuk mengendalikan motor dc, Arduino akan memberikan tiga buah logic output sebagai pengatur arah putaran dan kecepatan motor dc. Berikut langkah-langkah pembuatan program untuk mengontrol arah putaran dan kecepatan motor dc menggunakan Arduino.

Pada bagian atas program daftarkan variabel-variabel data untuk digunakan sebagai nilai analog PWM motor, kemudian pada void setup(), masukkan konfigurasi tombol dan output logic sesuai dengan rangkaian yang telah dibuat seperti gambar dibawah

```
int sensorPin = A0;
int sensorValue = 0;

void setup() {
  // put your setup code here, to run once:

  pinMode(2, OUTPUT);
  pinMode(3, OUTPUT);
  pinMode(4, OUTPUT);
  pinMode(7, INPUT_PULLUP);
  pinMode(8, INPUT_PULLUP);
```

Pin output diatas digunakan sebagai input logic yang akan diterima oleh driver motor sebagai perintah untuk menjalankan motor dc.

Kemudian tambahkan nilai analog output PWM pada void-void gerakan motor yang telah dibuat seperti pada materi sebelumnya

```
void clockwise()
{
   digitalWrite(2, HIGH);
   digitalWrite(4, LOW);
   analogWrite(3, sensorValue);

}
void cclockwise()
{
   digitalWrite(2, LOW);
   digitalWrite(4, HIGH);
   analogWrite(3, sensorValue);

}
```

```
void freestop()
{
   digitalWrite(2, LOW);
   digitalWrite(4, LOW);
   analogWrite(3, sensorValue);

}
void faststop()
{
   digitalWrite(2, HIGH);
   digitalWrite(4, HIGH);
   analogWrite(3, sensorValue);

}
```

Pada void loop(), masukkan perintah-perintah gerakan motor berdasarkan input dari tombol dan potensiometer.

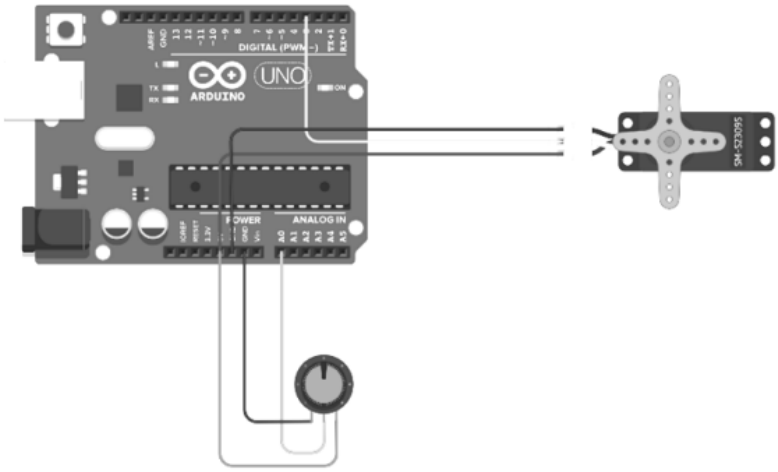
```
void loop() {
  // put your main code here, to run repeatedly:

sensorValue = analogRead(sensorPin);

  int sensorVal1 = digitalRead(7);
  int sensorVal2 = digitalRead(8);

  if (sensorVal1 == LOW) {
  clockwise();
 }
  else if (sensorVal2 == LOW) {
  cclockwise();
 }
  else {
  freestop();
 }

}
```

Pengertian Motor Servo

Motor servo adalah sebuah perangkat atau aktuator putar (motor) yang dirancang dengan sistem kontrol umpan balik loop tertutup (servo), sehingga dapat di set-up atau di atur untuk menentukan dan memastikan posisi sudut dari poros output motor. motor servo merupakan perangkat yang terdiri dari motor DC, serangkaian gear, rangkaian kontrol dan potensiometer. Serangkaian gear yang melekat pada poros motor DC akan memperlambat putaran poros dan meningkatkan torsi motor servo, sedangkan potensiometer dengan perubahan resistansinya saat motor berputar berfungsi sebagai penentu batas posisi putaran poros motor servo.

Penggunaan sistem kontrol loop tertutup pada motor servo berguna untuk mengontrol gerakan dan posisi akhir dari poros motor servo. Penjelasan sederhananya begini, posisi poros output akan di sensor untuk mengetahui posisi poros sudah tepat seperti yang di inginkan atau belum, dan jika belum, maka kontrol input akan mengirim sinyal kendali untuk membuat posisi poros tersebut tepat pada posisi yang diinginkan. Untuk lebih jelasnya mengenai sistem kontrol loop tertutup, perhatikan contoh sederhana beberapa aplikasi lain dari sistem kontrol loop tertutup, seperti penyetelan suhu pada AC, kulkas, setrika dan lain sebagainya.

Rangkaian Motor Servo



Mengendalikan Sudut Putaran Motor Servo menggunakan Arduino

Setelah membuat rangkaian Motor Servo, langkah selanjutnya adalah pembuatan program agar arduinodapat mengendalikan posisi sudut putaran . Langkah-langkah pembuatan program adalah sebagai berikut:

Buka Software Arduino

Buatlah program seperti ini:

```
#include <Servo.h>

Servo myservo;   // create servo object to control a servo

int potpin = 0;  // analog pin used to connect the potentiometer
int val;         // variable to read the value from the analog pin

void setup() {
  myservo.attach(3);  // attaches the servo on pin 3 to the servo object
}

void loop() {
  val = analogRead(potpin);            // reads the value of the potentiometer (value between 0 and 1023)
  val = map(val, 0, 1023, 0, 180);     // scale it to use it with the servo (value between 0 and 180)
  myservo.write(val);                  // sets the servo position according to the scaled value
  delay(15);                           // waits for the servo to get there
}

```

Program diatas bertujuan untuk mengatur sudut putaran sebuah motor servo berdasarkan sudut putaran pada poteniometer.