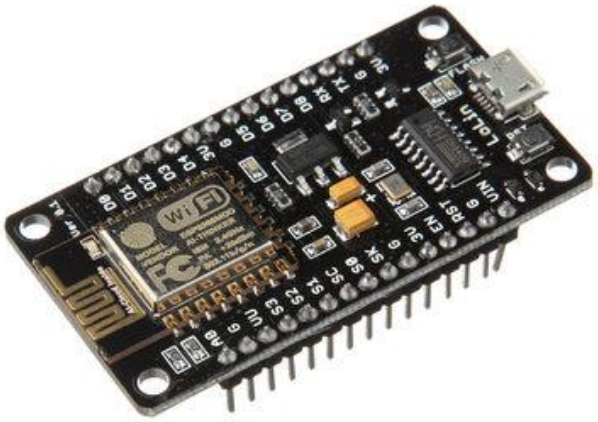
TUTORIAL PEMROGRAMAN NODEMCU ESP8266 DENGAN ARDUINO

NodeMCU adalah sebuah board elektronik yang berbasis chip ESP8266 dengan kemampuan menjalankan fungsi mikrokontroler dan juga koneksi internet (WiFi). Terdapat beberapa pin I/O sehingga dapat dikembangkan menjadi sebuah aplikasi monitoring maupun controlling pada proyek IoT.

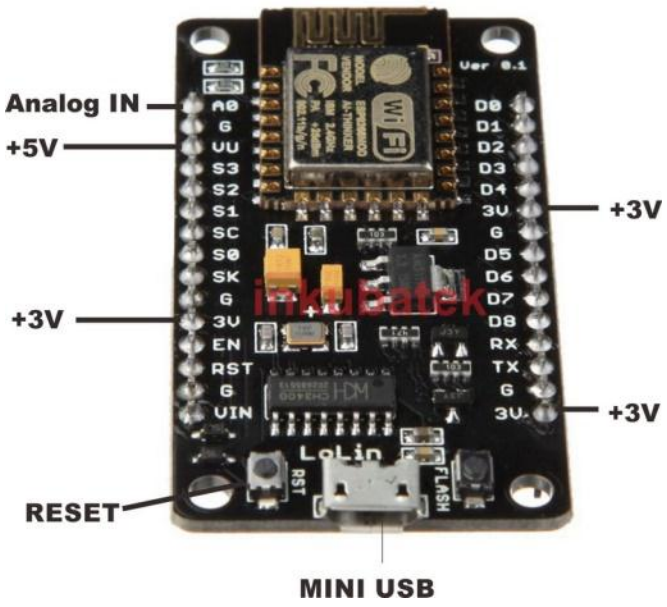
Bentuk fisik dari modul NodeMCU V3 tampak pada gambar berikut :



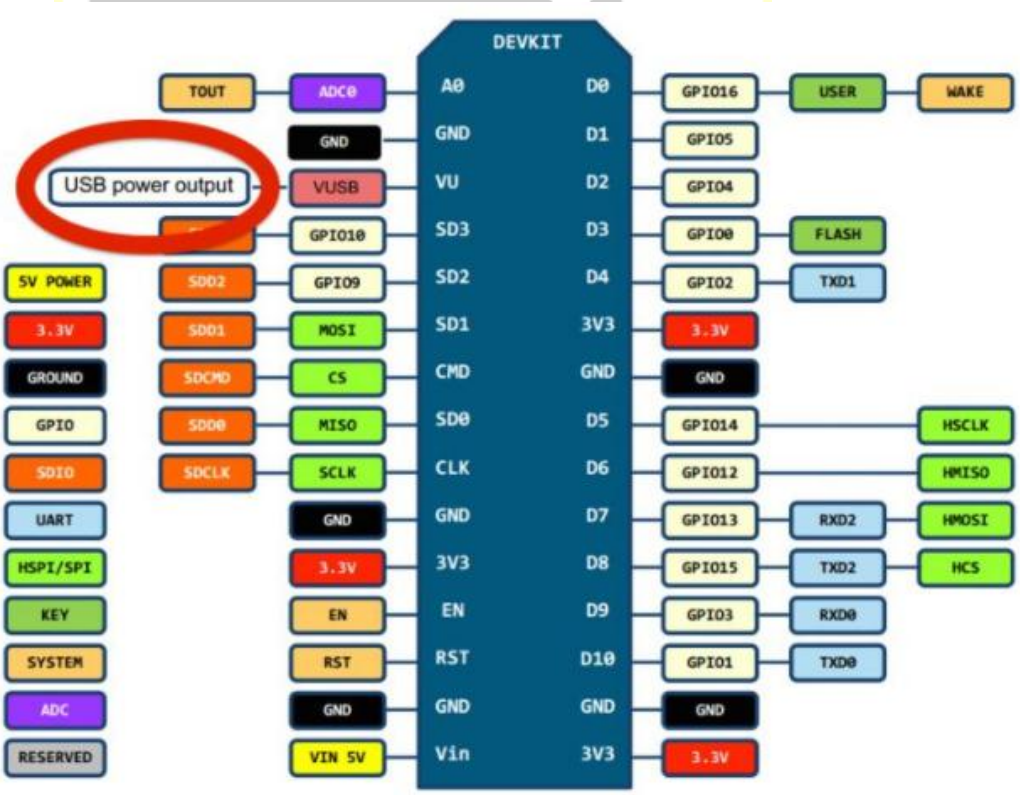
Perhatikan pada NodeMCU V3 ini terdapat port USB (mini USB) sehingga akan memudahkan dalam pemrogramannya nanti.

Spesifikasi :

- Tegangan kerja : 3,3 V
- Flash memori : 16 MB
- Terintegrasi dengan protocol TCP/IP
- Processor : Tensilica L106 32 bit
- Kecepatan : 80 – 160 Mhz
- Jumlah pin GPIO : 17



Konfigurasi pin NodeMCu V3 :



Pemrograman NodeMCU V3

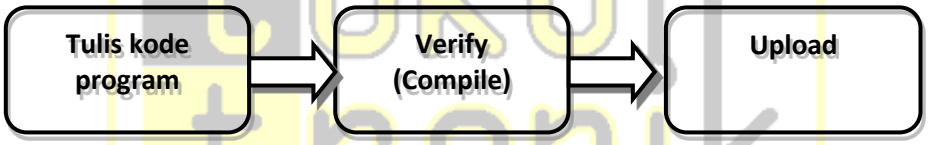
NodeMCU V3 dapat diprogram dengan *compiler*-nya Arduino, menggunakan Arduino IDE. Tentu saja platform pemrogramannya memakai bahasa C.

Bagi anda yang belum pernah sama sekali memprogram Arduino, sebaiknya pelajari dahulu dasar – dasar pemrograman Arduino, karena nanti akan sangat menunjang pemahaman dan pengembangan aplikasi IoT dengan modul NodeMCU ini.

Jadi pemrograman Arduino sama juga untuk pemrograman NodeMCU.

Tidak serumit bahasa pemrograman C untuk ATmega seperti Codevison-AVR (walaupun CodeVision AVR juga lebih mudah dibandingkan bahasa assembly), pemrograman Arduino menjadi lebih mudah. Struktur pemrogramannya memang bahasa C, bagi anda yang sudah menguasai bahasa C/C++ maka akan menjadi lebih gampang memprogram Arduino. Bagi yang belum pernah apalagi menguasai jangan khawatir, karena nanti akan kita buktikan bahwa ternyata memang sangat mudah.

Mekanisme permogramannya sama dengan memprogram mikrokontroler, mulai dari menulis program kemudian mengeksekusi (*compile*) selanjutnya proses *upload* yaitu mengisikan program kedalam memori program Arduino.



Struktur dasar

Sebuah program Arduino/NodeMCU minimal terdiri dari 2 bagian :

1. **Inisialisasi.**

Inisialisasi merupakan proses mengatur hardware seperti port I/O, PWM, serial dan peripheral lain. Struktur ini ditulis diawal program. Sebagai contoh port I/O mempunyai beberapa fungsi : digital input, digital output, serial komunikasi dan PWM. Sebuah port hanya dapat berfungsi untuk 1 tujuan, jadi jika kita hendak menggunakan port tersebut sebagai digital output maka harus diinisialisasi terlebih dahulu sebagai port output. Inisialisasi menggunakan struktur ***setup()***. Sebagai contoh :

```
void setup()
{
Serial.begin(9600);
```

```
pinMode(buttonPin, INPUT);

}
```

2. **Program utama.**

Setelah melakukan inisialisasi selanjutnya program yang dikerjakan adalah program utama, tergantung dari aplikasi yang dibuat, isi dari program utama berbeda antara satu program dengan program yang lain. Struktur yang digunakan adalah ***loop()***. Sebagai contoh :

```
void loop()

{
}
```

Harap diperhatikan bahwa bahasa pemrograman Arduino (sebagaimana bahasa C) termasuk **case sensitive**, artinya ***huruf besar dengan huruf kecil akan dianggap berbeda.***

Komentar

Komentar digunakan untuk memberi keterangan pada program yang dibuat, sifatnya tidak wajib dan tidak akan berpengaruh pada jalannya program karena tidak ikut dieksekusi. Komentar dapat dibuat untuk 1 baris maupun lebih dari 1 baris, berikut ini caranya :

```
//komentar untuk 1 baris diawali dengan 2 garis miring.

/*komentar untuk lebih dari 1 baris diawali dengan tanda garis miring + asterisk atau bintang (/*)
```

Serta diakhiri dengan tanda asterisk + garis miring */

Ekspresi Bilangan

Dalam pemrograman Arduino, bilangan dapat diekspresikan dalam beberapa format.

- **Desimal**

  Ditulis biasa tanpa tambahan apapun. Contoh : 234.

- **Oktal**

  Ditulis dengan awalan angka ‘0’ (nol) didepan. Contoh : 0631.
- **Biner**

  Penulisan diawali dengan huruf ‘B’. Contoh : B11100011.
- **Heksadesimal**

  Diawali dengan ‘0x’ . Contoh : 0x8C.

Kontrol Program

Sebuah program yang kita buat membutuhkan suatu kontrol, misalnya pengujian kondisi, melompat pada perintah yang lain dan sebagainya.

Pengujian Kondisi.

if

Digunakan untuk menguji kondisi, jika kondisi tersebut benar maka perintah didalam If akan dikerjakan.

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

Contoh :

```
if(x>7)
{
x=x+1;
}
```

if – else

Hampir sama dengan if, hanya saja ada 2 pilihan pernyataan/perintah. Jika kondisi benar maka perintah didalam blok if yang dikerjakan, jika kondisi salah maka pernyataan di dalam else yang dikerjakan.

```
if (kondisi)
```

```
{
Pernyataan/perintah 1:
}
else
{
Pernyataan/perintah 2;
}
```

Contoh :

```
if (x>7)
{
x=x+1;
}
else
{
x=x+2;
}
```


if – else if

Untuk pengujian dengan banyak kondisi maka digunakan if – else if.

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

```
}

else if (kondisi ke-n)

{

Pernyataan/perintah ke-n;

}
```

Contoh :

```
if(tombol=='1'){

Serial.println("Angka 1");

}

else if(tombol =='2'){

Serial.println("Angka 2");

}

else if(tombol =='3'){

Serial.println("Angka 3");

}
```


switch case

Seperti pernyataan if, digunakan untuk memilih kondisi yang sesuai untuk kemudian mengerjakan perintahnya. Bedanya adalah kondisi yang diuji berupa nilai sebuah veriabel.

```
switch (variabel)//variabel yang diuji nilainya

{

case 1: //pernyataan/perintah 1

break;

case 2: //pernyataan/perintah 2

break;
```

```
case n: //pernyataan/perintah n

break;

default : //pernyataan/perintah

}
```

Jika variabel bernilai ‘1’ maka pernyataan 1 yang dikerjakan. Break berfungsi untuk mengakhiri pernyataan/perintah pada setiap case. Default berfungsi jika semua nilai dalam case tidak ada yang sama dengan variabel.

Contoh :

```
switch (j)

{

case 1: digitalWrite(ledPin1, HIGH); break;

case 2: digitalWrite(ledPin2, HIGH); break;

case 3: digitalWrite(ledPin3, HIGH); break;

default: digitalWrite(ledPin4, LOW); break;

}
```


Perulangan.

while

Untuk membuat perulangan yang tidak terbatas selama kondisi di dalam while benar.

```
while(kondisi)

{

//pernyataan/perintah

}
```

Contoh :

```
while(j<100)

{
```

```
J++;
}
```

do ... while.

