

# Basic Arduino #1

Zamisyak Oby

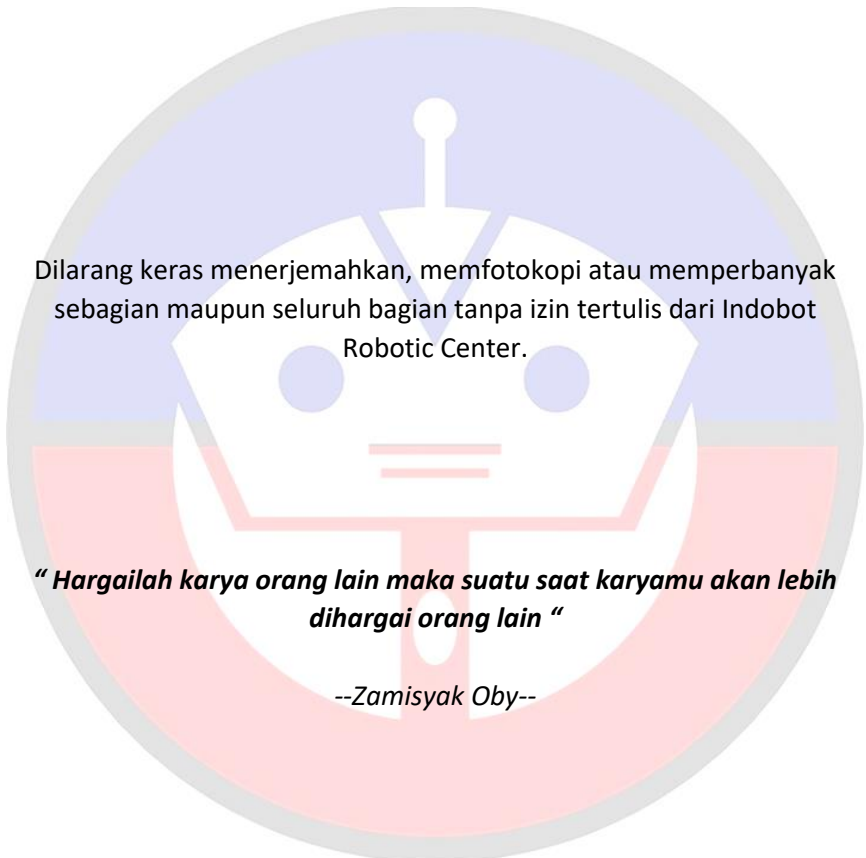


Yogyakarta, 2017

## Basic Arduino #1

Zamisyak Oby | Konsultan Project Robotic

© 2017, Indobot Robotic Center



Dilarang keras menerjemahkan, memfotokopi atau memperbanyak sebagian maupun seluruh bagian tanpa izin tertulis dari Indobot Robotic Center.

***“ Hargailah karya orang lain maka suatu saat karyamu akan lebih dihargai orang lain “***

*--Zamisyak Oby--*

# Kata Pengantar

Syukur Alhamdulillah penulis panjatkan pada Tuhan yang Maha Kuasa. Hanya karena anugerah-NYA Basic Arduino #1 ini dapat terselesaikan. Semoga tulisan ini dapat menjadi pedoman tambahan dalam belajar pemrograman Arduino.

Pemrograman sudah menjadi kebutuhan penting di era teknologi informasi ini. Meskipun sekarang sudah banyak alat – alat elektronik yang sifatnya otomatis sudah banyak beredar dipasaran, namun membuat program sendiri untuk kebutuhan yang lebih spesifik akan tetap diperlukan. Program komesil belum tentu sesuai dengan permasalahan yang akan dihadapi. Belajar elektronika dan coding tidak butuh waktu yang lama.

Pemrograman bukan pekerjaan yang sulit karena dengan Arduino semua bisa. Arduino sudah memiliki wadah komunitas secara global, dimana belajar pemrograman sudah bisa diakses secara global hanya dengan melalui internet.

Semakin hari kegiatan memprogram sudah menjadi kebutuhan tersendiri bukan hanya karena hobi, tapi karena tuntutan permasalahan yang sekarang ini harus diselesaikan dengan otomatisasi. Sebagian besar orang belajar memprogram dari *Learning by doing*. Memprogram membutuhkan metode yang baik supaya hasil program bisa maksimal. Buku Basic Arduino #1 akan mengajarkan tentang pemrograman yang baik dan mengupas tentang dunia Arduino.

Yogyakarta, November 2017

Zamisyak Oby

## Ucapan Terima Kasih

1. Alloh SWT, berkat izin dan hidayahnya sehingga mampu menyelesaikan buku ini.
2. Rasulullah SAW, yang menjadi suri tauladan dan panutan hingga akhir hayat nanti.
3. Ibukku tercinta, Darwiyah, yang selalu mendukung untuk menjadikan anak yang kuat dan mandapatkan yang terbaik.
4. Ayahku tercinta, Sukardi yang selalu mendukung sampai saat ini.
5. Kakakku, Iken Jhonatra dan Laode Kurnia Sandi yang luar biasa.
6. Tim Indobot Robotic Center, yang selalu kebersamai dalam mewujudkan mimpi besar bersama.
7. Mentor bisnis saya Coach Mugihardi, berkat bimbingan dan dukunganya bisnis ini menjadi lebih tumbuh.
8. Guru kehidupan saya, Pak Alfian yang sudah mengajari saya banyak hal sebagai bekal kehidupan nanti.
9. Guru imajiner saya, Dewa Eka Prayoga, Saptuari Sugiharto, Jaya Setiabudi dan seluruh penulis buku bisnis yang pernah saya baca. Berkat pemikiran anda yang luar biasa, pikiran saya jadi tercerahkan.
10. Arduino yang sudah merubah hidup saya menjadi lebih mudah. Luar biasa Arduino.
11. Anda, para pembaca buku ini yang luar biasa karena anda telah melakukan selangkah perubahan lebih baik lagi untuk menjadi progamer.

