

# Arduino UNO R3 Starter Kit



Building with the “Arduino UNO R3 Starter Kit”

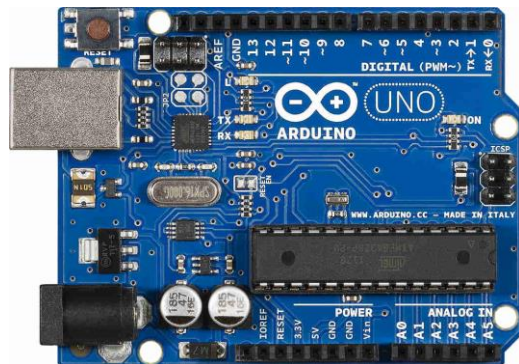
# Bagian 1. Pengenalan Arduino

## 1.1 Apa itu Mikrokontroler?

Menurut wikipedia:

*A microcontroller (sometimes abbreviated  $\mu C$ ,  $uC$  or  $MCU$ ) is a small computer on a single integrated circuit containing a processor core, memory, and programmable input/output peripherals.*

Dalam diskusi sehari-hari dan di forum internet, mikrokontroler sering dikenal dengan sebut  $\mu C$ ,  $uC$ , atau  $MCU$ . Terjemahan bebas dari pengertian tersebut, bisa dikatakan bahwa mikrokontroler adalah komputer yang berukuran mikro dalam satu chip IC (integrated circuit) yang terdiri dari processor, memory, dan antar muka yang bisa diprogram. Jadi disebut komputer mikro karena dalam IC atau chip mikrokontroler terdiri dari CPU, memory, dan I/O yang bisa kita kontrol dengan memprogramnya. I/O juga sering disebut dengan GPIO (*General Purpose Input Output Pins*) yang berarti : pin yang bisa kita program sebagai input atau output sesuai kebutuhan.

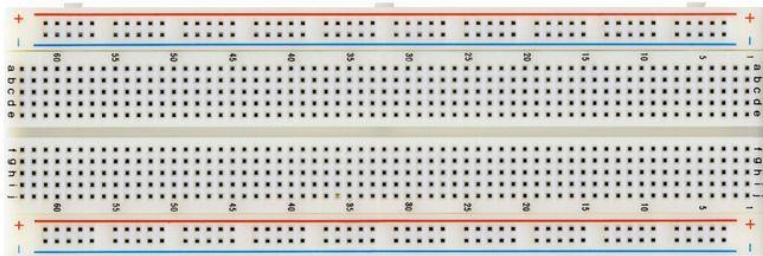


Gambar 1.1 Board Arduino Uno

Dalam ebook ini kita akan menggunakan *board* Arduino Uno (Gambar 1.1). Board Arduino terdiri dari hardware / modul

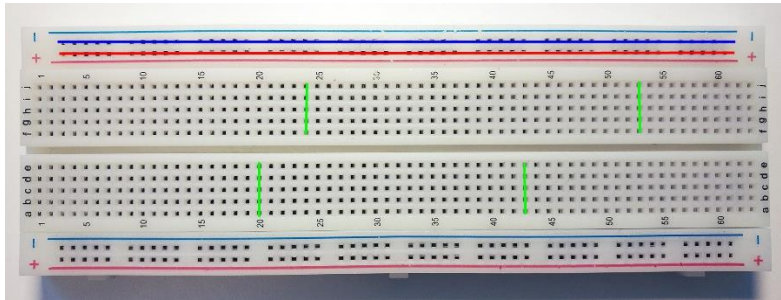
mikrokontroler yang siap pakai dan *software* IDE yang digunakan untuk memprogram sehingga kita bisa belajar dengan mudah. Kelebihan dari Arduino yaitu kita tidak direpotkan dengan rangkaian minimum sistem dan *programmer* karena sudah *built in* dalam satu *board*, oleh sebab itu kita bisa fokus ke pengembangan sistem.

Board Arduino memiliki 20 GPIO, meliputi 14 pin digital (0-13) dan 6 pin analog (A0-A5) yang bisa kita gunakan untuk keperluan pemrograman, dari 14 pin digital terdapat 2 pin serial RX dan TX yaitu pin 0 dan 1 serta 6 pin PWM yaitu pin 3, 5, 6, 9, 10 dan 11, pin PWM ditandai dengan tanda ~ pada board.



Gambar 1.2 Project board

Untuk praktek, kita akan menggunakan *project board* (ada yang menyebutnya dengan istilah *bread board*) (Gambar 1.2). Dengan *project board* kita tidak perlu menyolder rangkaian sehingga relatif mudah dan cepat dalam merangkai. *Project board* memungkinkan kita untuk membangun dan membongkar rangkaian dengan cepat sehingga sangat cocok untuk eksperimen. Tapi jika kita ingin membuat rangkaian yang permanen, maka kita harus menggunakan PCB.



### Gambar 1.3 Peta jalur pada project board

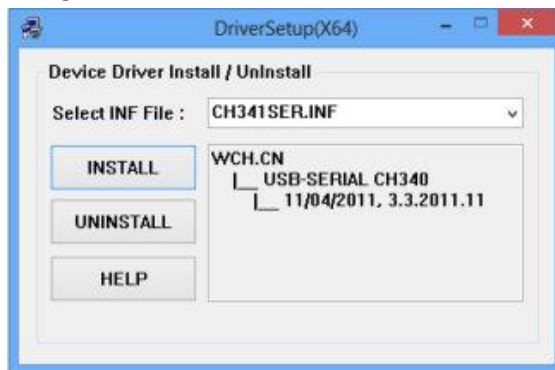
Yang terpenting adalah, kita harus memahami jalur-jalur pada *project board*. *Project board* terdiri dari jalur vertikal dan jalur horisontal. Jalur vertikal ada di bagian tengah yang terdiri dari 2 x 64 jalur. Masing- masing jalur terdiri dari 5 titik vertikal, misal jalur 1A- 1B- 1C- 1D-1E dan jalur 1F-1G-1H-1I-1J yang kedua tidak saling tersambung. Jalur horisontal sebanyak 8 jalur, 4 jalur ada di bagian atas dan 4 jalur lagi di bagian bawah. Jalur ini bisa digunakan untuk power supply (VCC dan GND) untuk rangkaian. Untuk lebih jelasnya, silakan perhatikan Gambar 1.3. Garis merah, biru dan hijau menunjukkan bahwa lubang tersebut terhubung secara fisik.

## 1.2 Instalasi Arduino IDE

Anda bisa mendownload Arduino IDE di website Arduino atau software yang telah disediakan di paket CD. Software Arduino ada yang versi installer (hanya untuk Windows) dan versi terkompres dalam zip. Jika memilih versi tanpa install (format .zip), maka Anda hanya perlu mengekstraknya di folder mana saja dan Anda bisa langsung menjalankannya. Untuk pengguna Windows, Anda bisa menginstall dengan mengikuti instruksi dalam ebook ini.

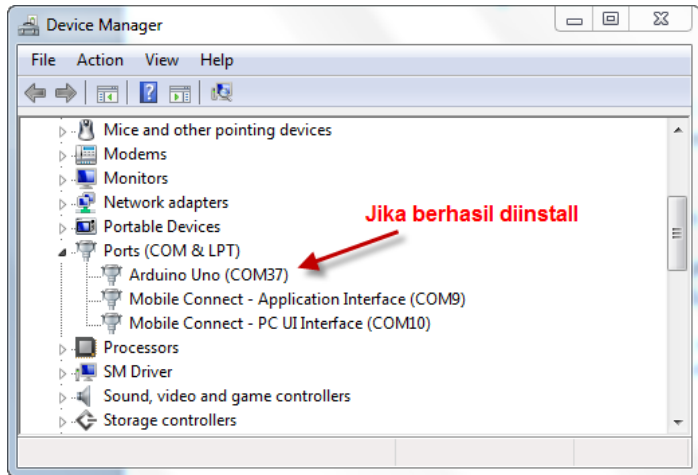
### 1.2.1 Instalasi di Windows

1. Pasang board Arduino Anda ke port USB pada komputer atau laptop, kemudian tunggu hingga Windows mencoba untuk menginstall driver sendiri. Biasanya dia gagal menginstall *driver* jika belum memiliki *driver* tersebut. (Silakan lanjutkan ke step berikutnya)
2. Install Software Arduino IDE yang ada di paket CD.
3. Jika anda menggunakan Arduino Versi SMD maka masuk dulu ke step ini, jika menggunakan Arduino versi DIP silahkan lewati.
  - Extract file CH340.rar yang ada di paket CD di folder driver dan software
  - Lalu instal seperti file biasa, maka akan tampil seperti gambar di bawah ini.



Gambar 1.4 Instalasi Driver CH340

4. Klik install lalu tunggu hingga selesai.
5. Jika berhasil, dan arduino terinisialisasi berarti instalasi selesai. Maka ketika membuka device manager akan tampil port arduino seperti berikut



Gambar 1.5 Device Manager

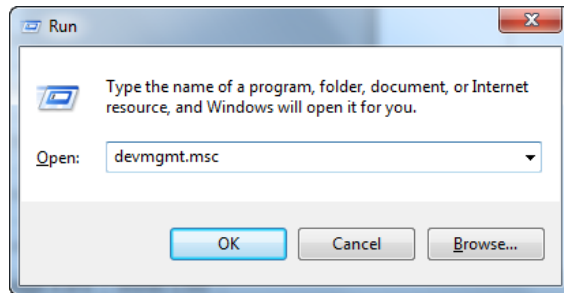
6. Jika berhasil, berarti instalasi selesai. Tapi jika gagal, lanjutkan ke step selanjutnya.
7. Anda harus install dari *device manager*. Untuk masuk ke *device manager*, Anda bisa melakukannya dengan dua cara:



Gambar 1.4 Posisi tombol Windows

- Tekan tombol ("Windows" + R) secara bersamaan. Tombol

"Windows" adalah tombol pada keyboard dengan logo Windows (gambar logo windows, biasanya terletak di sebelah kiri atau kanan spasi, lihat Gambar 1.4). Setelah Anda menekan tombol "Windows" + R, maka akan muncul "Run", ketikkan "*devmgmt.msc*" (tanpa tanda petik), kemudian tekan tombol ENTER. Jika benar, maka akan muncul window *Device Manager*.



Gambar 1.6 Window yang muncul setelah menekan  
(Windows + R)

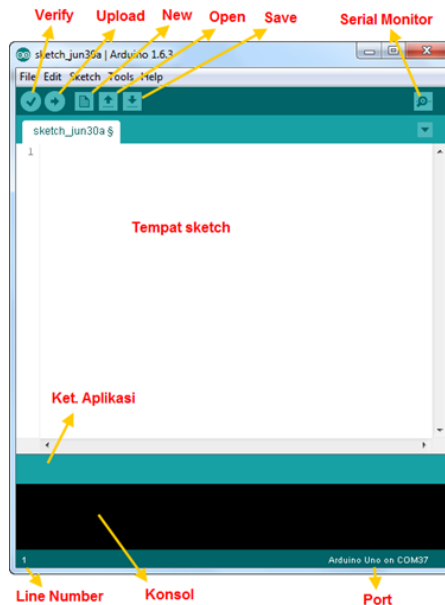
- Jika *Device Manager* Anda sudah keluar, Anda bisa lanjut ke point 4, jika tidak, coba cara berikut untuk menampilkan *device manager*
  - Klik Start - pilih *Control Panel*. Di dalam Control Panel, pilih *System and Security*, lalu pilih *System*. Selanjutnya pilih *Device Manager*.
8. Pada *Device Manager*, perhatikan bagian Ports (COM & LPT), akan muncul device baru dengan nama "Arduino UNO (COMxx)"
  9. Klik kanan pada "Arduino UNO (COMxx)", kemudian pilih "*Update Driver Software*".
  10. Selanjutnya pilih "*Browse my computer for Driver software*".
  11. Cara folder software Arduino Anda, kemudian cari file *arduino.inf* (khusus untuk Arduino UNO REF.3) pada folder Drivers. Jika Anda menggunakan versi IDE di bawah 1.0.3, Anda bisa memilih *driver* dengan nama file *ArduinoUNO.inf*

12. Jika berhasil, berarti instalasi *driver* sudah selesai.
13. Selanjut mari kita coba untuk mengupload sampel code yang ada pada *software* Arduino
14. Jalankan Aplikasi Arduino (arduino.exe), pada pojok kanan bawah akan ada tulisan "Arduino UNO on COMxx". Berarti port yang digunakan Arduino adalah COMxx, jika tulisan tersebut tidak muncul, berarti instalasi *driver* belum berhasil atau board Arduino belum disambungkan ke komputer. Selanjutnya, silakan buka sampel led flip-flop dengan cara Klik menu *File > Examples > 1.Basic > Blink*
15. Setting board Arduino dengan cara : Klik menu *Tools > Board > Arduino UNO*
16. Pilih port yang digunakan Arduino dengan cara mengklik menu *Tools > Ports > (pilih yang ada Arduino-nya)*
17. Klik tombol upload (tombol dengan panah ke kanan)
18. Setelah berhasil diupload, akan muncul tulisan "*Done uploading*" di bagian bawah. Jika berhasil, maka LED dengan tulisan "L" pada board Arduino akan kelap-kelip.

### 1.3 Arduino IDE

Untuk memprogram board Arduino, kita butuh aplikasi IDE (Integrated Development Environment) bawaan dari Arduino. Aplikasi ini berguna untuk membuat, membuka, dan mengedit source code Arduino (Sketches, para programmer menyebut source code arduino dengan istilah "sketches"). Selanjutnya, jika kita menyebut source code yang ditulis untuk Arduino, kita sebut "sketch". Sketch merupakan *source code* yang berisi logika dan algoritma yang akan diupload ke dalam IC mikrokontroler (Arduino).