Hampir sama dengan while, untuk do – while pernyataan/perintah dikerjakan terlebih dahulu baru melihat kondisi didalam ekspresi while, jika benar maka pernyataan/perintah akan diulangi lagi.

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

Contoh :

```
do
{
J++;
}
while (j<1000);
```


for

Digunakan untuk perulangan yang terbatas, misalnya mengulang perintah sebanyak 10 kali.

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

Inisialisasi : nilai awal variabel untuk proses perulangan.

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

Step : tahap perulangan, biasanya dalam bentuk penambahan.

Contoh :

```
for (x=0; x<100; x++)
{
Serial.println(x);
}
```

Kontrol program yang lain.

goto.

Untuk melompat/menuju perintah yang telah diberi label.

Goto label;

Contoh :

```
while(1)
{
digitalWrite(ledPin1, HIGH);
delay(1000);
digitalWrite(ledPin1, LOW);
delay(1000);
if (digitalRead(inPin)==LOW)
{goto keluar;}
}
keluar:
```

return

Digunakan untuk memberikan nilai balik dari sebuah fungsi.

Contoh :

```
int checkSensor(){
  if (analogRead(0) > 400) {
```

```
    return 1;
  else{
    return 0;
  }
}
```

continue

Untuk melewatkan perulangan yang tersisa dari struktur looping (do, for atau while).

Contoh :

```
for (x = 0; x < 255; x ++)
{
  if (x > 40 && x < 120){
    continue;
  }
  digitalWrite(PWMpin, x);
  delay(50);
}
```


break

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

Operator Aritmatika.

Operator	Keterangan
=	Pemberian nilai
+	Operasi penjumlahan
-	Operasi pengurangan
*	Operasi perkalian
/	Operasi pembagian

%	Operasi sisa pembagian

Operator Perbandingan.

Operator	Keterangan
==	Operator persamaan. Jika kedua nilai yang dibandingkan sama 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.

Operator Boolean.

Operator	Keterangan
&&	AND
||	OR
!	NOT

Operator Bitwise.

Operator	Keterangan
<<	Geser kiri
>>	Geser kanan
&	AND
|	OR
^	XOR
~	NOT

Operator Penaikan dan Penurunan.

Operator	Keterangan	Contoh	Arti
++	Penaikan 1 / increment	n++	n= n+1
--	Penurunan 1 / decrement	n--	n= n-1

Variabel

Dalam pembuatan program aplikasi Arduino, sering kita memerlukan tempat atau wadah untuk menampung atau menyimpan suatu data, tempat atau wadah tersebut adalah variabel.

Nama variabel sebenarnya bebas, boleh 1 huruf abjad seperti ‘i’ (tanpa petik), dapat juga sebuah kata misalnya ‘hasil’. Yang jelas tidak boleh mengandung spasi, maksimal 32 karakter dan tidak boleh menggunakan istilah baku yang sudah dipakai dalam bahasa C.

Sebelum dipakai, variabel terlebih dahulu dideklarasikan dengan cara :

[Tipe data] [nama variabel]

Contoh :

Int bilangan_1;

Atau bisa juga langsung diberi nilai

[tipe data] [nama variabel][=][nilai]

Contoh :

int bilangan_1=20;//variabel bilangan tipe integer dengan nilai awal 20

Tipe Data

Setiap variabel yang dibuat harus mempunyai tipe data yang menunjukkan kemampuan variabel tersebut menampung data.

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/d3.4028235E+38

PIN INPUT – OUTPUT

Pada board NodeMCU Arduino UNO terdapat pin Input dan Output (D0 – D10), dari namanya kita tahu bahwa fungsinya dapat sebagai pin input (masukan) maupun pin output (keluaran).

Inisialisasi Fungsi Pin I/O

Sebuah pin pada saat yang sama hanya mempunyai satu fungsi, sebagai input saja atau output saja, untuk itu harus ditentukan dulu fungsinya, yaitu ketika inisialisasi (*setup()*), dengan cara :

pinMode(pin, mode);

- ✓ ***pin*** : nomor pin yang akan dikonfigurasi (nomor pin pada board NodeMCU, D0 – D8).
- ✓ ***mode*** : INPUT atau OUTPUT.

Sebagai contoh jika pin no D2 akan dibuat sebagai pin output, maka :

pinMode(D2, OUTPUT);

Contoh yang lain pin no D4 dibuat sebagai input :

pinMode(D4, INPUT);

Perhatikan penulisan huruf, besar kecilnya sangat berpengaruh.

Menulis Data Digital di Pin Output

Setelah membuat pin sebagai digital output dengan fungsi pinMode(pin,OUTPUT) selanjutnya untuk menulis atau mengeluarkan data digital dengan perintah :

digitalWrite(pin, value);

- ✓ ***pin*** : nomor pin digital output.
- ✓ ***value*** : HIGH atau 1 (3,3 volt) atau LOW atau 0 (0 volt/ground).

Sebagai contoh, pin D3 sebagai pin digital output akan diberi logika 1 :

pinMode(D3, OUTPUT);

digitalWrite(D3, HIGH);

Atau dapat juga ditulis dengan cara lain, yaitu :

digitalWrite(D3, 1);

Membaca Data Digital di Pin Input

Jika sebuah pin dibuat sebagai pin input, maka kita masih harus menentukan tipe inputnya : *floating* atau *pullup*. Jika kita pilih pullup maka resistor *pullup* internal (pada setiap pin) akan aktif. Caranya adalah :

digitalWrite(pin, value);

- ✓ **pin** : nomor pin yang diset sebagai pin input.
- ✓ **value** : HIGH atau 1 (pullup aktif) atau LOW atau 0 (floating).

Contoh pin no 4 akan dibuat sebagai pin input dengan pullup :

pinMode(D4,INPUT);
digitalWrite(D4,HIGH);

Setelah diset sebagai pin input, fungsi pembacaan data digitalnya adalah :

Var=digitalRead(pin);

Misalnya akan dibaca pin no D4, hasil pembacaan disimpan dalan variabel baca :

baca=digitalRead(D4);

SIAPKAN HARDWARE DAN SOFTWARE

Setelah kita pelajari sekilas tentang konfigurasi dari board NodeMCU, selanjutnya kita siapkan instalasinya, sehingga siap untuk kita program.

Kebutuhan Hardware

Tidak perlu banyak hardware yang harus disiapkan untuk awal pembelajaran kita, cukup beberapa saja.

- 1 board NodeMCU
- Kabel micro USB (kabel data HP)
- 4 LED dan 4 resistor 330 ohm.

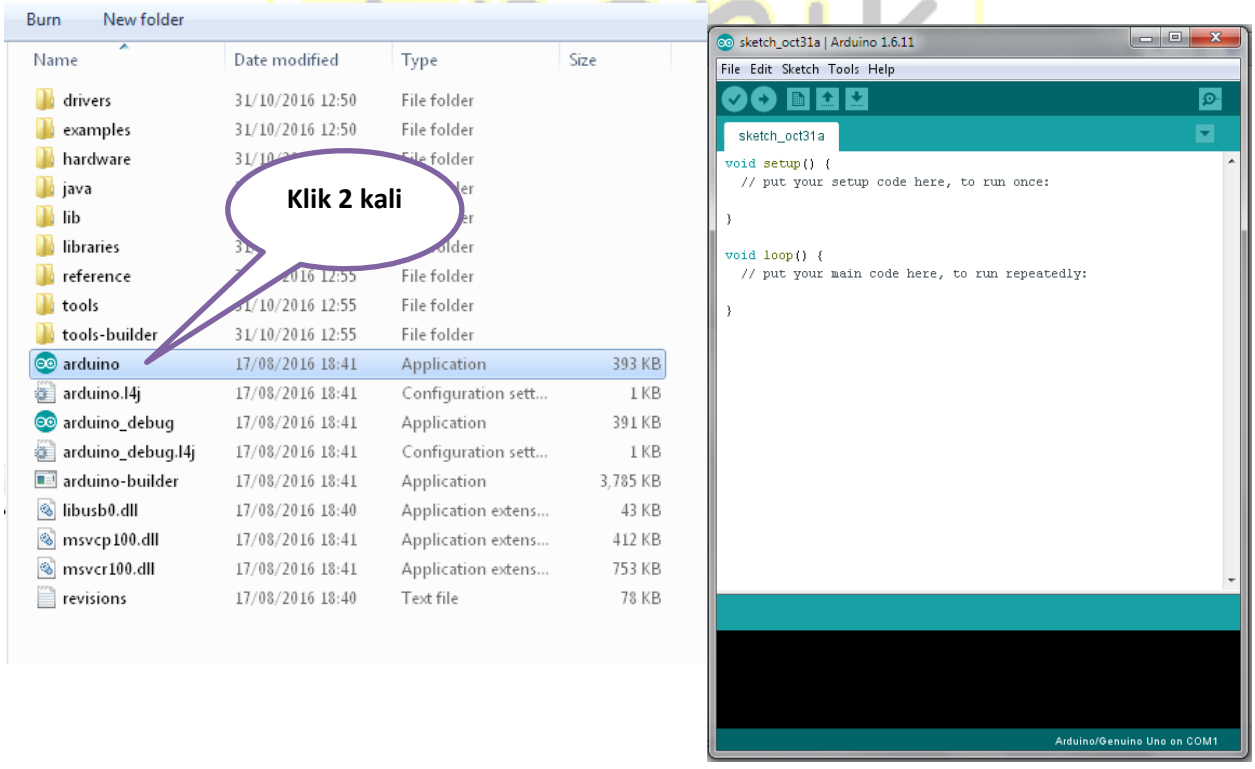
Power supply juga sudah dapat kita peroleh melalui jalur tegangan dari port USB (komputer)

Instalasi driver

Instalasi driver silakan baca di CD

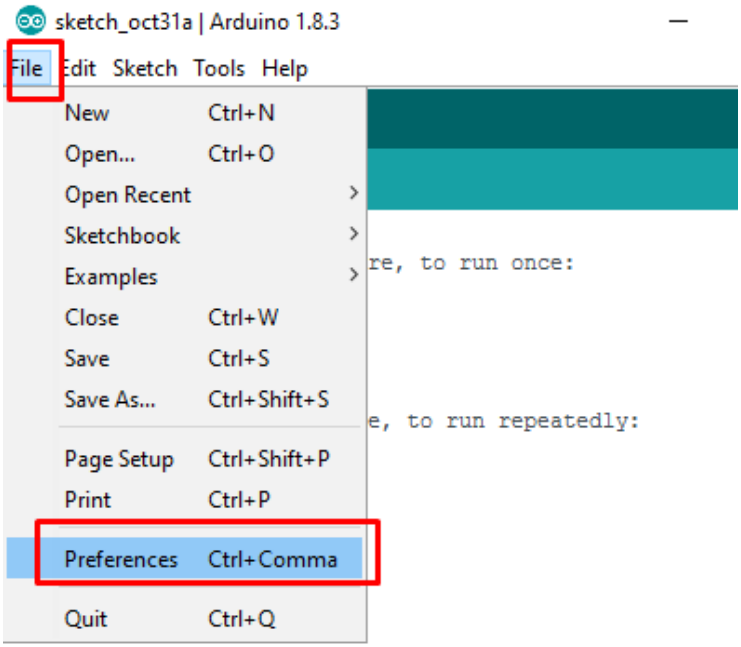
Instalasi Software Compiler

Software yang digunakan untuk membuat program berbasis board Arduino sifatnya *free*, dan instalasinya juga sangat gampang. Masih menggunakan OS Windows, silahkan explore folder arduino-1.6.11 (untuk versi terbaru bisa download di http://arduino.cc/en/Main/Software). Klik 2 kali (klik kanan terus open) file **Arduino.exe.**

