# CODING BERBASIS ARDUINO

*Versi 1.0*

APA ITU CODING?

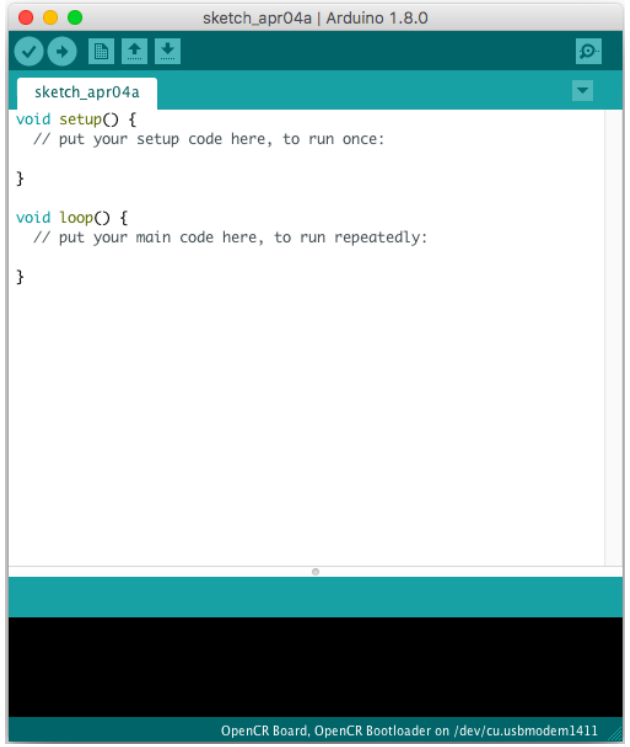
# CODING

adalah instruksi atau perintah kepada computer untuk menjalankan rancangan yang kita buat.

# BAHASA CODING

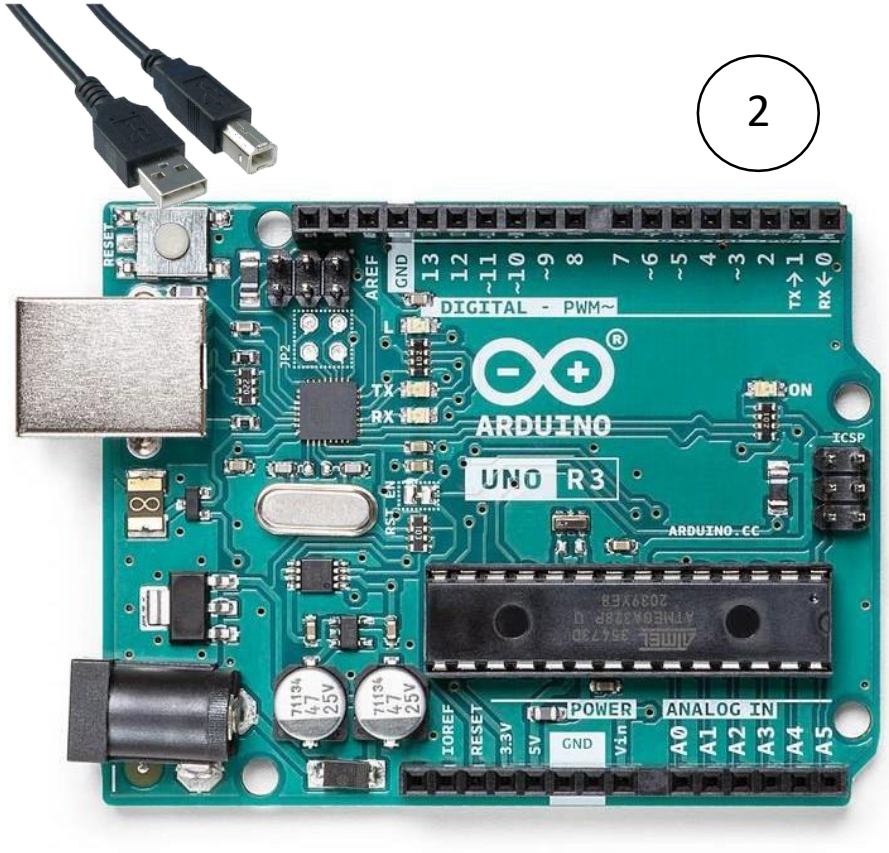
# TAHAPAN PEMBUATAN PROGRAM MICROCONTROLLER ARDUINO

1




Software editing/
compiler/ IDE

2




HARDWARE

# KELEBIHAN ARDUINO

✓ Open Source Hardware

Software ataupun hardware Arduino bersifat open source sehingga kita akan dapat membuat tiruan

✓ **Tidak Butuh Chip Programming/ downloader**

Chip yang ada pada Arduino dilengkapi bootloader yang nantinya akan dapat menangani proses upload dari komputer

✓ **Koneksi USB**

Sambungan dari board Arduino ke komputer dapat menggunakan USB dan tidak memerlukan parallel port ataupun serial port. Ini akan membuat sambungannya semakin mudah terutama bagi laptop yang tidak dilengkapi serial ataupun parallel port.

✓ **Fasilitas Chip Yang Lengkap**

Arduino telah menggunakan chip AVR AT mega 168/328 dengan kapasitas PWM, ADC, timer, komunikasi serial, interupt, I2C serta SPI. Ini berarti, Arduino dapat digabung bersama modul maupun alat lain yang protokolnya berbeda-beda.

# KELEBIHAN ARDUINO

✓ **Ukuran Kecil**

Arduino memiliki ukuran yang relatif kecil sehingga dapat dengan mudah dibawa kemanapun kita pergi. Arduino bisa dibawa bersama laptop ataupun dimasukkan dalam saku.

✓ **Bahasa Pemrograman Relatif Mudah**

Walau bahasa pemrograman yang digunakan ialah bahasa C/C++ namun dengan ditambahkannya library serta beragam fungsi standar membuat Ardunio bisa dipelajari dengan lebih mudah.

✓ **Library Gratis**

Tersedia library gratis yang akan menghubungkan Arduino dengan beragam sensor, modul komunikasi serta aktuator. Sebagai contoh adalah library khusus keyboard, mouse, GPS, servo dan lain-lain. Mengingat Arduino bersifat open source, maka library yang ada juga bisa didownload secara gratis di website resmi Arduino.

# KELEBIHAN ARDUINO


✓ **Pengembangan Aplikasinya Mudah**

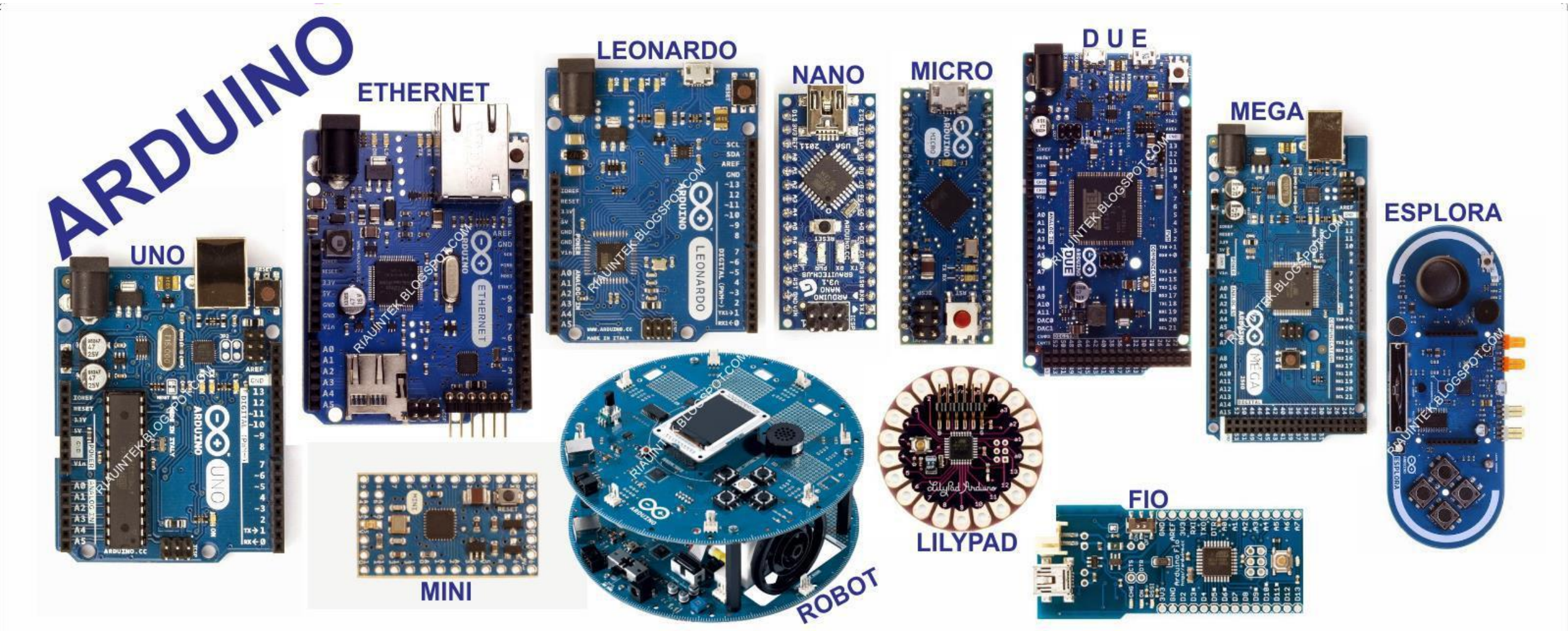
Mengembangkan aplikasi elektronik satu ini terbilang mudah, apalagi dengan adanya bahasa pemrograman yang mudah serta library dasar yang terbilang lengkap. Contohnya adalah apabila kita hendak membuat sebuah sensor suhu. Cukup beli IC sensor suhu dan sambungkan ke Arduino. Apabila suhu hendak ditampilkan di LCD, belilah LCD dan tambahkan ke library.

✓ **Komunitas Open Source Saling Mendukung**

Ada komunitas saling mendukung di dalamnya demi pengembangan proyek. pengembangan software maupun hardwarenya didukung para pecinta elektronika di seluruh dunia.

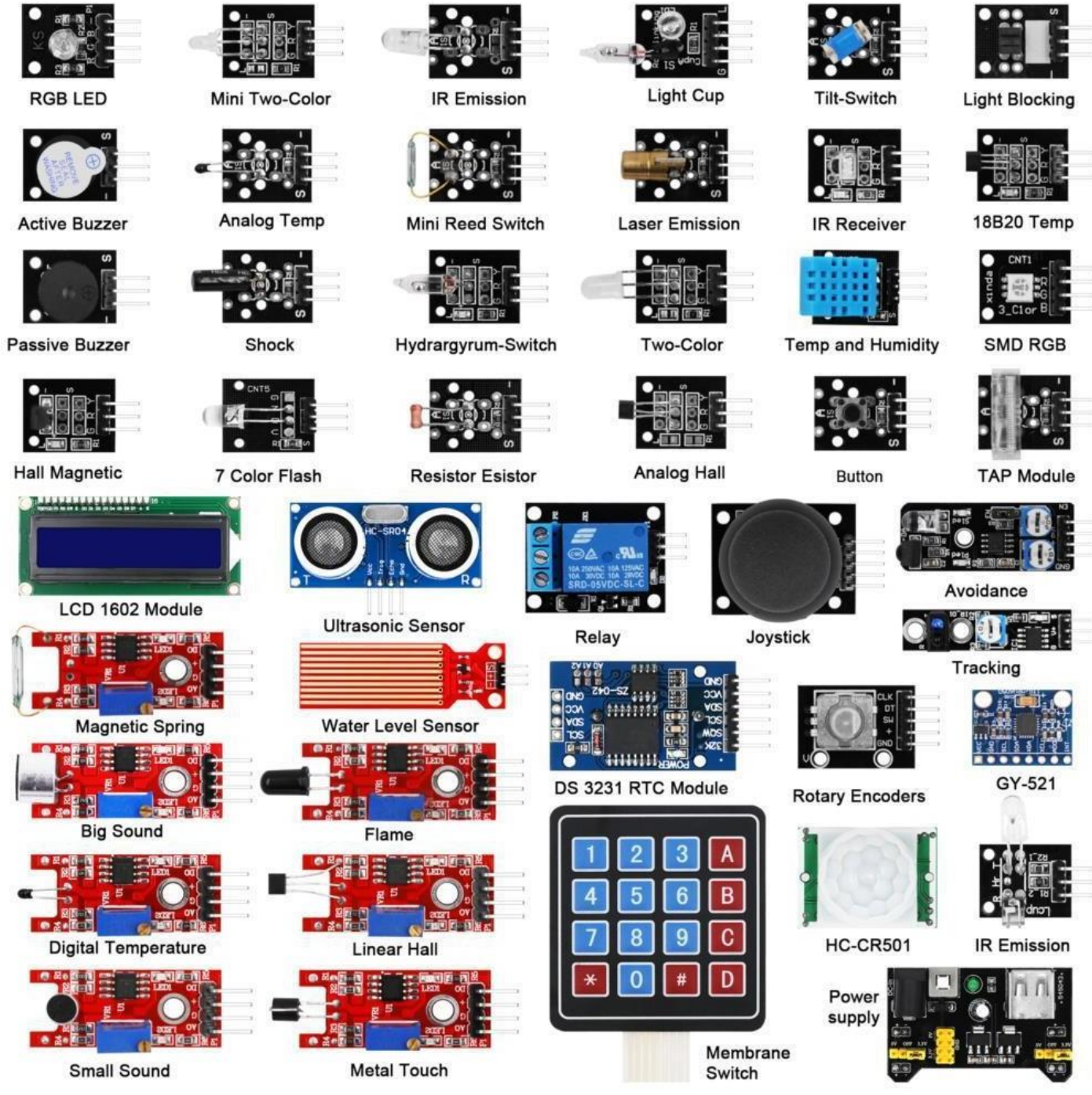
# VARIAN ARDUINO

BOARD ARDUINO



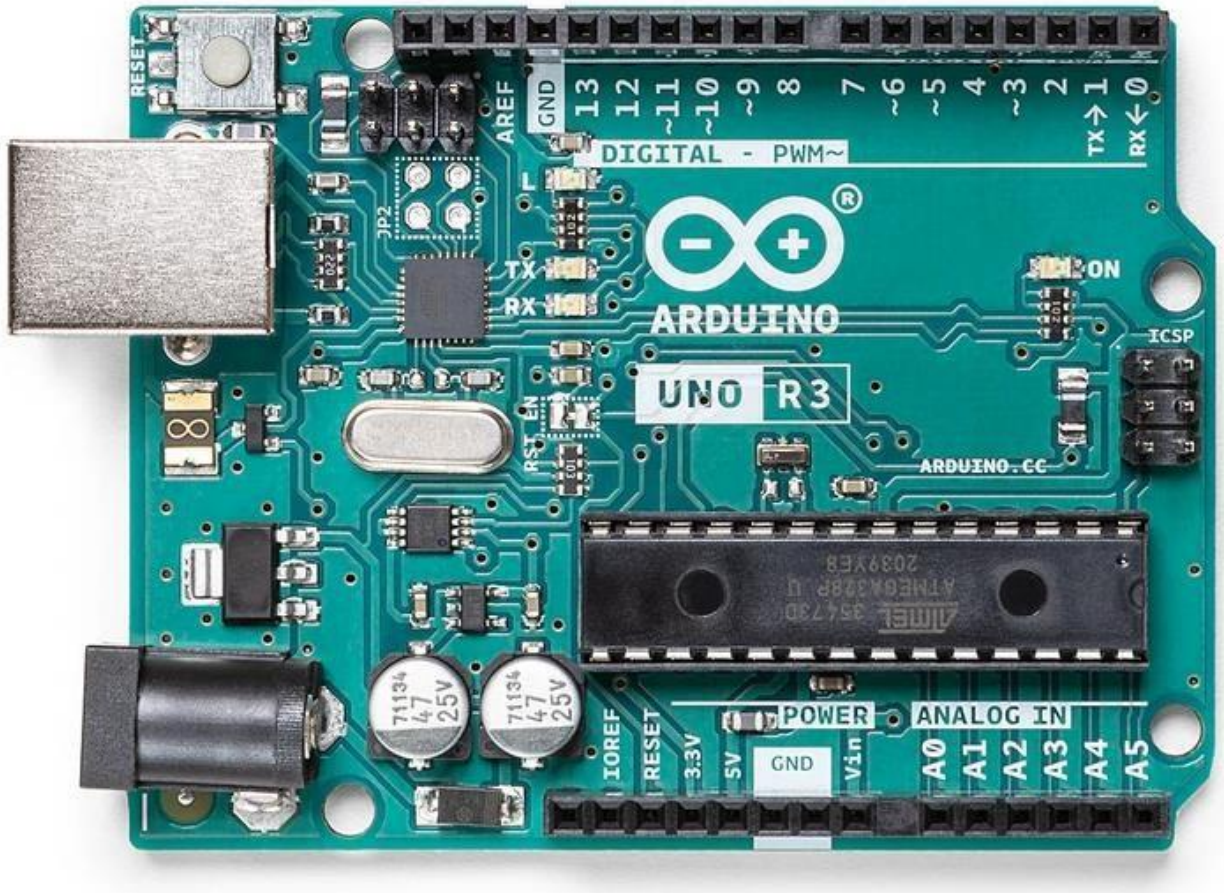
SHIELDS ARDUINO

# MODUL SENSOR ARDUINO



# Arduino UNO

| | |
|---|---|
| MICROCONTROLLER | ATmega328P |
| OPERATING VOLTAGE | 5V |
| INPUT VOLTAGE (RECOMMENDED) | 7-12V |
| INPUT VOLTAGE (LIMIT) | 6-20V |
| DIGITAL I/O PINS | 14 (of which 6 provide PWM output) |
| PWM DIGITAL I/O PINS | 6 |
| ANALOG INPUT PINS | 6 |
| DC CURRENT PER I/O PIN | 20 mA |
| DC CURRENT FOR 3.3V PIN | 50 mA |
| FLASH MEMORY | 32 KB (ATmega328P) of which 0.5 KB used by bootloader |
| SRAM | 2 KB (ATmega328P) |
| EEPROM | 1 KB (ATmega328P) |
| CLOCK SPEED | 16 MHz |
| LED_BUILTIN | 13 |
| LENGTH | 68.6 mm |
| WIDTH | 53.4 mm |
| WEIGHT | 25 g |