Gambar 1.7 Interface Arduino IDE

Interface Arduino IDE tampak seperti gambar 1.7. Dari kiri ke kanan dan atas ke bawah, bagian-bagian IDE Arduino terdiri dari:

- **Verify** : pada versi sebelumnya dikenal dengan istilah *Compile*. Sebelum aplikasi diupload ke *board* Arduino, biasanya untuk memverifikasi terlebih dahulu *sketch* yang dibuat. Jika ada kesalahan pada *sketch*, nanti akan muncul error. Proses Verify / *Compile* mengubah *sketch* ke *binary code* untuk diupload ke mikrokontroler.
- **Upload** : tombol ini berfungsi untuk mengupload *sketch* ke *board* Arduino. Walaupun kita tidak mengklik tombol *verify*, maka *sketch* akan di-*compile*, kemudian langsung diupload ke *board*. Berbeda dengan tombol *verify* yang hanya berfungsi untuk memverifikasi *source code* saja.
- **New Sketch** : Membuka window dan membuat *sketch* baru
- **Open Sketch** : Membuka *sketch* yang sudah pernah dibuat. *Sketch* yang dibuat dengan IDE Arduino akan disimpan dengan ekstensi file

## **.ino**

- **Save Sketch** : menyimpan *sketch*, tapi tidak disertai mengcompile.
- **Serial Monitor** : Membuka *interface* untuk komunikasi serial, nanti akan kita diskusikan lebih lanjut pada bagian selanjutnya
- **Keterangan Aplikasi** : pesan-pesan yang dilakukan aplikasi akan muncul di sini, misal "*Compiling*" dan "*Done Uploading*" ketika kita mengcompile dan mengupload *sketch* ke *board* Arduino
- **Konsol** : Pesan-pesan yang dikerjakan aplikasi dan pesan-pesan tentang *sketch* akan muncul pada bagian ini. Misal, ketika aplikasi mengcompile atau ketika ada kesalahan pada *sketch* yang kita buat, maka informasi *error* dan baris akan diinformasikan di bagian ini.
- **Baris Sketch** : bagian ini akan menunjukkan posisi baris kursor yang sedang aktif pada *sketch*.
- **Informasi Port** : bagian ini menginformasikan *port* yang dipakai oleh *board* Arduino.

## Bagian 2. Arduino Programming

### 2.1 Addition, Subtraction, Multiplication, & Division

Operasi ini memberikan hasil penambahan, pengurangan, pembagian dan perkalian dari dua operan. Operasi adalah pengaturan menggunakan dua tipe data dalam operan seperti 9/4 adalah 2 dimana 9 dan 4 adalah integer. Selain itu bisa juga diartikan operasi tersebut merupakan dapat menangani hasil yang mempunyai *range* dalam sebuah tipe data ( 1 integer mempunyai nilai 32,767 sampai -32,768). Jika beberapa operan mempunyai perbedaan tipe data yang “besar” yang digunakan untuk perhitungan. Salah satu operan yang digunakan adalah tipe *float* atau tipe *double*, floating point dapat digunakan untuk penghitungan

#### Contoh

```
y = y + 3;
x = x - 7;
i = j * 6;
r = r / 5;
```

#### Syntax

```
result = value1 + value2; result = value1
- value2; result = value1 * value2;
result = value1 / value2;
```

#### Parameter

Value1 : merupakan variable atau konstan

Value2 : merupakan variable atau konstan

#### Tips pemrograman :

- Ketahui bahwa integer constan default untuk **int** sehingga beberapa konstanta penghitungan bisa overflow (60\*1000 akan menghasilkan hasil yang negative)
- Pilih besaran variable yang besarnya cukup untuk menangani hasil dari penghitungan anda
- Ketahui bahwa tanda operasi bilangan yang akan Anda gunakan harus langsung sesuai arah matematika contoh 0-1 atau 0 - 32768

- Untuk pemrograman yang memerlukan pecahan, gunakanlah variabel *float*, tapi perlu disadari hal ini mempunyai kelemahan dalam memiliki ukuran yang besar dan kecepatan komputasi yang rendah
- Gunakanlah operasi pengalihan seperti (int)myFloat untuk merubah sebuah variable menjadi variable lain yang belum kita ketahui.

## 2.2 Assignment Operator (Single Equalsign)

Menyimpan nilai variable sebelah kanan ke dalam variable sebelah kiri. Single equal sign dalam pemrograman C adalah pemanggilan operator tugas (assignment operator). Hal ini berbeda dengan kelas aljabar yang mengindikasikan persamaan atau penyamaan. Operator penugasan ini memberitahu mikrokontroler untuk mengevaluasi nilai apapun atau ekspresi yang berada di sebelah kanan dan menyimpannya ke dalam variable sebelah kiri.

### Contoh

```
int senVal; // mendeklarasikan senVal sebagai variable
integer
senVal = analogRead(0); //menyimpan (digitasi) voltase
yang masuk pada analog pin ke 0 dalam sensVal
```

### Tips pemrograman

Variable yang ada di sebelah kiri dari assignment operation (= sign) dibutuhkan untuk digunakan menjaga nilai yang sudah tersimpan di dalamnya. Jika hal itu tidak cukup besar untuk menangani sebuah nilai, nilai yang sudah tersimpan di variable tersebut akan menjadi salah. Jangan terkecoh dengan assignment operator [=] (sama dengan) dengan operator komparasi [==] (dobel sama dengan) yang mengevaluasi dua ekspresi menjadi hasil.

### 2.2.1 TIPE DATA

#### 1. Integer (int)

*Integer* merupakan tipe data yang utama dari macam penyimpanan dan terdiri dari 2 nilai byte. Mempunyai *range* dari -32,768 sampai 32,762 (nilai minimal adalah  $-2^{15}$  dan nilai maksimal  $(2^{15}-1)$ )

### Contoh

```
int ledPin = 13;
```

### Syntax

```
int var = val;
```

var – nama variable integer Anda

val – nilai yang diberikan dalam variable Anda

### Tips pemrograman

Saat beberapa variable dibuat untuk melebihi kapasitas maksimal, maka nilai tersebut akan membalik ke minimal kapasitas, begitu juga sebaliknya.

```
int x
x = -32,768;
x = x - 1; // x sekarang menjadi 32,767-berbalik ke arah
negatif.
```

```
x = 32,767;
x = x + 1; // x sekarang menjadi 32,768-berbalik ke arah
positif
```

## 2. Unsigned Int

Unsigned int sama seperti **int**, namun dalam pemberian angka negative tidak diperkenankan, sebagai gantinya nilai negative digantikan nilai positif sehingga mempunyai range 0 sampai 65,535 ( $2^{16}-1$ ).

### Contoh

```
unsigned int ledPin = 13;
```

### Syntax

```
unsigned int var = val;
```

var – nama variable unsigned Anda

val – nilai yang berada dalam variable tersebut.

### Tips pemrograman

Ketika membuat variable yang melebihi kapasitas, maka akan membalik ke

kapasitas minimum

```
unsigned int x x = 0;  
x = x - 1; // x sekarang          65535 x = x + 1; // x  
sekarang 0
```

### 3. Word

Sebuah word menyimpan 16 bit angka dari 0 sampai 65535 sama seperti unsigned int

#### Contoh

```
word w = 10000;
```

### 4. Byte

byte menyimpan sebuah 8-bit dari 0 samapi 255

#### Contoh

```
byte b = B10010;    // "B" is the binary formatter (B10010  
= 18 decimal)
```

### 5. Float

Tipe data untuk angka floating point, sebuah angka yang mempunyai titik decimal. *Floating-point* sering digunakan pada nilai yang mendekati analog dan nilai yang berkesinambungan karena mempunyai resolusi yang lebih tinggi dari pada integer. *Floating-point* mempunyai besar  $3.4028235E+38$  dan  $-3.4028235E+38$ . Sehingga dapat menyimpan 32bit (4byte. ) informasi.

Float hanya mempunyai kepresisian 6 – 7 digit decimal. Yang maksudnya total angka dari digitnya bukan angka dari titik kanan setelah decimal. Tidak seperti platform, dimana Anda dapat mendapatkan akurasi dari penggunaan tipe data double (sampai 15 digit) dalam arduino, double adalah besarnya sama seperti float.

Angka *Floating-point* belum tentu tepat, dan bisa menghasilkan hasil yang tidak biasa saat di komparasi.

Seperti misalkan  $6.0 / 3.0$  hasilnya mungkin bukan 2.0 Anda harus mengganti

dan mengecek nya karena nilai absolute berbeda dengan angka yang memiliki ketelitian.

### Contoh

```
float myfloat;  
float sensorCalbrate = 1.117;
```

### Syntax

```
float var = val;  
var – nama variable anda val – nilai yang diberikan
```

### Contoh Kode

```
int x; int y; float z;  
x = 1;  
y = x / 2; // y sekarang 0, int tidak dapat menganinya  
z = (float)x / 2.0; // z sekarang .5 (Anda memiliki hasil  
2.0, bukan 2)
```

## 6. Double

Dalam Arduino, Double sama seperti float namun memiliki kepresisian yang sedikit berbeda.

### Tips Pemograman

Pengguna yang membawa kode dari *source code* dari luar yang didalamnya terdapat variable double lebih baik diuji terlebih dahulu untuk mengetahui perbedaan ketepatan yang terdapat pada Arduino.

## 7. Long

Variable long adalah variable yang mempunyai penyimpanan angka yang lebar yaitu 32 bit (4byte) dari

-2,147,483,648 sampai 2,147,487,483,647

### Contoh

```
long speedOfLight = 186000L; // see Integer Constants for  
explanation of the 'L'
```

### Syntax

```
long var = val;
```

## 8. Unsigned Long

Unsigned Long adalah variable yang mempunyai lebar 32bit dan tidak seperti **long**, **unsigned long** tidak dapat menyimpan angka negative, mempunyai range antara 0 sampai 4,294,967,295 ( $2^{32} - 1$ ).

### Contoh

```
unsigned long time;
void setup()
{
  Serial.begin(9600);
}
void loop()
{
  Serial.print("Time: "); time = millis();
  //mencetak waktu saat program dijalankan
  Serial.println(time);
  // besaran waktu tunggu saat tidak mengirim sejumlah
  data delay(1000);
}
```

### Syntax

```
unsigned long var = val;
```

## 9. Boolean

Boolean menangani dua nilai yaitu *true* atau *false* sedangkan variable menempati satu byte dari memori.

### Contoh

```
int LEDpin = 5; // LED pada pin 5 int
switchPin = 13;
/*saat switch terpasang pada pin 13, yang lain terkoneksi
pada ground*/
boolean running = false;
void setup()
{
  pinMode(LEDpin, OUTPUT);
  pinMode(switchPin, INPUT);
  digitalWrite(switchPin, HIGH);
  // menghidupkan pullup resistor
}
```



```

void loop()
{
if (digitalRead(switchPin) == LOW)
{
// switch ditekan - pullup membiarkan pin high normally
delay(100);          // delay untuk switch
running = !running; //toggle variable running
digitalWrite(LEDpin, running) // indikasi via LED
}
}

```

## 10. Char

Sebuah tipe data yang mengambil 1 byte memori untuk menyimpan nilai karakter. Karakter huruf tertulis dalam tanda kutip satu seperti "A" (untuk karakter yang banyak, menggunakan tanda petik dua : "ABC"). Anda dapat melihat *encoding* yang lebih spesifik dalam **ASCII CHART**. Ini dimaksudkan dapat memungkinkan untuk menjalankan aritmatika dalam nilai ASCII yang menggunakan karakter misalkan „A“ +1 memiliki nilai 66, karena nilai ASCII dari huruf A adalah 65.

Tipe data char adalah angka encoding dari -128 sampai 127

### Contoh

```

char myChar = 'A';
char myChar = 65;
// both are equivalent

```

## 11. UnsignedChar

Tipe data yang membutuhkan 1 byte dari memori. Seperti pada tipe data byte, unsigned char mempunyai panjang 0 sampai 255.

Dalam Arduino, tipe data ini jarang digunakan.

### Contoh

```

unsigned char myChar = 240;

```

## 12. Arrays

Array adalah kumpulan dari beberapa variable yang saling berkaitan dengan sebuah nomor indeks.

## Membuat (deklarasi) sebuah Array

Dibawah ini adalah membuat sebuah metode array yang benar.

```
int myInts[6];
int myPins[] = {2, 4, 8, 3, 6};
int mySensVals[6] = {2, 4, -8, 3, 2};
char message[6] = "hello";
```

Anda dapat mendeklarasikan array tanpa menginisialisasikan nya seperti dalam myInts.

## Mengakses sebuah array

Array merupakan **zero indexed** yang menunjuk pada array yang telah diinisialisasi sebelumnya, elemen pertama dari array adalah indeks ke 0

```
mySensVals[0] == 2,
mySensVals[1] == 4, dan sampai ke empat.
```

Seperti halnya array dengan 10 elemen, indeks 9 adalah elemen terakhir

```
int myArray[10]={9,3,2,4,3,2,7,8,9,11};
// myArray[9]      contains 11
/*myArray[10] is invalid and      contains      random
information (other memory address)*/
```

Tidak seperti BASIC atau JAVA, dalam compiler C tidak mengecek untuk mengetahui jika akses array sudah sesuai batas besaran array yang Anda deklarasikan sebelumnya.

Sebuah penugasan dalam sebuah array

```
mySensVals[0] = 10;
```

untuk mendapatkan sebuah nilai dari array

```
x = mySensVals[4];
```

### **Array dan perulangan OR**

Array dapat dimanipulasikan dalam perulangan for, dimana penghitungan perulangan digunakan sebagai indeks pada beberapa elemen array. Seperti contoh untuk mencetak beberapa elemen sebuah array pada serial port, Anda dapat melakukan sesuatu seperti berikut :

```
int i;
for (i = 0; i < 5; i = i + 1) {
  Serial.println(myPins[i]);
}
```

### **13. Void**

Kata kunci void hanya digunakan dalam mendelarasikan fungsi. Hal ini untuk mendandakan bahwa fungsi itu diharapkan untuk memberi informasi dari fungsi yang telah dipanggil.

#### **Contoh**

```
void setup()
{
  // ...
}

void loop()
{ // ... }
```

## 14. String

Text string dapat dituliskan ke dalam dua cara, Anda dapat menggunakan tipe data String yang merupakan bagian dari dasar versi 0019, atau Anda dapat menggunakan sebuah string luar dari sebuah array tipe char dan *null-terminatenya*.