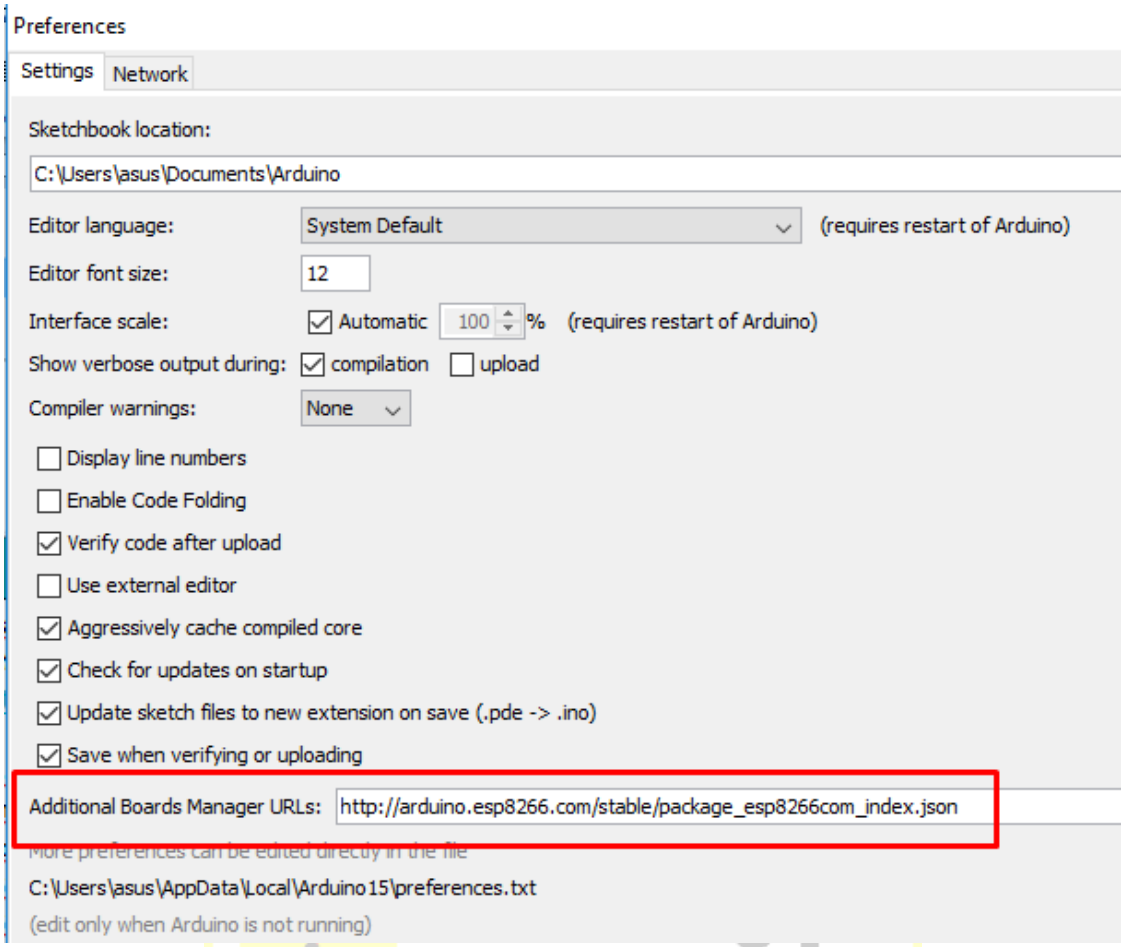


Selanjutnya kita seting Arduino IDE agar dapat dipakai untuk NodeMCu V3 kita.

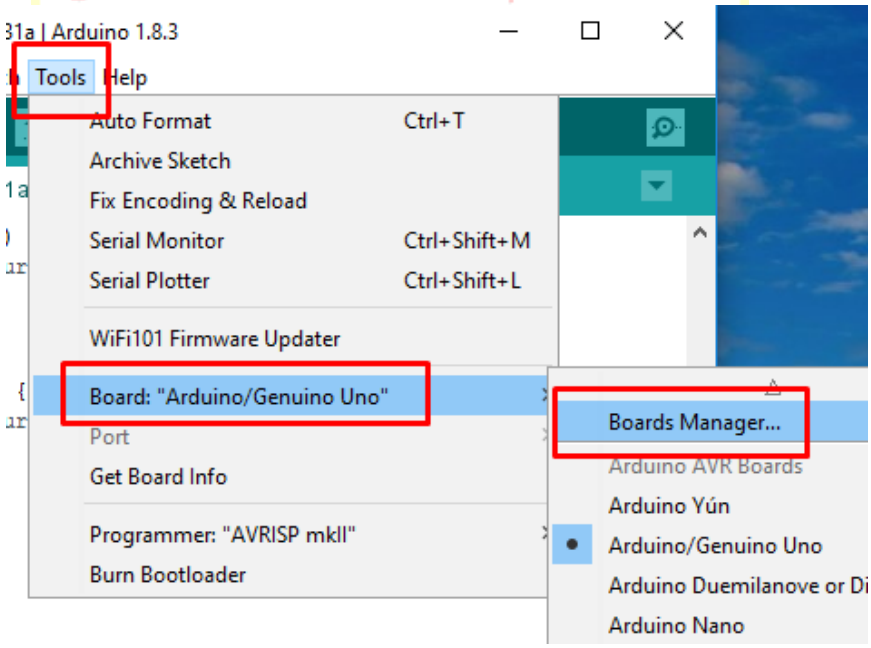
- Jalankan Arduino IDE. Dari menu ***File → Preference***.



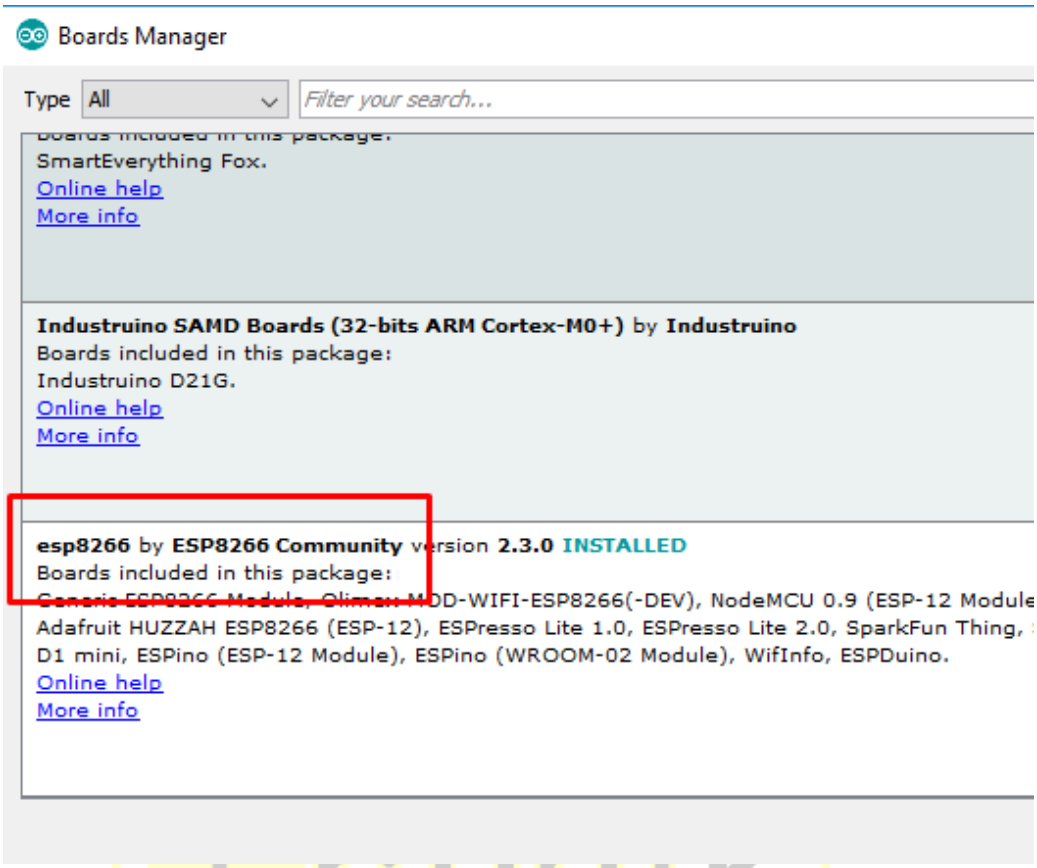
- Pada jendela Preference, di bagian bawah terdapat kolom ***Additional Boards Manager URLs*** kemudian copy paste link berikut : http://arduino.esp8266.com/stable/package_esp8266com_index.json



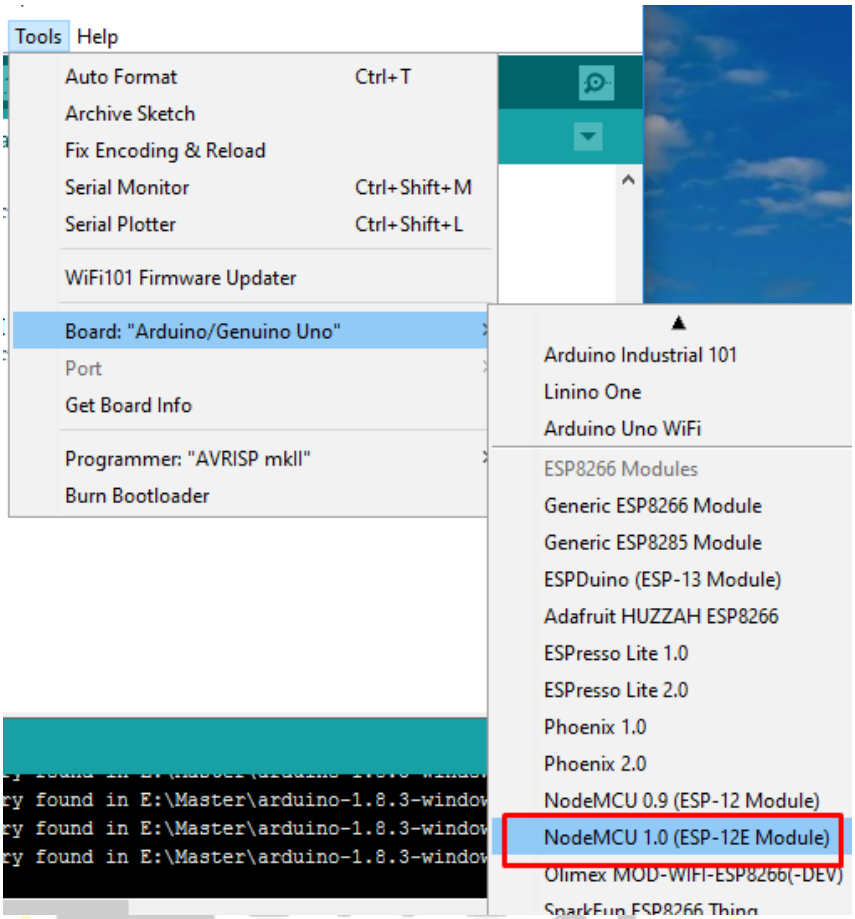
- Kemudian klik ***OK***.
- Selanjutnya kita update boardnya. Dari menu ***Tool → Board → Board manager***



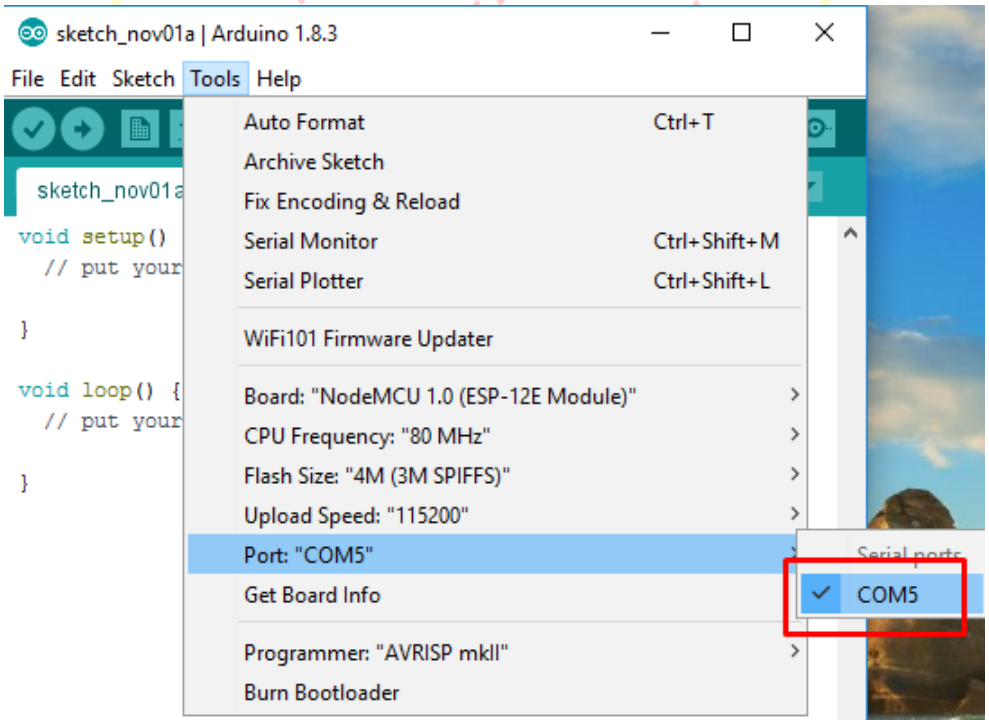
- Akan muncul jendela ***Boards Manager***. Pada bagian bawah cari esp8266 kemudian klik Install.