# AYO KITA MULAI

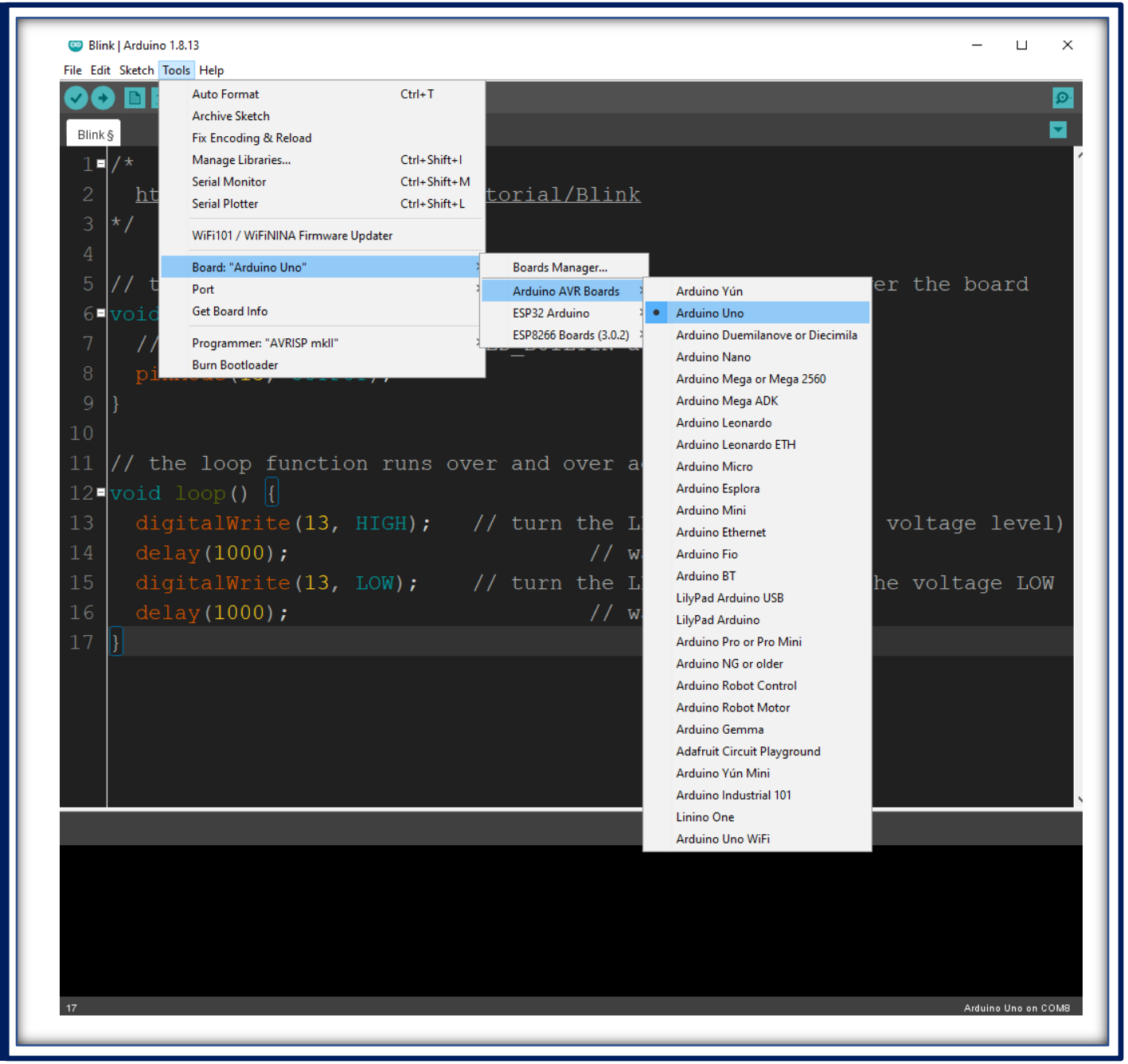
- **Download & install the Arduino environment (IDE (**Integrated Development Environment))
  www.arduino.cc
- **Sambungkan hardware ke computer melalui port USB**
- **Jika dibutuhkan, Install Driver Board**
  https://learn.sparkfun.com/tutorials/how-to-install-ch340-drivers/all
- **Jalankan Program Arduino IDE**
- **Open code sample pilih → file → example → basic → Blink**
- **Pilih Tipe Board**
- **Pilih Serial Port**
- **Upload the program**


Komputer + Software Arduino IDE


ARDUINO UNO

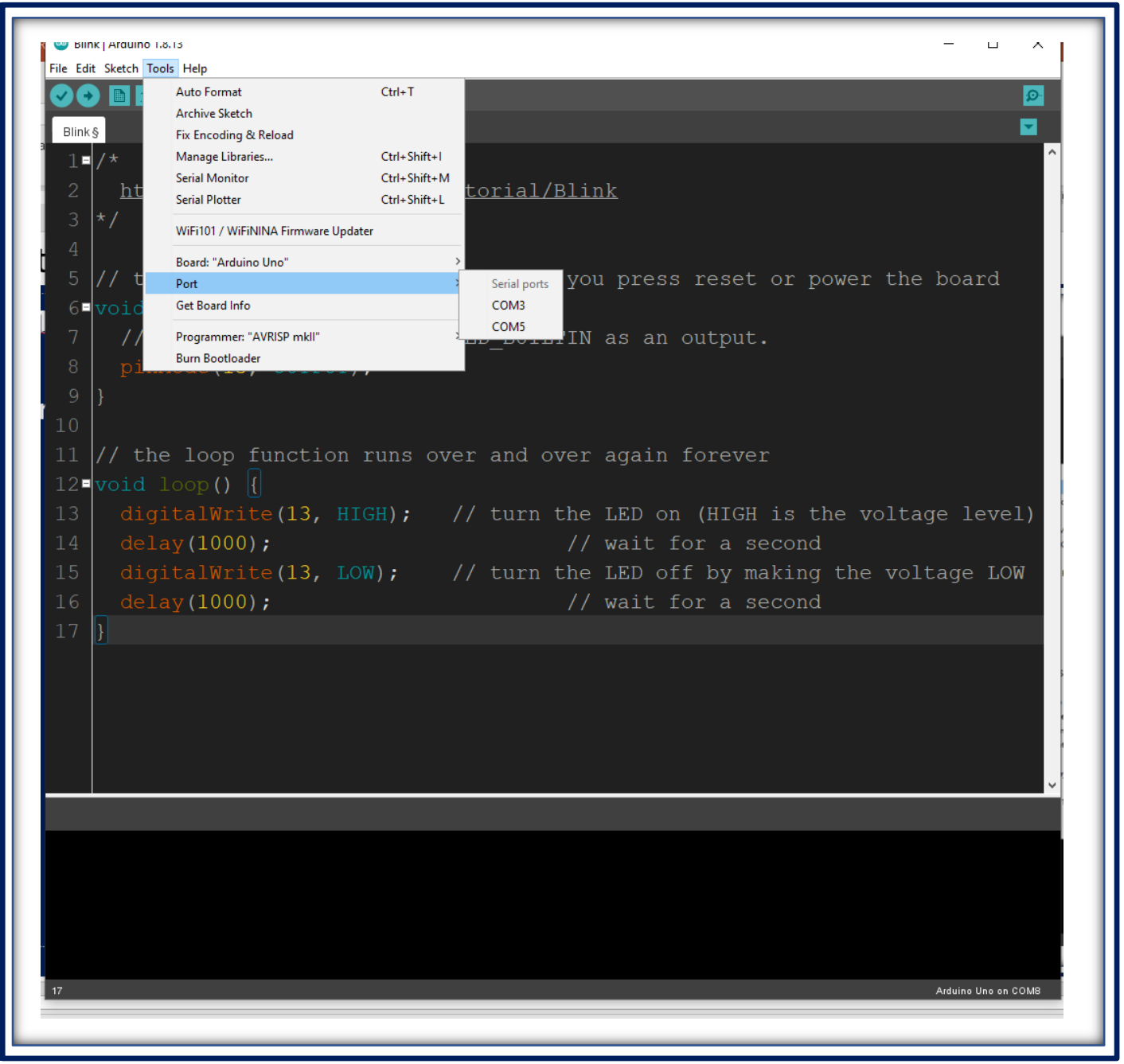
# Memilih tipe Board

1. Klik **Tool**
2. Pilih **Board “…….”**
3. Pilih **Arduino AVR Board**
4. Pilih board sesuai dengan yang digunakan, contoh karena menggunakan Arduino uno maka pilih **Arduino uno**

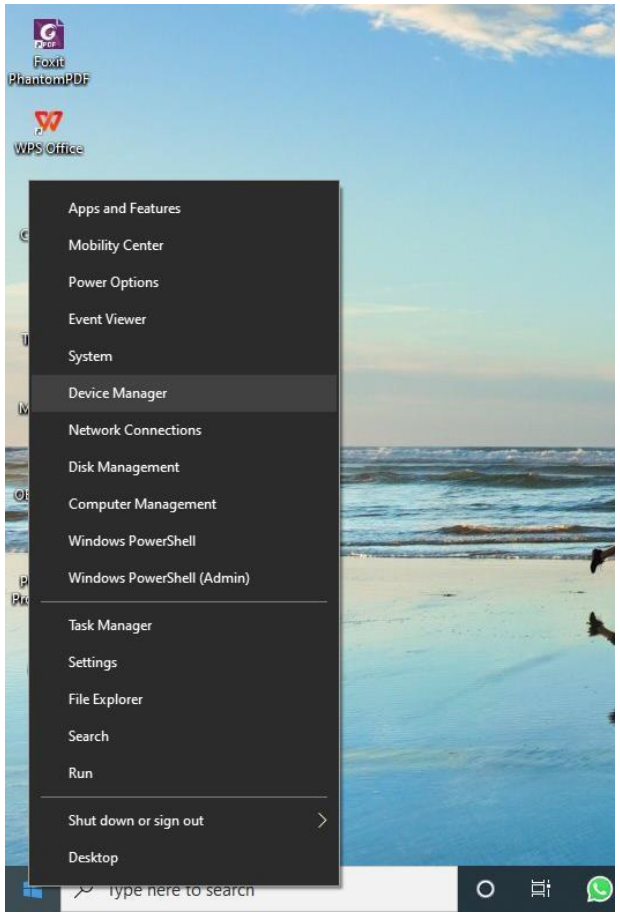


# Memilih Port USB

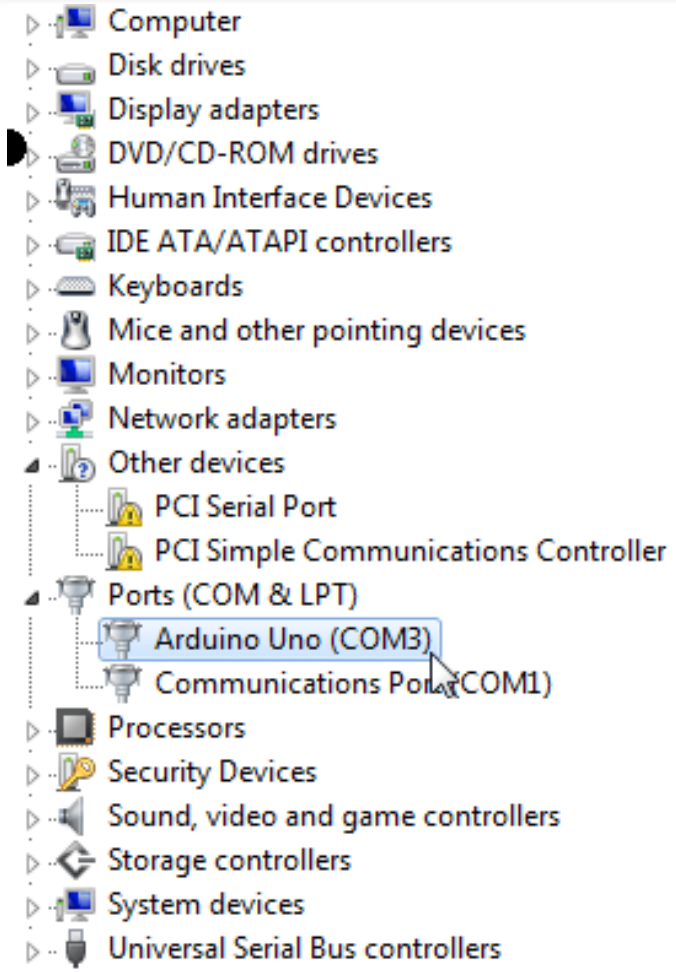
1. Klik **Tool**
2. Pilih **Port**
3. Pilih **Serial port** sesuai dengan **COMx**