# Daftar Isi

Kata Pengantar .....	3
Ucapan Terima Kasih .....	4
Daftar Isi .....	5
Bab 1. Apa Itu Arduino?.....	7
1.1 Pengenalan Macam Arduino Board.....	9
1.2 Penggunaan Board Arduino Uno .....	10
1.3 Cara Instal Software Arduino IDE .....	13
1.4 Pengenalan Software Arduino IDE .....	16
Bab 2. Pemrograman Arduino.....	18
2.1 Struktur Utama .....	18
2.2 Ekspresi Bilangan .....	20
2.3 Struktur Kontrol .....	21
2.4 Perulangan.....	24
2.5 Syntax .....	27
2.6 Operasi Aritmatika.....	28
2.7 Operator Perbandingan .....	28
2.8 Operator Boolean .....	28
2.9 Operator Bitwise.....	29
2. 10 Operator Pertambahan dan Pengurangan .....	29
2. 11 Variabel.....	29
2.12 Tipe Data.....	30
2.13 Pin Input dan Output .....	31
2.13.1 Inisialisasi Fungsi Pin I/O.....	31

2.13.2 Menulis Data Digital di Pin Output .....	31
2.13.1 Membaca Data Digital pada Pin Input .....	32
2.13.3 Menulis Data Analog di Pin Output PWM .....	33
2.13.4 Membaca Data Analog di Pin Input ADC ( Analog to Digital Converter ) .....	33
2.14 Time .....	34
2.14.1 millis() .....	34
2.14.2 micros() .....	34
2.14.3 delay() .....	34
2.14.4 delayMicroseconds() .....	35
2.15 External Interrupts .....	35
2.15.1 attachInterrupt() .....	35
2.15.2 detachInterrupt() .....	38
2.16 Interrupts .....	38
2.16.1 interrupts() .....	38
2.16.2 noInterrupts() .....	38
2.17 Communication .....	39
2.17.1 Serial .....	39

# Bab 1. Apa Itu Arduino?



Arduino merupakan *platform prototyping open-source hardware* yang mudah digunakan dalam membuat suatu proyek berbasis pemrograman. Arduino Board mampu membaca inputan berupa sensor, tombol dan mengolah menjadi outputan seperti mengaktifkan motor, menyalakan LED dan sebagainya. Anda dapat memprogram Arduino Board dengan memberikan set instruksi tertentu dengan menggunakan *Arduino programming language*, dan Software Arduino (IDE).

## Mengapa Arduino?

Arduino dapat bekerja di Mac, Windows, dan Linux. Semua orang bisa membuat instrumen ilmiah menggunakan Arduino untuk membuktikan prinsip – prinsip kimia dan fisika, atau untuk memulai dengan pemrograman robotika. Jadi Arduino adalah salah satu kunci untuk belajar hal – hal baru. Siapapun seperti anak – anak, seniman, progamer dan penghobi elektronika dapat memulai menggunakan arduino hanya dengan mengikuti tutorial, kit maupun berdiskusi secara online dengan anggota komunitas Arduino baik di Facebook, Twitter maupun web [arduino.cc](http://arduino.cc) dan komunitas di daerah anda.

## Apa saja keuntungan menggunakan Arduino?

Arduino menyederhanakan proses bekerja dengan mikrokontroler, tetapi menawarkan berbagai keuntungan bagi guru, siswa, dan semua orang yang tertarik dengan Arduino seperti :

- **Murah** – Arduino Board relatif murah kalau di Indonesia dr harga Rp 100.000 – Rp 400.000, harga tersebut bisa kalian temukan dari Arduino yang clone sampai Arduino yang asli.
- **Cross-platform** – Software Arduino (IDE) lebih fleksibel karena dapat digunakan di Windows, Macintosh OSX, dan sistem operasi Linux. Kebanyakan software mikrokontroler hanya tersedia di Windows.
- **Sederhana untuk dipelajari** – Software Arduino (IDE) mudah digunakan untuk pemula dan tingkat lanjut.
- **Open Source dan Software extensible** – Perangkat lunak Arduino diterbitkan sebagai alat Open Source. Bahasa yang digunakan ialah bahasa C untuk AVR dan dapat dikembangkan lagi untuk membuat library melalui C++.
- **Open source dan hardware extensible** – Arduino Board diterbitkan di bawah lisensi Creative Commons, sehingga desainer sirkuit yang berpengalaman dapat membuat versi mereka sendiri, dan mengembangkan sendiri. Bahkan pengguna yang relatif tidak berpengalaman dapat membangun versi papan arduino untuk memahami cara kerjanya dan menghemat uang.