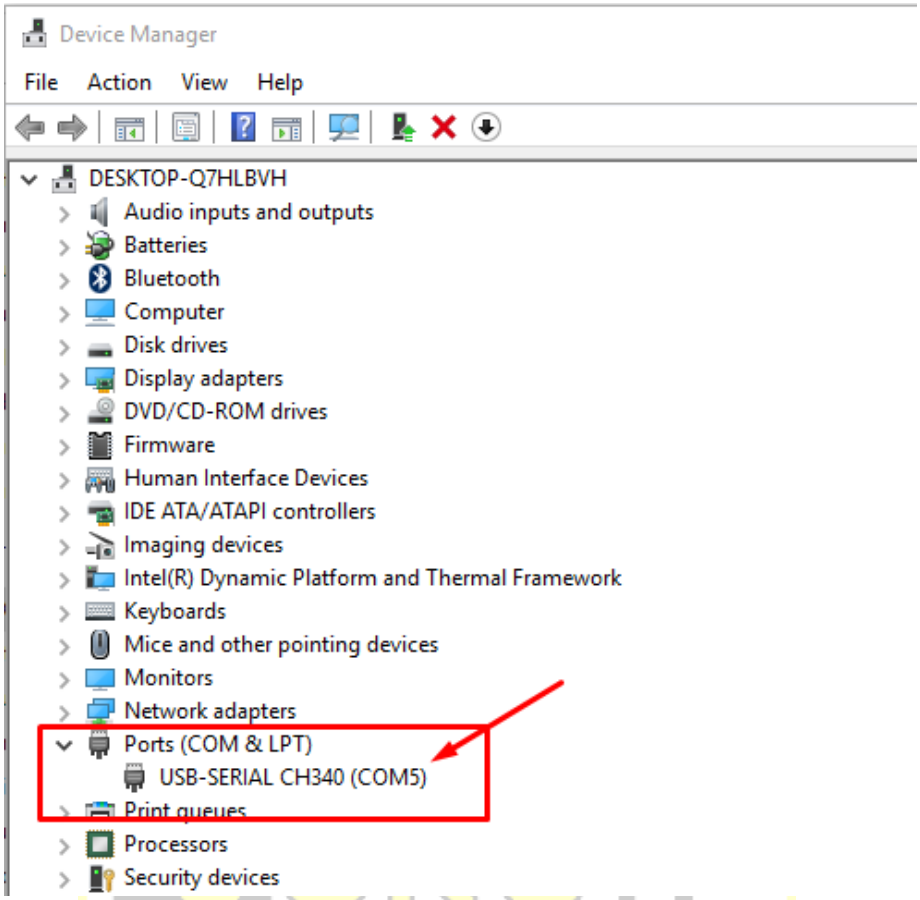
- Ok, sekarang kita cek apakah NodeMCU sudah terinstal di Arduino IDE kita atau belum. Dari menu ***Tools → Board → NodeMCU*** seperti pada gambar.



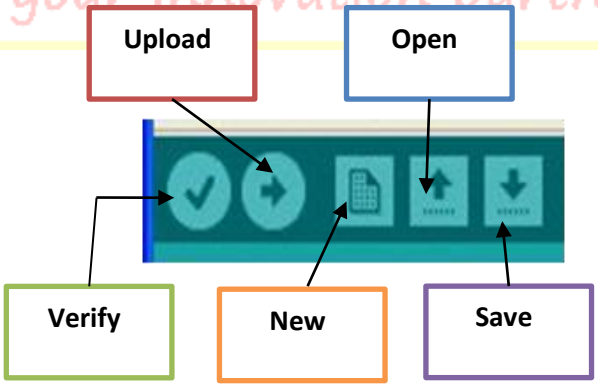
- Sekarang Arduino IDE sudah siap dipakai untuk pemrograman NodeMCU V3. Selanjutnya kita seting nomor com (USB) nya



- Untuk nomor port-nya silakan cek di Device Manager computer anda.



Untuk penggunaannya kita akan sering memakai toolbar, keterangan tombol toolbar terlihat pada gambar berikut.



Gambar tersebut adalah tampilan software Arduino yang nantinya akan kita pakai untuk membuat program dan mengisikannya (*upload*) ke memori program NodeMCU. Sekarang kita pelajari sebentar fungsi tombol pada toolbar.

- ***Verify***.

  Berfungsi untuk mengeksekusi (proses *compile*) program.

- ***New***.

  Membuat lembar kerja baru.
- ***Open***.

  Membuka file yang pernah dibuat & disimpan.
- ***Save***.

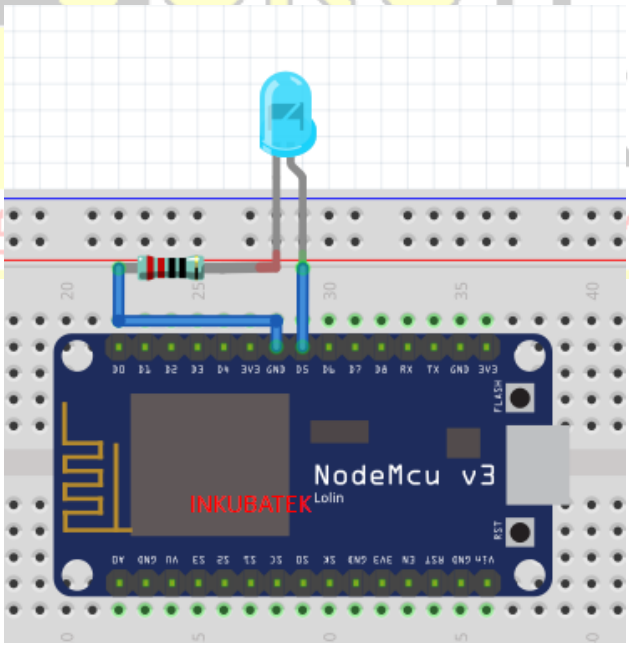
  Menyimpan program, secara otomatis akan berekstensi *.pde.
- ***Upload***.

  Berfungsi untuk mentransfer program yang telah dieksekusi ke dalam memori NodeMCU, dalam beberapa software disebut dengan istilah program, ada juga yang menyebut dengan istilah download.

PRAKTEK DIGITAL OUTPUT

Aplikasi digital output memanfaatkan pin I/O yang difungsikan sebagai pin output.

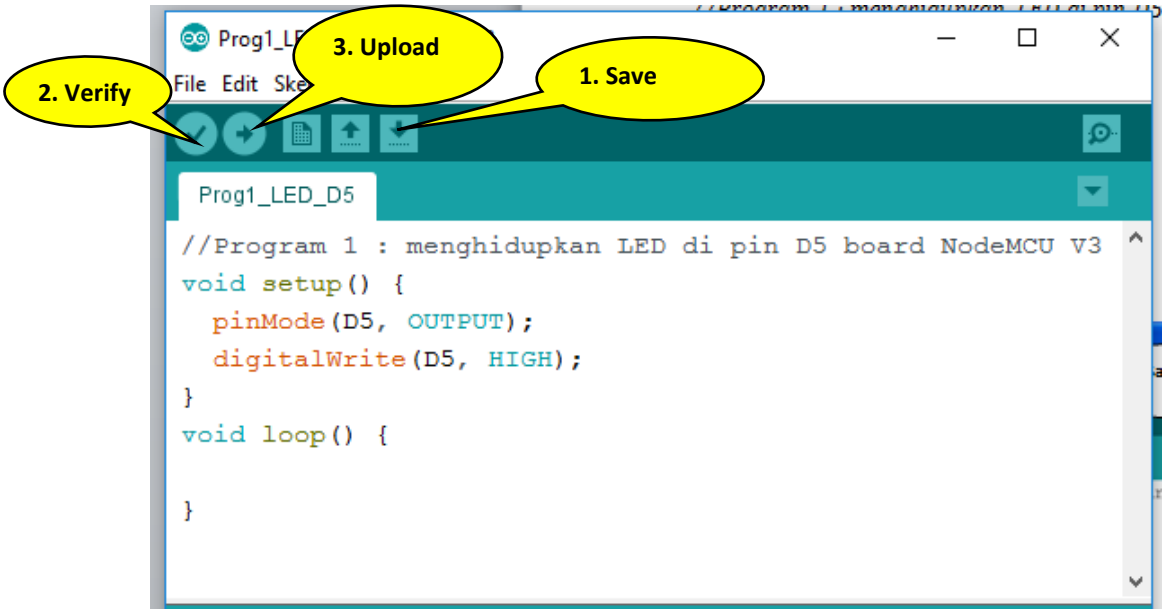
Buatlah rangkaian LED sederhana dengan tambahan resistor seperti gambar berikut.



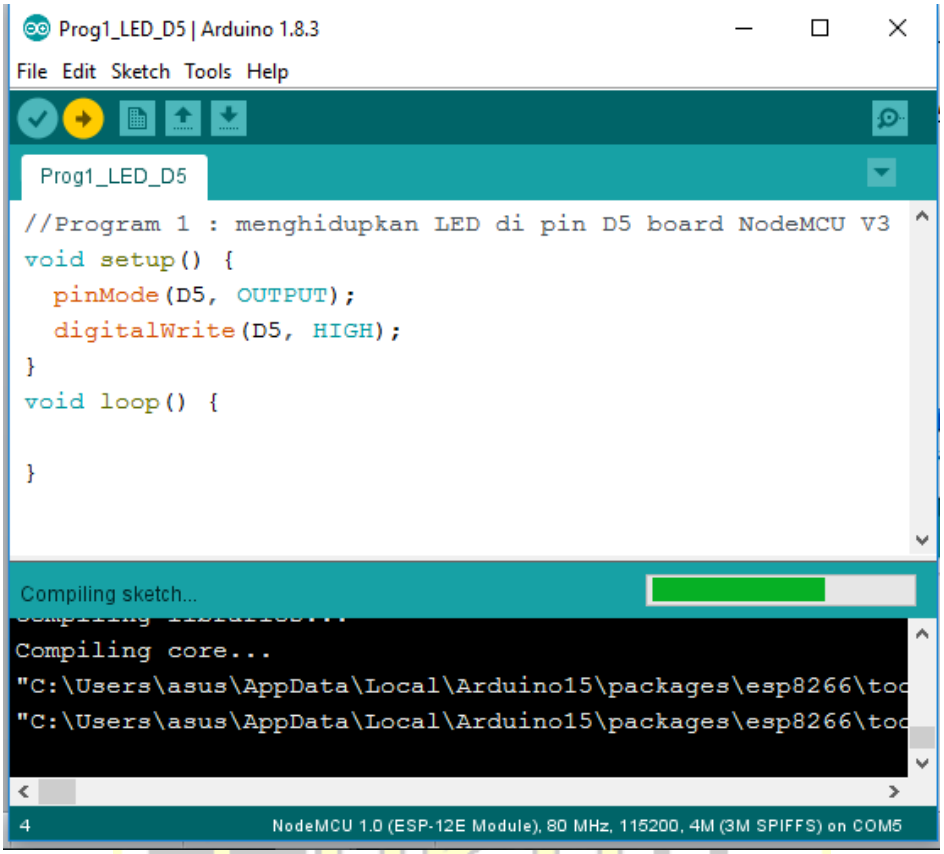
Perhatikan rangkaian LED-nya yang terhubung dengan pin D5, jika pada pin D5 terdapat tegangan 3,3 volt (HIGH) maka terdapat perbedaan tegangan antara kaki anoda (3,3 volt) dengan kaki katoda (0 volt, ground) sehingga akan mengalir arus dan LED nyala (ON). Tegangan 0 volt (LOW) akan membuat LED padam (off).

Baik, sekarang buka program Arduino.exe, kemudian tulis program berikut.

```
//Program 1 : menghidupkan LED di pin D5 board NodeMCU V3
void setup() {
  pinMode(D5, OUTPUT);
digitalWrite(D5, HIGH);
}
void loop() {


}
```



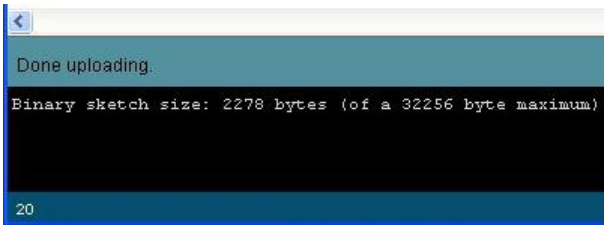
Setelah selesai, simpan (***Save***) dan beri nama, misalnya ‘program1’ (tanpa tanda petik). Secara otomatis terbentuk folder dengan nama ‘program1’.Didalamya terdapat file dengan nama ‘program1.pde’. Compile langsung atau klik toolbar ***Verify***.