### Contoh

```
char Str1[15];
char Str2[8] = {'a', 'r', 'd', 'u', 'i', 'n', 'o'};
char Str3[8] = {'a', 'r', 'd', 'u', 'i', 'n', 'o', '\0'};
char Str4[ ] = "arduino";
char Str5[8] = "arduino";
char Str6[15] = "arduino";
```

### Kemungkinan – Kemungkinan Untuk Deklarasi String

- Deklarasi sebuah array dari chars tanpa inisialisasi seperti dalam Str1
- Deklarasi sebuah array dari chars (dengan satu extra char) dan compiler akan menambah karakter null seperti dalam Str2
- Menambahkan karakter null secara langsung, Str3
- Inisial dengan sebuah konstanta string dalam tanda kutipan, compiler akan menghitung lebar konstanta array untuk panjang konstanta string dan sebuah pengaliran karakter null, Str4
- Inisial array dengan menuliskan besar dan konstanta string, Str5
- Inisial array, memberi ruang ekstra untuk string yang lebih besar, Str6

### 14.1 Akhiran Null

Biasanya, banyak string yang diakhiri dengan sebuah karakter null (kode ASCII 0). Hal ini membolehkan fungsi (seperti Serial.print() ) untuk memanggil diakhir string. Selain itu, mereka akan melanjutkan pembacaan byte subsekuensial dari memori yang bukan merupakan bagian dari string. Hal ini dimaksudkan saat Anda membutuhkan string yang memiliki ruang untuk karakter yang lebih banyak dari teks yang Anda inginkan. Ini

kenapa Str2 dan Str5 membutuhkan 8 karakter sedangkan “arduino” hanya 7, posisi yang terahir secara otomatis akan ditempai oleh karakter null, Str4 akan otomatis sebesar 8 karakter, satu dari external null.

## 14.2 Menyatukan string yang panjang

Anda dapat menyatukan string yang panjang seperti berikut :

```
char myString[] = "This is the first line" "second line"
```

## 14.3 Array dari string

```
char* myStrings[]={ "This is string 1", "This is string  
2", "This is string 3", "This is string 4", "This is string  
5", "This is string 6"};
```

```
void setup(){ Serial.begin(9600);  
}  
void loop(){  
  for (int i = 0; i < 6; i++)  
  {  
    Serial.println(myStrings[i]); delay(500);}  
}
```

## 15. Penghitungan

### 15.1 abs(x)

Menghitung nilai absolute dari sebuah cacah

#### Parameter

x : cacah

#### Perhatian!

karena tentang abs() berfungsi untuk penerapan, hindari penggunaan fungsi yang terdapat tanda kurang, hal akan menghasilkan hasil yang salah

```
abs(a++); // hindari tanda ini (-) yields incorrect results  
a++; // use this instead -  
abs(a); // keep other math outside the function
```

### 15.2 constrain (x,a,b)

constrain merupakan angka batasan dalam sebuah *range*

#### Parameter

x : angka dari constrain ; semua tipe data  
a : batas bawah dari *range*; semua tipe data  
b : batas atas dari *range*; semua tipe data

### Kebalikan

x : jika x diantara a dan b  
a : jika x lebih kecil dari a  
b : jika x lebih dari b

### Contoh

```
sensVal = constrain(sensVal, 10, 150);  
// batas range dari nilai sensor antara 10  
//dan 150
```

## 15.3 map(value, fromLow, fromHigh, toLow, toHigh)

Memetakan ulang sebuah angka dari sebuah range ke yang lainnya. Ini adalah sebuah nilai dari fromLow yang sudah dipetakan ke **toLow**, sebuah nilai dari **fromHigh** pada **toHigh**.

### Contoh

```
y = map(x, 1, 50, 50, 1);  
fungsi ini juga menangani angka negative  
y = map(x, 1, 50, 50, -100);
```

### Contoh

```
//Map an analog value to 8 bits (0 to 255)  
void setup() {}  
void loop()  
{  
  int val = analogRead(0);  
  val = map(val, 0, 1023, 0, 255);  
  analogWrite(9, val);  
}
```

### Catatan tambahan

Untuk penghitungan matematika, berikut adalah fungsinya

```
long map(long x,, long in_min, long in_max, long  
out_min, long out_max)
```

```
{
  return(x - in_min)*(out_max - out_min)/(in_max - in_min)
  + out_min;}

```

## 15.4 min(x,y)

Menghitung minimum dari dua angka

```
sensVal = min(sensVal, 100);
/*memberikan perintah agar nilai sensVal lebih kecil dari
sensVal atau 100. Memastikan bahwa nilai sensVal tidak
pernah mendapatkan 100*/

```

max() juga ikut digunakan untuk konstrain yang lebih rendah dari variable range, sedangkan min() digunakan untuk konstrain yang mempunyai range lebih tinggi.

```
sensVal = max(sensVal, 20);
/*memberikan perintah agar nilai sensVal lebih besar dari
sensVal atau 20*/

```

## 16. Struktur

### 16.1 setup()

Fungsi setup() untuk memulai sketsa pemrograman, inisialisasi variable, pin mode, memulai menggunakan library dan lainnya. Fungsi setup akan dijalankan sekali setelah arduino dihidupkan atau di reset.

#### Contoh

```
int buttonPin = 3;
void setup()
{
  Serial.begin(9600);
  pinMode(buttonPin, INPUT);
}
void loop()
{
  //...
}

```

### 16.2 loop()

Setelah membuat fungsi setup (), yang harus dilakukan adalah menginisialisasi dan mengeset nilai inisial dari fungsi loop() dengan tepat

apa nama yang akan diberikan. Gunakan ini untuk mengaktifkan board Arduino.

### Contoh

```
//konfigurasi serial dan pin button
int buttonPin = 3;
void setup()
{
  beginSerial(9600);
  pinMode(buttonPin, INPUT);
}
// melakukan cek perulangan pada pin button setiap waktu
// dan serial akan mengirim data ketika button di tekan
void loop()
{
  if(digitalRead(buttonPin)== HIGH)
    serialWrite('H');
  else serialWrite('L');
  delay(1000);
}
```

## 17. Control Structure

### 17.1 IF

If yang digunakan dalam imbuhan dengan operator pembandingan.

Format untuk if :

```
if (someVariable > 50)
{
  // bisa digunakan untuk statemen yang lain
}
```

Untuk cara lain, statemen if selalu dalam keadaan true

```
if (x > 120) digitalWrite(LEDpin, HIGH);
if (x > 120) digitalWrite(LEDpin, HIGH);
if (x > 120){ digitalWrite(LEDpin, HIGH); }
if (x > 120){
  digitalWrite(LEDpin1, HIGH);
  digitalWrite(LEDpin2, HIGH);
}
// semua dianggap benar
```



## 17.2 Comparison Operators

```
x == y (x sama dengan y)
x != y (x tidak sama dengan y)
x < y (x lebih kecil dari y)
x > y (x lebih besar dari y)
x <= y (x lebih kecil sama dengan y)
x >= y (x lebih besar sama dengan y)
```

## 17.3 IF/ELSE

Struktur if/else memberikan kita control yang lebih besar untuk menangani berbagai statement dibandingkan dengan statemen dasar if. Jadi struktur if else merupakan gabungan pernyataan if bersarang yang membolehkan logika ya atau tidak. Untuk contoh, sebuah input analog dapat mengerjakan sebuah aksi yang mempunyai inputan kurang dari 500 dan yang lain beraksi jika inputan tersebut > 500, maka kodenya dapat dituliskan sebagai berikut:

```
if (pinFiveInput < 500)
{
    // action A
}
else
{
    // action B
}
```

Else juga dapat digunakan secara bertingkat sampai dengan tak terhingga selama proses yang kita inginkan belum menyatakan sebuah kebenaran atau berlogika true, berikut contoh kodenya

```
if (pinFiveInput < 500)
{
    // do Thing A
}
else if (pinFiveInput >= 1000)
{
    // do Thing B
}
```

```
}  
else  
{//do Thing C}
```

## 17.4 Switch/case

Seperti halnya if statemen, switch...case megontrol alir dari program yang membolehkan beberapa parameter yang berbeda untuk dieksekusi dalam kondisi yang bervariasi.

### Contoh:

```
switch (var) { case 1:  
//do something when var equals 1  
break;  
case 2:  
//do something when var equals 2  
break;  
default:  
// if nothing else matches, do the default  
// default is optional  
}
```

### Syntax

```
switch (var) {  
case label:  
// statements  
break;  
case label:  
//statements  
break;  
default:  
// statements  
}
```

**break** selalu ada dalam statemen case, dan digunakan untuk ahiran dari bermacam macam case, tanpa statemen break, statemen swich akan terus dijalankan sesuai ekpresi tanpa henti ("falling- through") sampai menemukan statemen break atau sampai statemen swich ditutup.

## 17.5 For

Pernyataan for digunakan untuk mengulang sebuah blok statemen yang terus berputar. Biasanya, pernyataan for digunakan untuk mengkombinasikan sebuah array dalam menyatakan sebuah pin atau data. Berikut header untuk perulangan for :

```
for (initialization; condition; increment) {  
    //statement(s);}
```

Inisialisasi dilakukan pertama dan sekali saja. Condition adalah berapa kali kita akan melakukan looping, increment adalah menunjukkan bahwa looping akan dijalankan secara bertingkat.

### Contoh :

```
// Dim an LED using a PWM pin  
// LED in series with 470 ohm resistor on pin 10  
int PWMpin = 10;  
void setup()  
{  
    // no setup needed  
}  
void loop()  
{  
    for (int i=0; i <= 255; i++)  
    {  
        analogWrite(PWMpin, i);  
        delay(10);  
    }  
}
```

### Tips Pemograman

Dalam C, statemen for sangat fleksibel di banding dengan for yang ada pada BASIC. Beberapa diantaranya adalah elemen header bisa saja dihilangkan, namun tanda titik koma harus tetap disertakan.

```
for(int x = 2; x < 100; x = x * 1.5)  
{ println(x);}
```

menghasilkan: 2,3,4,6,9,13,19,28,42,63,94

Contoh lain:

```
void loop()
{
  int x = 1;
  for (int i = 0; i > -1; i = i + x){
    analogWrite(PWMPin, i);
    if (i = 255) x = -1;
    // switch direction at peak delay(10);
  }
}
```

## 17.6 while

While akan melakukan perulangan secara kontinyu dan tidak terbatas samapi ekspresi tersebut menemukan kembali parenthesisnya, () menjadi false. Biasanya digunakan untuk mengetes sebuah sensor karena perulangan ini tak akan berahir sampai adanya kondisi eksternal yang menutupnya

### Syntax

```
while(expression)
{
  // statement(s)
}
```

### Parameters

expression - (boolean) statemen C mengkoreksi true atau false

### Contoh

```
var = 0;
while(var < 200){ //mengulang    sesuatu sampai 200
  kali var++;}
```

## 17.7 do-while

Perulangan **do** bekerja seperti cara **while** melakukan perulangan, perulangan **do** hanya mengerjakan sekali di ahir statemen.

```
do
{
  //statement block
```

```
}  
while (test condition);
```

### Contoh

```
do  
{  
  delay(50); //wait for sensors to stabilize  
  x = readSensors(); //check the sensors  
}  
while (x < 100);
```

### 17.8 break

**break** digunakan untuk mengahiri perulangan dari **do**, **for** atau **while**.

Dan juga digunakan untuk mengahiri statemen **switch**.

### Contoh

```
for (x = 0; x < 255; x ++)  
{  
  digitalWrite(PWMpin, x);  
  sens = analogRead(sensorPin);  
  if (sens > threshold){ //bail out on sensor detect  
    x = 0;  
    break;  
  }  
  delay(50);  
}
```

### 17.9 goto

Untuk melompati alur program ke program yang sudah kita beri label.

### Syntax label:

`goto label;` // mengirim alur ke program yang telah diberi label.

### Tips Pemograman

Dalam menggunakan goto, adalah tidak lazim dalam pemrograman C. beberapa buku memberikan gambaran bahwa statemen goto justru akan mengganggu jalannya alur program, karena tidak bisa statemen yang ada

didalam goto tidak dapat dikenali dan tidak dapat di debug, sehingga hasilnya pun tidak bisa kita ketahui. Berikut contoh programnya:

```
for(byte r = 0; r < 255; r++){
for(byte g = 255; g > -1; g--){
for(byte b = 0; b < 255; b++){
    if (analogRead(0) > 250){
        goto bailout;}
        // more statements ...
    }
}
}
```

### 17.10 bailout

Mengahiri sebuah fungsi dan mengembalikan sebuah nilai dari sebuah fungsi tersebut untuk memanggil fungsi yang lain jika diinginkan.

#### Syntax:

```
return; return value;
```

#### Contoh

Sebuah fungsi untuk membandingkan inputan sebuah sensor kepada sebuah threshold

```
int checkSensor(){
    if (analogRead(0) > 400) {
        return 1;
    }
    else{
        return 0;
    }
}
```

### 17.11 continue

Statemen **continue** membiarkan atau melanjutkan penghentian perulangan (**do**, **for** atau **while**). Digunakan untuk melanjutkan pada pengecekan sebuah ekspresi kondisi dari loop dan memprosesnya dengan beberapa subsequent yang berulang ulang.

#### Contoh

```
for (x = 0; x < 255; x ++){
```

```

    if (x > 40 && x < 120){
    // create jump in values continue;
    }
    digitalWrite(PWMpin, x);
    delay(50);
}

```

## 18. FUNCTION

### 18.1 pinMode()

Digunakan untuk mengkonfigurasi pin secara spesifik sebagai sebuah inputan atau outputan.