## 1.1 Pengenalan Macam Arduino Board



Arduino Uno



Arduino 101



Arduino Pro



Arduino Mega



Arduino Zero



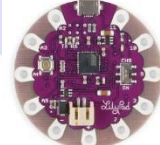
Arduino Due



Arduino Yun



Arduino Gemma



Arduino Lilypad USB



Arduino Pro Micro



Arduino Micro



Arduino Pro Mini

## 1.2 Penggunaan Board Arduino Uno

Arduino Uno adalah papan terbaik untuk memulai dengan belajar elektronik dan coding. Papan jenis ini yang paling kuat dan yang paling banyak digunakan dari seluruh keluarga Arduino.



Disini kita akan lebih sering menggunakan Arduino Uno karena lebih kuat dan banyak digunakan untuk memulai belajar elektronik dan coding.

Selain itu, beberapa pin memiliki fungsi khusus:

Uno memiliki 6 input analog, berlabel A0 melalui A5, yang masing-masing menyediakan 10 bit resolusi (yaitu 1024 nilai yang berbeda). Secara default mereka mengukur dari tanah ke 5 volt, meskipun adalah mungkin untuk mengubah batas atas dari kisaran mereka menggunakan pin AREF dan fungsi `analogReference()`.

AREF. tegangan referensi untuk input analog. Digunakan dengan `analogReference()`. Berikut spesifikasi lengkap Arduino Uno.

## Spesifikasinya

Mikrokontroler	ATmega328P
Tegangan operasi	5V
Input Voltage (dianjurkan)	7-9 V
Input Voltage (batas)	6-20V
Digital I / O Pins	14 (dimana 6 memberikan output PWM)
PWM Digital I / O Pins	6
Pins Masukan Analog	6
Arus DC per I / O Pin	20 mA
Arus untuk DC 3.3V Pin	50 mA
Flash Memory	32 KB (ATmega328P) yang 0,5 KB digunakan oleh bootloader
SRAM	2 KB (ATmega328P)
EEPROM	1 KB (ATmega328P)
Kecepatan jam	16 MHz

## Tegangan Kerja Arduino

Untuk Arduino yang mudah digunakan untuk belajar pertama kali adalah Arduino Uno. Cara menyakan arduino cukup mudah yaitu dengan menghubungkan port USB pada USB tipe B arduino dengan PC/Laptop atau bisa menggunakan tegangan eksternal melalui DC IN dengan tegangan yang dianjurkan 7 sampai 9V.

## Input dan Output

Setiap papan Arduino memiliki jumlah input dan output yang berbeda-beda. Kali ini yang akan dibahas adalah Arduino Uno. Berikut adalah table 1.1 untuk pin I/O Arduino Uno :

Tabel 1.1 pin I/O Arduino Uno

No Pin	Fungsi	Fungsi Lain
0	Digital I/O	Rx ( Serial Receiver )
1	Digital I/O	Tx ( Serial Transmitter )
2	Digital I/O	Interupsi Eksternal
3	Digital I/O	Interupsi Eksternal / PWM Timer 2
4	Digital I/O	-
5	Digital I/O	PWM Timer 0
6	Digital I/O	PWM Timer 0
7	Digital I/O	-
8	Digital I/O	-
9	Digital I/O	PWM Timer 1
10	Digital I/O	SPI – SS / PWM Timer 1
11	Digital I/O	SPI – MOSI / PWM Timer 1
12	Digital I/O	SPI – MISO
13	Digital I/O	SPI – SCK / LED

No Pin	Fungsi	Fungsi Lain
A0	Analog I/O	-
A1	Analog I/O	-
A2	Analog I/O	-
A3	Analog I/O	-
A4	Analog I/O	TWI – SDA
A5	Analog I/O	TwI – SCK

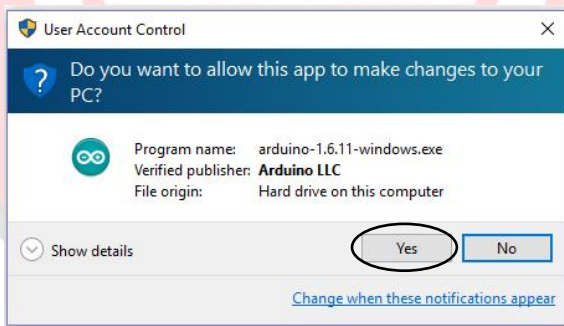
### 1.3 Cara Instal Software Arduino IDE

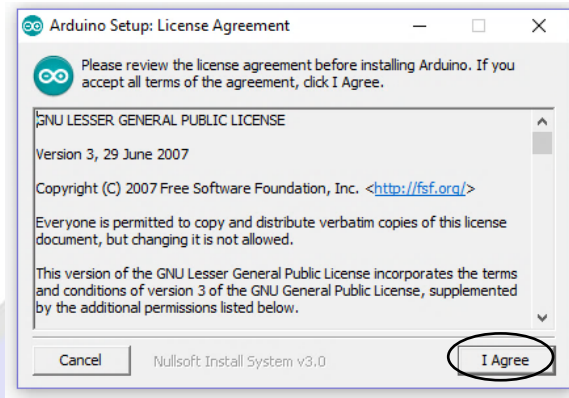
Cara menginstal Arduino IDE bisa langsung membuka file yang sudah disediakan dalam CD maupun download di <https://www.arduino.cc/en/Main/Software>.