## Melakukan Cek PORT COM USB

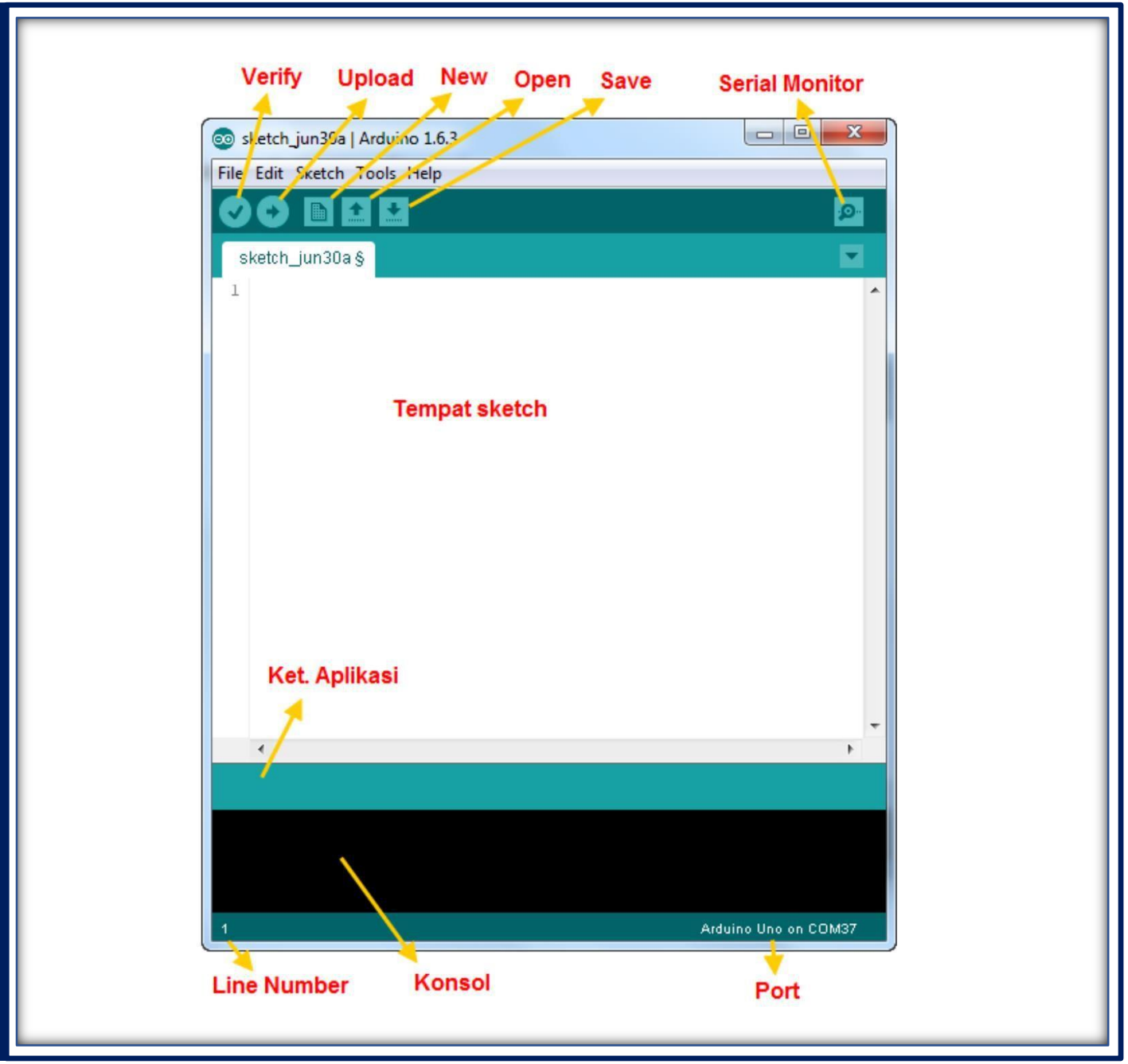



1. Klik kanan windows start
2. Pilih Device manager

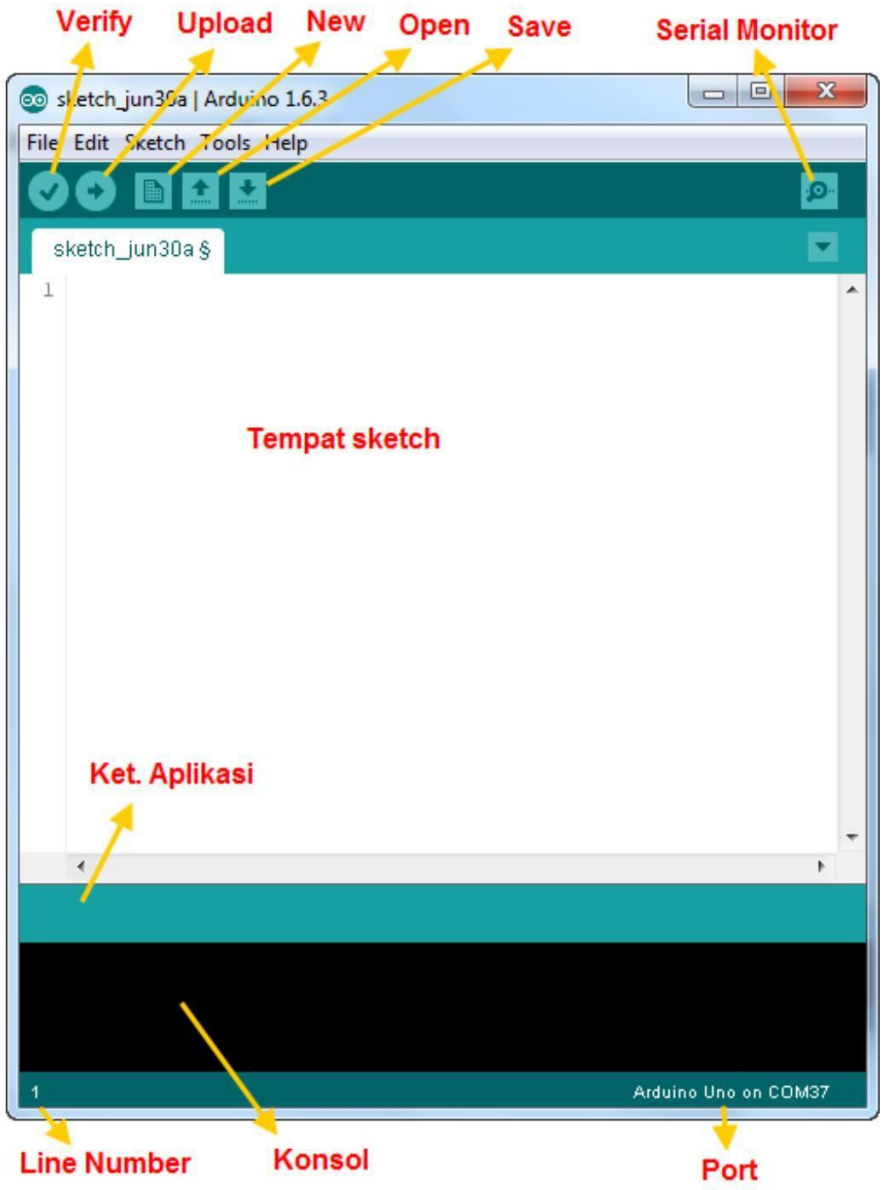



1. Ditailkan PORT (COM & LPT)
2. Lihat Arduino uno COMx

**Verify** : pada versi sebelumnya dikenal dengan istilah *Compile*. Sebelum aplikasi diupload ke *board* Arduino, biasakan untuk memverifikasi terlebih dahulu *sketch* yang dibuat. Jika ada kesalahan pada *sketch*, nanti akan muncul error. Proses Verify / *Compile* mengubah *sketch* ke *hex code* untuk diupload ke mikrokontroller.



**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.



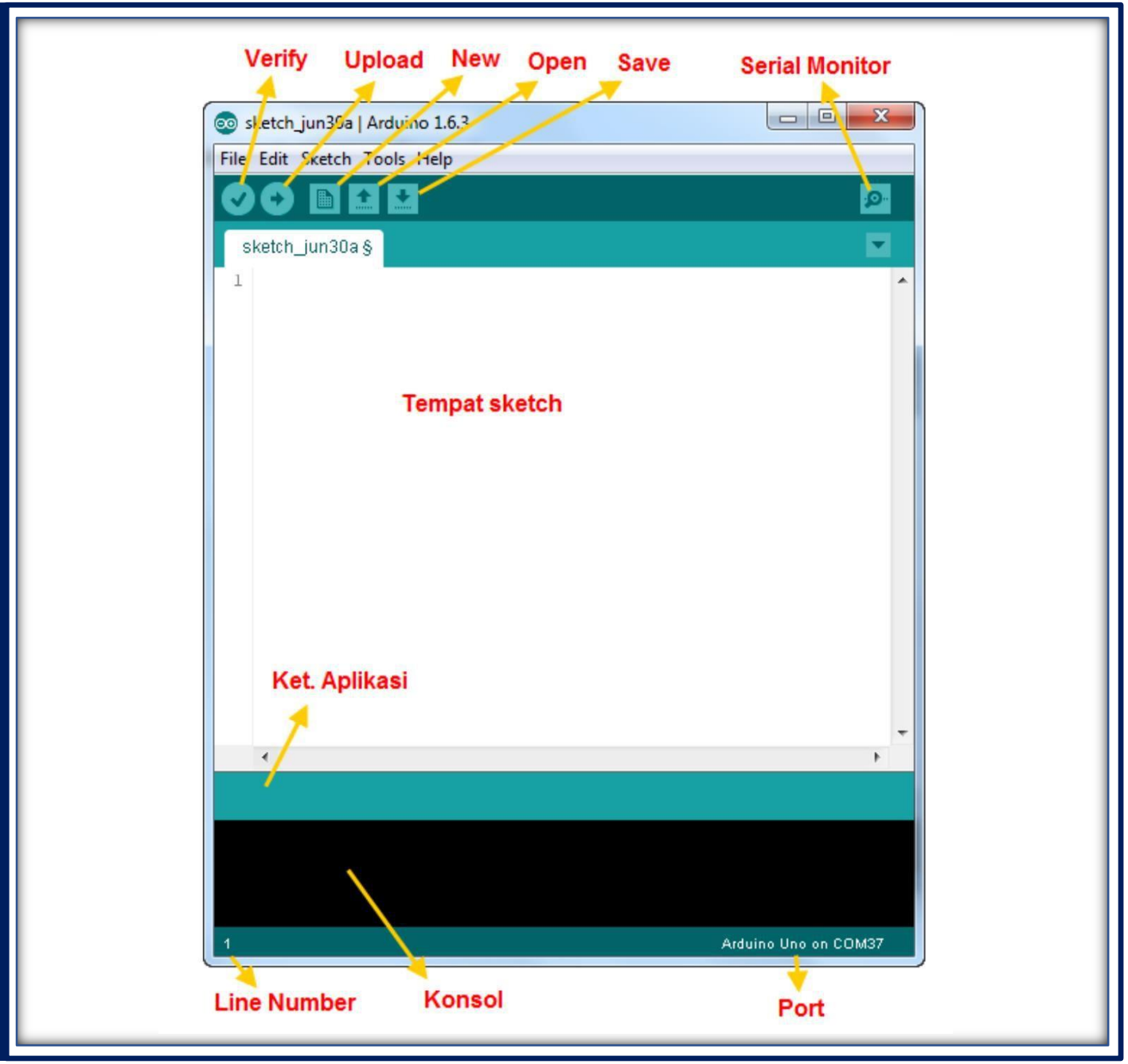
**New Sketch** : Membuka window dan membuat *sketch* baru

**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

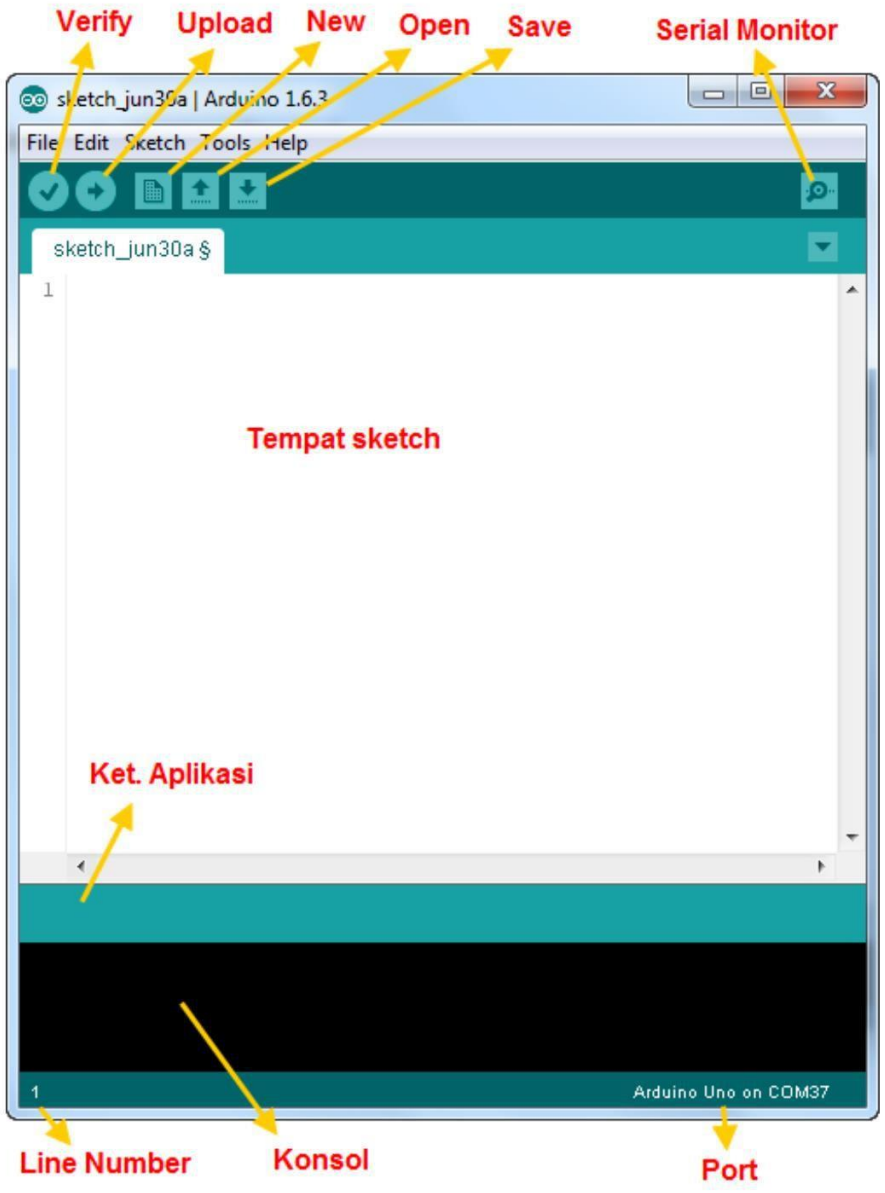
**Keterangan Aplikasi** : pesan-pesan yang dilakukan aplikasi akan muncul di sini, misal "*Compiling*" dan "*Done Uploading*" ketika kita mengcompile dan mengupload *sketch* ke *board* Arduino

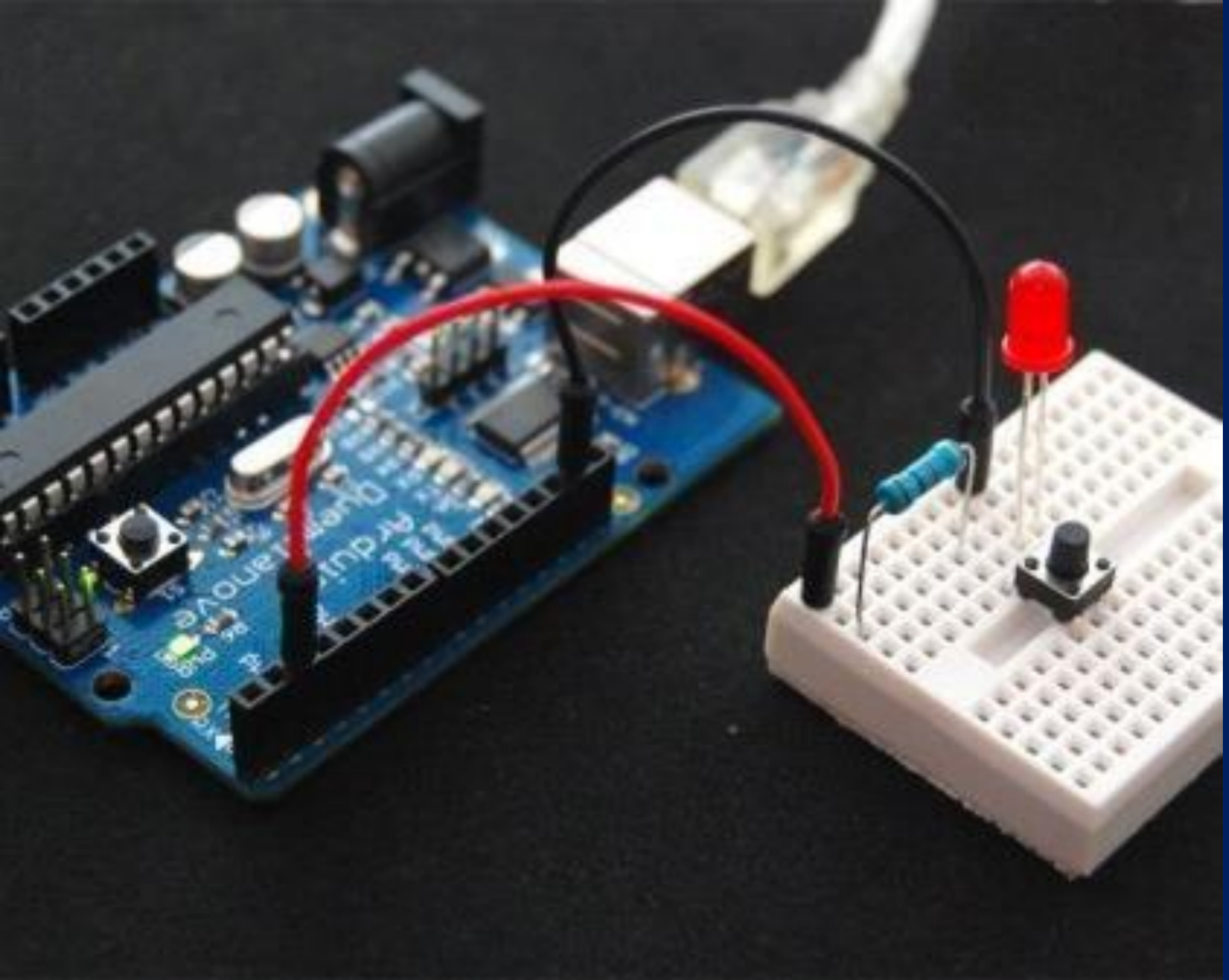


**Konsol** : Pesan-pesan yang dikerjakan aplikasi dan pesan-pesan tentang *sketch* akan muncul pada bagian ini. Misal, Ketika aplikasi mengcompile atau ketika ada kesalahan pada *sketch* yang kita buat, maka informasi *error* dan baris akan diinformasikan di bagian ini.

**Baris Sketch** : bagian ini akan menunjukkan posisi baris kursor yang sedang aktif pada *sketch.*

**Informasi Port** : bagian ini menginformasikan *port* yangdipakah oleh *board* Arduino.





# BAHASA PEMROGRAMAN ARDUINO

*Versi 1.0*

APA ITU BAHASA?

**Language Reference/** Referensi Bahasa

Bahasa pemrograman Arduino dapat dibagi menjadi tiga bagian utama:

1. ***functions*** /fungsi,
2. ***Value*** /nilai (***variables*** /variable/ nilai berubah ubah dan **constants** /konstanta/ nilai tetap),
3. structure /struktur.

## 1. *FUNCTIONS* /FUNGSI,

Untuk mengendalikan papan Arduino dan melakukan perhitungan.

- **Digital I/O**
- **Analog I/O**
- **Advanced I/O**
- **Time**
- **Communication**
- **Math**
- **Trigonometry**
- **Characters**
- **Random Numbers**
- **Bits and Bytes**
- **External Interrupts**
- **Interrupts**
- **USB**

- **Digital I/O**
  - pinMode()
  - digitalWrite()
  - digitalRead()

- **pinMode()**

  mengkonfigurasikan Pin menjadi **input** atau **Output;**

  Syntax

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

  pin : Pin Arduino yang dipilih; contoh 1, 2, 3
  mode : INPUT, OUTPUT, atau INPUT_PULLUP.

**Notes and Warnings**

The analog input pins can be used as digital pins, referred to as A0, A1, etc

- **digitalWrite()**

  Untuk menjadikan Output PIN menjadi HIGH atau LOW,

  Jika pin telah dikonfigurasi sebagai OUTPUT dengan pinMode(),

  HIGH = 5V (atau 3.3V pada board 3.3V),

  LOW = 0V (ground).

  Syntax

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

  Pin : Pin pada Arduino yang dipilih, contoh 1, 2, 3

  Value : HIGH or LOW.

**Notes and Warnings**

The analog input pins can be used as digital pins, referred to as A0, A1, etc

- **digitalRead()**

  Membaca nilai dari kondisi digital pin arduino, antara HIGH atau LOW.

  Syntax

  ```
  digitalRead(pin);
  ```

  Pin : Pin pada Arduino yang akan dibaca, contoh 1, 2, 3

**Notes and Warnings**

The analog input pins can be used as digital pins, referred to as A0, A1, etc

- **Analog I/O**
  - analogRead()
  - analogWrite()
  - analogReference()

- **analogRead()**

Membaca nilai dari pin analog, ADC (analog to Digital Converter); contoh 10-bit ADC, berarti merubah tegangan 0 ~ (5V atau 3.3V) menjadi *integer 0 ~1023*.
Ini menghasilkan resolusi 5V/1024=0,0049V (4,9mV);

Syntax

```
analogRead(pin);
```

pin : Pin Arduino yang dipilih; contoh A0, A1, ect

| BOARD | OPERATING VOLTAGE | USABLE PINS | MAX RESOLUTION |
|---|---|---|---|
| Uno | 5 Volts | A0 to A5 | 10 bits |
| Mini, Nano | 5 Volts | A0 to A7 | 10 bits |
| Mega, Mega2560, MegaADK | 5 Volts | A0 to A14 | 10 bits |
| Micro | 5 Volts | A0 to A11* | 10 bits |
| Leonardo | 5 Volts | A0 to A11* | 10 bits |
| Zero | 3.3 Volts | A0 to A5 | 12 bits** |
| Due | 3.3 Volts | A0 to A11 | 12 bits** |
| MKR Family boards | 3.3 Volts | A0 to A6 | 12 bits** |

*A0 through A5 are labelled on the board, A6 through A11 are respectively available on pins 4, 6, 8, 9, 10, and 12

**The default `analogRead()` resolution for these boards is 10 bits, for compatibility. You need to use analogReadResolution() to change it to 12 bits.