#### Syntax

```
pinMode(pin, mode)
```

#### Parameters

pin: set nomor pin mode

mode: INPUT or OUTPUT

#### Contoh

```

int ledPin = 13; // LED terkoneksi pada digital pin 13
void setup()
{
    pinMode(ledPin, OUTPUT);
    //set digital pin sebagai output
}
void loop()
{
    digitalWrite(ledPin, HIGH); //set LED on
    delay(1000);
    // waktu tunggu dalam second
    digitalWrite(ledPin, LOW);
    // set LED off
    delay(1000); //waktu tunggu dalam second
}

```

## 19. Digital I/O

### 19.1 digitalWrite()

Menuliskan logika HIGH atau LOW pada pin digital. Jika pin telah dikonfigurasi sebagai OUTPUT dengan pinMode(), itu berarti arus akan diset dengan nilai 5V (atau 3.3V dalam board) untuk HIGH, 0V (ground) untuk LOW. Jika pin dikonfigurasi sebagai INPUT, maka penulisan logika HIGH dengan digitalWrite() akan membutuhkan sebuah resistor pullup internal 20K, penulisan LOW akan mendisable pullup nya. Pullup resistor cukup untuk menyalakan LED dengan redup, jadi jika LED bekerja namun samar, maka harus memperbaiki set pin ke output dengan fungsi pinMode()

### **Syntax**

digitalWrite(pin, value)

### **Parameters**

pin: nomor pin

value: HIGH atau LOW

### **Contoh**

```
int ledPin = 13; // LED connected to digital pin 13
void setup()
{
  pinMode(ledPin, OUTPUT); //sets the digital pin as output
}
void loop()
{
  digitalWrite(ledPin, HIGH); // sets the LED on
  delay(1000);                // waits for a second
  digitalWrite(ledPin, LOW);  // sets the LED off
  delay(1000);                // waits for a second
}
```

## **19.2 digitalWrite()**

Membaca nilai dari spesifikasi digital pin yang bernilai HIGH atau LOW.

### **Syntax**

digitalRead(pin)

### **Parameters**

pin: nomor dari digital pin yang ingin dibaca (*int*)

### **Contoh**

```
int ledPin = 13;
```



```
// LED terkoneksi pada digital pin 13
int inPin = 7;
// pushbutton terkoneksi digital pin 7
int val = 0;
// variable to store the read value void setup()
{
    pinMode(ledPin, OUTPUT);
    // set digital pin 13 sebagai output
    pinMode(inPin, INPUT);
    // set digital pin 7 sebagai input
}
void loop()
{
    val = digitalRead(inPin); // baca input pin
    digitalWrite(ledPin, val);
    // set LED untuk nilai pada button}
```

Set pin 13 seperti nilai dari pin 7 yang menjadi inputan. Jika pin tidak terkoneksi pada digitalRead() maka dapat dikembalikan pada HIGH atau LOW dan dapat berganti secara acak.

Input pin analog dapat digunakan sebagai digital pin, tertunjuk sebagai A0, A1, dsb.

## 20 Analog I/O

### 20.1 analogReference(type)

Fungsi ini digunakan untuk mengkonfigurasi voltase acuan yang digunakan untuk input analog, sebagai pilihannya :

- DEFAULT : default acuan analog sebesar 5V (dalam 5V Arduino board) atau 3.3 V (dalam 3.3V board Arduino)
- INTERNAL : sebuah acuan built-inn sama dengan 1.1V dalam ATmega168 atau ATmega328v dan 2.56V pada ATmega8 (tidak terdapat pada Arduino Mega)
- INTERNAL1V1 : acuan built-in 1.1V (hanya Arduino Mega)
- INTERNAL2V56 : acuan built-in 2.56V (hanya Arduino Mega)
- EXTERNAL : voltase yang digunakan pada pin AREF

### Parameters

type: acuan yang digunakan (DEFAULT, INTERNAL, INTERNAL1V1, INTERNAL2V56, atau EXTERNAL).

Setelah mengganti acuan analog, pembacaan pertama pada `analogRead()` mungkin tidak bisa akurat. Jika Anda menggunakan acuan voltase external (digunakan untuk pin AREF), Anda harus menyetting acuan analog pada EXTERNAL sebelum memanggil `analogRead()`. Selain itu, Anda bisa mengaktifkan acuan voltase bersamaan (internally generated) dan pada pin AREF, dapat merusak mikrokontroler pada board Arduino.

## 20.2 `analogRead()`

Membaca nilai dari pin analog. Board arduino menampung 6 channel (8 channel pada Mini dan Nano, 16 pada Mega), 10-bit ADC. Ini berarti akan memetakan voltase antara 0 sampai 5 volt dalam nilai integer antara 0 dan 1023. Sehingga menghasilkan resolusi pembacaan dari 5volt/1024 unit atau 0.0049volt (4.9mV) per unit. Panjang input dan resolusi dapat digantikan dengan `analogReference()`. Hal ini membutuhkan 100milisecond (0.0001s) untuk membaca input analog, jadi maksimum rate pembacaan sekitar 10000 kali per detik.

### Syntax

```
analogRead(pin)
```

### Parameter

pin : nomor dari pin input analog untuk dibaca dari (0 sampai 5 pada umumnya, 0 sampai 7 pada Mini dan Nano, 0 sampai 15 pada Mega) bisa juga dituliskan sebagai int (0 sampai 1023) .

### Contoh

```
int analogPin = 3;
// potentiometer wiper (middle terminal)
// connected to analog pin 3
// outside leads to ground and +5V int val = 0;
// variable to store the value read
void setup()
{
    Serial.begin(9600);
```

```

        // setup serial
    }
    void loop()
    {
        val = analogRead(analogPin);
        // read the input pin
        Serial.println(val);
        // debug value
    }

```

### 20.3 analogWrite()

`analogWrite()` dapat digunakan untuk menulis nilai analog (gelombang PWM) untuk sebuah pin. Dapat juga digunakan menyalakan LED dengan cahaya yang bervariasi atau mengontrol motor dengan kecepatan yang bervariasi. Frekuensi sinyal PWM bisa mencapai 490 Hz. Di dalam board Arduino ATmega168 atau ATmega328, pin yang berfungsi adalah 3,5,6,9,10 dan 11.

Di dalam Arduino Mega bekerja pada pin 2 bersama 13, dalam Arduino ATmega8 hanya mendukung `analogWrite()` pada pin 9,10 dan 11. Anda tidak dapat memanggil `pinMode()` untuk menyeting pin sebagai output sebelum memanggil `analogWrite()`. Fungsi dari `analogWrite` tidak bisa bekerja tanpa pin analog atau fungsi `analogRead`.

#### Syntax

```
analogWrite(pin, value)
```

#### Parameter

pin: pin yang akan digunakan

value : putaran penugasan : antara 0 (selalu off) dan 255 (selalu on).

Pada output PWM yang dihasilkan pada pin 5 dan 6 akan mempunyai putaran penugasan yang lebih tinggi. Hal ini dikarenakan interaksi dengan fungsi `millis()` dan `delay()` yang membagi sesama penggunaan internal timer yang digunakan oleh output PWM.

#### Contoh

Set output pada LED dengan nilai yang disesuaikan dari nilai potensiometer

```

int ledPin = 9; //LED connected to digital pin 9
int analogPin = 3;

```

```
//potentiometer connected to analog pin 3
int val = 0;
// variable to store the read value
//analogWrite values from 0 to 255
}
```

## **20. Advance I/O**

### **21.1 pulseIn()**

Fungsi ini digunakan untuk membaca sebuah pulsa (HIGH atau LOW) pada sebuah pin

#### **Syntax**

```
pulseIn(pin,value)
pulseIn(pin,value,timeout)
```

#### **Parameter**

pin : nomor pin yang akan dibaca pulsa nya (int)

value : tipe dari pulsa (HIGH atau LOW) (int)

timeout (optional) : angka dalam mikrosekon waktu tunggu pulsa untuk memulai, defaultnya adalah 1 detik (unsigned long)

#### **Contoh**

```
int pin = 7;
unsigned long duration;
void setup()
{
    pinMode(pin,INPUT;
void setup()
{
    pinMode(ledPin,OUTPUT); // sets the pin as output
}
void loop()
{
    val = analogRead(analogPin); //read the input pin
    analogWrite(ledPin, val/4);
    //analogRead values go from 0 to 1023
    duration = pulseIn(pin, HIGH);
}
```

## 21.2 shiftOut()

Shift out adalah sebuah byte dari data dalam satu bit per satuan waktu.

### Syntax

```
shiftOut (dataPin, clockPin, bitOrder, value)
```

### Parameter

dataPin : pin yang menjadi output bit (int)

clockPin : pin yang sekali dilewati dengan dataPin (int)

bitOrder : bit yang akan diberikan untuk shift out berupa **MSBFIRS** atau **LSBFIRST** (Most significant Bit First atau Least Significant Bit First)

value : data untuk shift out (byte)

**dataPin** dan **clockPin** harus selalu dikonfigurasi sebagai output untuk memanggil pinMode()

```
// Do this for MSBFIRST serial
int data = 500;
// shift out highbyte
shiftOut(dataPin, clock, MSBFIRST, data >> 8));
// shift out lowbyte
shiftOut(data, clock, MSBFIRST, data);
// Or do this for LSBFIRST serial data = 500;
// shift out lowbyte
shiftOut(dataPin, clock, LSBFIRST, data);
// shift out highbyte
shiftOut(dataPin, clock, LSBFIRST, (data >> 8));
```

### Contoh

*Untuk contoh ini silahkan lihat tutorial mengkontrol IC shift register 74HC595*

```
//Pin connected to ST_CP of 74HC595
int latchPin = 8;
//Pin connected to SH_CP of 74HC595
int clockPin = 12;
//Pin connected to DS of 74HC595
int dataPin = 11;
void setup() {
  /*set pins to output because they are addressed in the
  main loop*/
  pinMode(latchPin, OUTPUT);
```

```

pinMode(clockPin, OUTPUT);
pinMode(dataPin, OUTPUT);
}
void loop() {
//count up routine
for (int j = 0; j < 256; j++) {
//ground latchPin and hold low for as long as you are
//transmitting
digitalWrite(latchPin, LOW);
shiftOut(dataPin, clockPin, LSBFIRST, j);
//return the latch pin high to signal chip that it
//no longer needs to listen for information
digitalWrite(latchPin, HIGH);
delay(1000);
}
}

```

### 21.3 tone()

Dapat menghasilkan gelombang kotak dengan spesifikasi frekuensi (50% siklus kerja) dalam sebuah pin. Biasanya pin ini dikoneksikan pada buzzer atau speaker untuk memutar bunyi, bunyi ini akan diputar terus menerus sebelum ada pemanggilan noTone(). Jika tone() sedang bekerja pada sebuah pin, maka pin yang lain tidak akan mempunyai efek apapun, jadi bila kita akan membunyikan suara pada beberapa pin, maka kita harus menyertakan fungsi noTone() pada setiap akhir ekspresinya.

#### Syntax

```

tone(pin, frequency)
tone(pin, frequency, duration)

```

## 21. Communication

### 22.1 Serial

Untuk berkomunikasi dengan komputer atau device lain, semua board arduino mempunyai serial port (UART atau USART). Untuk berkomunikasi menggunakan digital pin 0 (RX) dan 1 (TX) seperti computer dengan USB. Dalam Arduino Mega, mempunyai tambahan 3 serial port : serial 1 pada pin 19 (RX) dan 18 (TX), serial 2 pada pin 17 (RX) dan 16 (TX), serial 13 pada pin 15(RX) dan 14 (TX), untuk berkomunikasi dengan PC, Anda membutuhkan

USB to Serial adaptor, untuk mengkomunikasikan dengan external TTL serial device. Jangan menyambungkan pin pada RS232 secara langsung, karena mempunyai +/- 12v yang bisa merusak board Arduino.

## 22.2 Interrupt()

Mengaktifkan kembali interupsi (setelah nonaktif oleh `noInterrupt()`). interupsi memperbolehkan program lain yang lebih penting untuk berjalan terlebih dahulu. Beberapa fungsi tidak akan dapat bekerja jika interupsi tersebut disable, dan berkomunikasi kembali ketika diabaikan.

### Contoh

```
void setup()
{
}

void loop()
{
  noInterrupts();
  // critical, time-sensitive code here
  interrupts();
  // other code here
}
```

## 22.3 noInterrupts()

Mematikan interupsi (Anda dapat mengaktifkan kembali dengan `interrupts()`). fungsi ini adalah kebalikan dari fungsi `interrupts()`.

### Contoh

```
void setup()
{
}

void loop()
{
  noInterrupts();
  // critical, time-sensitive code here
  interrupts();
  // other code here
}
```

## 22.4 External Interrupt (attachInterrupt(interrupt, function, mode))

Fungsi yang lebih spesifik untuk memanggil external interrupt. Mengganti beberapa fungsi sebelumnya yang telah disertakan. Kebanyakan board Arduino mempunyai 2 external interrupt. Nomor 0 (dalam digital pin 2) dan 1 (dalam digital pin 3). Arduino Mega mempunyai 4 tambahan: nomor 2 (pin 21), 3 (pin 20), 4 (pin 19) dan 5 (pin 18)

### Parameter

interrupt : nomor interupsi (int)

function : fungsi untuk memanggil interupsi yang terjadi, menjadi acuan bagi interrupt service routine.

mode : mendefinisikan jika terjadi trigger interupsi, ada 4 parameter yang menjadi validasi nilainya :

- LOW untuk trigger interupsi saat pin low.
- CHANGE untuk trigger interupsi saat pin mengganti nilai.
- RISING untuk trigger saat pin berpindah dari low menjadi high.
- FALLING untuk pin yang berpindah dari high ke low.

Interupsi digunakan untuk membantu menangani masalah penjadwalan waktu pada program mikrokontroler.

### Contoh

```
int pin = 13;
volatile int state = LOW;
void setup()
{
    pinMode(pin, OUTPUT);
    attachInterrupt(0, blink, CHANGE);
}
void loop()
{
    digitalWrite(pin, state);
}
void blink()
{
    state = !state;
}
```



## **22. TIME**

### **23.1 delay()**

Memberi jeda pada program untuk beberapa waktu ( dalam millisecond)

#### **Syntax**

`delay(ms)`

#### **Parameter**

**ms** : waktu dalam millisecond (unsigned long)

#### **Contoh**

```
int ledPin = 13; // LED connected to digital pin 13
void setup(){
  pinMode(ledPin,OUTPUT);
  // sets the digital pin as output
}
void loop()
{
  digitalWrite(ledPin, HIGH); //sets the pin on
  delay(1000); //waits for a second
  digitalWrite(ledPin, LOW); // sets the pin off
  delay(1000); //waits for a second
}
```

### **23.2 delayMicroseconds()**

Memberi jeda program dalam beberapa waktu (mikrodetik), 1000 microsecond adalah 1 millisecond dan 1juta microsecond adalah 1 detik.