Setelah itu mulai dengan step pertama. Buka Arduino.exe

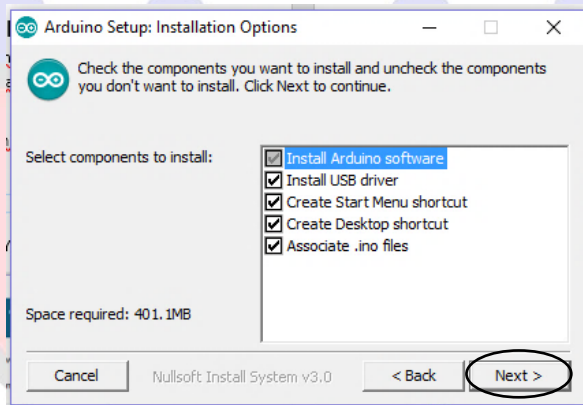
Name	Date modified	Type	Size
arduino-1.6.11-windows.exe	8/18/2016 9:45 PM	Application	86,047 KB

Lalu akan tampil seperti ini. Lalu klik Yes.

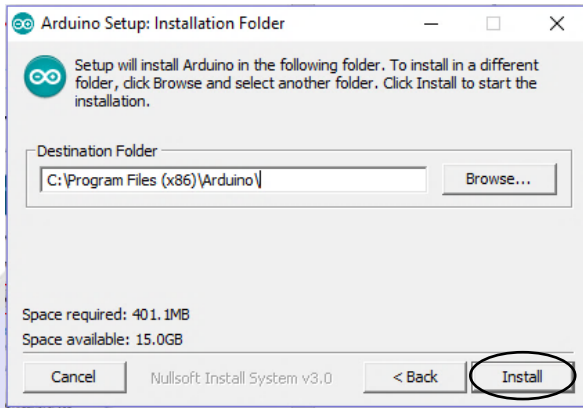




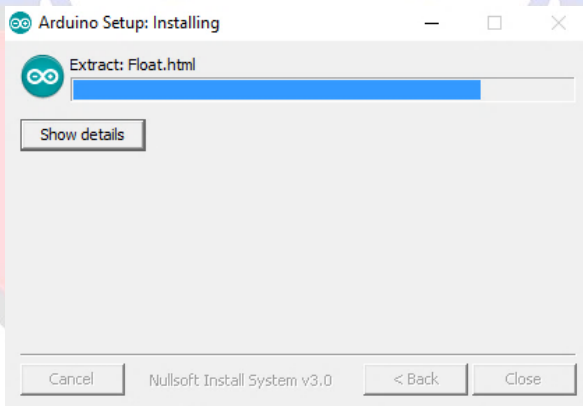
Setelah ini akan muncul seperti diatas Klik I Agree.



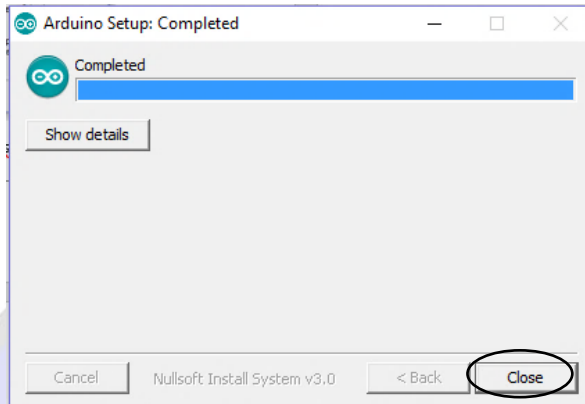
Kemudian centang semua dan klik next



Kemudian tentukan Destination Folder untuk Hasil Instalasinya. Disini bisa dipilih default pada C. Setelah itu klik install. Tunggu prosesnya hingga selesai.

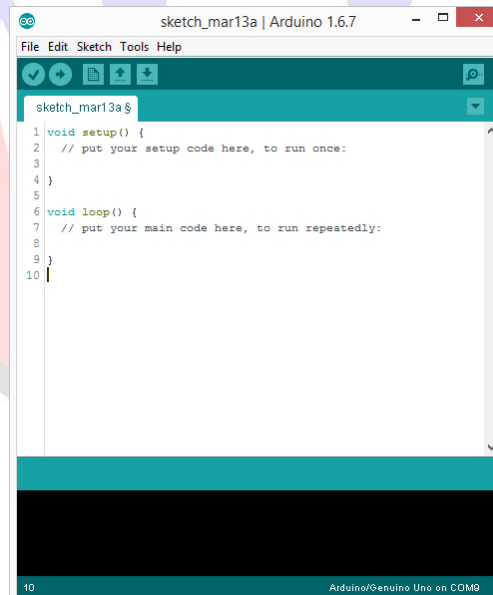


Setelah itu akan muncul completed jika sudah selesai.



Klik close setelah selesai dan sudah mulai bisa dibuka software arduinonya.

## 1.4 Pengenalan Software Arduino IDE



Gambar Tampilan Awal Arduino IDE



*Arduino Integrated Development Environment* - atau Arduino Software (IDE) - berisi editor teks untuk menulis kode, area pesan, konsol teks, toolbar dengan tombol untuk fungsi-fungsi umum dan serangkaian menu. Termasuk menghubungkan ke perangkat keras Arduino untuk meng-upload program dari komputer.



Verify

Memeriksa kode Anda untuk kesalahan kompilasi itu.



Upload

Mengkompilasi kode Anda dan upload ke papan dikonfigurasi. Lihat upload di bawah ini untuk rincian.