- **analogWrite()**

  Menulis nilai analog (gelombang PWM *pulse width modulation*) ke pin. Dapat digunakan untuk menyalakan LED pada berbagai kecerahan atau menggerakkan motor pada berbagai kecepatan.

  Syntax

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

  pin : Arduino pin;

  pin : the duty cycle: between 0 ~ 255

| BOARD | PWM PINS | PWM FREQUENCY |
|---|---|---|
| Uno, Nano, Mini | 3, 5, 6, 9, 10, 11 | 490 Hz (pins 5 and 6: 980 Hz) |
| Mega | 2 - 13, 44 - 46 | 490 Hz (pins 4 and 13: 980 Hz) |
| Leonardo, Micro, Yún | 3, 5, 6, 9, 10, 11, 13 | 490 Hz (pins 3 and 11: 980 Hz) |
| Uno WiFi Rev2, Nano Every | 3, 5, 6, 9, 10 | 976 Hz |
| MKR boards * | 0 - 8, 10, A3, A4 | 732 Hz |
| MKR1000 WiFi * | 0 - 8, 10, 11, A3, A4 | 732 Hz |
| Zero * | 3 - 13, A0, A1 | 732 Hz |
| Nano 33 IoT * | 2, 3, 5, 6, 9 - 12, A2, A3, A5 | 732 Hz |
| Nano 33 BLE/BLE Sense | 1 - 13, A0 - A7 | 500 Hz |
| Due ** | 2-13 | 1000 Hz |
| 101 | 3, 5, 6, 9 | pins 3 and 9: 490 Hz, pins 5 and 6: 980 Hz |

- **Time**
  - delay()
  - delayMicroseconds()
  - micros()
  - millis()

- **delay()**

Menjeda program untuk jumlah waktu (dalam milidetik). 1000 milidetik = 1 detik.

Syntax

```
delay(ms);
```

ms : nilai milliseconds; contoh 1000 = 1000ms = 1S

- **delayMicroseconds()**

Menjeda program untuk jumlah waktu (dalam Microdetik).

1000 microdetik dalam satu milidetik.

Syntax

```
delayMicroseconds(us);
```

ms : nilai microseconds; contoh 1000 = 1000us = 1mS

- **Advanced I/O**
  - noTone()
  - pulseIn()
  - pulseInLong()
  - shiftIn()
  - shiftOut()
  - tone()
- **Math**
  - abs()
  - constrain()
  - map()
  - max()
  - min()
  - pow()
  - sq()
  - sqrt()
- **Trigonometry**
  - cos()
  - sin()
  - tan()
- **Communication**
  - Serial
  - Stream

- **Character**
  - isAlpha()
  - isAlphaNumeric()
  - isAscii()
  - isControl()
  - isDigit()
  - isGraph()
  - isHexadecimalDigit()
  - isLowerCase()
  - isPrintable()
  - isPunct()
  - isSpace()
  - isUpperCase()
  - isWhitespace()
- **Random Number**
  - random()
  - randomSeed()
- **Bits dan Bytes**
  - bit()
  - bitClear()
  - bitRead()
  - bitSet()
  - bitWrite()
  - highByte()
  - lowByte()
- **External Interrupts**
  - attachInterrupt()
  - detachInterrupt()
- **Interrupts internal**
  - interrupts()
  - noInterrupts()
- **USB**
  - Keyboard
  - Mouse

# *VARIABLE*

**Variabel** adalah kode program yang digunakan untuk menyimpan suatu nilai pada sebuah nama.
Nama tidak boleh ada spasi

Contoh

```
Int  hasilBacaSensor;
Int  hasil_baca_sensor;
float  nilai_kalkulasi;
Unsigned char kondisi=0;
```

Int, float, unsigned char adalah **tipe data**

# TYPE DATA

| Type Data | Ukuran Memori byte | Range |
|---|---|---|
| boolean | 1 | True or flase |
| char | 1 | -128 to +127, tipe data digunakan untuk penyimpanan nilai character merujuk pada tabel ASCII |
| unsigned Char | 1 | 0 to 255 |
| byte | 1 | 0 to 255 |
| int | 2 | -32768 to 32767 |
| unsigned int | 2 | 0 to 65535 |
| word | 2 | 0 to 65535 |
| long | 4 | -2.147.483.648 to 2.147.483.647 |
| unsigned long | 4 | 0 to 4.294.967.295 |
| float | 4 | -3.4028235E+38 to 3.4028235E+38 |
| double | 4 | -3.4028235E+38 to 3.4028235E+38 |

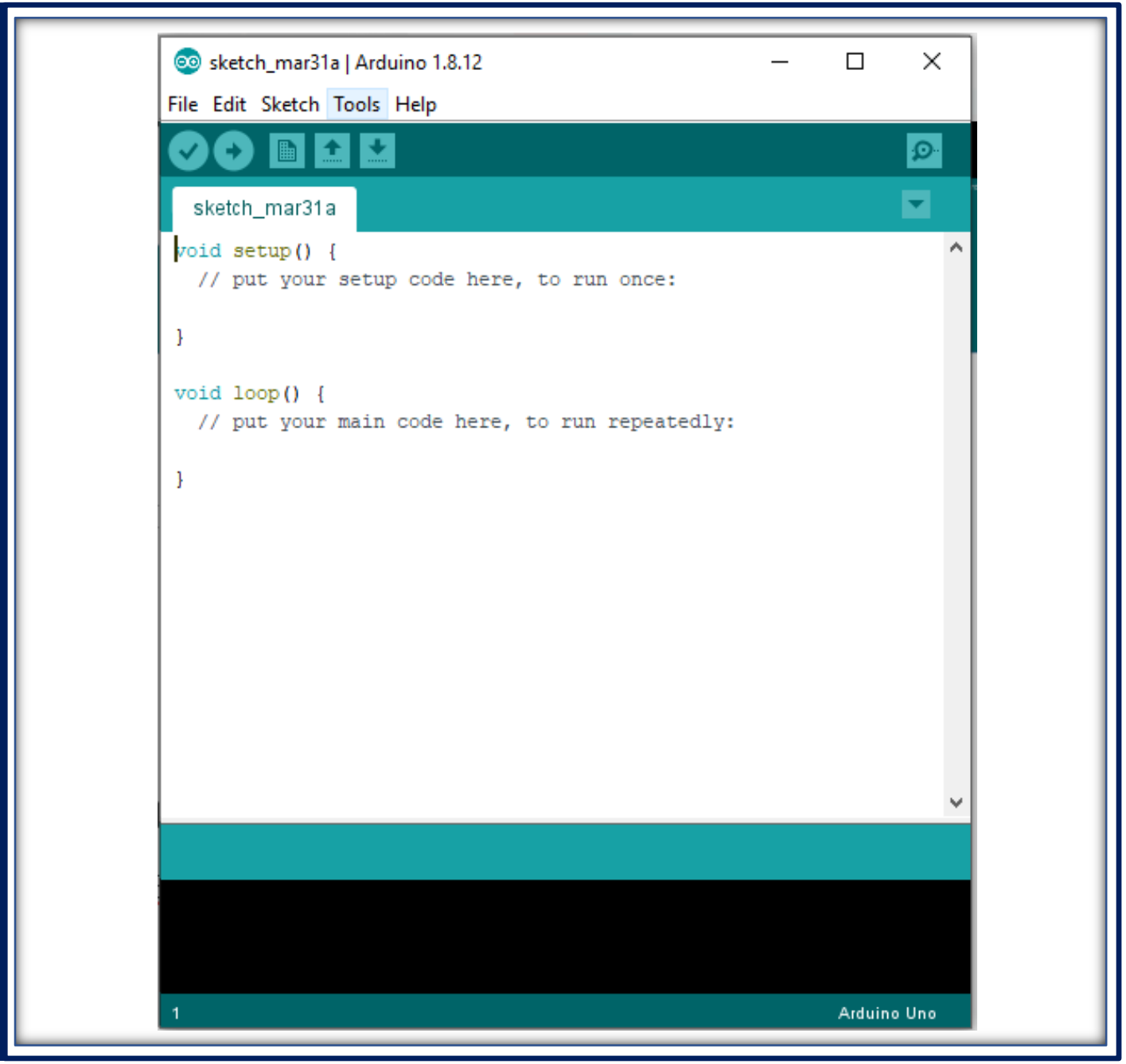
```
Int  hasilBacaSensor;     //mendeklarasikan nama variable hasilBacaSensor dengan type
                            data integer (Int)

Int  hasilBacaSensor = 0;     //mendeklarasikan nama variable hasilBacaSensor dengan
                                type data integer (Int) dengan nilai awal 0;
```

# structure /struktur

**void** digunakan untuk diklarasi fungsi

Fungsi **setup ()** dipanggil saat sketsa dimulai. Gunakan untuk menginisialisasi variabel, mode pin, mulai menggunakan library, dll. Fungsi **setup()** hanya akan berjalan satu kali, setelah setiap powerup atau reset board Arduino.

fungsi **loop()** melakukan pengulang program secara berurutan/ Gunakan untuk secara aktif mengontrol papan Arduino./ tempat penulisan program control.



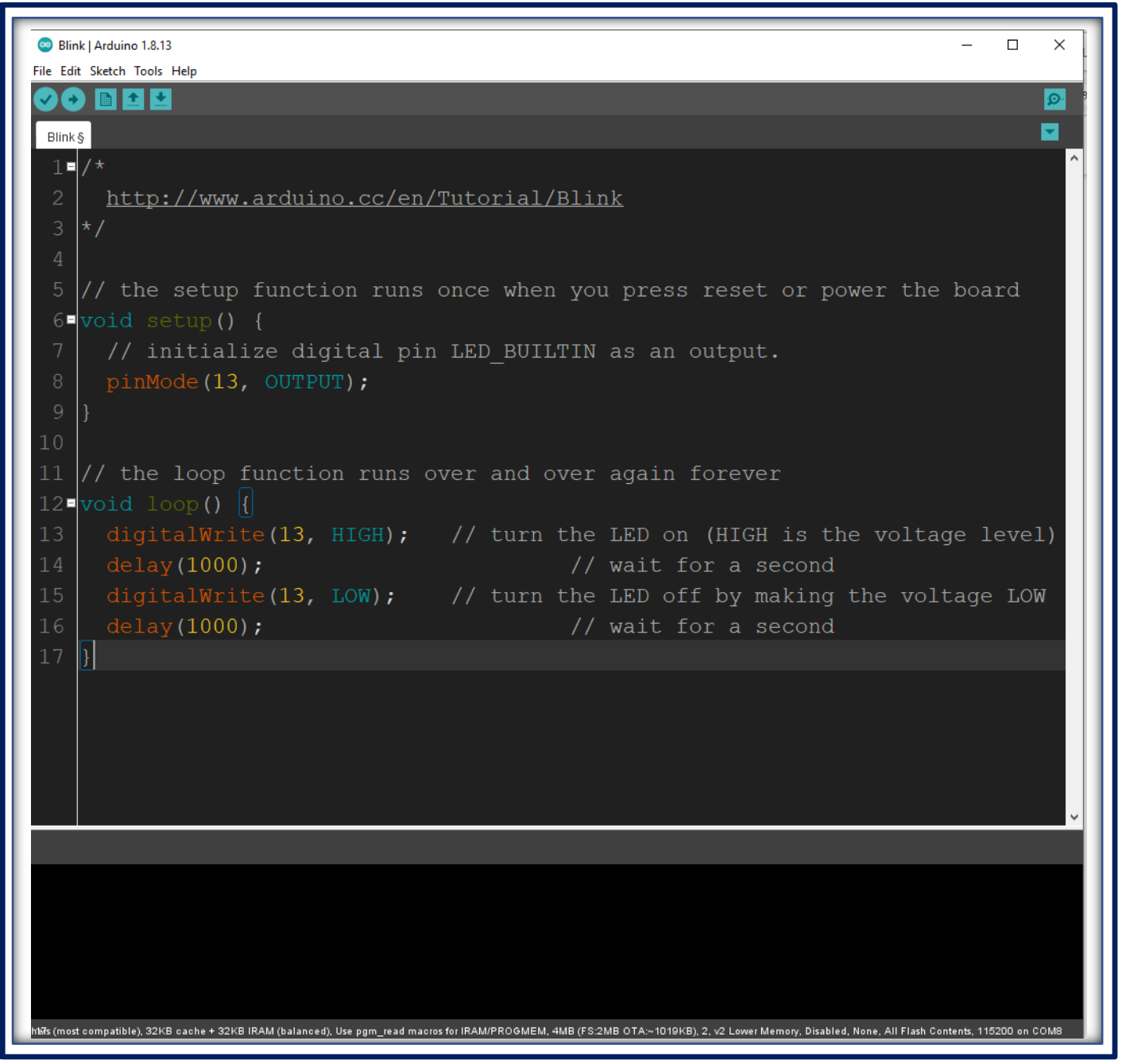
# structure /struktur

**Semicolon ;**

Digunakan untuk mengahiri statement atau fungsi, kecuali pada kurung kurawal

**Single line comment //**

Komentar adalah baris dalam program yang digunakan untuk memberi tahu diri sendiri atau orang lain tentang cara kerja program. Mereka diabaikan oleh compiller, dan tidak diekspor ke prosesor, sehingga tidak memakan ruang di memori flash mikrokontroler.



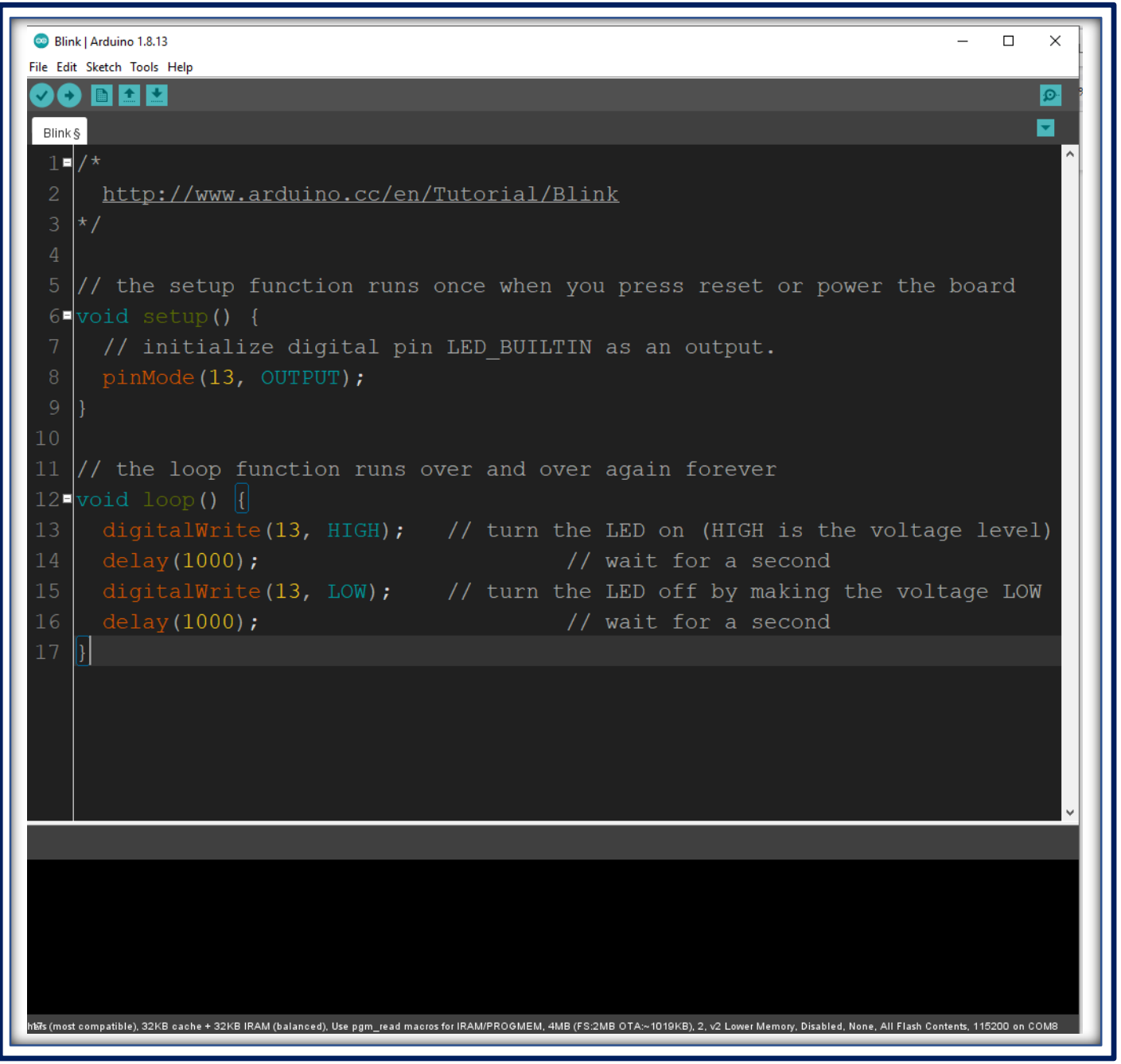
# structure /struktur

**Block comment /* */**

Komentar beberapa baris

Curly braces { }

**Curly braces** atau **kurung kurawal awal** { digunakan untuk memulai fungsi suatu blok code. Dan **kurung kurawal akhir** } digunakan untuk mengakhiri fungsi suatu blok code program



#define adalah komponen C++ yang berguna yang memungkinkan pemrogram untuk memberi nama pada nilai konstan (tetap) sebelum program dikompilasi. Konstanta (nilai tetap) yang ditentukan dalam program Arduino ini tidak menggunakan ruang memori program apa pun pada chip.