#### **Syntax**

`delayMicroseconds(us)`

#### **Parameter**

**us** : angka untuk jeda dalam microsecond (unsigned int)

#### **Contoh**

```
int outPin = 8; // digital pin 8
void setup()
{
  pinMode (outPin, OUTPUT);
  //sets the digital pin as output
}
void loop()
```





```
{  
    digitalWrite(outPin, HIGH); //sets the pin on  
    delayMicroseconds(50); //pauses for 50 microseconds  
    digitalWrite(outPin, LOW); // sets the pin off  
    delayMicroseconds(50); //pauses for 50 microseconds  
}
```

**Mengkonfigurasi pin nomor 8 sebagai pin output, memberikan pulsa dengan waktu 100 microsecond per periode.**

## **Bagian 3. Latihan Programing**

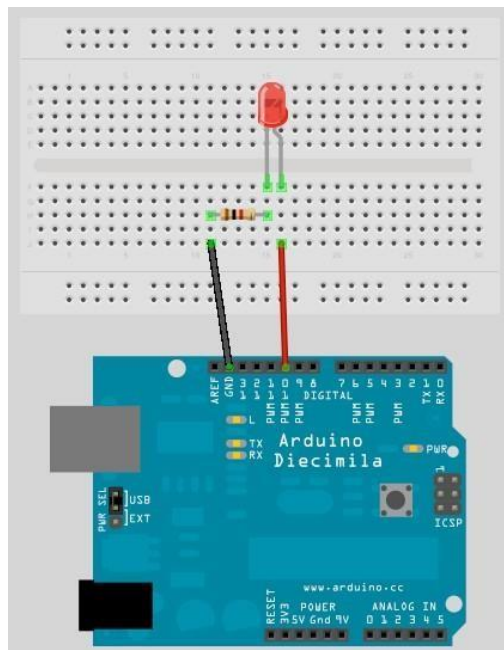
# Latihan 1-LED Blinking

## Komponen yang diperlukan

			
Breadboard	LED	Resistor 220 $\Omega$	Kabel Jumper

## Rangkaian

Setelah memastikan software arduino sudah terinstall dengan baik dan benar selanjutnya kita hubungkan semuanya seperti gambar dibawah ini:




Tidak masalah jika menggunakan kabel jumper yang berbeda atau pun arduino yang berbeda selama koneksi antara komponen seperti gambar.

Pastikan posisi Anoda dan Katoda LED benar, LED dengan kaki yang panjang terhubung ke pin digital 10, kaki ini adalah kaki Anoda LED yang harus terhubung ke pin positif, sedangkan kaki LED yang pendek terhubung ke ground (GND) yang sebelumnya di seri menggunakan resistor 220  $\Omega$ , karena kaki LED ini merupakan kaki Katoda LED.

Setelah semuanya benar maka hubungkan USB ke komputer anda lalu buka software Arduino IDE.

## Program

Masukan kode berikut ke halaman Arduino IDE :

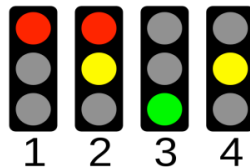


```
//LED Blinking
int ledPin = 10;
void setup()
{
    pinMode(ledPin, OUTPUT);
}
void loop()
{
    digitalWrite(ledPin, HIGH);
    delay(1000);
    digitalWrite(ledPin, LOW);
    delay(1000);
}
```




Setelah itu klik Verify/Compile pada software Arduino IDE, tunggu hingga selesai dan sukses mengcompile program, jika pada notifikasi muncul error maka perlu di tinjau ulang, mungkin ada kesalahan pada tipe board yang di pakai, port yang sedang digunakan ataupun source code yang tidak tertulis. Setelah itu klik tombol Upload, pada proses ini program akan di transfer ke board Arduino, tunggu hingga program terupload dan muncul notifikasi “Done uploading” maka Led akan berkedip dengan interval waktu hidup 1 detik dan mati 1 detik, jika masih kurang paham silahkan buka kembali materi tentang delay(ms).

# Latihan2- Traffic Light

Pada kali ini kita akan membuat sebuah program lampu jalan atau *Traffic Light*, cara kerja program ini yaitu LED pada posisi warna merah kemudian akan berpindah ke LED berwarna hijau yang sebelumnya melalui LED warna kuning dan kembali lagi ke LED merah melalui LED kuning terlebih dahulu.

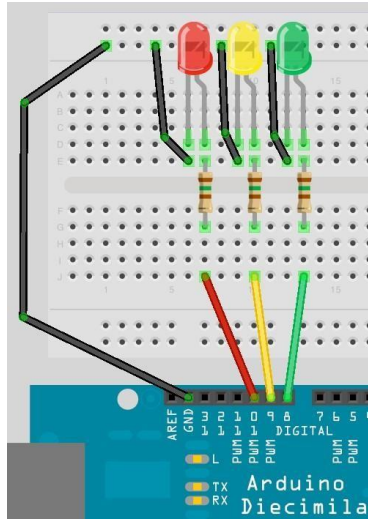


## Komponen yang diperlukan

					
Breadboard	LED Merah	LED Kuning	LED Hijau	Resistor 220 $\Omega$	Kabel Jumper

## Rangkaian

Kali ini kita menghubungkan 3 LED sekaligus dengan pin Anoda LED terhubung dengan pin 8, 9 dan 10 papan arduino melalui resistor 220  $\Omega$  sebelumnya. Untuk memudahkan merangkai maka kita hubungkan semua pin katoda led pada satu jalur *breadboard* menuju pin *ground* papan arduino



## Program

Masukan kode berikut, cek ulang lalu upload.

```
//Traffic Lights
int ledDelay = 5000; // delay in between changes
int redPin = 10;
int yellowPin =9;
int greenPin =8;

void setup()
{
  pinMode(redPin,OUTPUT);
  pinMode(yellowPin,OUTPUT);
  pinMode(greenPin,OUTPUT);
}
void loop() {
// turn the red light on
digitalWrite(redPin, HIGH);
delay(ledDelay); // wait 5 seconds
digitalWrite(yellowPin, HIGH); // turn on yellow
delay(2000); // wait 2 seconds
digitalWrite(greenPin, HIGH); // turn green on
```

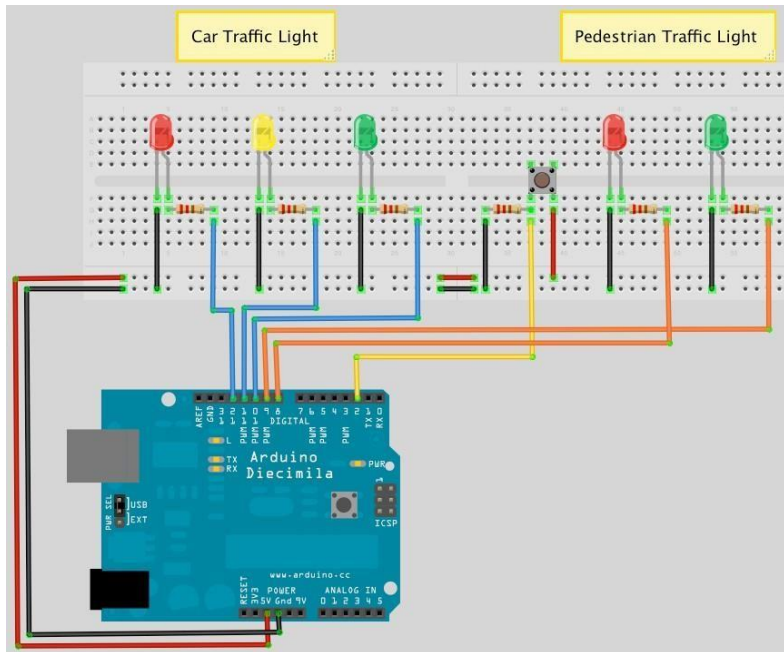
```
digitalWrite(redPin, LOW); // turn red off
digitalWrite(yellowPin, LOW); // turn yellow off
delay(ledDelay); // wait ledDelay milliseconds
digitalWrite(yellowPin, HIGH); // turn yellow on
digitalWrite(greenPin, LOW); // turn green off
delay(2000); // wait 2 seconds
digitalWrite(yellowPin, LOW); // turn yellow off
// now our loop repeats
}
```

Program tersebut menjalankan LED merah selama *ledDelay* detik (pada program tersebut di set *ledDelay* = 5 detik), kemudian berpindah ke LED kuning selama 2 detik lalu menuju LED hijau selama *ledDelay* detik.

Pada project selanjutnya kita akan menambahkan lampu pedestrian dan *push button* untuk membuat *Traffic Light Interactive*.











# Latihan 3-Interactive Traffic Light



Kali ini kita membuat *Interactive Traffic Light*, program yang digunakan sama dengan *traffic light* pada latihan 2 hanya kali ini kita akan menambahkan program lampu *pedestrian*/pejalan kaki yang digunakan untuk menyebrang jalan. Lampu *pedestrian* akan berfungsi ketika *push button* ditekan, ini akan mengubah keadaan *traffic light* untuk mobil menjadi merah dan *traffic light* untuk pejalan kaki akan berwarna hijau.

## Komponen yang diperlukan

							
Breadboard	2x LED Merah	1x LED Kuning	2x LED Hijau	5x Resistor 220 Ω	1x Resistor 10k	Push Button	Kabel Jumper

## Rangkaian

Hubungkan LED dan *push button* sesuai dengan diagram yang ada pada halaman sebelumnya, lalu hubungkan LED pada pin 8,9,10,11 dan 12 serta *push button* pada pin 2 yang di *pullup* menggunakan resistor 10k

## Program

Masukan kode pada halaman arduino IDE, cek kembali lalu klik upload.



```
//Interactive Traffic Lights
int carRed = 12; // assign the car lights
int carYellow = 11;
int carGreen = 10;
int pedRed = 9; // assign the pedestrian lights
int pedGreen = 8;
int button = 2; // button pin
int crossTime = 5000; // time allowed to cross
unsigned long changeTime; // time since button pressed

void setup()
{
    pinMode(carRed, OUTPUT);
    pinMode(carYellow, OUTPUT);
    pinMode(carGreen, OUTPUT);
    pinMode(pedRed, OUTPUT);
    pinMode(pedGreen, OUTPUT);
    pinMode(button, INPUT); // button on pin 2
    // turn on the green light
    digitalWrite(carGreen, HIGH);
    digitalWrite(pedRed, HIGH);
}

void loop() {
    int state = digitalRead(button);
    /* check if button is pressed and it is over 5 seconds
    since last button press */
    if (state==HIGH && (millis()- changeTime) >5000)
    {
        // Call the function to change the lights
        changeLights();
    }
}
```

```

void changeLights()
{
    digitalWrite(carGreen, LOW); // green off
    digitalWrite(carYellow, HIGH); // yellow on
    delay(2000); // wait 2 seconds

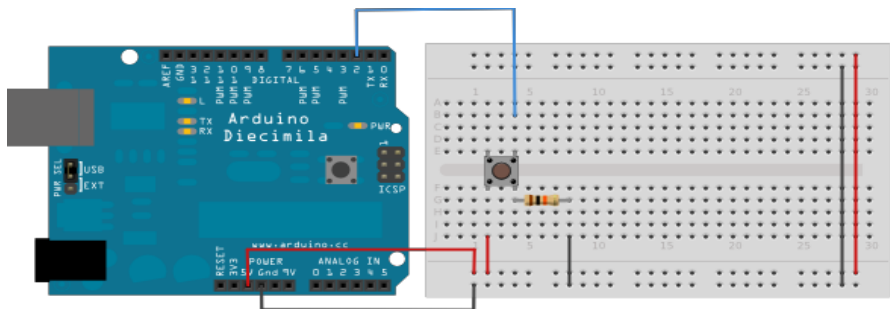
    digitalWrite(carYellow, LOW); // yellow off
    digitalWrite(carRed, HIGH); // red on
    delay(1000); // wait 1 second till its safe
    digitalWrite(pedRed, LOW); // ped red off
    digitalWrite(pedGreen, HIGH); // ped green on
    delay(crossTime); // wait for preset time period
    // flash the ped green
    for (int x=0; x<10; x++)
    {
        digitalWrite(pedGreen, HIGH);
        delay(250);
        digitalWrite(pedGreen, LOW);
        delay(250);
    }
    // turn ped red on
    digitalWrite(pedRed, HIGH);
    delay(500);
    digitalWrite(carYellow, HIGH); // yellow on
    digitalWrite(carRed, LOW); // red off
    delay(1000);
    digitalWrite(carGreen, HIGH);
    digitalWrite(carYellow, LOW); // yellow off
    // record the time since last change of lights
    changeTime = millis();
    // then return to the main program loop
}

```

Setelah berhasil, ketika menjalankan program kita akan melihat traffic light untuk mobil akan berwarna hijau, dan traffic light untuk pejalan kaki berwarna merah. Ketika push button kita tekan, program akan mengecek keadaan traffic light untuk mobil agar berubah menjadi merah, setelah itu giliran lampu untuk pejalan kaki yang akan berubah warna menjadi hijau,







waktu tunggu yang di tentukan dari mulai penekanan tombol hingga perubahan trafic light menjadi warna merah di atur oleh syntax `changeLight()` sedangkan untuk lamanya waktu yang dibutuhkan untuk lampu pejalan kaki dari berwarna merah menjadi hijau diatur oleh syntax `changeTime()` pada source code di halaman 2. Lampu pejalan kaki akan berwarna hijau lalu berkedip maka waktu tunda untuk lampu pejalan kaki akan habis, dan lampu akan berwarna merah dan *traffic light* untuk mobil akan berubah menjadi warna kuning dan hijau kembali.

# Latihan 4 – *Pushbutton*



*Pushbutton* merupakan komponen yang menghubungkan atau memutuskan dua titik di dalam sebuah rangkaian ketika kita menekannya. Terdapat jenis-jenis *pushbutton* tapi yang akan kita gunakan yaitu tactile switch *pushbutton* normally open artinya pada kondisi tidak di tekan kondisi *pushbutton* terbuka begitu sebaliknya ketika *pushbutton* di tekan maka kondisinya terhubung, contohnya LED akan menyala ketika kita menekan *pushbutton*. Pada percobaan kali ini kita akan membuat LED menyala ketika *pushbutton* ditekan, dan akan mati ketika *pushbutton* dilepas.