Catatan: Jika Anda menggunakan programmer eksternal dengan papan Anda, Anda dapat tekan "shift" tombol pada komputer Anda saat menggunakan ikon ini maka teks akan berubah menjadi *"Upload Using Programmer"*



New

Membuat sketsa baru.



Open

Membuka file yang sudah ada



Save

Mengamankan sketsa Anda.

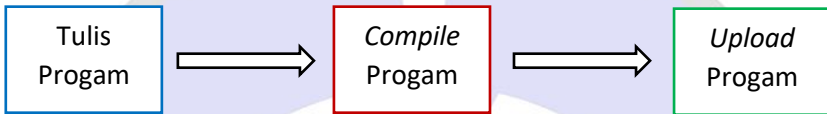


Serial Monitor

Membuka Monitor serial .

## Bab 2. Pemrograman Arduino

Pemrograman arduino menggunakan struktur Bahasa C. Mekanisme pemrogramannya arduino sama dengan mikrokontroler pada umumnya. Mulai dari membuat sket progam, meng-*compile*, selanjutnya proses *upload* pada papan arduino. Pengisian progam dengan metode upload ialah mengisi papan arduino dengan progam yang sudah berbentuk Hex atau hasil *compile* dari bahasa C ke bahasa mesin.



Program Arduino dapat dibagi dalam tiga bagian utama: struktur, nilai-nilai (variabel dan konstanta), dan fungsi.

### 2.1 Struktur Utama

- **Setup()**

Fungsi `setup()` dipanggil ketika sketsa progam dimulai. Fungsi ini digunakan untuk menginisialisasi variabel, mode pin, penggunaan librari, dll. Fungsi `setup()` hanya akan berjalan sekali, setelah power arduino dinyalakan atau saat mereset papan Arduino.

Contoh : Program 1.1

```
int ledPin = 13;
void setup(){
  pinMode(ledPin, OUTPUT);
  digitalWrite(ledPin, HIGH);
  delay(5000);
  digitalWrite(ledPin, HIGH);
}
void loop(){
  // ...
}
```

Program 1.1 akan menyalakan LED pada pin 13 selama 5 detik lalu mati. Eksekusi ini dilakukan hanya sekali.

- Loop()

Setelah membuat fungsi setup(), maka berikutnya adalah fungsi loop(). Fungsi loop() akan melakukan loop berturut-turut dimana program akan dijalankan terus menerus secara berurutan dan loop untuk mengontrol papan Arduino.

Contoh : Program 1.2

```
const int buttonPin = 3;

setup ()
{
  Serial.begin(9600);
  pinMode(buttonPin, INPUT);
}

loop()
{
  if (digitalRead (buttonPin) == HIGH)
    Serial.write ( 'H');
  else
    Serial.write ( 'L');
  delay (1000);
}
```

Program 1.2 pada serial monitor akan menampilkan huruf H ketika tombol pada pin 3 ditekan dan bila dilepaskan akan tampil huruf L.

- **//Komentar**

Komentar digunakan untuk memberikan keterangan pada program yang dibuat. Komentar tidak dieksekusi maka komentar tidak menambah ukuran file hasil *compile*. Cara membuat komentar ialah sebagai berikut :

```
//komentar segaris diawali dengan dua garis miring  
/*komentar untuk lebih dari satu baris diawali  
dengan garis miring lalu tanda bintang serta  
diakhiri dengan bintang lalu garis miring*/
```

## 2.2 Ekspresi Bilangan

Dalam pemrograman bahasa C pada arduino, bilangan dapat diekspresikan dalam beberapa format, yaitu :

- **Biner**  
Ditulis dengan awalan huruf '0b'. Contoh : b11110010
- **Desimal**  
Ditulis biasa tanpa awalan. Contoh : 435
- **Oktal**  
Ditulis dengan awalan angka '0'. Contoh : 0753
- **Heksadesimal**  
Diawali dengan '0x'. Contoh : 0x5A

## 2.3 Struktur Kontrol

Setiap program yang dibuat membutuhkan suatu kontrol. Tak hanya perulangan namun suatu eksekusi dengan syarat tertentu juga diperlukan.

### Pengujian Kondisi :

- **if**

Digunakan untuk mengecek suatu kondisi. Jika benar maka perintah didalam **if** akan dikerjakan.

```
if(kondisi){  
    Pernyataan / perintah  
}
```

Contoh :

```
if(x==6) {  
    a=a+5;  
}
```

- **if - else**

seperti dengan **if**, hanya saja ada 2 pilihan pernyataan / perintah. Jika kondisi benar maka perintah didalam **if** akan dikerjakan, jika kondisinya salah maka pernyataan didalam **else** lah yang akan dikerjakan

```
if(kondisi){  
    Pernyataan / perintah 1  
}  
else {  
    Pernyataan / perintah 2  
}
```

Contoh :

```
if(x==1) {  
    a=1;  
}  
else {  
    a=0;  
}
```

- **if - else if**

Untuk melakukan pengecekan suatu kondisi lebih dari satu maka bisa menggunakan **if - else if**.

```
if(kondisi1){  
    Pernyataan / perintah 1  
}  
else if(kondisi2){  
    Pernyataan / perintah 2  
}  
else if(kondisi ke-n){  
    Pernyataan / perintah ke-n  
}
```

Contoh :

```
if(x==1) {  
    a=1;  
}  
else if(x==2){  
    a=2;  
}  
else if(x==3){  
    a=3;  
}
```

- **switch case**

Pernyataan ini digunakan untuk memilih kondisi yang sesuai untuk kemudian mengerjakan perintahnya. Bedanya adalah kondisi yang diuji berupa sebuah nilai variable.

```
switch(variabel){ //variable yang diuji
  case 1  : //pernyataan/perintah 1
    break;
  case 2  : //pernyataan/perintah 2
    break;
  case n   : //pernyataan/perintah n
    break;
  default : //pernyataan/perintah default
}
```

Jika variable memenuhi syarat dari salah satu case maka dia akan mengerjakan pernyataan/perintah tersebut. Misal nilai variable = 2 maka dia kan mengerjakan pernyataan/perintah 2. Jika tidak memenuhi maka dia akan mengerjakan default.