# #define

## Syntax

```
#define constantName value
```

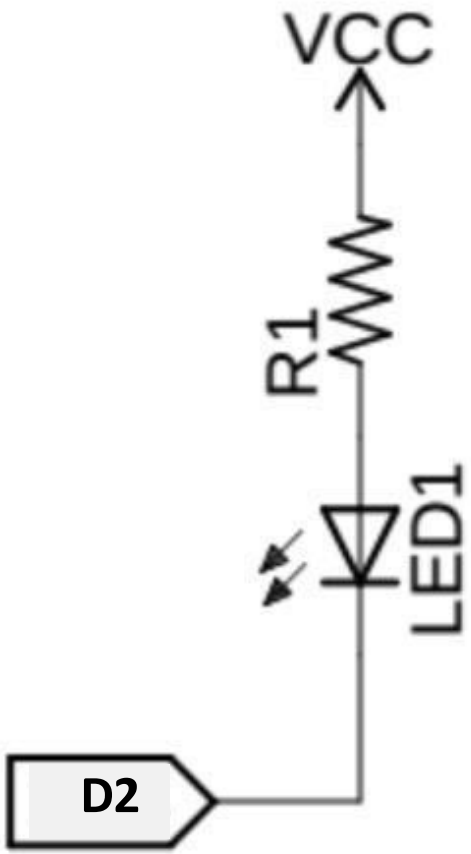
constantName: nama makro yang akan di tentukan.
value: nilai yang akan ditetapkan ke makro.

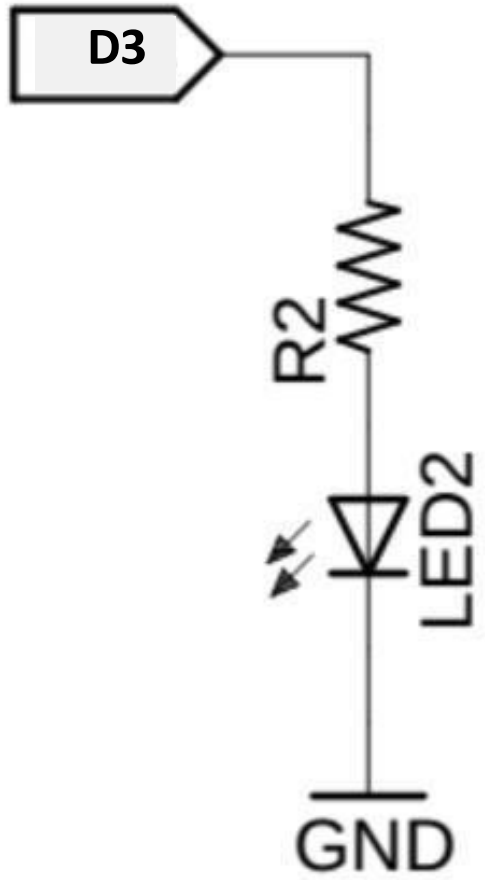
#include digunakan untuk menyertakan library luar dalam sketsa Anda. Ini memberikan akses programmer ke sekelompok besar library C standar (kelompok fungsi yang dibuat sebelumnya), dan juga library yang ditulis khusus untuk Arduino.

# #include

## Syntax

```
#include <LibraryFile.h>
#include "LocalFile.h"
```



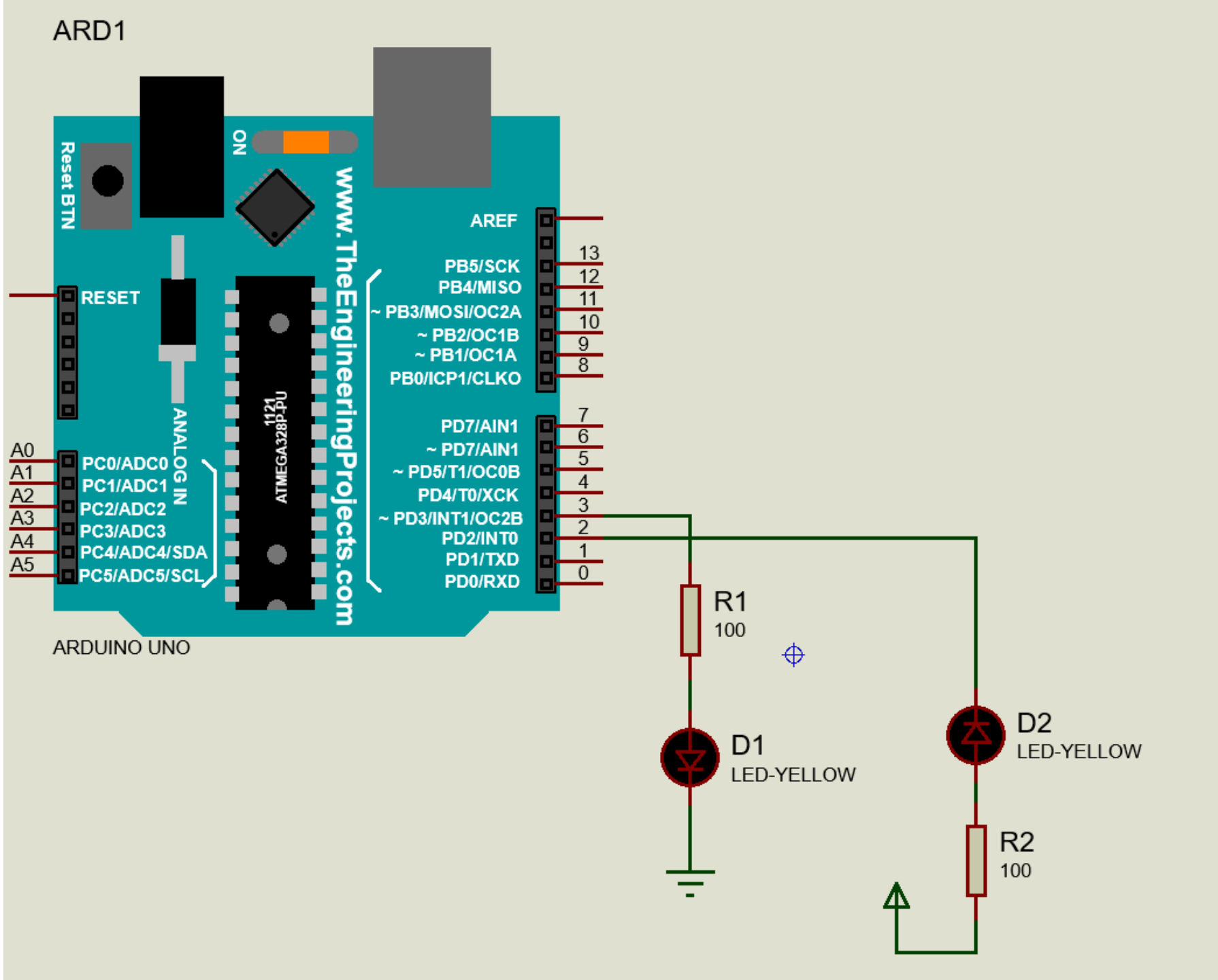

Jika D2 HIGH maka LED OFF

Jika D2 LOW maka LED ON

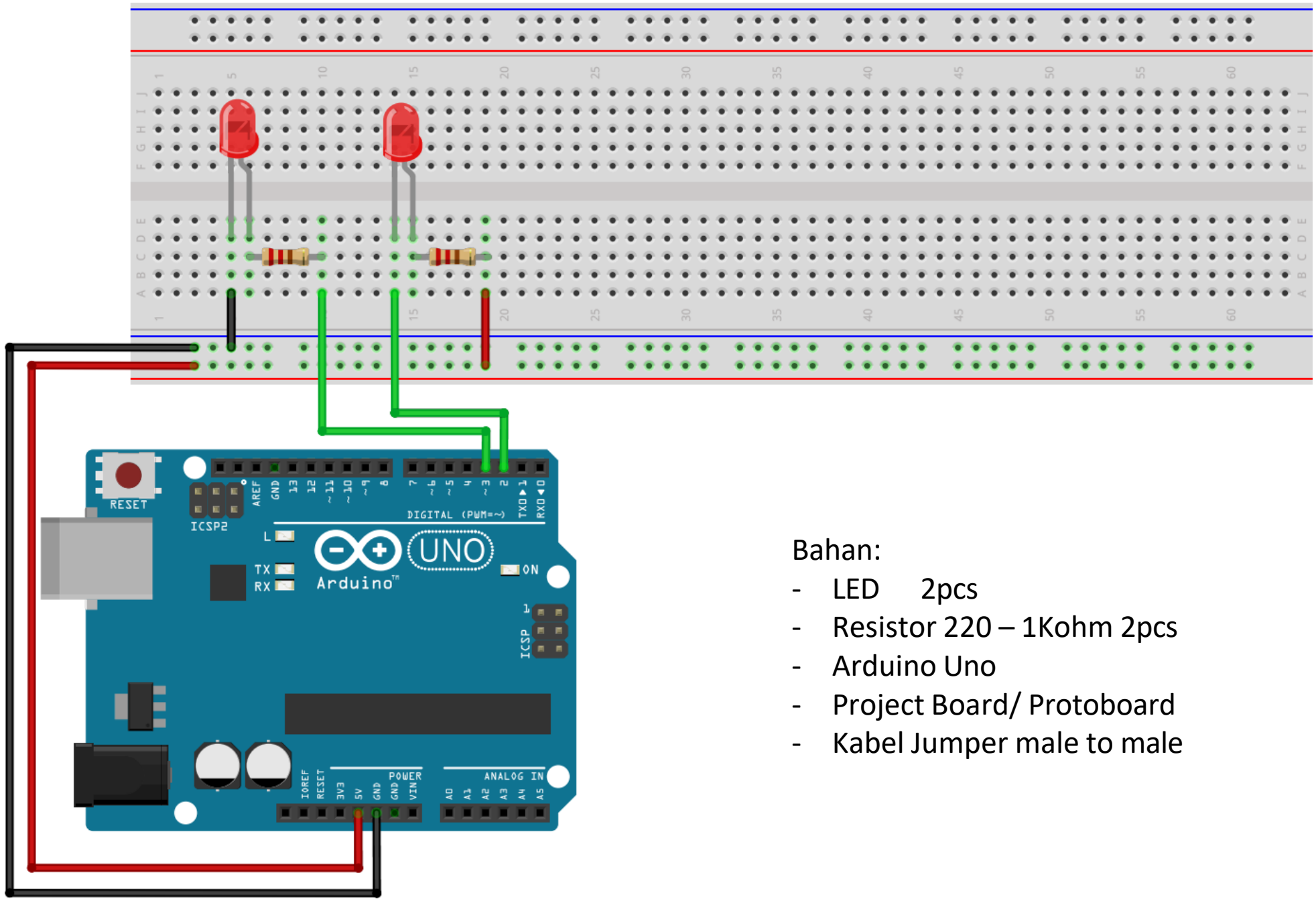
Jika D3 HIGH maka LED ON

Jika D3 LOW maka LED OFF

LATIHAN 1


ARD1
Reset BTN
ON
www.TheEngineeringProjects.com
AREF
RESET
13
12
11
10
9
8
PB5/SCK
PB4/MISO
~ PB3/MOSI/OC2A
~ PB2/OC1B
~ PB1/OC1A
PB0/ICP1/CLKO
ANALOG IN
ATMEGA328P-PU
7
6
5
4
3
2
1
0
PD7/AIN1
~ PD7/AIN1
~ PD5/T1/OC0B
PD4/T0/XCK
~ PD3/INT1/OC2B
PD2/INT0
PD1/TXD
PD0/RXD
A0
A1
A2
A3
A4
A5
PC0/ADC0
PC1/ADC1
PC2/ADC2
PC3/ADC3
PC4/ADC4/SDA
PC5/ADC5/SCL
ARDUINO UNO
R1
100
D1
LED-YELLOW
D2
LED-YELLOW
R2
100

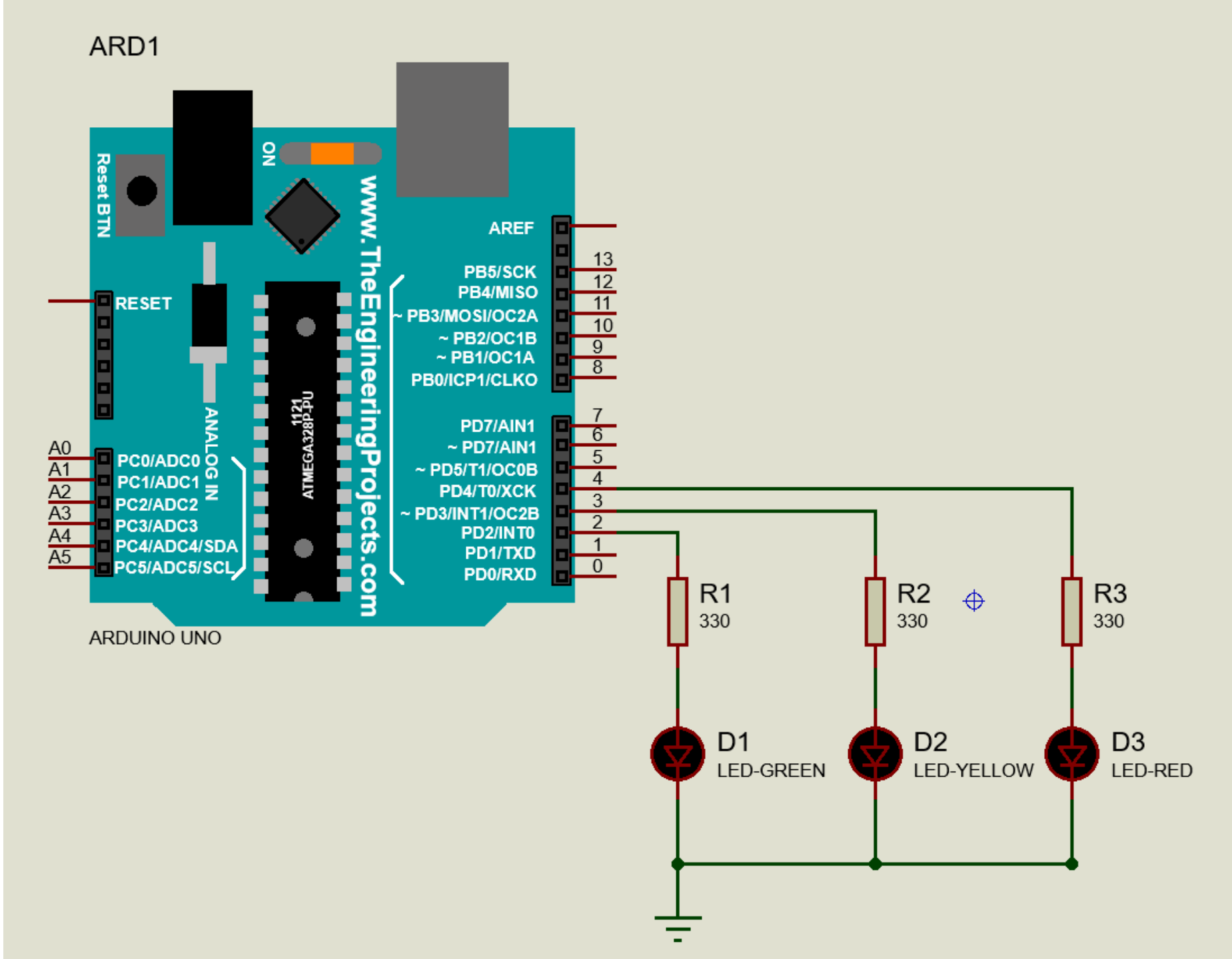
PROTOBOARD
LATIHAN 1




Bahan:

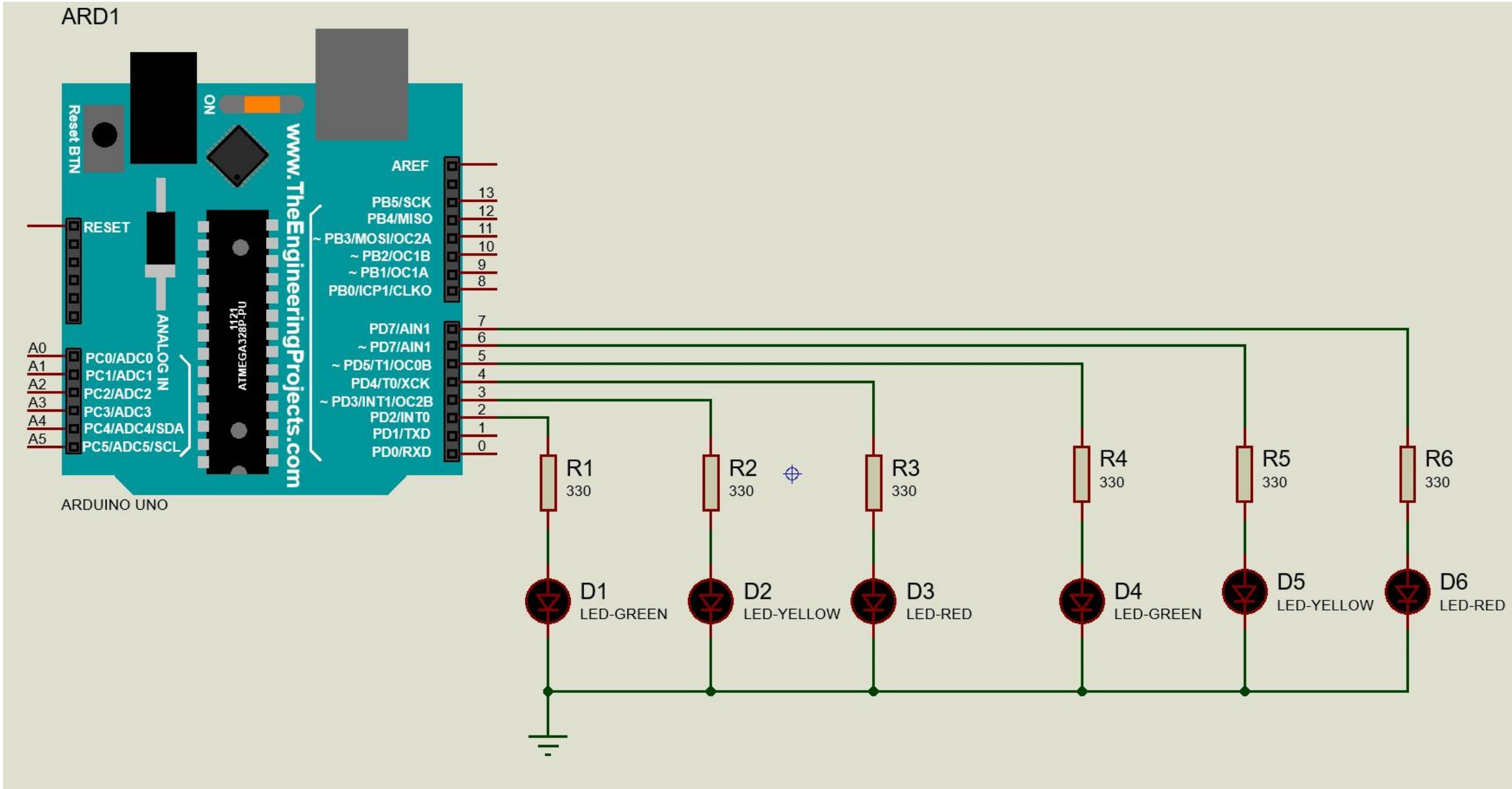
- LED 2pcs
- Resistor 220 – 1Kohm 2pcs
- Arduino Uno
- Project Board/ Protoboard
- Kabel Jumper male to male

LATIHAN 2


ARD1
Reset BTN
ON
www.TheEngineeringProjects.com
AREF
RESET
PB5/SCK
PB4/MISO
~ PB3/MOSI/OC2A
~ PB2/OC1B
~ PB1/OC1A
PB0/ICP1/CLKO
13
12
11
10
9
8
ANALOG IN
ATMEGA328P-PU
1121
PD7/AIN1
~ PD7/AIN1
~ PD5/T1/OC0B
PD4/T0/XCK
~ PD3/INT1/OC2B
PD2/INT0
PD1/TXD
PD0/RXD
7
6
5
4
3
2
1
0
A0
A1
A2
A3
A4
A5
PC0/ADC0
PC1/ADC1
PC2/ADC2
PC3/ADC3
PC4/ADC4/SDA
PC5/ADC5/SCL
ARDUINO UNO
R1
330
R2
330
R3
330
D1
LED-GREEN
D2
LED-YELLOW
D3
LED-RED

Buatlah rangkaian LATIHAN 2 pada protoboard

LATIHAN 3

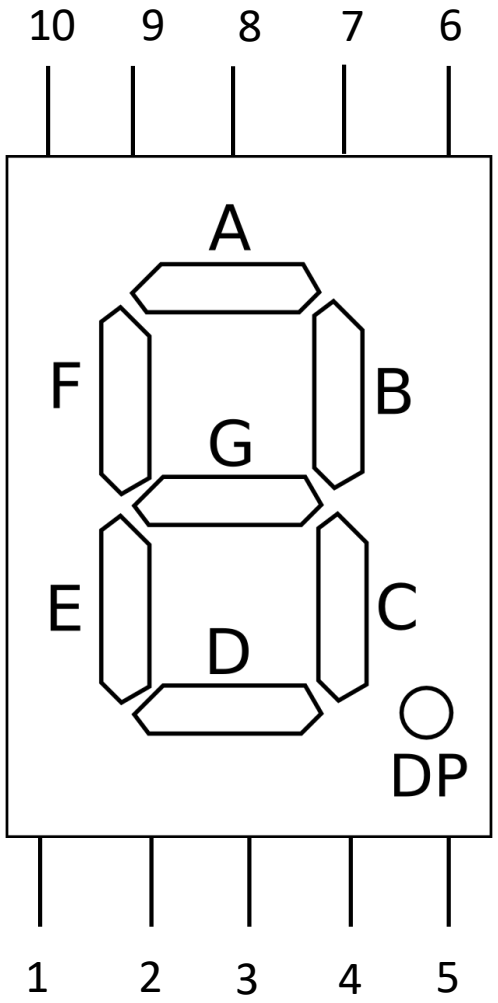

ARD1
Reset BTN
ON
www.TheEngineeringProjects.com
AREF
RESET
PB5/SCK
PB4/MISO
~ PB3/MOSI/OC2A
~ PB2/OC1B
~ PB1/OC1A
PB0/ICP1/CLKO
13
12
11
10
9
8
ANALOG IN
ATMEGA328P-PU
PD7/AIN1
~ PD7/AIN1
~ PD5/T1/OC0B
PD4/T0/XCK
~ PD3/INT1/OC2B
PD2/INT0
PD1/TXD
PD0/RXD
7
6
5
4
3
2
1
0
A0
A1
A2
A3
A4
A5
PC0/ADC0
PC1/ADC1
PC2/ADC2
PC3/ADC3
PC4/ADC4/SDA
PC5/ADC5/SCL
ARDUINO UNO
R1
330
R2
330
R3
330
R4
330
R5
330
R6
330
D1
LED-GREEN
D2
LED-YELLOW
D3
LED-RED
D4
LED-GREEN
D5
LED-YELLOW
D6
LED-RED

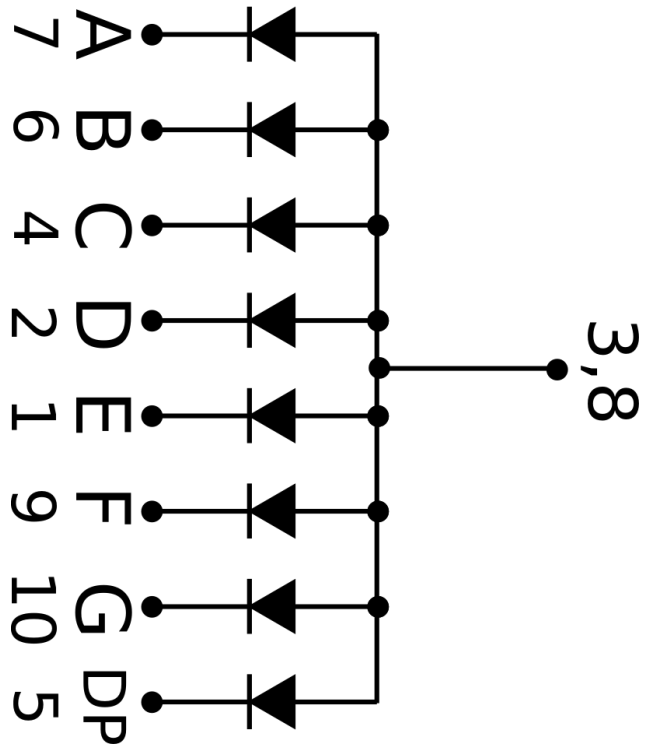
Buatlah rangkaian LATIHAN 3 pada protoboard

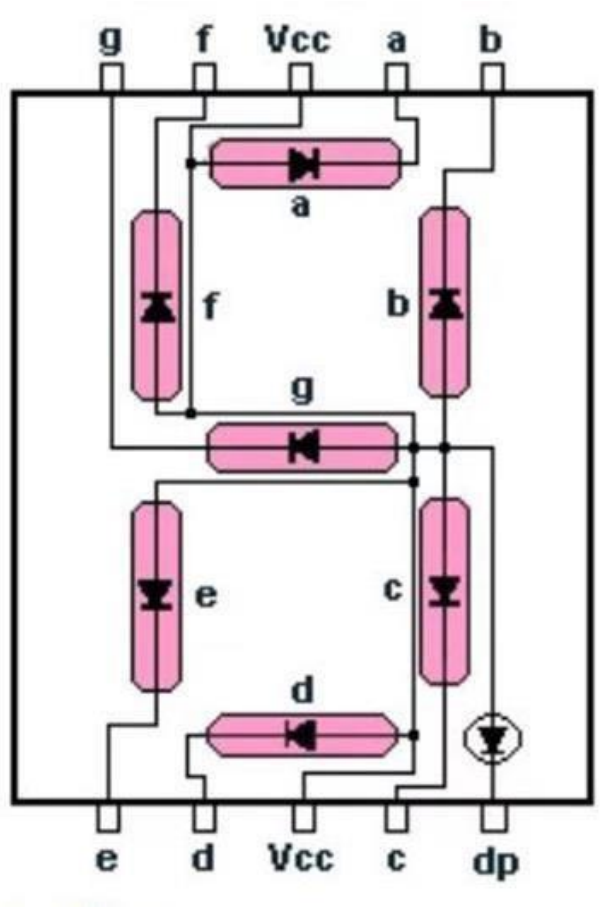
| LINE 1 | | | LINE 2 | | | |
|---|---|---|---|---|---|---|
| LED D1/ **pin arduino 2**/ LED hijau | LED D2/ **pin arduino** 3/ LED kuning | LED D3/ **pin arduino** 4/ LED merah | LED D4/ **pin arduino** 5/ LED hijau | LED D5/ **pin arduino** 6/ LED kuning | LECD6/ **pin arduino** 7/ LED merah | delay |
| ON | OFF | OFF | OFF | OFF | ON | 6000mS |
| OFF | ON | OFF | OFF | OFF | ON | 1500mS |
| OFF | OFF | ON | OFF | ON | OFF | 700mS |
| OFF | OFF | ON | ON | OFF | OFF | 6000mS |
| OFF | OFF | ON | OFF | ON | OFF | 1500mS |
| OFF | ON | OFF | OFF | OFF | ON | 700mS |
| Kembali ke awal/ mengulang | | | | | | |

Buatlah program traffic light sesuai dengan table diatas

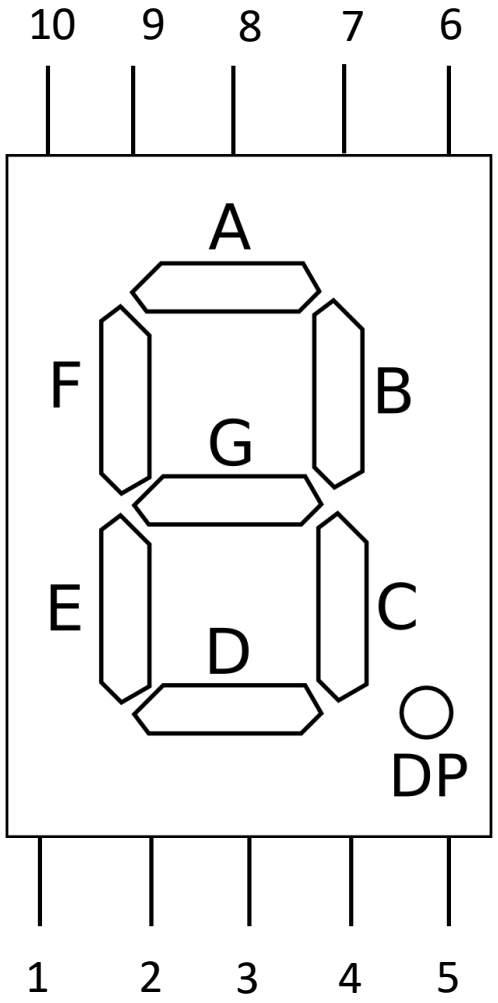
# COMMON ANODA (CA)

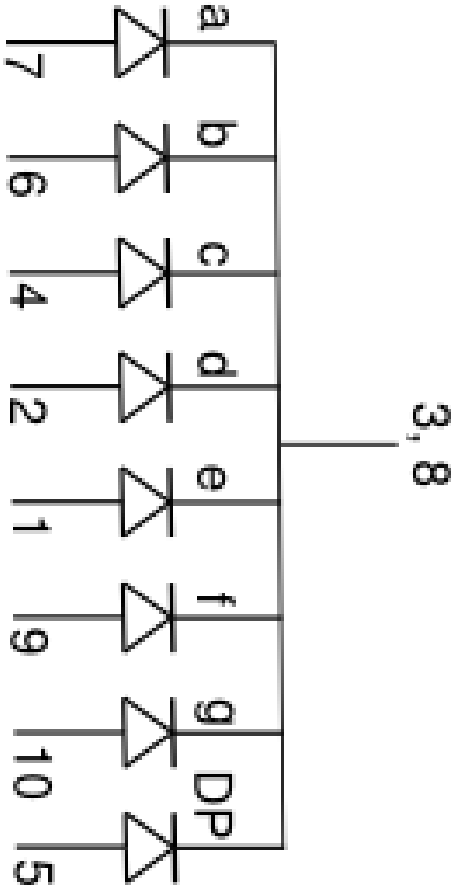


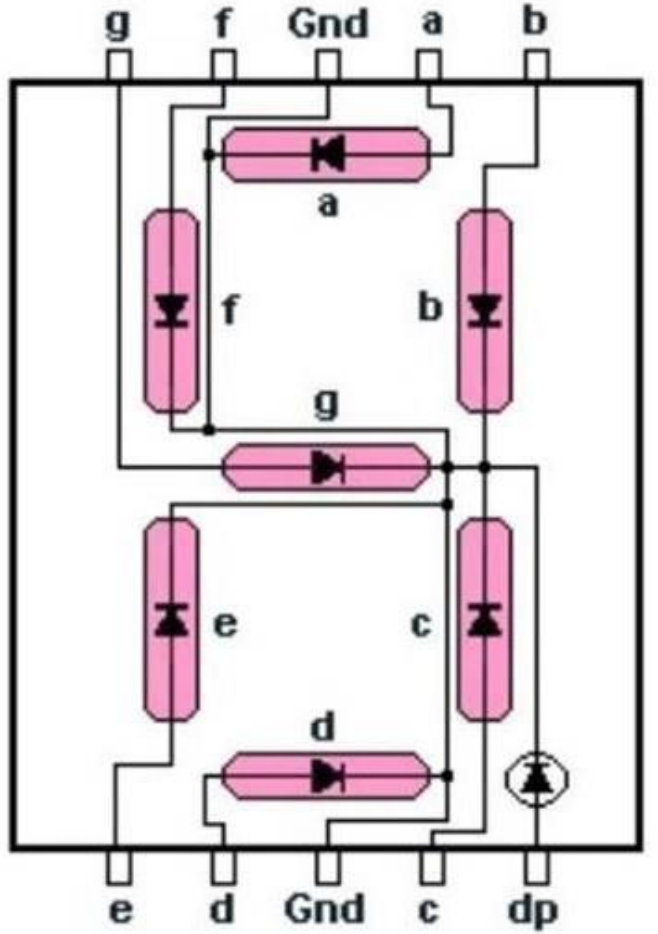



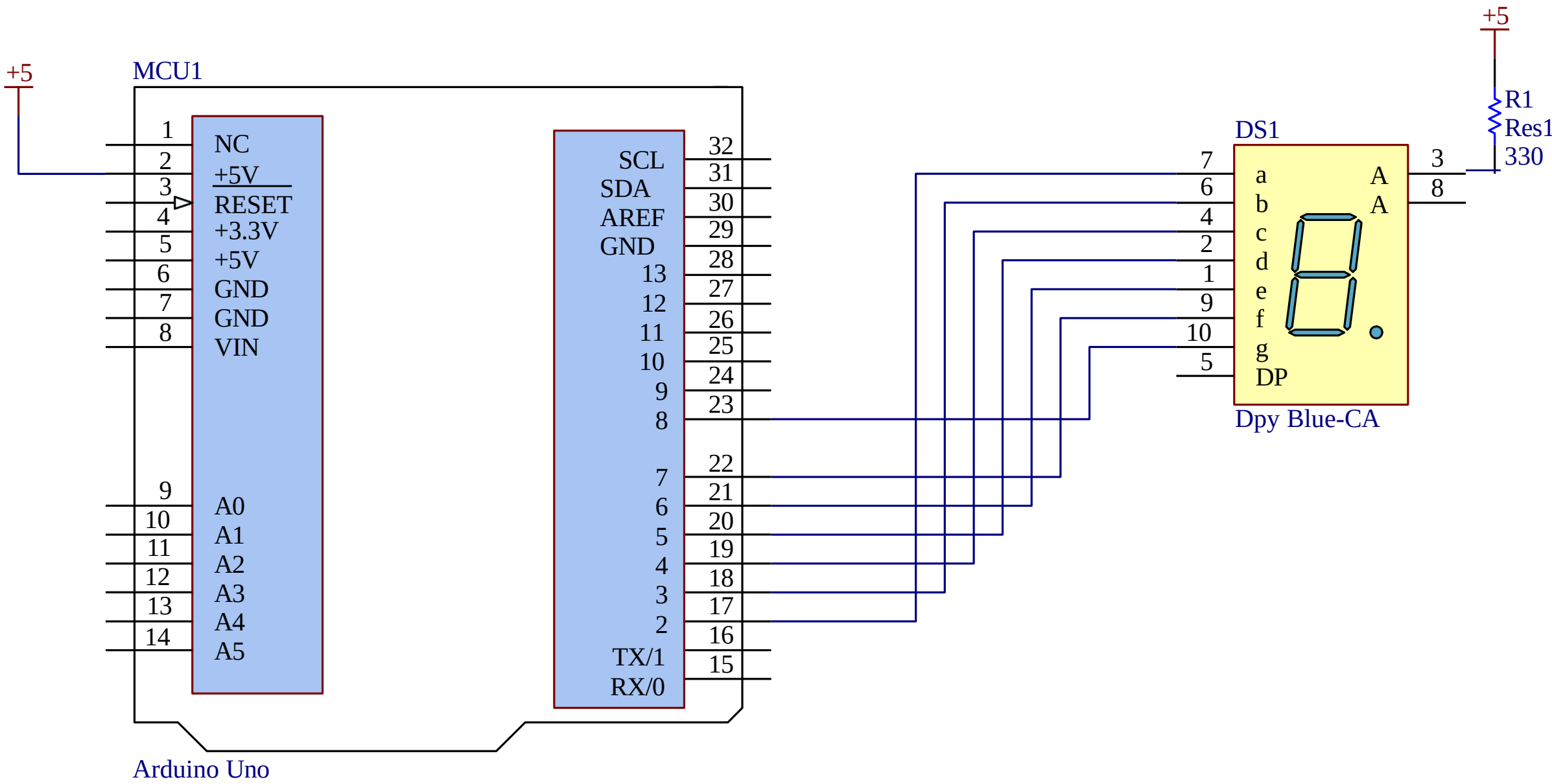
# COMMON CATHODE (CC)