## Komponen yang diperlukan

					
Breadboard	1x LED Hijau	1x Resistor 220 $\Omega$	1x Resistor 10k	Push Button	Kabel Jumper

## Rangkaian

Pertama hubungkan 1 kaki pushbutton dihubungkan dengan 1 kaki resistor dan pin digital 2, kaki pushbutton yang lain dihubungkan dengan VCC dan satu kaki resistor yang lain di hubungkan dengan *ground*. Lihat contoh diagram rangkaian di atas. Sedangkan untuk led kita bisa menghubungkannya dengan ke kaki 13 setelah melalui resistor 220 $\Omega$ , sedangkan kaki katoda LED

dimasukan ke ground arduino, pada project ini bisa digunakan juga LED internal yang telah disediakan di papan arduino, sehingga kita tidak perlu menambahkan led lagi.

## Program

Masukan kode berikut ke Arduino IDE



```
//Pushbutton

const int buttonPin = 2; //inisialisasi pin button
const int ledPin = 13; //inisialisasi pin LED
int buttonState = 0; //variable
// variable digunakan untuk pembacaan status button

void setup()
{
  pinMode(ledPin, OUTPUT);
  pinMode(buttonPin, INPUT);
}







void loop() {
  //membaca nilai dari pin button
  buttonState = digitalRead(buttonPin);

  //mengecek jika pushbutton ditekan
  //buttonState bernilai HIGH atau di tekan maka,
  if (buttonState == HIGH)
  {
    //LED Menyala
    digitalWrite(ledPin, HIGH);}
  else {
    // turn LED off:
    digitalWrite(ledPin, LOW);
  }
}
```

# Latihan 5 - Pushbutton Switch

Latihan ini hampir sama dengan latihan sebelumnya, komponen yang digunakanpun sama dengan komponen sebelumnya, akan tetapi pada kali ini kita akan menggunakan pushbutton sebagai switch yang dimana akan menghidupkan LED (atau apapun yang terhubung ke pin LED) dari keadaan mati atau mematikan LED dari keadaan hidup hanya dengan menggunakan 1 pushbutton.


## Komponen yang diperlukan

					
Breadboard	1x LED Hijau	1x Resistor 220 $\Omega$	1x Resistor 10k	Push Button	Kabel Jumper

## Rangkaian

Pertama hubungkan 1 kaki pushbutton dihubungkan dengan 1 kaki resistor dan pin digital 2, kaki pushbutton yang lain dihubungkan dengan VCC dan satu kaki resistor yang lain di hubungkan dengan ground. Lihat contoh diagram rangkaian Latihan 4. Sedangkan untuk led kita bisa menghubungkannya dengan ke kaki 13 setelah melalui resistor 220 $\Omega$ , sedangkan kaki katoda LED dimasukan ke ground arduino, pada project ini bisa digunakan juga LED internal yang telah disediakan di papan arduino, sehingga kita tidak perlu menambahkan led lagi.

## Program



```
//Pushbutton Switch

int inPin = 2;
int outPin = 13;
int state = HIGH;
int reading;
int previous = LOW;
long time = 0;
long debounce = 200;

void setup()
{
    pinMode(inPin, INPUT);
    pinMode(outPin, OUTPUT);
}

void loop(){
    reading = digitalRead(inPin);
    if (reading == HIGH && previous == LOW && millis() - time
> debounce){
        if (state == HIGH)
            state = LOW;
        else
            state = HIGH;
        time = millis();
    }
    digitalWrite(outPin, state);
    previous = reading;
}
```



# Latihan 6 - Two Button and LED







Pada latihan sebelumnya kita sudah mencoba menggunakan pushbutton sebagai switch, pada latihan kali ini kita akan memodifikasi program dan skematik dari program pushbutton.

Latihan ini kita akan belajar cara mengatur tingkat kecerahan dari LED, program sebelumnya LED hanya menyala ketika posisi pin HIGH atau mati pada posisi LOW. Lalu bagaimana mengatur tingkat kecerahannya?

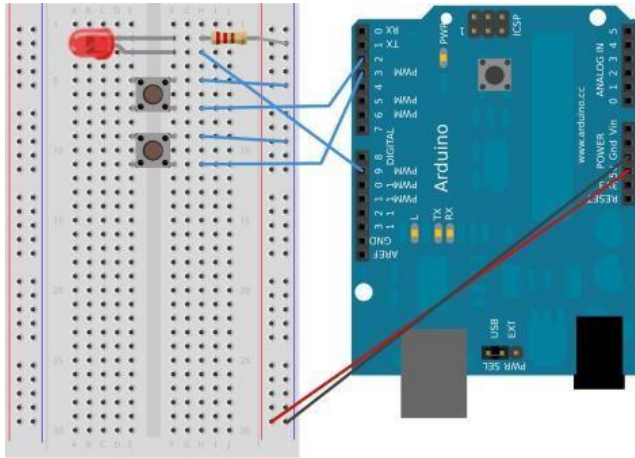
1. Dengan mengubah nilai resistor yang terhubung sebagai pembatas arus untuk LED
2. Menggunakan fungsi yang disebut dengan PWM (Pulse Width Modulation)

Untuk cara pertama bisa digunakan dengan tambahan komponen yaitu potensiometer atau variable resistor, sedangkan cara kedua kita hanya memanfaatkan fungsi dari board Arduino yaitu fungsi PWM, pada Arduino UNO sudah dilengkapi dengan pin PWM, pin yang terdapat simbol ~ pada papan Arduino yaitu pin 3, 4, 5, 9, 10 dan 11. Kita dapat memberikan rentang nilai 0 sampai 255. 0 berarti nilai tegangan pada pin tersebut bernilai 0V, sedangkan 255 tegangan pada pin tersebut bernilai 5v. Pada bahasa pemrograman Arduino nilai tersebut dipanggil menggunakan fungsi `analogWrite()`.

## Komponen yang diperlukan

					
Breadboard	1x LED	1x Resistor 220 $\Omega$	1x Resistor 10k	Push Button	Kabel Jumper

## Rangkaian



Made with  Fritzing.org

## Program

Rangkailah pushbutton, LED dan Resistor seperti pada rangkaian diatas, pushbutton 1 ke pin digital 2, pushbutton 2 ke pin digital 2 dan kaki Anoda LED ke pin PWM 9.

```
//Two Button and LED
const int kPinButton1 = 2;
const int kPinButton2 = 3;
const int kPinLed = 9;
int ledBrightness = 128; //nilai pertama LED

void setup()
{
  Serial.begin(9600);
  pinMode(kPinButton1, INPUT);
  pinMode(kPinButton2, INPUT);
  pinMode(kPinLed, OUTPUT);
  digitalWrite(kPinButton1, HIGH); //hidupkan pullup resistor
  digitalWrite(kPinButton2, HIGH); //hidupkan pullup resistor
}

void loop()
{
```

```

if(digitalRead(kPinButton1) == LOW){
    ledBrightness--;
}
else if(digitalRead(kPinButton2) == LOW){
    ledBrightness++;
}
ledBrightness = constrain(ledBrightness,0,255);
analogWrite(kPinLed, ledBrightness);
delay(20);
Serial.println(ledBrightness);
}





```

Untuk melakukan pembuktian setelah di upload buka serial monitor arduino IDE, setting baud rate ke 9600, maka akan terlihat nilai awal 128, nilai berikut merupakan nilai awal LED yang sudah di definisikan di program int ledBrightness = 128, lalu tekan salah satu button lihat hasilnya di serial monitor arduino.

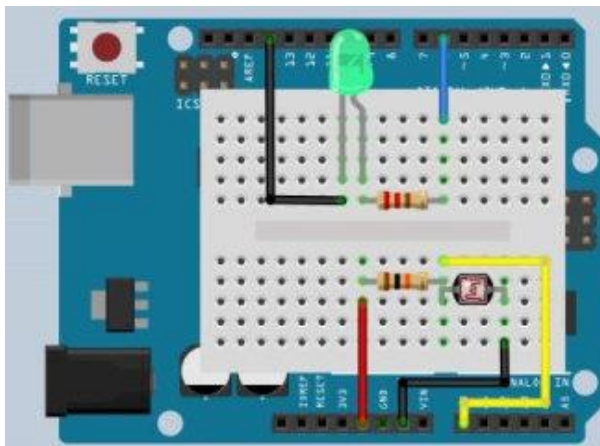
# Latihan 7 - Light Sensor

Pada latihan ini kita akan menggunakan LDR ( Light Dependent Resistor) untuk membaca nilainya dan merubah kecepatan kedipan pada led.

## Komponen yang diperlukan

				
Breadboard	LED Merah	LED Hijau	Resistor 220 $\Omega$	Kabel Jumper


## Rangkaian



Hubungkan seri LDR dan resistor 10K lalu diantara keduanya terhubung ke input analog A0, berikut merupakan sebuah pembagi tegangan, seperti gambar diatas, selanjutnya hubungkan LED kaki anoda ke pin digital 6 melalui resistor, dan kaki katoda ke ground.

## Program

Masukan kode berikut lalu upload. Setelah sukses kita akan melihat LED berkedip on dan off. Jika kita tutupi LDR (dalam keadaan gelap) maka LED akan berkedip lebih lambat dari sebelumnya dan jika kita terangi LDR menggunakan lampu maka LED akan terlihat berkedip lebih cepat dari sebelumnya.



```
//Light Sensor
int ledPin = 6; //pin LED
int ldrPin = 0; //pin LDR
int lightVal = 0; //nilai yang akan di baca LDR






void setup()
{
  pinMode(ledPin, OUTPUT);
}
void loop()
{
  //membaca nilai dari LDR
  lightVal = analogRead(ldrPin);
  digitalWrite(ledPin, HIGH); //Hidupkan LED
  //delay di tentukan dari pembacaan nilai LDR
  delay(lightVal);
  digitalWrite(ledPin, LOW); //Matikan LED
  delay(lightVal);
}
```

Setelah melakukan percobaan diatas maka coba tukar posisi LDR dengan resistor 10k, posisi kaki LDR langsung ke supply sedangkan kaki resistor terhubung ke ground. Amati hasilnya.

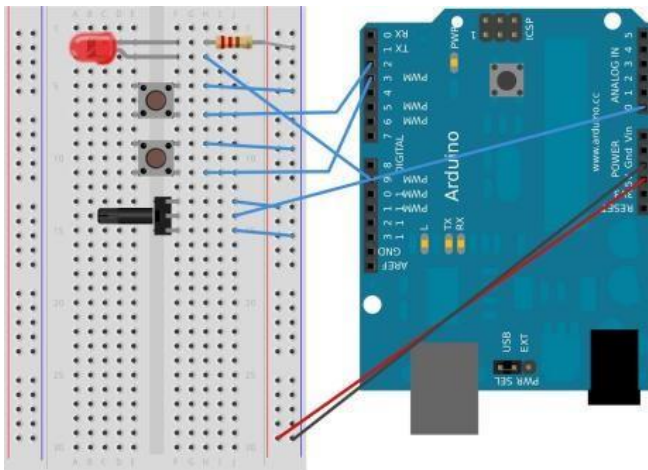
# Latihan 8 - Potensiometer

Jika tadi kita menggunakan pushbutton maka pada latihan kali ini kita akan menggunakan potensio sebagai pengatur nilai PWM. Potensio merupakan sebuah resistor yang nilainya bisa diatur berdasarkan putaran pada knob potensio, nilai yang dihasilkan bisa bervariasi tergantung nilai maksimum yang tertera pada potensio, contoh jika nilai yang tertera pada potensio bernilai 10k, maka itu merupakan nilai maximum dari sebuah potensio, dengan kata lain jika potensio diputar berlawanan arah nilai maximum maka nilai akan berkurang sampai menuju 0 ohm.

## Komponen yang diperlukan


				
Breadboard	Potensiometer 10k	LED	Resistor 220 $\Omega$	Kabel Jumper

## Rangkaian



Hubungkan kaki anoda led ke pin 9 dan kaki katoda LED ke kaki resitor yang dilanjutkan ke VCC, dan pin tengah potensiometer ke pin Analog 0 (A0).

## Program



```
//Potensiometer
const int PinPot = A0;
const int PinLed = 9;
void setup()
{
  pinMode(PinPot, INPUT);
  pinMode(PinLed, OUTPUT);
}
void loop()
{
  int ledBrightness;
  int sensorValue = 0;
  sensorValue = analogRead(PinPot);
  ledBrightness = map(sensorValue, 0, 1023, 0, 255);
  analogWrite(PinLed, ledBrightness);
}
```

Pada code diatas terdapat code `map(sensorValue, 0, 1023, 0, 255);` kode tersebut mengubah nilai 0 sampai 1023 ke 0 sampai 255, berikut fungsi dari code diatas `map(value, fromLow, fromHigh, toLow, toHigh)` mengapa dirubah ? karena pada pin PWM hanya membaca nilai dari rentan 0 sampai 255, sedangkan pada input analog A0 bisa membaca nilai dari 0 sampai 1023.

# Latihan 9 - Servo

Para projek kali ini kita akan menggunakan motor servo sebagai aktuator, hal pertama yang perlu kita ketahui adalah bagaimana cara menggerakkan motor servo menggunakan arduino.

## Komponen yang di perlukan

		
Breadboard	Servo SG9g/90	Kabel Jumper

## Rangkaian

Pada servo umumnya terdapat 3 kabel yaitu kabel supply, ground dan data, warna kabel setiap servo terkadang berbeda, tapi pada kali ini kita akan menggunakan motor servo Merk Toper Pro 90/9g, pada servo ini kabel warna coklat untuk ground, merah untuk supply dan orange untuk data, hubungkan ketiga kabel tersebut ke arduino seperti pada gambar.

Pada program kali ini kita akan memutar kondisi servo dari 0 derajat sampai 180 derajat, kanan dan kiri, begitu seterusnya.