Contoh :

```
switch(a){
  case 1  : digitalWrite(pin1,HIGH)
    break;
  case 2  : digitalWrite(pin2,HIGH)
    break;
  case 3  : digitalWrite(pin3,HIGH)
    break;
  default : digitalWrite(pin4,LOW)
}
```

## 2.4 Perulangan

- **while**

Perulangan ini digunakan untuk membuat perulangan yang tidak terbatas selama kondisi dalam **while** benar.

```
while(kondisi) {  
    //pernyataan/perintah  
}
```

Contoh :

```
while(a<200) {  
    a++;  
}
```

Perulangan while akan berhenti atau keluar setelah a mencapai angka 200.

- **do ... while**

Perulangan ini akan melakukan pernyataan /perintah lalu akan melihat kondisi dalam **while**. Jika benar maka pernyataan / perintah akan dieksekusi kembali.

```
do{  
    //pernyataan/perintah  
}  
while(kondisi);
```

Contoh :

```
do{  
    a++;  
}  
while(a<200);
```

Perulangan pertambahan a+1 akan dilakukan sampai nilai a=200.



- **for**

Digunakan untuk perulangan yang sifatnya terbatas.

```
for(inisialisasi;kondisi;step){  
    //pernyataan/perintah  
}
```

Contoh :

```
for(a=0;a<=10;a++){  
    Serial.println(a);  
}
```

**Inisialisasi** : nilai awal suatu variable untuk proses perulangan.

**Kondisi** : kondisi yang menentukan proses perulangan, jika benar perulangan dikerjakan.

**Step** : tahap perulangan bisa dalam bentuk perkalian, pertambahan, pengurangan dan pembagian.

Program tersebut akan menampilkan nilai a dari 0 sampai 10.

- **goto**

Perintah ini digunakan untuk melompat/menuju perintah yang telah diberi label.

```
goto label;
```

Contoh :

```
while(1) {  
    digitalWrite(pin0, HIGH);  
    delay(1000);  
    digitalWrite(pin0, LOW);  
    delay(1000);  
    if(digitalRead(pin1)==HIGH);  
    {goto keluar;}  
}  
  
keluar:
```

- **return**

Digunakan untuk memberikan nilai balik dari sebuah fungsi.

Contoh :

```
int data() {  
    if(analogRead(A0)>100) {  
        return 1;  
    }  
    else  
        return 0;  
}
```

- **continue**

untuk melewati perulangan yang tersisa dari struktur looping (do, for, atau while).

Contoh :

```
for (a=0; a<=255; a+10) {  
    if (digitalRead(pin0) == HIGH) {  
        continue;  
    }  
    digitalWrite(pwm1, a);  
    delay(100);  
}
```

- **break**

Perintah 'keluar' dari pernyataan perulangan do, for, atau while. Juga digunakan untuk mengakhiri pernyataan dalam switch – case.

## 2.5 Syntax

- **; (semicolon)**

Digunakan untuk mengakhiri sebuah pernyataan.

- **{ } (curly braces)**

Bagian utama dari bahasa pemrograman C yang digunakan dalam beberapa konstruksi yang berbeda dalam beberapa fungsi.

- **#define**

Komponen C yang berguna yang memungkinkan programmer untuk memberi nama untuk nilai konstan sebelum program dikompilasi.

- **#include**

Digunakan untuk memasukkan perpustakaan atau library di luar di sketsa program.

## 2.6 Operasi Aritmatika

Operator	Keterangan
=	Pemberian Nilai
+	Penjumlahan
-	Pengurangan
*	Perkalian
/	Pembagian
%	Sisa Bagian

## 2.7 Operator Perbandingan

Operator	Keterangan
==	Persamaan. Jika kedua nilai yang dibandingkan sama maka hasilnya 'true'
!=	Pertidaksamaan. Jika kedua nilai yang dibandingkan tidak sama hasilnya 'true'
>	Lebih Besar
<	Lebih Kecil
>=	Lebih Besar atau Sama Dengan
<=	Lebih Kecil atau Sama Dengan

## 2.8 Operator Boolean

Operator	Keterangan
&&	AND
	OR
!	NOT

## 2.9 Operator Bitwise

Bitwise Operator = Digunakan untuk operasi bit per bit pada nilai integer. Terdiri dari operator NOT, AND, OR, XOR, Shl, Shr. Type : int atau char

Bitwise operator, dari namanya sudah jelas diketahui pasti berhubungan dgn bit. Biasanya digunakan utk memanipulasi data bertipe bit dari tipe data integer atau byte.