Tunggu sampai proses compiling selesai.

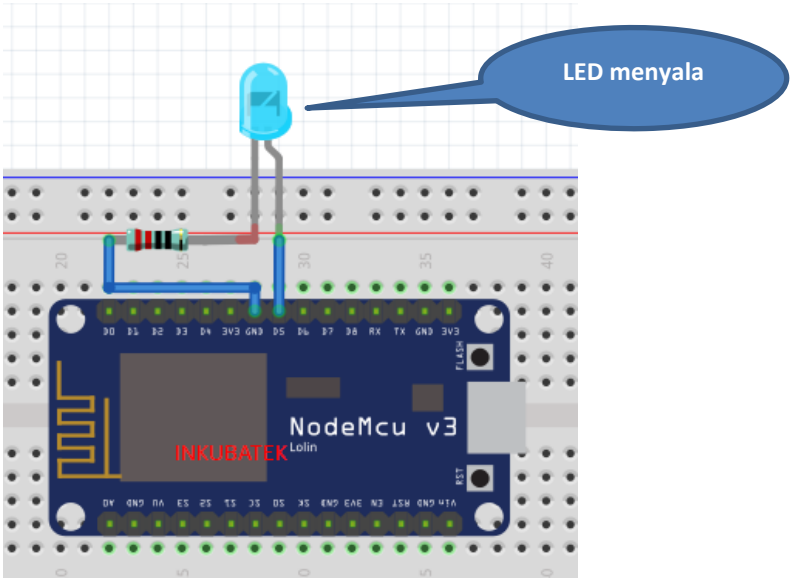
Jika ada error silahkan perbaiki dulu, perhatikan penulisannya, karena pemrograman juga tentang ketelitian. Selanjutnya upload program ke board NodeMCU, klik ***Upload***.

Tunggu sampai proses upload selesai.



Jalannya program :

Lihat hasilnya pada led akan nyala.



Penjelasan program :

Pada baris atas ada sebuah komentar (diawali dg //) :

//Program 1 : menghidupkan LED di pin D5 board NodeMCU

Komentar sifatnya sebagai penjelas saja. Tidak wajib. Tidak berpengaruh terhadap jalannya program.

Program dimulai dengan inisialisasi : *void setup()*.

pinMode(D5, OUTPUT); → pin no D5 difungsikan sebagai pin output.

digitalWrite(D5, HIGH); → pin no D5 ditulis HIGH, artinya pin 13 mengeluarkan logika HIGH/"1" (3,3 volt), maka kita lihat LED akan nyala (ON).

Selanjutnya program utama : *void loop()* tidak ada perintah karena memang tidak perlu perulangan program disini.

```
//Program 1 : menghidupkan LED di pin 13 board Arduino
void setup() {
  pinMode(D5, OUTPUT);
digitalWrite(D5, HIGH);
}
void loop() {
}
```

Komentar

Fungsi : Output

No pin :

Pin D5 mengeluarkan logika HIGH

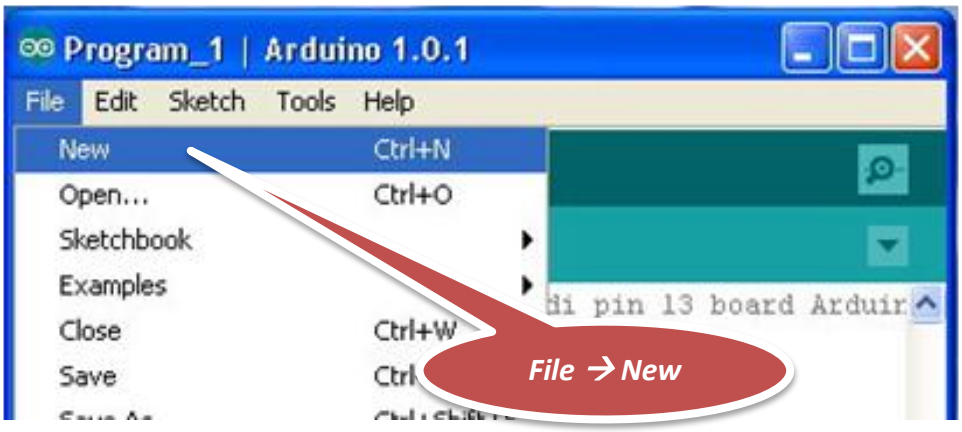
Sebenarnya ada cara lain untuk membuat logika pin menjadi 'HIGH' atau 'LOW'. Dengan cara langsung ditulis nilai '1' untuk 'HIGH' dan '0' untuk 'LOW. Langsung contoh aja ya, pada program 1 :

```
//Program 1 : menghidupkan LED di pin D5 board NodeMCU
void setup() {
  pinMode(D5, OUTPUT);
digitalWrite(D5, 1);
}
void loop() {
}
```

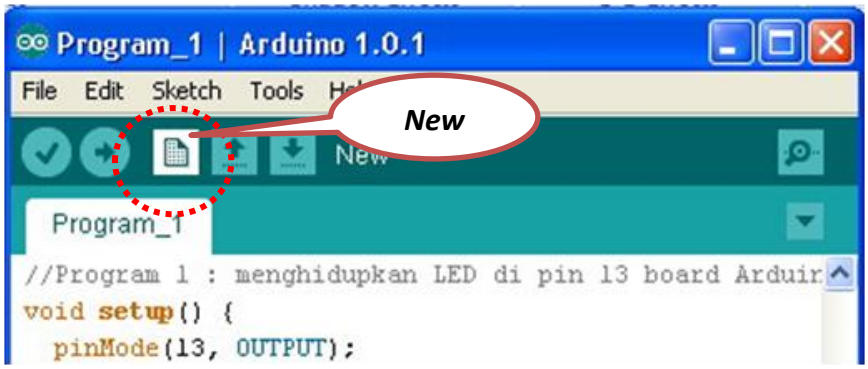
Pin D5 mengeluarkan logika '1' (HIGH)

Program 2 : LED berkedip.

Masih menggunakan LED, kita mencoba led berkedip. Buat lembar kerja baru, ***File → New*** **(ctrl+N)**.



Atau bisa juga lewat toolbar *New*.



Sudah ? *Next* tulis program berikut ya ..

```
//Program 2 : LED berkedip
void setup() {
  pinMode(D5, OUTPUT);
}
void loop() {
 digitalWrite(D5, HIGH);
 delay(1000);
 digitalWrite(D5, LOW);
 delay(1000);
}
```

Jalannya program :

LED berkedip. Satu detik nyala kemudian satu detik berikutnya padam.

Penjelasan program :

Inisialisasi sama dengan program 1. Untuk program utama (*loop*) terdapat 4 baris perintah yang akan diulang terus (*looping*).

digitalWrite(D5, HIGH)→ pin D5 berlogika '1' sehingga LED akan ON.

delay(1000)→ fungsi tunda, program akan menunggu disini sampai 1000 ms atau 1 detik.

digitalWrite(D5, LOW) →pin D5 berlogika '0' sehinggaLED akan OFF.

delay(1000)→ fungsi tunda, program akan menunggu disini sampai 1000 ms atau 1 detik.

Jika akan mengganti waktu tunda tinggal anda ganti nilai yang ada di dalam kurung *delay()*, misal akan dibuat penundaan 1,5 detik maka cukup tulis *delay(1500)*.

DIGITAL INPUT

Pin I/O yang sudah kita gunakan semua difungsikan sebagai pin output, sinyal digital dikeluarkan oleh NodeMCU melalui pin yang dipilih. Nah, sekarang kebalikannya, NodeMCU **membaca** sinyal digital melalui pin yang ditentukan. Untuk membuat sebuah pin menjadi pin digital input kita atur sebuah pin dengan 2 tahap :

- Tentukan nomor pin-nya :

  pinMode(pin, INPUT)

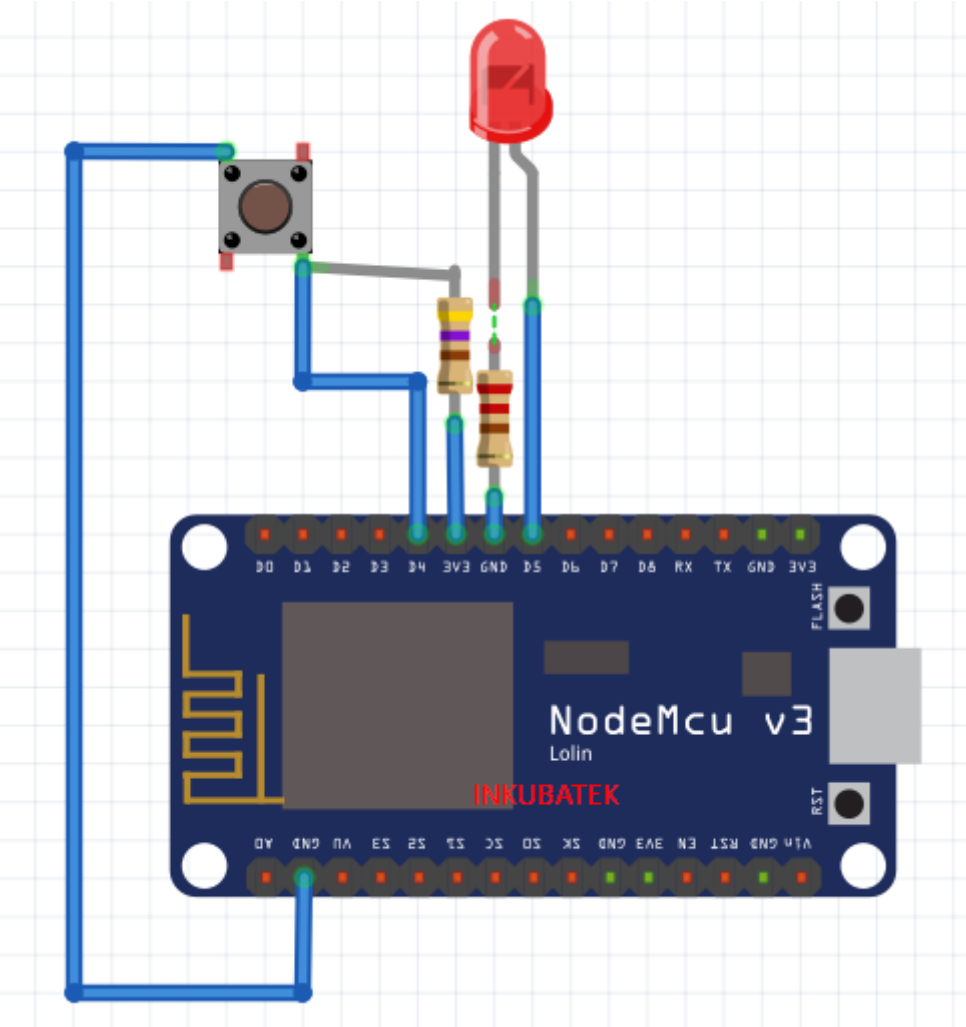
  pin : nomor pin NodeMCU

- Kemudian untuk membaca data digital yang masuk, gunakan perintah berikut :
  var=digitalRead(pin);

  var adalah variabel, nilainya tergantung logika input. Bernilai ‘1’ (HIGH) atau ‘0’ (LOW).
  pin adalah nomor pin NodeMCU yang dibaca inputannya.

Sekarang langsung praktek digital input, aplikasinya dengan tombol + LED.

Nah rangkaiannya akan jadi seperti ini :



LED masih terhubung dengan pin D5, tombol push on terhubung dengan pin D4 dan tersambung dengan pullup VCC sehingga dalam kondisi tombol tidak ditekan (OFF) maka logikanya ‘1’ dan ketika ditekan maka logikanya ‘0’.