## Program



```
//Servo Sweep
#include <Servo.h>
Servo myservo; //definisikan variabel myservo
sebagai servo

int pos = 0;    //variabel untuk memposisikan servo

void setup() {
  myservo.attach(8); //pin servo
}

void loop() {
  /*mulai dari step 0 derajat sampai step 180
  derajat, setiap step bertambah 1*/
  for (pos = 0; pos <= 180; pos += 1) {
    //tuliskan nilai variabel 'pos' pada servo
    myservo.write(pos);
    delay(15);
  }
  /*mulai dari step 180 derajat sampai step 0
  derajat, setiap step berkurang 1*/
  for (pos = 180; pos >= 0; pos -= 1) {
    //tuliskan nilai variabel 'pos' pada servo
    myservo.write(pos);
    delay(15);
  }
}
```

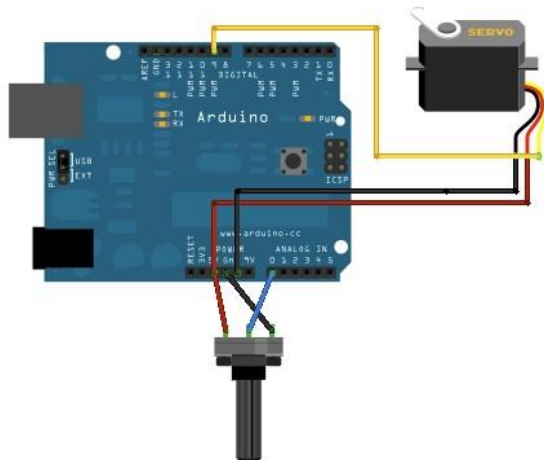
# Latihan 10 - Servo Control

Latihan selanjutnya lanjutan dari latihan servo sebelumnya, akan tetapi jika pada latihan sebelumnya posisi servo sudah di atur untuk menempati posisi tertentu, kali ini kita akan mengatur posisi servo menggunakan potensiometer.

## Komponen yang diperlukan


			
Breadboard	Servo SG9g/90	Potensiometer 10k	Kabel Jumper

## Rangkaian



Hubungkan pin data servo ke pin 8, lalu hubungkan kaki 2 potensio (Posisi tengah) ke pin analog 0 arduino

## Program



```
//Servo Control
#include <Servo.h>
Servo myservo;
int potpin = 0; //pin analog terkoneksi ke potensiometer
int val; //variabel untuk membaca nilai dari pin analog

void setup() {
    myservo.attach(8); //pin servo
}

void loop() {
    val = analogRead(potpin);
    //membaca nilai servo diantara 0 sampai 1023
    val = map(val, 0, 1023, 0, 180);
    //perubahan skala di perlukan untuk mengkonversi
    //dari 0 - 1023 ke 0 - 180
    myservo.write(val);
    delay(15);
}
```




## LATIHAN

1. Gunakan sensor analog lain untuk merubah posisi servo, implementasikan!

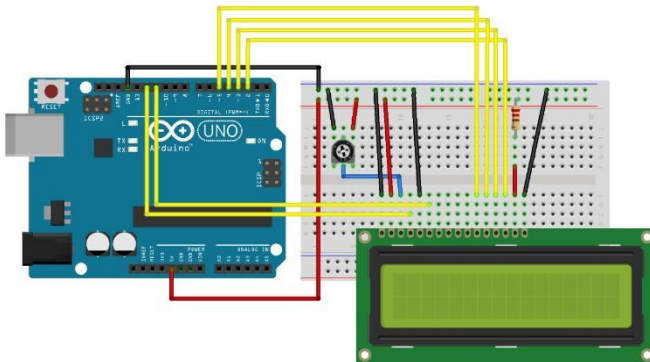
# Latihan 11 - LCD 16x2

Latihan kali ini akan menggunakan LCD sebagai penampil data, LCD yang digunakan pada starter kit ini yaitu menggunakan LCD karakter 16x2, 16x2 berarti terdapat 16 kolom dan 2 baris.

## Komponen yang diperlukan

			
Breadboard	LCD 16x2	Potensiometer 10k	Kabel Jumper

## Rangkaian



Koneksikan pin LCD ke pin Arduino sesuai urutan berikut

1. LCD RS pin (pin 4 lcd) to digital pin 12
2. LCD R/W pin (pin 5 lcd) to ground
3. LCD Enable pin (pin 6 lcd) to digital pin 11
4. LCD D4 pin (pin 11 lcd) to digital pin 5
5. LCD D5 pin (pin 12 lcd) to digital pin 4
6. LCD D6 pin (pin 13 lcd) to digital pin 3

7. LCD D7 pin (pin 14 lcd) to digital pin 2
8. LCD VSS pin (pin 1 lcd) to ground
9. LCD VCC pin (pin 2 lcd) to 5V
10. LCD Brightness pin (pin 3 lcd) to potensio pin 2
11. LCD Backlight + pin (pin 15 lcd) to +5V
12. LCD Backlight - pin (pin 16 lcd) to ground

### Penting!

Perlu diperhatikan posisi kaki pertama pin LCD, jangan sampai terbalik menghubungkan pin LCD, pin satu dan 16 LCD biasanya diberi tanda pada bagian setiap ujung pin pada LCD

### Program



```
//LCD 16x2
#include <LiquidCrystal.h> //Library LCD


//inisialisasi nomor pin LCD
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);

void setup() {
    //set kolom dan baris
    lcd.begin(16, 2);
    //tampilkan kata
    lcd.print("hello, world!");
}

void loop() {
    lcd.setCursor(0, 1);
    //cursor di posisikan pada kolom 0 baris 1
    lcd.print(millis() / 1000);
}
```

Maka hasil yang di dapat yaitu LCD pada kolom 0 baris 0 akan menampilkan tulisan hello, world!, (karena kolom dan baris pertama dimulai dari angka 0, jadi kolom/baris 1 = kolom/baris 0, kolo/baris 2 = kolom/baris 1) dan di kolom pertama baris 1 akan menampilkan hitungan dalam detik. Seperti program lainnya, LCD dijalankan menggunakan perintah perintah khusus yang sudah di definisikan di library LiquidCrystal.h seperti perintah lcd.print() untuk menampilkan data dan perintah lcd.setCursor(0,1) yaitu menempatkan kursor pada kolom 0 baris 1, berikut program-program LCD lainnya. Untuk skematik sama seperti skematik program LCD.

## LCD - Blink



```
#include <LiquidCrystal.h>

LiquidCrystal lcd(12, 11, 5, 4, 3, 2);

void setup() {
  //set kolom dan baris
  lcd.begin(16, 2);
  // tampilkan kata
  lcd.print("hello, world!");
}

void loop() {
  //matikan cursor blinking
  lcd.noBlink();
  delay(3000);
  //hidupkan cursor blinking
  lcd.blink();
  delay(3000);
}
```

## LCD – Serial Input



```
//LCD Serial Input
#include <LiquidCrystal.h>
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);

void setup() {
  lcd.begin(16, 2);
  Serial.begin(9600);
}

void loop() {
  //jika karakter di terima oleh serial port
  if (Serial.available()) {
    //menunggu pesan masuk
    delay(100);
    lcd.clear(); //menghapus LCD
    // membaca semua karakter yang masuk
    while (Serial.available() > 0) {
      //menulis ke LCD dari hasil bacaan serial yang masuk
      lcd.write(Serial.read());
    }
  }
}
```

Setelah terupload buka serial monitor lalu tulis sesuatu di halaman serial, setelah dikirim maka tulisan tersebut akan tampil di LCD

## LCD – Auto Scroll



```
#include <LiquidCrystal.h>
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);

void setup() {
  lcd.begin(16, 2);
}

void loop() {
  //set kursor pada kolom dan baris 0
  lcd.setCursor(0, 0);
  // print dari 0 sampai 9:
  for (int thisChar = 0; thisChar < 10; thisChar++) {
    lcd.print(thisChar);
    delay(500);
  }




  //set kursor pada kolom 16 dan baris 1
  lcd.setCursor(16, 1);
  //set LCD untuk autoscroll
  lcd.autoscroll();
  // print dari 0 sampai 9:
  for (int thisChar = 0; thisChar < 10; thisChar++) {
    lcd.print(thisChar);
    delay(500);
  }
  //matikan automatic scrolling
  lcd.noAutoscroll();
  //bersihkan LCD untuk loop selanjutnya
  lcd.clear();
}
```



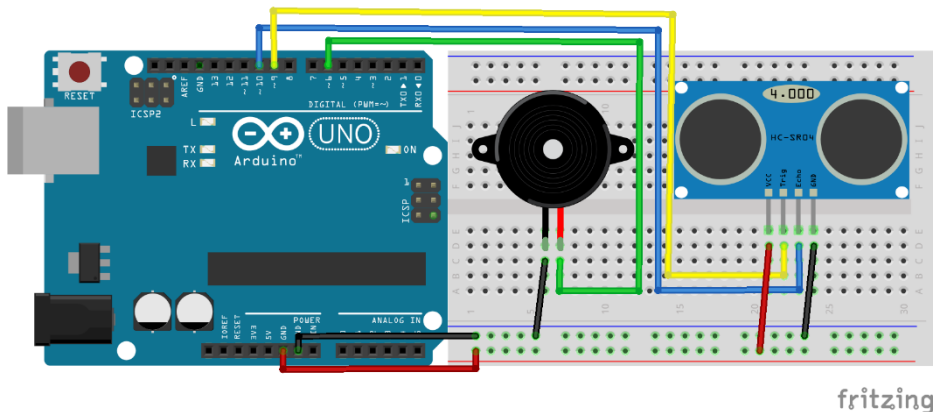
# Latihan 12 - Ultrasonic

Kali ini kita akan menggunakan sensor ultrasonic sebagai penghitung jarak, jarak akan di tampilkan menggunakan ukuran cm atau inch, jarak yang bisa terhitung normal oleh sensor ini berkisar 3 cm sampai 300 cm lebih dari itu terkadang mendapat error yang lebih besar.

## Komponen yang diperlukan

		
Breadboard	Sensor Ultrasonik	Kabel Jumper

## Rangkaian

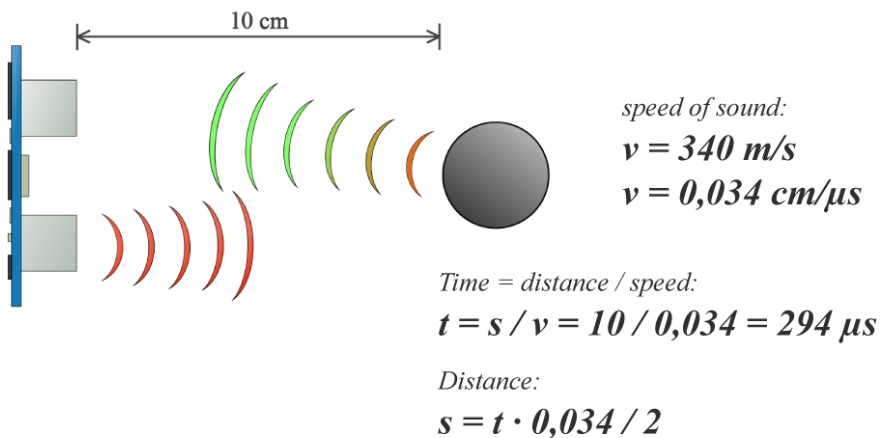


Pada sensor Ultrasonik HC-SR04 memiliki 4 pin yaitu pin vcc, ground,

trigger dan echo. Hubungkan pin vcc dan ground dari sensor ke arduino, pin trigger ke pin 9 dan pin echo ke pin 10 arduino.

Cara kerja singkat dari sensor ini yaitu ketika pin trigger di beri sinyal pendek (5v) dari mikrokontroler maka sensor mengeluarkan gelombang suara, dan ketika gelombang tersebut memantul mengenai dinding atau objek lainya gelombang tersebut kembali, sensor akan mengirim sinyal balik ke arduino melalui pin echo. Arduino mengukur waktu antara transmisi dan kembalinya gelombang suara dan mengonversi waktu tersebut menjadi jarak.

Perhitungan untuk mendapatkan jarak dari sensor ultrasonik ini sebagai berikut



## Program




```
const int trigPin = 9; //pin trigger
const int echoPin = 10; //pin echo
long duration;
int distance;

void setup() {
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);
  Serial.begin(9600);
}

void loop() {
  // Clears pin trigger
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  //set pin trigger selama 10 microsecond
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);
  //membaca jarak yang dihasilkan
  duration = pulseIn(echoPin, HIGH);
  //memperkirakan jarak
  distance= (duration*0.034)/2;
  //tampilkan diserial monitor
  Serial.print("Distance: ");
  Serial.print(distance);
  Serial.println("cm");
}
```

Project selanjutnya masih menggunakan jarak akan tetapi kita akan menambah komponen yaitu buzzer sebagai tanda pada L ini. Kita akan menentukan jarak tertentu yang di mana jika jarak tersebut di lewati maka buzzer akan berbunyi.


### Komponen yang diperlukan

			
Breadboard	Sensor Ultrasonik	Buzzer	Kabel Jumper

### Rangkaian

Sama seperti skematik latihan sebelumnya, akan tetapi kita akan menambahkan buzzer, kaki positif buzzer dihubungkan ke pin digital 6 arduino, dan pin negatif ke ground. Perhatikan lambang positif pada buzzer menunjukan kaki positif pada buzzer.

### Program



```

const int trigPin = 9; //pin trigger
const int echoPin = 10; //pin echo
int piezo=6; //Piezo speaker di pin 6.
long duration;
int distance;

void setup() {
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);
  pinMode(piezo, OUTPUT);
  Serial.begin(9600);
}

```

```




void loop() {
  // Clears pin trigger
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  //set pin trigger selama 10 microsecond
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);
  //membaca jarak yang dihasilkan
  duration = pulseIn(echoPin, HIGH);
  //memperkirakan jarak
  distance= (duration*0.034)/2;
  //tampilkan diserial monitor

  if (distance >= 200 || distance <= 0)
  {
    Serial.println("No measurement");
  }
  else
  {
    Serial.print("Distance: ");
    Serial.print(distance);
    Serial.println("cm");
  }
  if (distance <= 6)
  //Jika jarak kurang atau sama dengan 6cm
  {
    digitalWrite(piezo,HIGH);
    //piezo akan berbunyi
  }
  else //jika tidak
  {
    digitalWrite(piezo,LOW);
    //piezo berhenti berbunyi
  }
}

```

# Latihan 13 - Ultrasonic Reverse Warning System

## Komponen yang diperlukan

		
Breadboard	Sensor Ultrasonik	Kabel Jumper

## Rangkaian

Masih menggunakan skematik yang sama akan tetapi ditambahkan LED sebagai penanda, hubungkan led ke pin digital 13 pada arduino, atau bisa menggunakan LED internal yang terdapat di Arduino.