Operator	Keterangan
<<	Geser Kiri
>>	Geser Kanan
&	AND
	OR
^	XOR
~	NOT

## 2. 10 Operator Pertambahan dan Pengurangan

Operator	Keterangan	Contoh	Keterangan
++	Pertambahan 1 / <i>increment</i>	a++	a = a + 1
--	Pengurangan 1 / <i>decrement</i>	a--	a = a - 1

## 2. 11 Variabel

Variabel adalah suatu wadah untuk menyimpan atau menampung data. Nama variable dibebaskan namun ada peraturan tersendiri seperti tidak boleh ada spasi, maksimal 32 karakter dan tidak boleh menggunakan istilah baku dalam bahasa C arduino karena dapat tercaji progam yang error.

Cara emndeklarasikan variable sebelum digunakan sebagai berikut :

```
int nilai_1;
```

atau bisa diisi dengan nilai :

```
[tipe data][spasi][nama variable][=][nilai]
```

Contoh :

```
int nilai_1=17; //variabel bilangan tipe  
integer diisi nilai 17
```

## 2.12 Tipe Data

Tipe data yang berbeda – beda memiliki kapasitas penyimpanan yang berbeda – beda pula. Berikut tipe data tersebut :

tipe data	Lebar Data	Jangkauan
char	1 byte	-128 s/d 127
unsigned char	1 byte	0 s/d 255
byte	1 byte	0 s/d 255
word	2 byte	0 s/d 65535
int	2 byte	-32768 s/d 32767
unsigned int	2 byte	0 s/d 65535
long	4 byte	-2147438648 s/d 2147438647
unsigned long	4 byte	0 s/d 4294967295
float	4 Byte	-3.4028235E+38 s/d 3.4028235E+38

## 2.13 Pin Input dan Output

Pada oaoan Arduino Uno terdapat 20 pin I/O yaitu 14 pin digital dan 6 pin analog.

### 2.13.1 Inisialisasi Fungsi Pin I/O

Pada saat ynagn sama, sebuah pin hanya bisa memiliki satu fungsi saja baik input maupun output. Inisialisasi ini dilakukan pada fungsi `setup()`, dengan cara :

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

- Pin : nomor pin yang dikonfigurasi dari papan arduino.
- Mode : INPUT, INPUT\_PULLUP, OUTPUT.

Sebagai Contohnya jika pin no 3 akan dibuat menjadi Inputan maka :

**`pinMode (3 , INPUT) ;`**

Bila pin 3 menjadi Outputan maka :

**`pinMode (3 , OUTPUT) ;`**

\*penulisan besar dan kecilnya huruf sangat berpengaruh. Perhatikan dengan seksama saat menulis progam.

### 2.13.2 Menulis Data Digital di Pin Output

Setelah membuat pin sebagai digital output, selanjutnya untuk menulis atau mengeluarkan logika data digital dengan perintah sebagai berikut :

**`digitalWrite(pin,value);`**

- Pin : nomor pin digital output.
- Value : HIGH atau LOW.

Sebagai Contoh :

```
pinMode(3,OUTPUT);  
  
digitalWrite(3,HIGH);
```

### 2.13.1 Membaca Data Digital pada Pin Input

Jika sebuah pin dibuat sebagai inputan maka kita harus menentukan aktif HIGH atau aktif LOW. Jika aktif HIGH maka dibutuhkan resistor pulldown. Jika memilih aktif LOW, cukup dengan memanggil resistor internal dengan pullup pada setiap pin arduino.

Sebelum melakukan pembacaan maka perlu disetting untuk Inputanya.

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

- Pin : nomor pin yang dikonfigurasi dari papan arduino.
- Mode : INPUT, INPUT\_PULLUP.

Setelah itu baru menuliskan ini :

```
digitalRead(pin) ;
```

- Pin : nomor pin arduino yang digunakan sebagai inputan.

Contoh : Pin 3 digunakan sebagai inputan pullup.

```
int baca;  
  
pinMode(3,INPUT_PULLUP) ;  
  
baca = digitalRead(3) ;  
  
Serial.println(baca) ;
```

Hasil pembacaan pin 3 maka disimpan pada variabel baca.



### 2.13.3 Menulis Data Analog di Pin Output PWM

Untuk menggunakan `analogWrite()`, tidak perlu menggunakan `pinMode()` untuk mengatur pin sebagai output.

Cara menggunakannya sebagai berikut :

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

- Pin : nomor pin arduino yang digunakan sebagai outputan lihat table 1.1.
- Value : nilai pwm mulai dari 0-255.

Contoh : Pin 5 digunakan sebagai outputan pwm.

```
analogWrite (5,100) ;
```

### 2.13.4 Membaca Data Analog di Pin Input ADC ( Analog to Digital Converter )

Untuk menggunakan `analogRead()`, tidak perlu menggunakan `pinMode()` untuk mengatur pin sebagai input.

Cara menggunakannya sebagai berikut :

```
analogRead (analogPin) ;
```

- analogPin : nomor pin arduino yang digunakan sebagai inputan analog (A0, A1, A2, A3, A4, A5).

Contoh : Membaca nilai analog pada analogPin A0 dan ditampilkan pada Serial.

```
int val = analogRead(A0) ;
```

```
Serial.println(val) ;
```

## 2.14 Time

### 2.14.1 millis()

Menghitung dengan satuan miliseconds sejak papan Arduino mulai menjalankan program hingga 50 hari setelah itu akan kembali ke nol begitu selanjutnya.