Programnya kek gini gan :

```
//Program 3 : membaca  tombol

byte tombol;

void setup()

{

 pinMode(D4,INPUT);

 pinMode(D5,OUTPUT);


 digitalWrite(D4,HIGH);
```

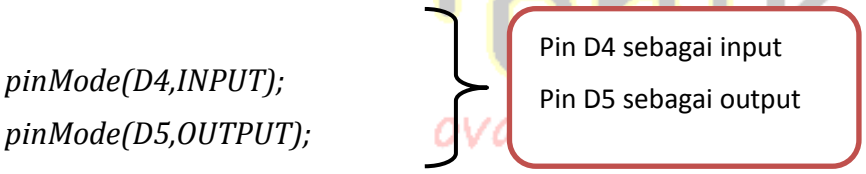
```
 digitalWrite(D5,LOW);
}
void loop()
{
  tombol=digitalRead(D4);
  if (tombol==LOW) digitalWrite(D5,1);
  else  digitalWrite(D5,0);
}
```

Jalannya program :

Led akan nyala sesuai dengan tombol yang ditekan. Coba tekan tombol 1 (*press and hold*), LED 1 akan nyala. Kalau dilepas LED akan padam lagi.

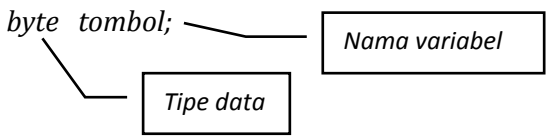
Penjelasan program :

Inisialisasi pada *void setup()* berupa seting pin D4 input untuk membaca sinyal masukan dari tombol SW sedangkan D5 sebagai output yang terhubung dengan LED

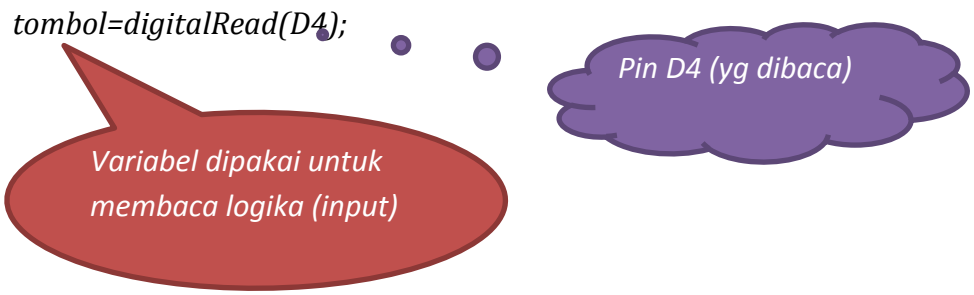


Program utama, seperti yang kita lihat pada blok *void loop()*, membaca pin input dan kemudian mengatur nyala – padam led . Ada 2 kondisi yang diuji/dibaca sehingga kita membutuhkan bentuk ***if – else if***.

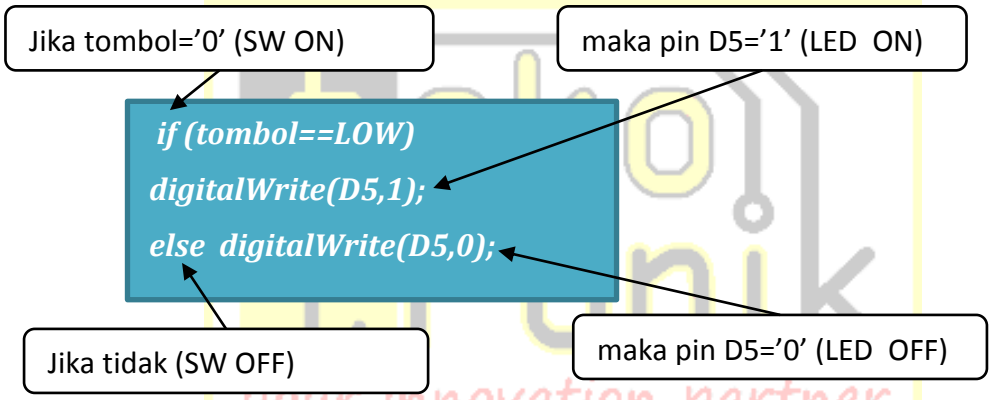
Karena kita membutuhkan variabel untuk menyimpan data hasil 'bacaan' input digital, maka kita perlu mendeklarasikan variabelnya terlebih dahulu. Deklarasi kita tulis di awal program. Tipenya cukup *byte*.



Pertama, baca dulu data digital (*input*) pada pin D4. Nah pin D4 terhubung ke SW1.



Jika SW ditekan ('ON'), maka pin D4 terhubung ke GND, sehingga berlogika 'LOW'. Secara otomatis isi variabel tombol juga 'LOW'. Kondisi ini akan membuat LED nyala. Jika SW 'OFF' maka pin D4 berlogika 'HIGH' (karena *pull up*). Nah isi variabel 'tombol' akan sama dengan kondisi logika di pin D4. Program akan menguji isi variabel tombol. Cukup memakai pengujian kondisi ***if else***.



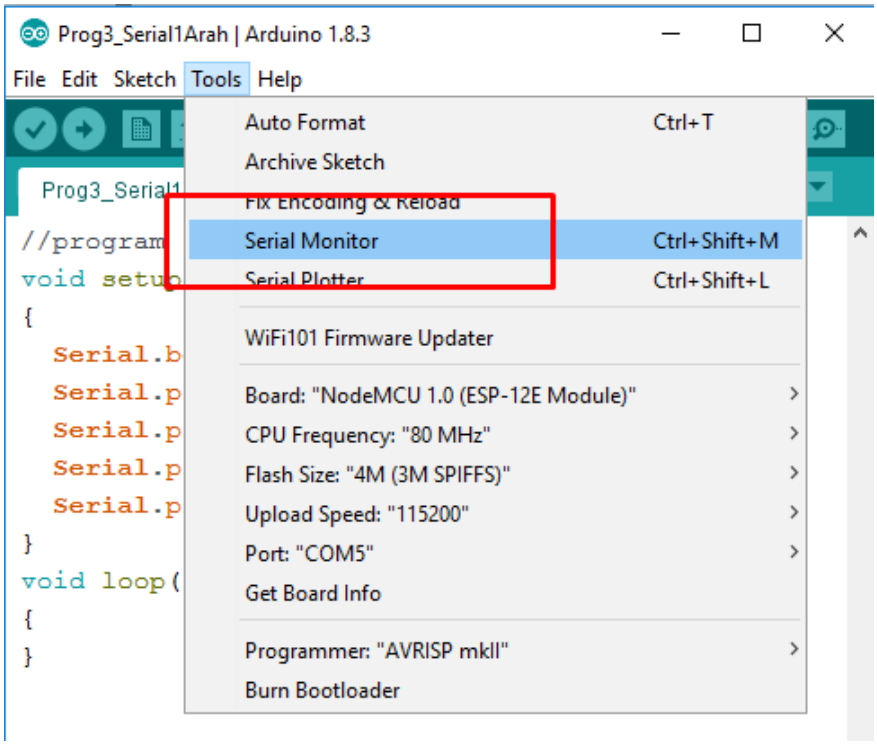
Komunikasi Serial

Untuk pemrograman serial komunikasi dengan NodeMCU juga sangat gampang, karena semua fungsi komunikasi serial sudah tersedia. Langsung aplikasi ya.

```
//program 4: komunikasi serial 1 arah

void setup()

{

 Serial.begin(115200);

 Serial.print("Coba ");

 Serial.print("Komunikasi...");
```

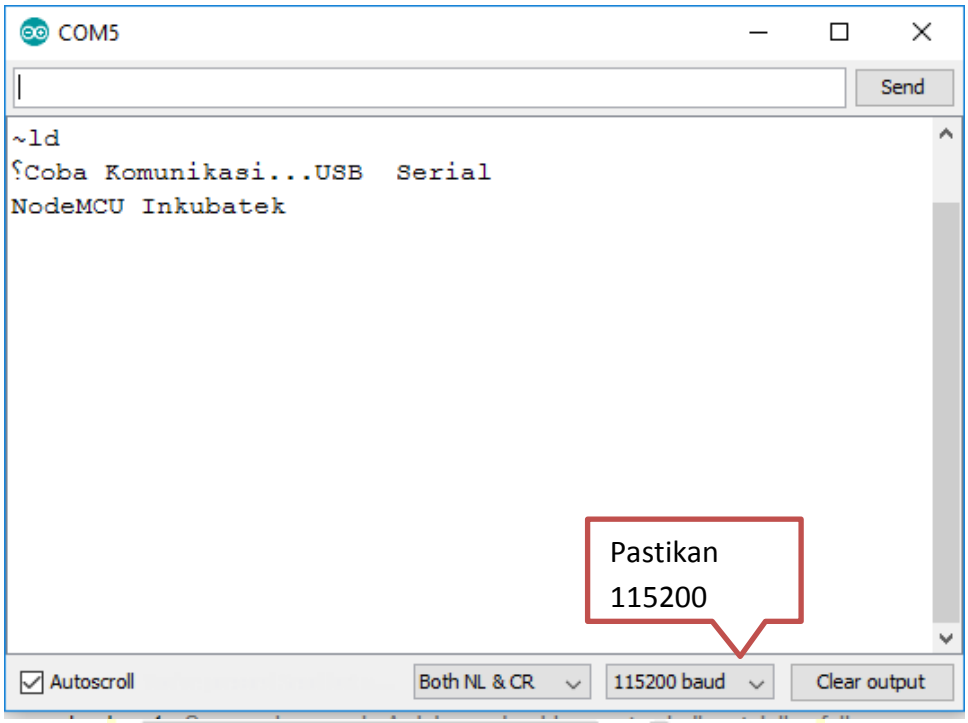
```
 Serial.println("USB  Serial");
 Serial.println("NodeMCU Inkubatek");
}
void loop()
{
}
```

Setelah anda simpan kemudian *verify* dan *upload*, silahkan buka program serial monitor pada software Arduino IDE. Klik ***Tools → Serial Monitor***.



Jalannya program :

Reset pada hardware (tekan tombol RESET), maka pada serial monitor akan tampil tulisan “Coba Komunikasi… USB 2 Serial” pada baris selanjutnya “NodeMCU Inkubatek”.



Penjelasan program :

Pada komunikasi serial yang penting nilai *baud rate* kedua device sama, dalam hal ini NodeMCU dengan komputer/laptop. Nilai *baud rate* yang dipilih adalah 115200 bps (*bit per second*), lihat pada pojok kanan bawah tampilan serial monitor (115200 baud). Pada program nilai baud rate ini ditulis ketika inisialisasi, pada setup() :

Serial.begin(115200);

Jika menggunakan nila baud rate yang lain tinggal mengganti nilai yang ada di dalam kurung begin(), misalnya 19200 bps, maka inisialisasinya menjadi :

Serial.begin(19200);

Sesuaikan juga nilai baud rate pada Serial Monitor, klik *scroll down* disamping nilai 9600 baud, pilih 19200 baud.

Baris berikutnya adalah mengirim data serial. Cukup kasih perintah *Serial.print()* data yang akan dikirim serial]).

Serial.print("Coba ");

Mudah bukan ? cukup dengan perintah *Serial.print()* maka data akan langsung terkirim. (“Coba ”) merupakan data string sehingga harus diapit tanda petik dua “ “.

Serial.print("Komunikasi...");

Selanjutnya NodeMCU mengirim lagi data string (“Komunikasi...”) dengan perintah yang sama, *Serial.print()* sehingga kita lihat pada Serial Monitor terlihat tulisan “Coba Komunikasi...” . (sebenarnya dapat sekali kirim dg perintah *Serial.print(“Coba Komunikasi…”),* dalam hal ini agar anda menjadi lebih paham).

Serial.println("USB 2 Serial");

Hampir sama dengan perintah sebelumnya, mengirim data serial, hanya saja perintahnya berbeda. Perhatikan untuk dua perintah sebelumnya menggunakan perintah *print()* dan terlihat data yang dikirim berikutnya berada disebelah kanannya. Perintah ***println()*** akan menyertakan tambahan karakter “***/n/r***” atau karakter ***Enter*** sehingga data yang dikirim berikutnya berada di baris baru dan kolom baru. Hal ini sama ketika kita menulis dengan Ms.Word, dengan menekan tombol Enter, maka tulisan berikutnya berada pada baris baru dan kolom baru.

Serial.println("NodeMCU Inkubatek");

Perhatikan setelah pengiriman data “NodeMCU Inkubatek” pada Serial Monitor tampil pada baris dan kolom baru. Perintah terakhir juga menggunakan *println().*

Program utama *loop()* tidak mengerjakan sesuatu sehingga kita kosongkan saja, anda dapat memodifikasinya sendiri.

Program berikutnya komunikasi serial 2 arah, Arduino akan mengirim dan menerima data komputer. Kali ini kita memerlukan 4 LED.