Cara kerjanya yaitu ketika posisi jarak ultrasonik sudah pada jarak tertentu maka led akan berkedip, jika makin dekat kedipan led akan semakin cepat, begitu sebaliknya.

## Program



```
const int trigPin = 9; //pin trigger
const int echoPin = 10; //pin echo
int piezo=6; //Piezo speaker di pin 5
int led=13; //Pin LED
long duration;
int distance;

void setup() {
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);
  pinMode(piezo, OUTPUT);
  pinMode(led, OUTPUT);
  Serial.begin(9600);
}
```

```

}

void loop() {
    digitalWrite(trigPin, LOW); //Clears pin trigger
    delayMicroseconds(2);
    //set pin trigger selama 10 microsecond
    digitalWrite(trigPin, HIGH);
    delayMicroseconds(10);
    digitalWrite(trigPin, LOW);
    //membaca jarak yang dihasilkan
    duration = pulseIn(echoPin, HIGH);
    distance= (duration*0.034)/2; //memperkirakan jarak




    if (distance >= 200 || distance <= 0){
        Serial.println("No measurement");
    }
    else{
        Serial.print("Distance: ");
        Serial.print(distance);
        Serial.println("cm");
    }
    if (distance <= 20){
        //Jika jarak kurang atau sama dengan 20cm
        //piezo dan led akan hidup
        digitalWrite(piezo,HIGH);
        digitalWrite(led, HIGH);
        delay(distance*10);
        digitalWrite(piezo,LOW);
        digitalWrite(led, LOW);
        delay(distance*10);
    }
    else { //jika tidak
        digitalWrite(piezo,LOW);
        digitalWrite(led, LOW);
    }
}

```

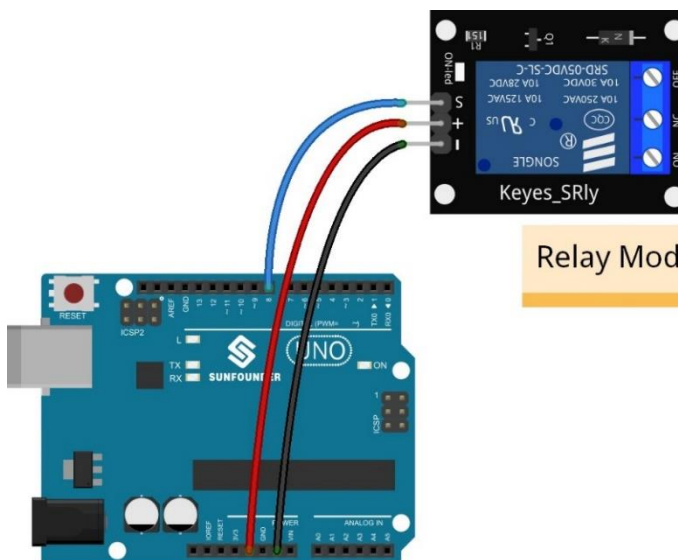
# Latihan 14 – Relay

Relay adalah salah satu aktuator yang banyak digunakan, fungsi relay digunakan untuk mengsaklar, yang bisa di saklar tidak hanya tegangan DC tapi juga tegangan AC dengan trigger yang kecil, penggunaanya hanya memberi logika High ata Low pada pin input relay.

## Komponen yang diperlukan

		
Breadboard	Relay Modul	Kabel Jumper

## Rangkaian




frit:



Hubungkan input vcc dan ground di modul relay ke arduino, lalu hubungkan pin IN ke pin 8 Arduino, pada modul relay ini mempunyai 3 pin input yaitu vcc, ground dan sinyal, sinyal ini diberi trigger oleh mikrokontroler 5V atau 0V.

Relay mempunyai 2 kondisi Normally Open (NO) dan Normally Close (NC), maksudnya adalah ketika posisi relay tidak mendapat tegangan input kondisi relay saat itu adalah tidak terhubung atau Normally open, sedangkan Normally Close sebaliknya ketika relay tidak mendapat tegangan input maka kondisi relay sudah terhubung.

## Program




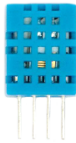

```
void setup()
{
    pinMode(8, OUTPUT);
}
void loop()
{
    digitalWrite(8, HIGH); //relay hidup
    delay(1000);
    digitalWrite(8, LOW); //matikan relay
    delay(1000);
}
```

Gunakan relay sebagai aktuator/output untuk latihan-latihan sebelumnya

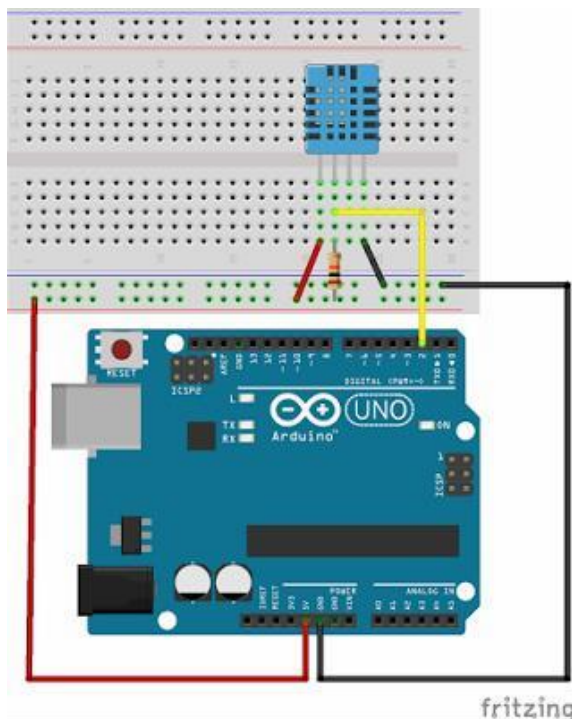
# Project 15 – DHT Temperature and Humidity

Pada latihan kali ini kita akan membuat sebuah pengukur suhu dan kelembaban menggunakan sensor DHT11.

## Komponen yang dibutuhkan

		
Breadboard	DHT11	Kabel Jumper

## Rangkaian



1. Hubungkan kaki ke-1 DHT11 ke 5V Arduino
2. Hubungkan kaki ke-2 DHT11 ke pin 2 Arduino

3. Dengan menggunakan **Resistor 10 Kilo Ohm**, hubungkan kaki ke-2 **DHT11** ke **5V Arduino**.
4. Hubungkan kaki ke-4 **DHT11** ke **GND Arduino**
5. Untuk kaki **Ke-3 DHT11** kita abaikan saja **Program**



```
#include <dht.h>
dht DHT;
#define DHT11_PIN 2 //pin dht
void setup()
{
  Serial.begin(9600);
  Serial.println("DHT11 PROGRAM ");
  Serial.print("LIBRARY VERSION: ");
  Serial.println(DHT_LIB_VERSION);
  Serial.println();
  Serial.println("Type,\tstatus,\tHumidity
(%) ,\tTemperature (C)");
}

void loop()
{
  // Membaca data dari DHT
  Serial.print("DHT11, \t");
  int chk = DHT.read11(DHT11_PIN);
  //cek data sensor DHT
  switch (chk)
  {
    case DHTLIB_OK:
      Serial.print("OK,\t");
```

```

break;
case DHTLIB_ERROR_CHECKSUM:
Serial.print("Checksum error,\t");
break;
case DHTLIB_ERROR_TIMEOUT:
Serial.print("Time out error,\t");
break;
case DHTLIB_ERROR_CONNECT:
    Serial.print("Connect error,\t");
    break;
case DHTLIB_ERROR_ACK_L:
    Serial.print("Ack Low error,\t");
    break;
case DHTLIB_ERROR_ACK_H:
    Serial.print("Ack High error,\t");
    break;
default:
Serial.print("Unknown error,\t");
break;
}
// Menampilkan data dari sensor DHT
Serial.print(DHT.humidity, 1);
Serial.print(",\t");
Serial.println(DHT.temperature, 1);
delay(2000);
}

```

Setelah di upload selanjutnya buka serial monitor maka akan tampil nilai kelembaban dan suhu.

## LATIHAN

1. Tampilkan kelembaban dan suhu pada LCD!
2. Buat program mengenai sensor DHT11 digabungkan LCD dan modul output (Motor, Relay, Buzzer, LED, dll)

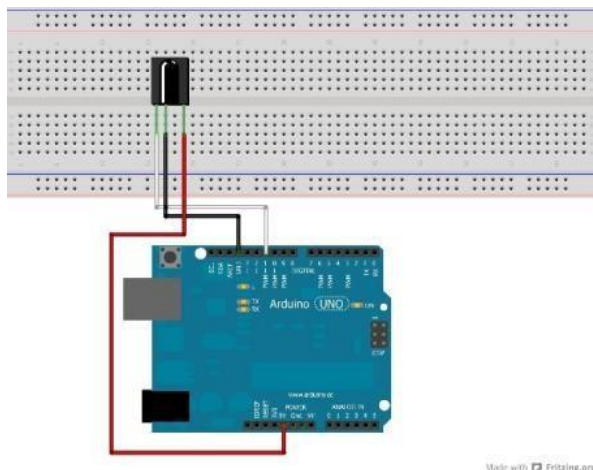
# Latihan 16 - Infrared

Pada latihan kali ini kita akan menggunakan sensor TL1838 sebagai penerima sinyal infrared, sensor ini memiliki 3 pin, konfigurasi dari setiap pin ini yaitu 1=out/signal, 2=ground dan 3=vcc (lihat datasheet). Hati-hati untuk sensor type lain susunan pin bisa berubah, posisi ground dan vcc bisa tertukar, jadi lihat datasheet untuk memastikan konfigurasi untuk setiap sensor.

## Komponen yang diperlukan

			
Breadboard	TL1838	Kabel Jumper	Remote Infrared (Remote Universal, Remot TV, dll)

## Rangkaian



Hubungkan pin 1 sensor ke pin 11 arduino, pin 2 ke ground serta 3 ke vcc.

## Program



```
//Infrared Receiver
#include <Irremote.h>
int RECV_PIN = 11; //pin sensor
IRrecv irrecv(RECV_PIN);
decode_results results;

void setup ()
{
  Serial.begin(9600);
  pinMode (13, OUTPUT);
  irrecv.enableIRIn();
}

void loop()
{
  if (irrecv.decode(&results)) {
    Serial.println(results.value, DEC);
    irrecv.resume();
  }
}
```

Setelah berhasil upload, silahkan buka serial monitor arduino lalu tekan sembarang tombol pada remote infrared dan arahkan ke sensor TL1838, maka akan tampil kode berbentuk angka desimal, setiap tombol yang ditekan akan menghasilkan angka desimal yang berbeda, angka itulah yang akan kita definisikan nanti.

Contoh ketika menekan tombol “1” maka muncul nilai desimal “1678996467”, Setiap remote infrared mempunyai nilai desimal yang berbeda, terkadang ada juga ketika tombol ditekan menghasilkan nilai yang tidak konstan atau bahkan muncul nilai “4294967295” ini bukan kode unik dari tombol tersebut, hal ini bisa terjadi biasanya apabila kita menekan tombol secara berkelanjutan/ditekan lama. Dengan kata lain sama seperti type sensor lain, remote pun punya spesifikasinya masing masing.

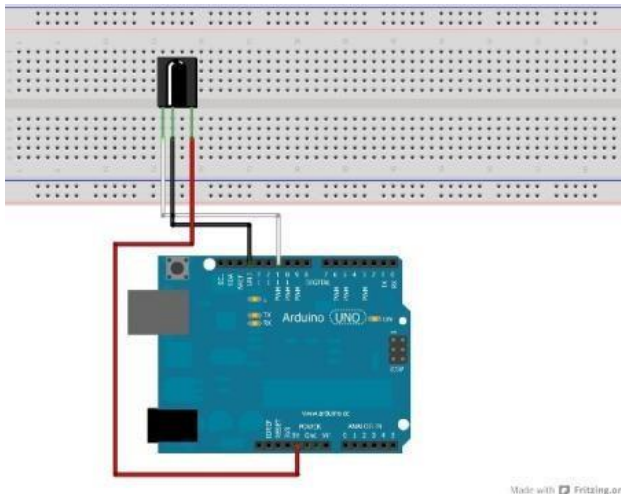
# Latihan 17 - Control LED Using Infrared

Pada latihan ini kita akan menggunakan remote infrared untuk mengendalikan LED.

## Komponen yang diperlukan

			
Breadboard	TL1838	Kabel Jumper	Remote Infrared (Remote Universal, Remot TV, dll)

## Rangkaian



Untuk skematik masih menggunakan skematik yang sama, kita juga bisa menggunakan LED internal pada arduino, yaitu pada pin digital 13 atau kita

bisa memasang LED lagi di pin yang lain seperti yang ada pada latihan sebelumnya.

## Program



```
#include <Irremote.h>
int RECV_PIN = 11;
IRrecv irrecv(RECV_PIN);
decode_results results;

void setup()
{
  Serial.begin(9600);
  pinMode (13, OUTPUT);
  digitalWrite(13, LOW);
  irrecv.enableIRIn();
}
void loop() {
  if (irrecv.decode(&results)) {
    Serial.println(results.value, DEC);
    if (results.value == 16724175){
      //nomor 16724175 didapatkan dari program
      sebelumnya
      //didefinisikan nomor 16724175 = nomor 1
      digitalWrite (13, HIGH);
    }//...led akan menyala

    if (results.value == 16718055)
      //nomor 16718055 didapatkan dari program
      sebelumnya
      //didefinisikan nomor 16718055 = nomor 1.
      //jika receiver menerima kode ini maka
      {digitalWrite (13, LOW);}
      //...led akan mati
    irrecv.resume(); // Receive the next value
  }
}
```