Contoh :

```
unsigned long time = millis();  
Serial.println(time);
```

### 2.14.2 micros()

Menghitung dengan satuan microseconds sejak papan Arduino mulai menjalankan program hingga 70 menit setelah itu akan kembali ke nol begitu selanjutnya.

Contoh :

```
unsigned long time = micros();  
Serial.println(time);
```

### 2.14.3 delay()

Jeda program untuk jumlah waktu (dalam milidetik). (Ada 1000 milidetik dalam satu detik.)

Contoh :

```
digitalWrite (ledPin, TINGGI); // set LED on  
delay (1000); // menunggu untuk kedua  
digitalWrite (ledPin, LOW); // set LED off  
delay (1000); // menunggu untuk kedua
```

#### 2.14.4 delayMicroseconds()

Jeda program untuk jumlah waktu (dalam mikrodetik). Ada seribu mikrodetik di milidetik, dan satu juta mikrodetik dalam detik.

Contoh :

```
digitalWrite (ledPin, TINGGI); // set LED on
delayMicroseconds(50); // menunggu
digitalWrite (ledPin, LOW); // set LED off
delayMicroseconds(50); // menunggu
```

### 2.15 External Interrupts

#### 2.15.1 attachInterrupt()

Anda harus menggunakan digitalPinToInterrupt (pin) untuk menerjemahkan pin digital sebenarnya untuk jumlah interrupt tertentu.

Misalnya, jika Anda terhubung ke pin 3, menggunakan digitalPinToInterrupt (3) sebagai parameter pertama yang attachInterrupt.

Board	Digital Pins Usable For Interrupts
Uno, Nano, Mini, other 328-based	2, 3
Mega, Mega2560, MegaADK	2, 3, 18, 19, 20, 21
Micro, Leonardo, other 32u4-based	0, 1, 2, 3, 7
Zero	all digital pins, except 4
MKR1000 Rev.1	0, 1, 4, 5, 6, 7, 8, 9, A1, A2
Due	all digital pins
101	all digital pins

Sintaknya adalah :

attachInterrupt (digitalPinToInterrupt (pin), ISR, mode);	(Disarankan)
attachInterrupt (interrupt, ISR, mode);	(Tidak disarankan)
attachInterrupt (pin, ISR, mode);	(Hanya Arduino Due, Zero, MKR1000,101 saja)

Parameternya :

interupsi:	jumlah interrupt ( <i>int</i> )	
pin:	nomor pin	(ArduinoZero, MKR1000 saja)
ISR:	ISR untuk panggilan ketika interupsi terjadi; fungsi ini harus ada parameter dan kembali apa-apa. Fungsi ini kadang-kadang disebut sebagai <i>rutin layanan interupsi</i> .	
modus:	<p>Interrupt harus dipicu. Empat konstanta yang telah ditetapkan sebagai nilai-nilai yang valid:</p> <p><b>LOW</b> untuk memicu interupsi setiap kali pin rendah,</p> <p><b>CHANGE</b> memicu interupsi setiap kali pin perubahan nilai</p> <p><b>RISING</b> untuk memicu ketika pin ganti dari rendah ke tinggi,</p>	

	<b>FALLING</b> ketika pin ganti dari tinggi ke rendah.	
	<p>The papan Arduino Due memungkinkan untuk:</p> <p><b>HIGH</b> untuk memicu interupsi setiap kali pin yang tinggi.</p>	<i>(ArduinoZero, MKR1000 saja)</i>

Contoh :

```
const byte ledPin = 13;
const byte interruptPin = 2;
volatile byte state = LOW;

void setup() {
  pinMode(ledPin, OUTPUT);
  pinMode(interruptPin, INPUT_PULLUP);

  attachInterrupt(digitalPinToInterrupt(interruptPin), blink, CHANGE);
}

void loop() {
  digitalWrite(ledPin, state);
}

void blink() {
  state = !state;
}
```

### 2.15.2 detachInterrupt()

Mematikan interupsi yang diberikan.

Sintak :

<code>detachInterrupt (interrupt)</code>	
<code>detachInterrupt (digitalPinToInterrupt (pin));</code>	
<code>detachInterrupt (pin)</code>	<i>(Arduino Karena, Nol saja)</i>

## 2.16 Interrupts

### 2.16.1 interrupts()

Mengaktifkan kembali interupsi (setelah dinonaktifkan oleh `noInterrupts()`). Interupsi memungkinkan melakukan tugas-tugas penting tertentu di belakang program utama dan diaktifkan secara default. Beberapa fungsi tidak akan bekerja saat interupsi dinonaktifkan, dan komunikasi yang masuk dapat diabaikan.

Contoh :

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

### 2.16.2 noInterrupts()

Menonaktifkan interupsi (Anda dapat mengaktifkan kembali mereka dengan interupsi `()`). Interupsi memungkinkan melakukan tugas-tugas penting tertentu di belakang program utama dan diaktifkan secara default.

Contoh :

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

## 2.17 Communication

### 2.17.1 Serial

Komunikasi serial merupakan komunikasi dua arah dari Transmitter dengan Receiver dan sebaliknya. Kita bisa melakukan komunikasi serial dengan memanfaatkan pin Rx dan Tx pada arduino maupun pada USB. Cara menggunakan serial sebagai berikut :

Contoh :

```
void setup() {  
    Serial.begin(9600);  
}  
void loop() {  
    unsigned char a = a++;  
    Serial.println(a);  
}
```