Langsung deh buat programnya.

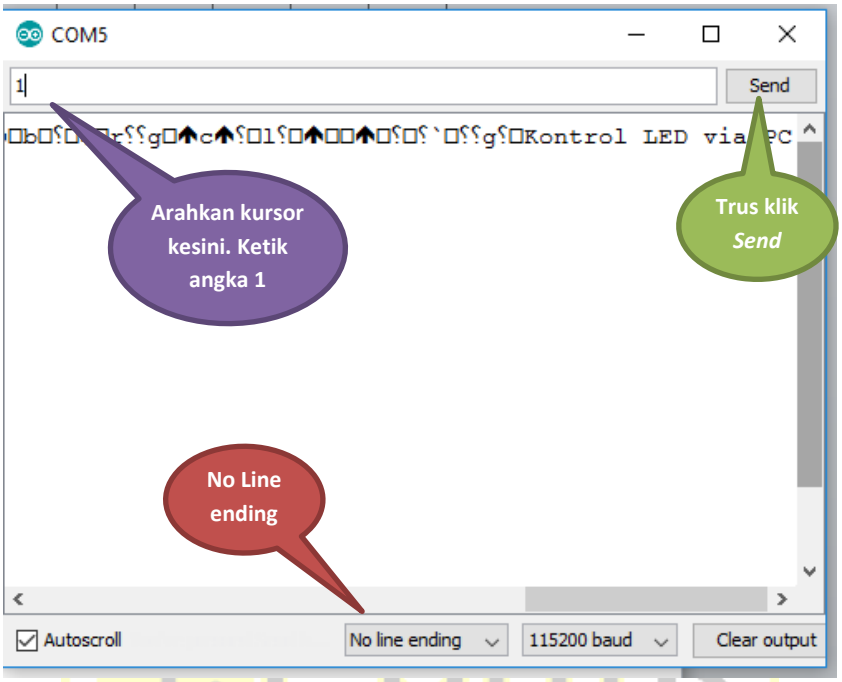
```
//Program 5 : kontrol LED via PC
byte in_serial,str;
void setup()
{
 pinMode(D5,OUTPUT);
 pinMode(D6,OUTPUT);
 pinMode(D7,OUTPUT);
 pinMode(D8,OUTPUT);
 Serial.begin(115200);
```

```
 Serial.println("Kontrol LED via PC");
}
void loop()
{
  in_serial=Serial.read(); //masih kode ASCII
  str=char(in_serial); //konversi data ASCII ke string
  switch(str){
    case '1': //LED 1 ON
            digitalWrite(D5,1);
            digitalWrite(D6,0);
            digitalWrite(D7,0);
            digitalWrite(D8,0);
   Serial.println("LED 1 ON"); break;
    case '2': //LED 2 ON
            digitalWrite(D5,0);
            digitalWrite(D6,1);
            digitalWrite(D7,0);
            digitalWrite(D8,0);
   Serial.println("LED 2 ON"); break;
    case '3': //LED 3 ON
            digitalWrite(D5,0);
            digitalWrite(D6,0);
            digitalWrite(D7,1);
            digitalWrite(D8,0);
   Serial.println("LED 3 ON"); break;
    case '4': //LED 4 ON
            digitalWrite(D5,0);
            digitalWrite(D6,0);
            digitalWrite(D7,0);
            digitalWrite(D8,1);
   Serial.println("LED 4 ON"); break;
```

```
  }
}
```

Jalannya program :

Setelah anda ***Upload***, buka Serial Monitor. Pada Serial Monitor akan tampil seperti pada gambar berikut.



Perhatikan juga bahwa semua led dalam keadaan padam. Arahkan kursor pada kolom paling atas, sebelah kiri tombol *Send*, kemudian ketik angka 1 terus klik tombol ***Send***. Led 1 akan nyala dan pada Serial Monitor tampil LED 1 ON. Cobalah untuk led 2, 3 dan led 4 dengan menulis angka 2, 3 atau 4 kemudian klik tombol Send.

Penjelasan program :

Pin D5 sampai dengan pin D8 digunakan sebagai pin output, dihubungkan dengan led, sehingga pinMode=OUTPUT.

pinMode(D5,OUTPUT);dan seterusnya…

Baud rate yang digunakan nilainya 115200 bps :

Serial.begin(115200);

Selanjutnya NodeMCU mengirim data string ke komputer, semua menggunakan perintah println().

Serial.println("Kontrol LED via PC");

Program utama kita buat untuk membaca data serial yang diterima, dalam hal ini kita membutuhkan variabel penampung data serial tersebut, ***in_serial*** dengan tipe data byte. Data yang masuk dibaca dengan perintah :

in_serial=Serial.read();

Format datanya masih ASCII sehingga perlu diubah kedalam tipe karakter :

str=char(in_serial);

Sebagai contoh jika kita mengirim karakter '1' dari komputer maka variabel in_serial = 48 (ASCII '1' = 48), nah agar menjadi karakter maka perlu perintah ***char()*** sehingga nilai variabel str='1'.

Karakter yang masuk (nilai variabel str) akan kita uji/cek untuk kemudian melakukan aksi berupa menghidupkan led sesuai dengan karakter yang diterima. Bentuk pengujian kondisi yang kita pakai adalah ***switch – case***.

```
switch(str){
    case '1':
    case '2':
    case '3':
    case '4':
}
```

Jika kita kirim karakter '1' maka nilai str='1' dan blok perintah pada ***case '1'*** yang dikerjakan.

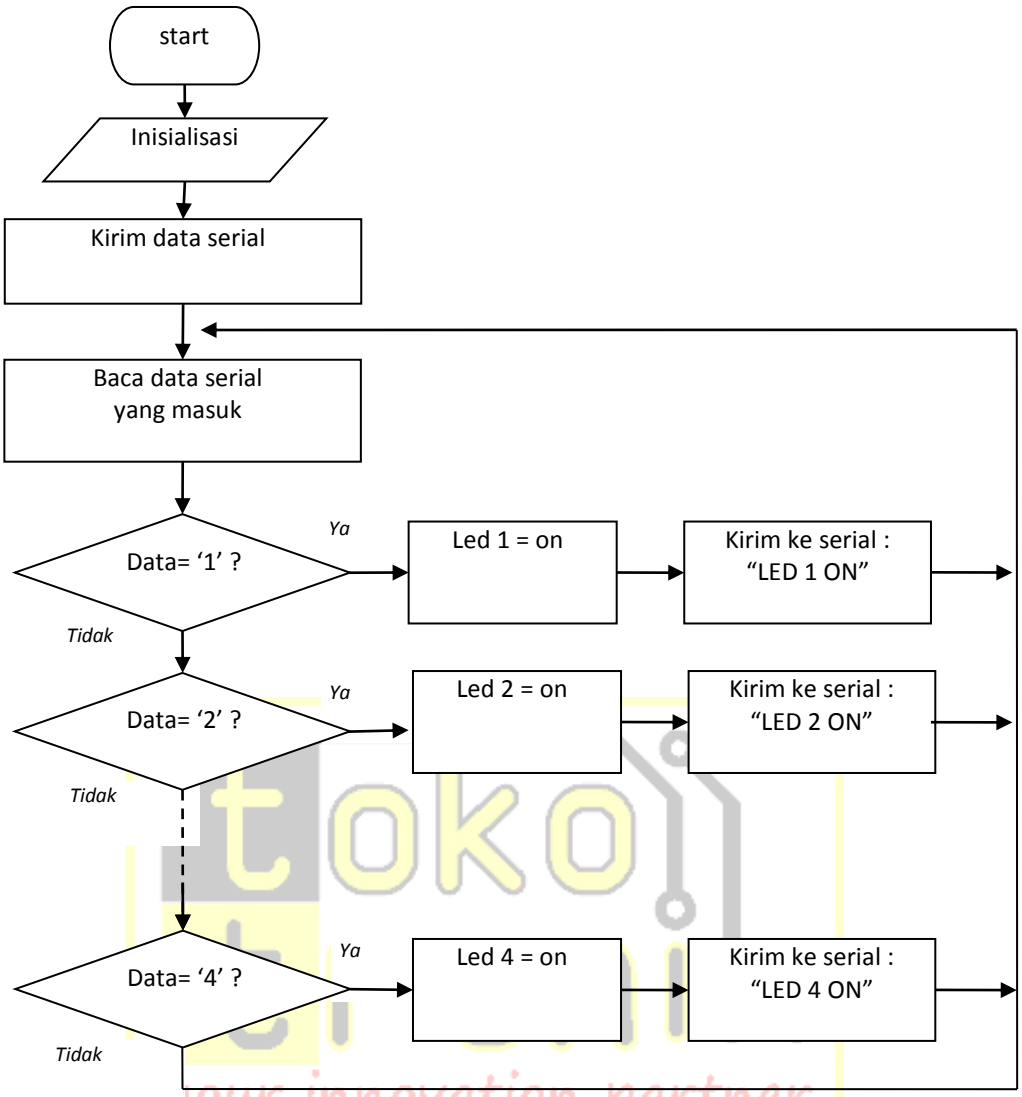
```
case '1': //LED 1 ON
      digitalWrite(D5,1);
      digitalWrite(D6,0);
      digitalWrite(D7,0);
      digitalWrite(D8,0);
     Serial.println("LED 1 ON"); break;
```

Inti dari perintah pada **case '1'** adalah menghidupkan LED 1 dan mengirim tulisan "LED 1 ON" ke komputer. Pernyataan ***'break'*** digunakan untuk mengakhiri setiap ***'case'***.

Demikian juga untuk *'case'* yang lain. Jika data yang diterima karakter '4' maka yang dikerjakan blok perintah pada ***'case 4'*****:**

```
case '4': //LED 4 ON
        digitalWrite(D5,0);
        digitalWrite(D6,0);
        digitalWrite(D7,0);
        digitalWrite(D8,1);
       Serial.println("LED 4 ON"); break;
```

Sekali lagi ini hanya contoh. Anda dapat menentukan sendiri karaketr untuk mengontrol LED-nya. Misal karakter 'a' untuk LED 1. Jumlah LED juga bebas ya. Silahkan berkreasi.



ANALOG INPUT ADC

Pada materi terdahulu kita sudah mencoba program pembacaan tombol. Semuanya termasuk digital input, NodeMCU membaca kondisi logika pada pin digital input. Hasil yang diperoleh hanya dua kemungkinan : HIGH (1) atau LOW (0). Analog input membaca sinyal masukan berupa tegangan analog. Kalau anda pernah mendengar istilah ADC (*Analog to Digital Converter*), maka ADC internal NodeMCU inilah yang melakukan tugas pembacaan sinyal analog tersebut. Terdapat 1 pin masukan analog, yaitu A0, jadi pin D2 misalnya, tidak bisa dipakai untuk membaca sinyal analog.