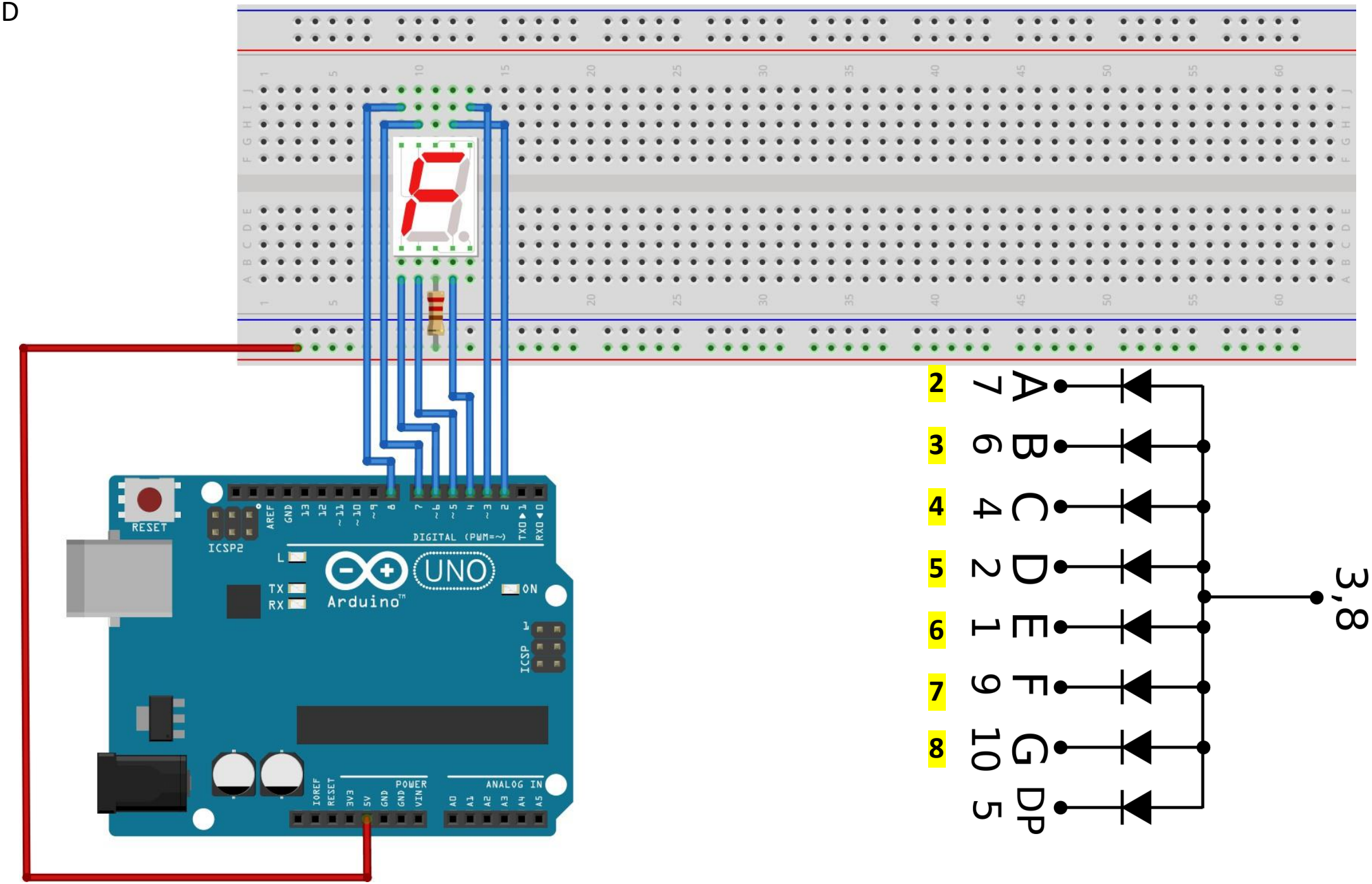



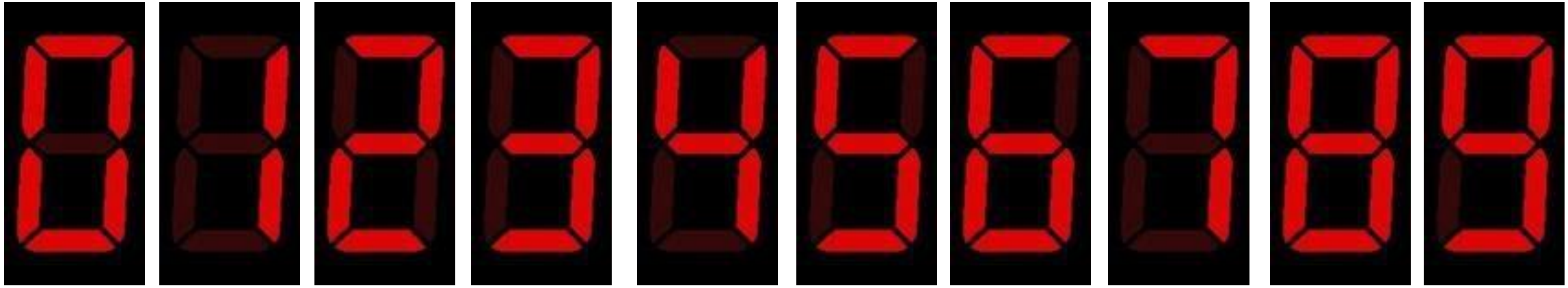
LATIHAN 4


+5
MCU1
1
2
3
4
5
6
7
8
NC
+5V
RESET
+3.3V
+5V
GND
GND
VIN
9
10
11
12
13
14
A0
A1
A2
A3
A4
A5
SCL
SDA
AREF
GND
13
12
11
10
9
8
7
6
5
4
3
2
TX/1
RX/0
32
31
30
29
28
27
26
25
24
23
22
21
20
19
18
17
16
15
Arduino Uno
DS1
7
6
4
2
1
9
10
5
a
b
c
d
e
f
g
DP
A
A
3
8
Dpy Blue-CA
+5
R1
Res1
330

PROTOBOARD
LATIHAN 4

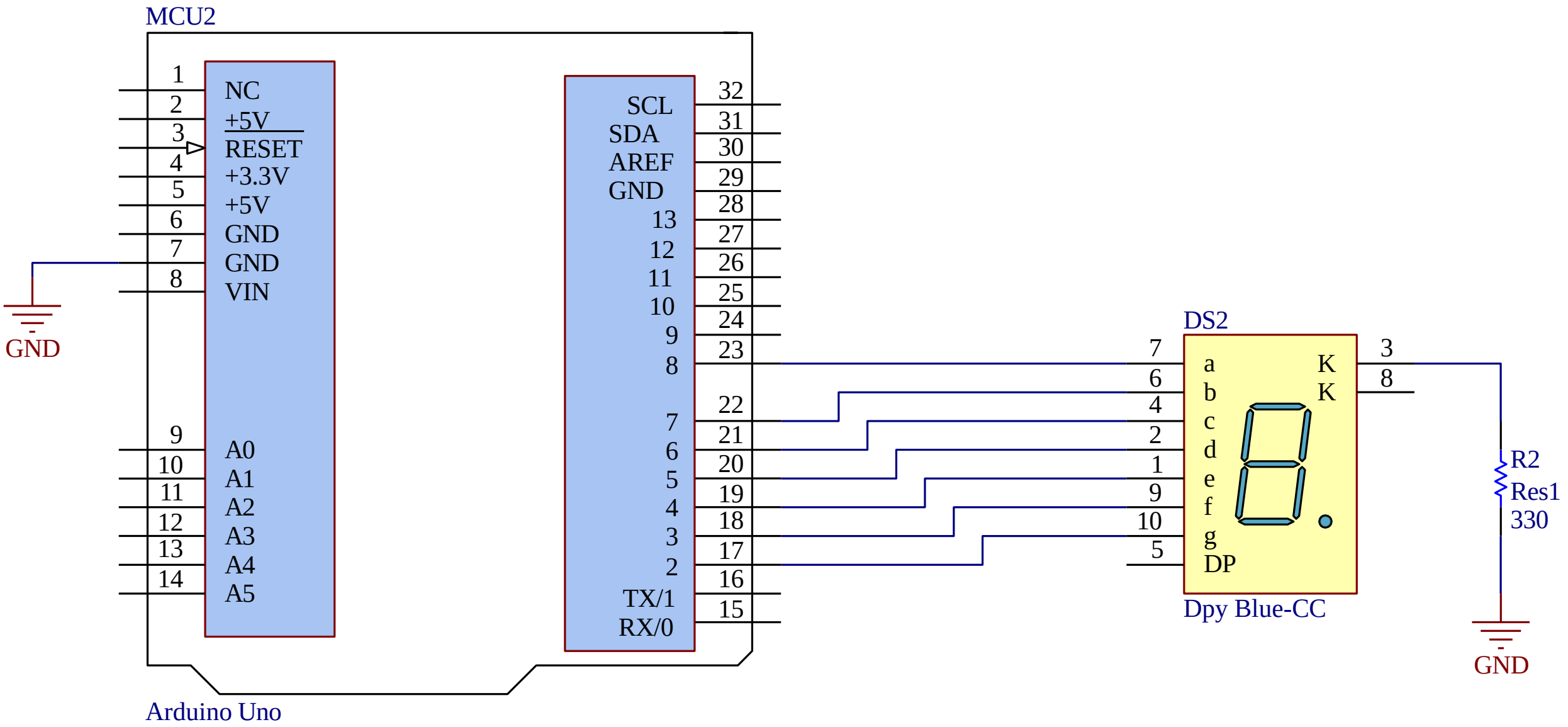


Buatlah Program untuk memunculkan nilai 0



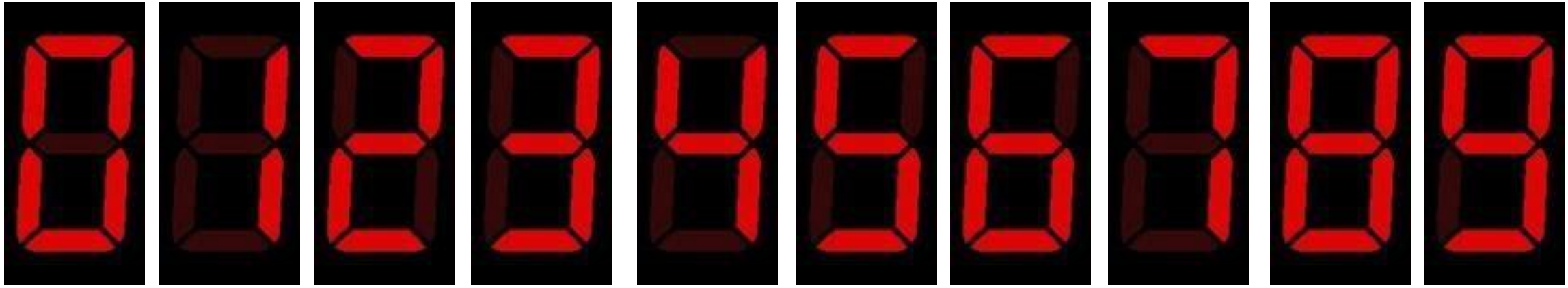
Buatlah Program memunculkan nilai 0 s.d 9 dengan interval 1000ms

LATIHAN 5


MCU2
1
2
3
4
5
6
7
8
NC
+5V
RESET
+3.3V
+5V
GND
GND
VIN
9
10
11
12
13
14
A0
A1
A2
A3
A4
A5
SCL
SDA
AREF
GND
13
12
11
10
9
8
7
6
5
4
3
2
TX/1
RX/0
32
31
30
29
28
27
26
25
24
23
22
21
20
19
18
17
16
15
GND
DS2
7
6
4
2
1
9
10
5
a
b
c
d
e
f
g
DP
K
K
3
8
Dpy Blue-CC
R2
Res1
330
GND
Arduino Uno

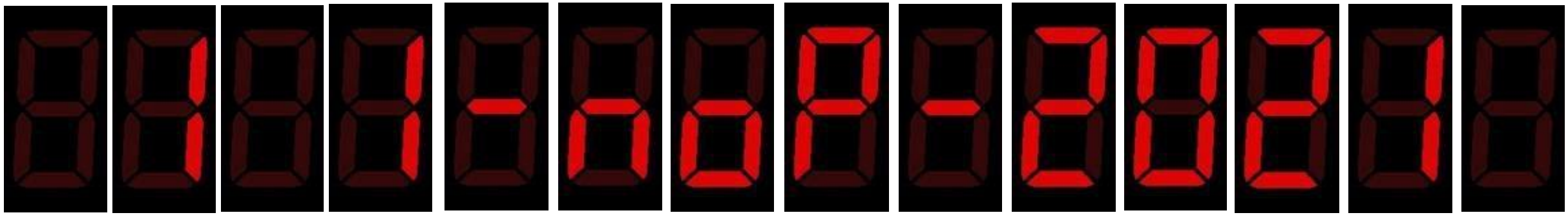
Buatlah rangkaian LATIHAN 5 pada protoboard

Buatlah Program untuk memunculkan nilai 0



Buatlah Program memunculkan nilai 0 s.d 9 dengan interval 1000ms

TUGAS



500mS 500mS 200mS 500mS 500mS 500mS 500mS 500mS 500mS 500mS 500mS 500mS 500mS 500mS 500mS

# PENGAMBILAN KEPUTUSAN

Bahasa C menyediakan beberapa jenis pernyataan pengambilan keputusan (decision marking) yaitu sebagai berikut:

- Pengambilan keputusan tipe **if**
- Pengambilan keputusan tipe **if-else**
- Pengambilan keputusan tipe **switch**

Sebuah keputusan dapat dilakukan jika persoalan tersebut memenuhi kriteria/ kondisi tertentu yaitu:
TRUE (BENAR) or FALSE (SALAH)

# PENGAMBILAN KEPUTUSAN

Dalam menentukan kondisi digunakan **operator relasi** dan **operator logika**.

**Operator relasi** digunakan untuk membandingkan dua buah nilai dengan hasil **TRUE** atau **FALSE**.

Sedangankan **operator logika** dipakai untuk menghubungkan ekspresi relasi.