Proses pembacaannya sendiri sangat mudah, cukup satu baris perintah :

var=analogRead(pin);

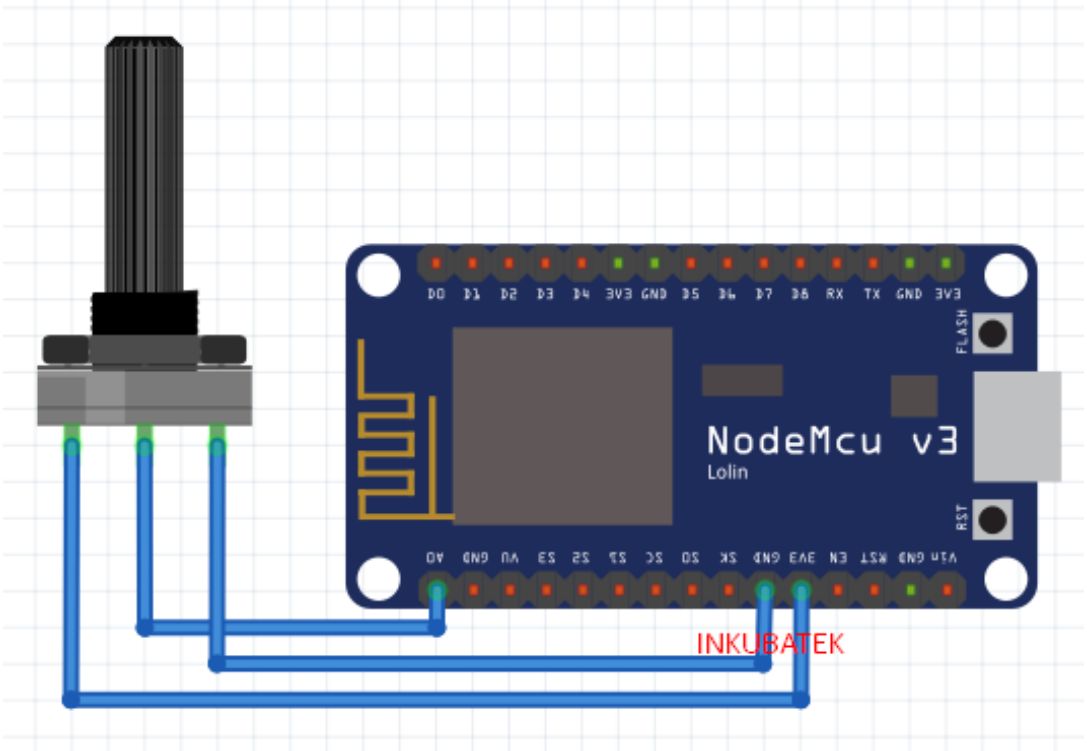
- ***var*** adalah variabel untuk menyimpan hasil pembacaan sinyal analog, bertipe integer (int).
- ***pin*** merupakan nomor pin analog, berisi A0

Resolusi ADC adalah 10 bit, artinya nilai terkecilnya 0 (tegangan masukan 0 volt) dan terbesar 1023 (tegangan masukan 3,3 volt). Tegangan masukan pada masing – masing pin analog input maksimal 3,3 volt. Nilai 1 bit dari hasil pembacaan ADC dapat kita hitung sebagai berikut :

$$1\ bit = \frac{3{,}3\ V}{1023} = 3{,}22\ mV$$

Artinya jika hasil pembacaan ADC misalnya 560, maka tegangan analog yang dibaca adalah 560 x 3,22 mV =1803 mV.

Supaya lebih mudah dipahami mari langsung kita praktekkan. Sebagai sumber tegangan analog kita ambil dari tegangan pada board NodeMCU (3,3 volt) yang dihubungkan dengan potensiometer, menjadi rangkaian *voltage devider*. Cukup koneksikan kaki tengah potensio dengan pin A0 serta masing- masing kaki yang lain di VCC dan GND.

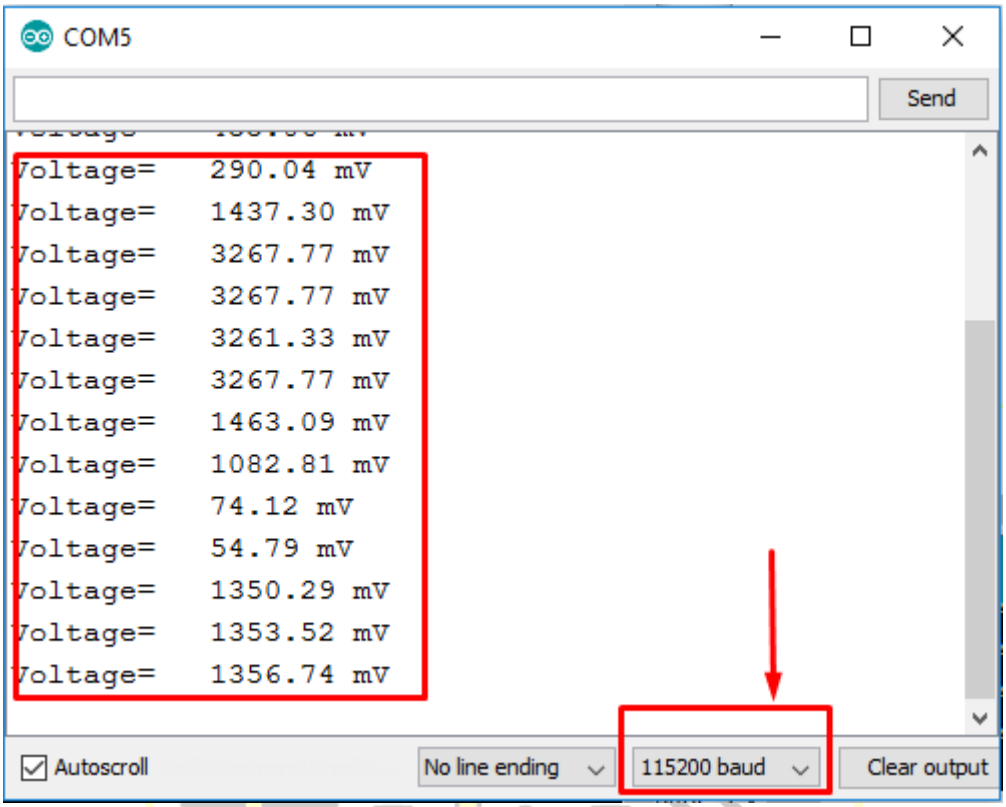


Programnya sebagai berikut :

```
//Program 6: membaca analog input A0
void setup()
{
 Serial.begin(115200);
}

void loop()
{
 int analogValue = analogRead(A0);
 float millivolts = (analogValue/1024.0) * 3300;
 Serial.print("Voltage=   ");
 Serial.print(millivolts);
 Serial.println(" mV");
 delay(1000);
}
```

Jalannya program :

Buka Serial Monitor (***Tools → Serial Monitor***) dengan baudrate 115200. Putar potensiometer dan amati hasilnya di Serial Monitor.



Penjelasan program :

Program dimulai dengan inisialisasi serial komunikasi.

Serial.begin(115200);

Program utama berisi looping pembacaan sinyal analog :

int analogValue = analogRead(A0);

Perintah tersebut membaca data analog di pin A0 karena output voltage devider (potensio) masuk ke pin A0.

*float millivolts = (analogValue/1024.0) * 3300;*

Selanjutnya nilai analog dikonversi ke tegangan (mV) dan hasilnya dikirim melalui serial komunikasi

Serial.print("Voltage= ");

Serial.print(millivolts);

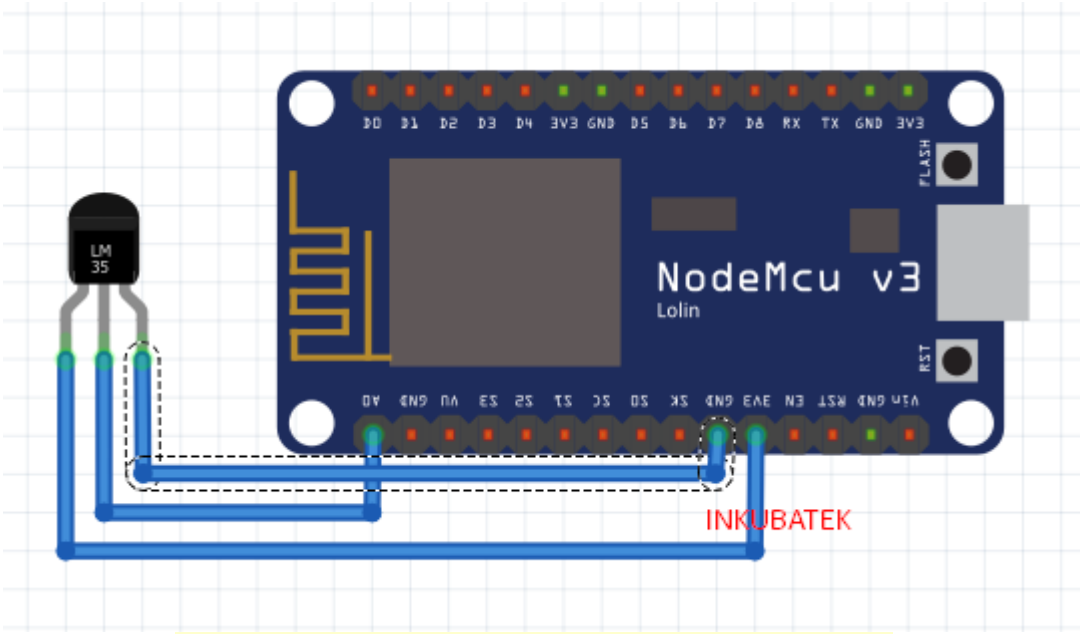
Serial.println(" mV");

delay(1000);

Bikin Thermometer Digital

Alat pengukur temperatur atau suhu lingkungan dapat kita buat secara mudah dengan Arduino ditambah sensor suhu. LM35 adalah sensor suhu yang biasa dipakai. Ada 3 kaki. Vcc (5V), Gnd dan Out. Untuk output berupa tegangan. Nilainya mewakili temperatur yang diukur. Perubahan outputnya mengikuti persamaan V_{out} = 10 mV/1^{0}C. Artinya, jika terbaca tegangan Vout = 300 mV, maka temperaturnya = 300 /10 = 30 ^{0}C.

Buat rangkaian sensor suhu LM35 dengan board NodeMCU.



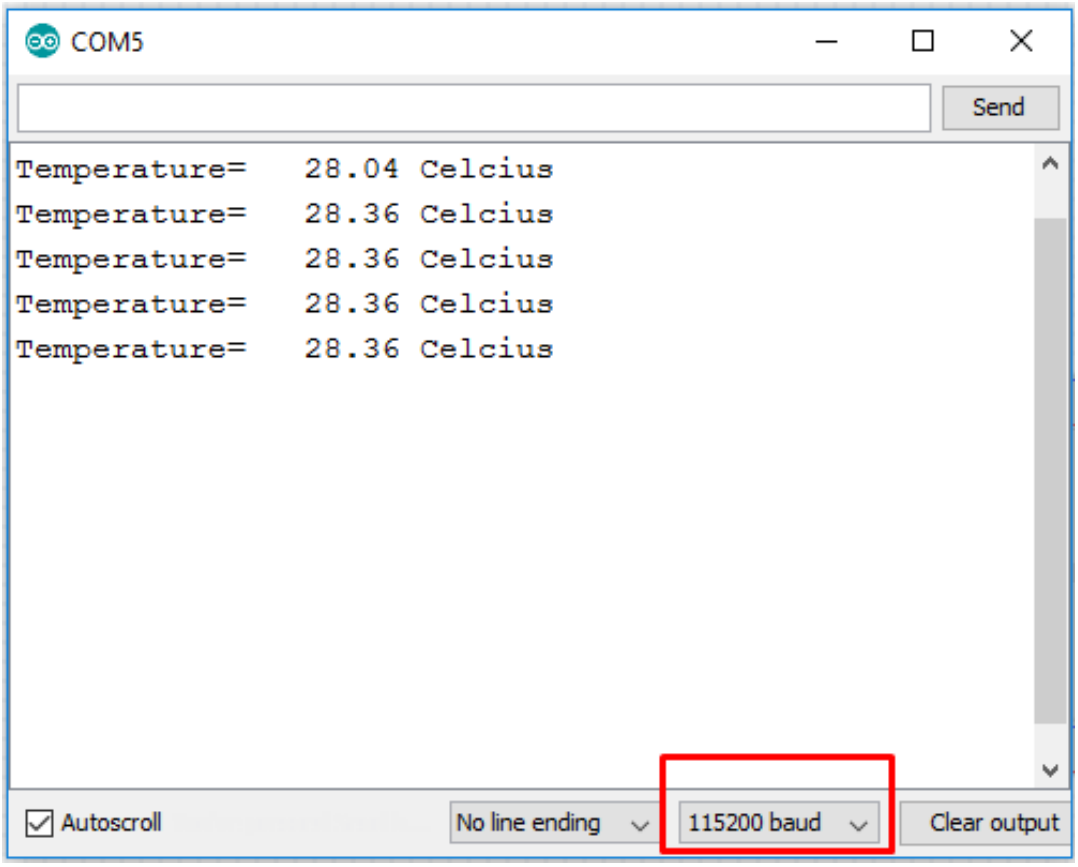
Kemudian siapkan programnya.

```
//Program 7: Digital Thermometer
void setup() {
Serial.begin(115200);
}
void loop() {
 int analogValue = analogRead(A0);
 float millivolts = (analogValue/1024.0) * 3300;
 float celsius = millivolts/10;
 Serial.print("Temperature=  ");
```

```
  Serial.print(celsius);
  Serial.println(" Celcius");
  delay(1000);
}
```

Jalannya program :

Buka Serial Monitor (***Tools → Serial Monitor***) dengan baudrate 115200. Putar potensiometer dan amati hasilnya di Serial Monitor.



Penjelasan program :

Program hamper sama dengan program 6, hanya saja ditambahi perhitungan suhu.

float celsius = millivolts/10;

Nilai suhu/temperature merupakan nilai tegangan hasil pembacaan ADC dibagi dengan 10.