Berikut tabel operator relasi:

| Operator | Keterangan |
|---|---|
| > | Lebih dari |
| >= | Lebih dari atau sama dengan |
| < | Kurang dari |
| <= | Kurang dari atau sama dengan |
| == | Sama dengan |
| != | Tidak sama dengan |

Contoh berikut table Operator Relasi

| Kondisi | BACA & Hasil |
|---|---|
| 1>2 | Dibaca : 1 lebih dari 2<br>Hasil **FALSE/ SALAH** |
| 1<2 | Dibaca 1 kurang dari 2<br>Hasil **TRUE/ BENAR** |
| Kondisi_Tombol==1 | Jika **Kondisi_Tombol = 1** maka hasilnya **TRUE,** jika tidak maka **FALSE** |

Berikut tabel operator logika:

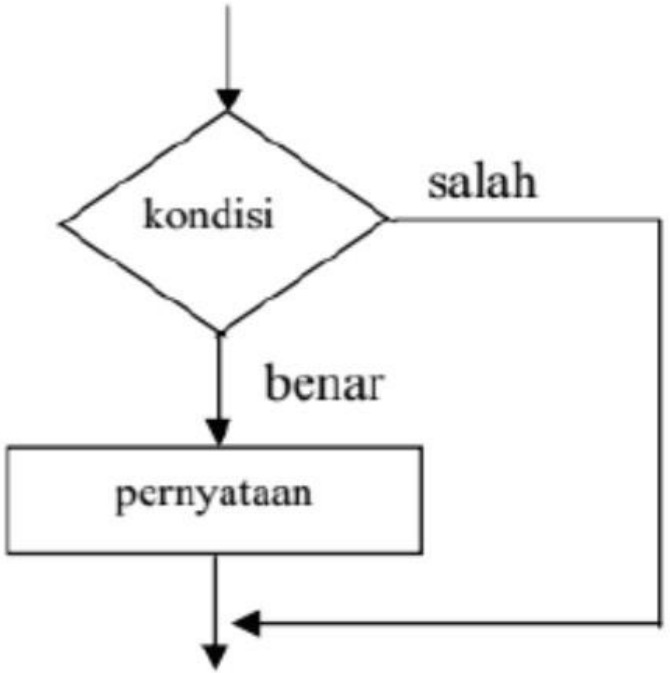
| Operator | Keterangan |
|---|---|
| && | dan (AND) |
| \|\| | atau (OR) |
| ! | tidak (NOT) |

If (digitalRead(1)==HIGH && digitalRead(2)==HIGH)

## Pernyataan if

Pernyataan if memiliki bentuk umum sebagai berikut:

```
If (kondisi) {
Pernyataan;
}
```



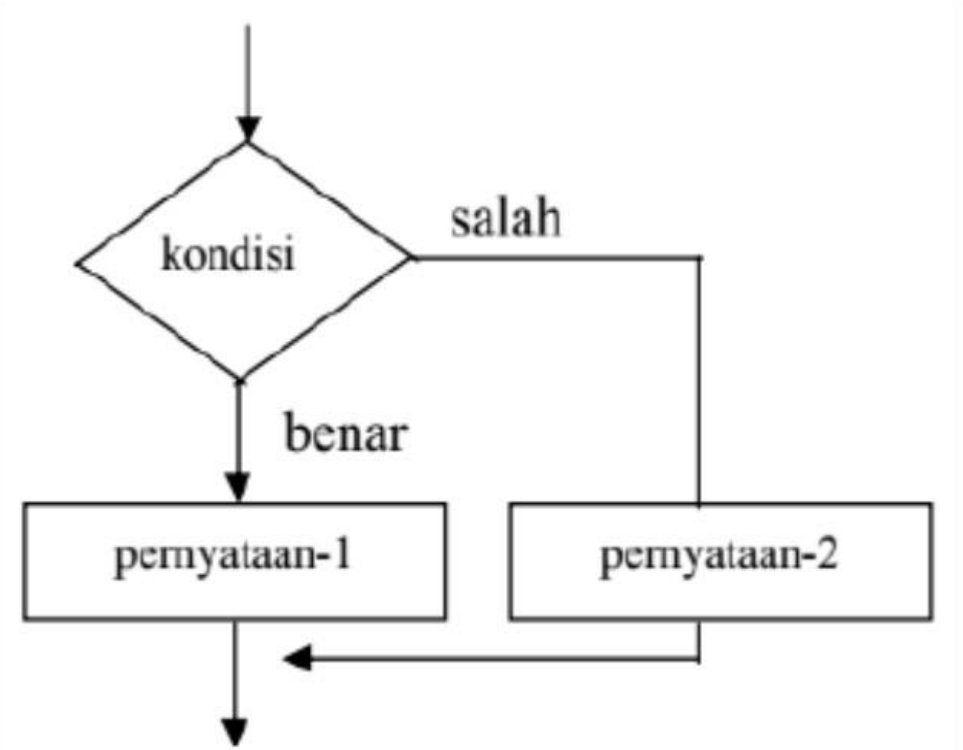
Jika didalam sebuah kondisi terdapat banyak pernyataan maka program ditulis seperti ini:

```
if (kondisi)
{                       //tanda awal pernyataan majemuk
    pernyataan_1;
    pernyataan_2;
    -
    -
    pernyataan_n;
}                       //tanda akhir pernyataan majemuk
```

## Pernyataan if else

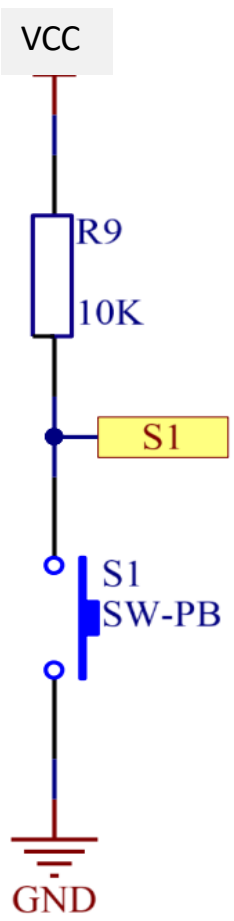
Pernyataan if else memiliki bentuk umum sebagai berikut:

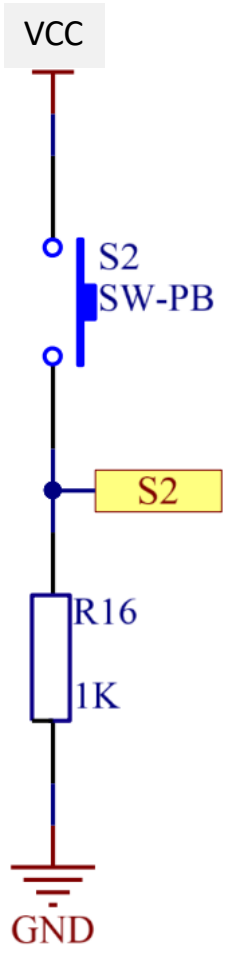
```
If (kondisi) {
Pernyataan;
}
else {
Pernyataan;
}
```

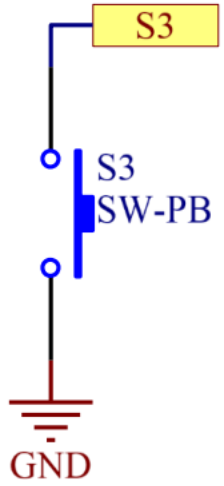


Bentuk pernyataan majemuk dari pengambilan keputusan ini sebagi berikut:

```
if (kondisi)
{                       //tanda awal blok if
    pernyataan_1;
    pernyataan_2;
    -
    -
    pernyataan_n;
}                       //tanda akhir blok if
else
{                       //tanda awal blok else
    pernyataan_1;
    pernyataan_2;
    -
    -
    pernyataan_n;
}                       //tanda akhir blok else
```




Jika tombol ditekan maka keluaran output S1 = LOW

Jika tombol tidak ditekan maka keluaran output S1 = HIGH

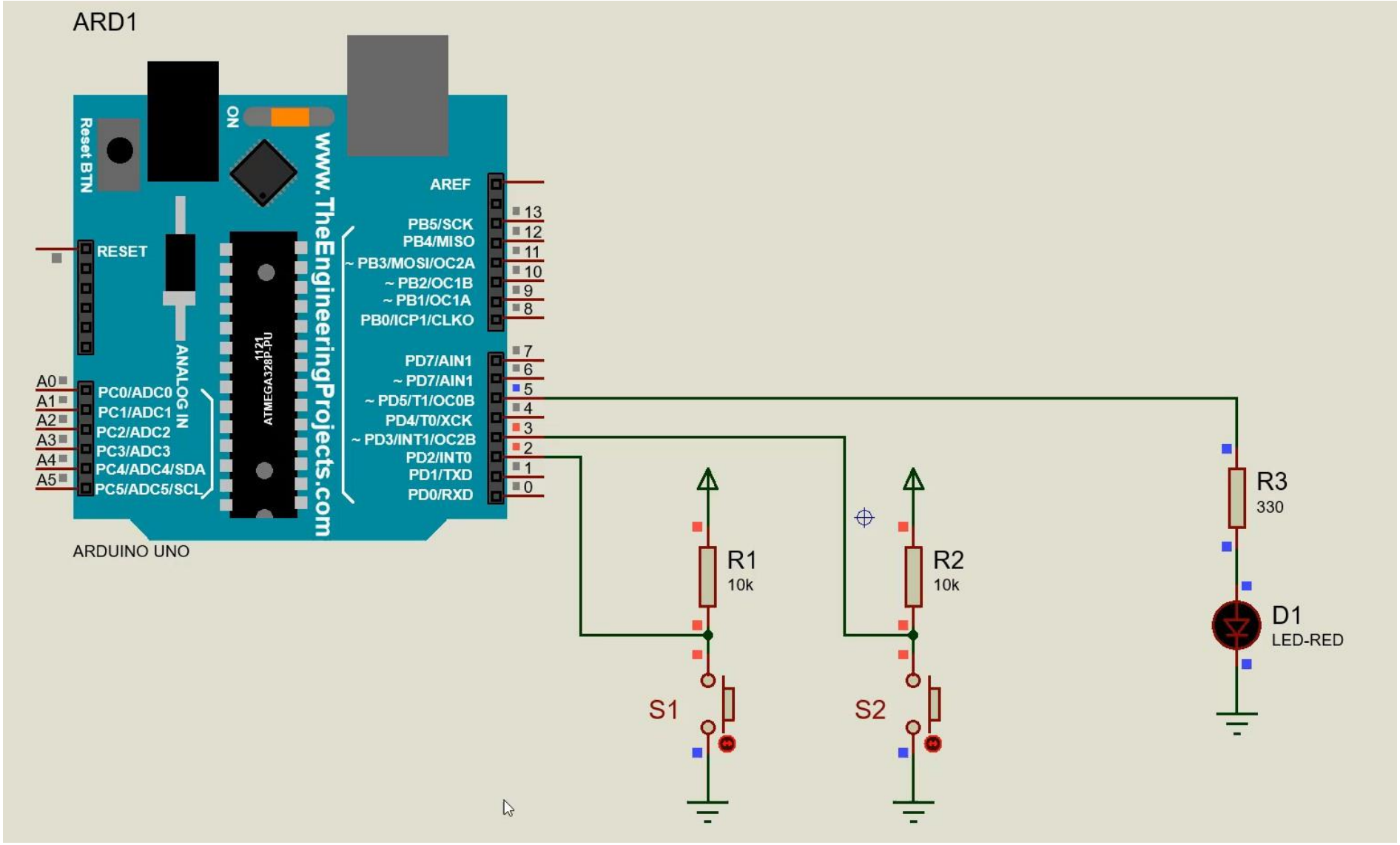
Jika tombol ditekan maka keluaran output S2 = HIGH

Jika tombol tidak ditekan maka keluaran output S2 = LOW

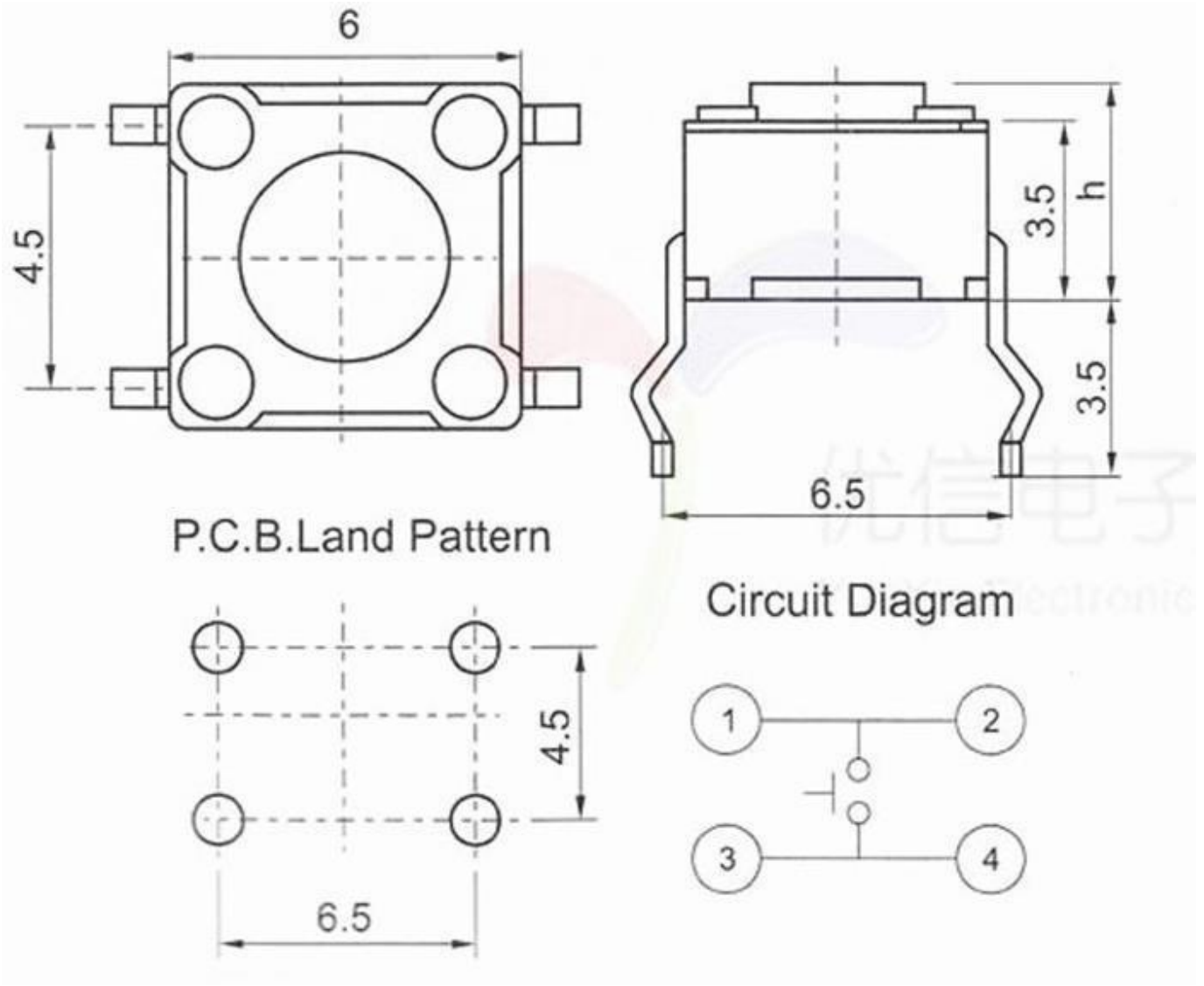
**PULL UP Internal**

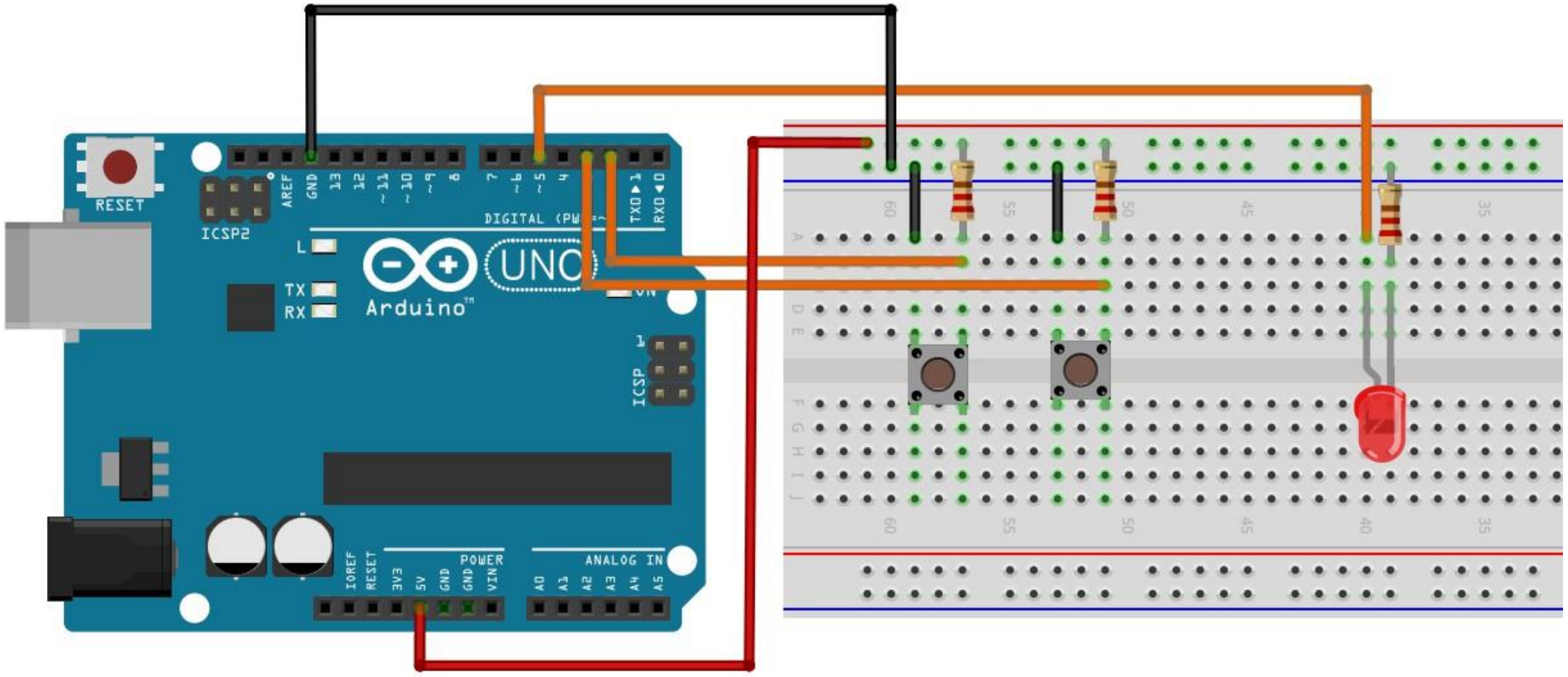
Jika tombol ditekan maka keluaran output S3 = LOW

Jika tombol tidak ditekan maka keluaran output S3 = HIGH


ARD1
Reset BTN
ON
www.TheEngineeringProjects.com
RESET
ANALOG IN
ATMEGA328P-PU
1121
AREF
PB5/SCK
PB4/MISO
~ PB3/MOSI/OC2A
~ PB2/OC1B
~ PB1/OC1A
PB0/ICP1/CLKO
PD7/AIN1
~ PD7/AIN1
~ PD5/T1/OC0B
PD4/T0/XCK
~ PD3/INT1/OC2B
PD2/INT0
PD1/TXD
PD0/RXD
13
12
11
10
9
8
7
6
5
4
3
2
1
0
A0
A1
A2
A3
A4
A5
PC0/ADC0
PC1/ADC1
PC2/ADC2
PC3/ADC3
PC4/ADC4/SDA
PC5/ADC5/SCL
ARDUINO UNO
R1
10k
R2
10k
R3
330
D1
LED-RED
S1
S2

# push button 4 pin




RESET
ICSP2
AREF
GND
13
12
~11
~10
~9
8
7
~6
~5
4
DIGITAL (PW
TX→1
RX←0
L
TX
RX
Arduino
UNO
ICSP
POWER
IOREF
RESET
3V3
5V
GND
GND
VIN
ANALOG IN
A0
A1
A2
A3
A4
A5