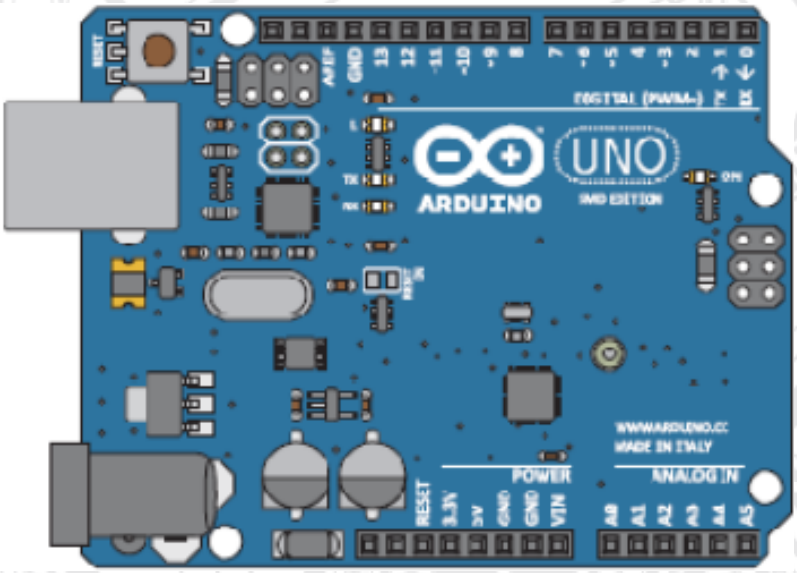
BBROBOTINDONESIA



PANDUAN PRAKTIKUM

DASAR ARDUINO

Praktikum 13 Belajar Arduino
Deteksi Perubahan Kondisi dan Operator Modulo

Pada praktikum sebelumnya, kita telah belajar menggunakan push Button sebagai input Arduino. Ketika kita menekan Button dan menahannya, maka Arduino akan menyalakan sebuah LED. Setelah kita lepaskan Button, maka Arduino akan memadamkan LED tersebut. Sekarang pertanyaannya adalah “bagaimana jika kita hanya menginginkan push Button tetap dapat menyalakan LED tanpa harus tetap menekan push Button?”

Button yang kita gunakan pada praktikum sebelumnya maupun pada praktikum ini adalah sebuah momentary push button. Button ini, hanya akan mengalirkan arus listrik jika kita tetap menekannya. Jika kita lepaskan, maka arus listrik tidak akan mengalir melalui push button ini. Untuk itulah, pada praktikum kali ini, kita akan mencari tahu bagaimana cara agar kita dapat menggunakan sebuah momentary push button berfungsi layaknya sebuah saklar on/off biasa.

Sebenarnya, untuk membuat aplikasi seperti membuat LED menyala atau padam, kita bisa saja secara fisik mengganti momentary push button dengan saklar biasa. Tetapi, pada praktikum kali ini sengaja tidak kita lakukan. Tujuannya adalah agar kita dapat mempelajari metode pemrograman dan juga dapat mempelajari salah satu operator yang disebut dengan operator modulo.

Baiklah, untuk lebih jelasnya kita langsung saja ke praktikumnya berikut ini.

Komponen yang Dibutuhkan

Arduino Uno 1x

Protoboard 1x

LED5mm 1x

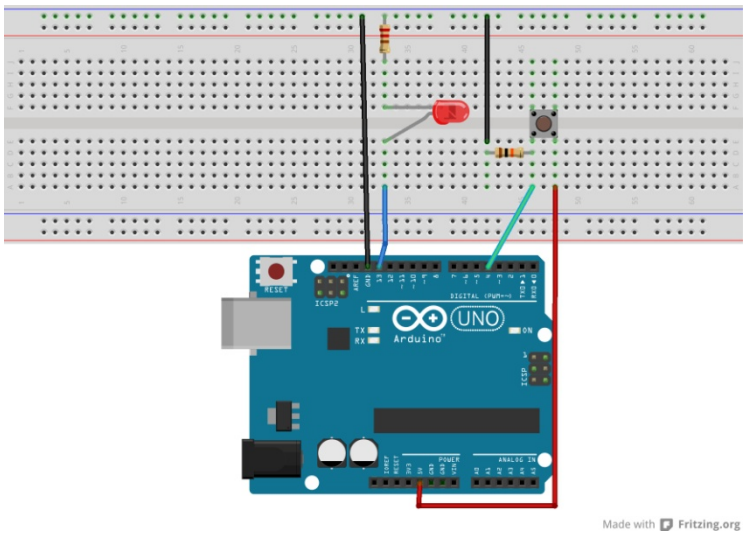
Kabel Jumper Male 5x

Resistor 220 ohm 1x

Resistor 10kOhm 1x

Pushbutton 1x

Langkah-Langkah Praktikum



Susunlah komponen-komponen praktikum seperti gambar di atas, caranya :

1. hubungkanlah kaki + Anoda LED ke Pin 13 Arduino menggunakan kabel jumper Male (warna biru, saya gunakan untuk membedakan saja. Bisa menggunakan warna apa saja)

2. hubungkan kaki - Katoda LED dengan kaki 1 Resistor (resistor tidak ada kaki + atau - nya).

3. hubungkan kaki 2 Resistor dengan Pin Gnd Arduino menggunakan kabel jumper Male (warna hitam, biasa digunakan untuk Gnd atau 0 Volt. Sedangkan Merah, biasa digunakan untuk Voltase +).

4. Hubungkan Kaki 1 Push Button dengan pin 5v Arduino menggunakan kabel jumper (gunakan warna merah)

5. Hubungkan Kaki 2 Push Button dengan salah satu kaki Resistor 10K Ohm dan pin digital 4 Arduino. Untuk ke pin 4 arduino gunakan kabel jumper (gunakan warna hijau)

6. Hubungkan kaki Resistor 10k ohm yang belum terhubung ke jalur GND protoboard menggunakan kabel jumper(gunakan warna hitam)

7. hubungkan board Arduino Uno dengan Komputer menggunakan kabel USB.

8. Bukalah IDE Arduino, kemudian ketikkan kode program/sketch berikut:

```
const int pinButton = 4;
const int pinLED = 13;

int penghitung = 0;
int kondisiSekarang = 0;
int kondisiTerakhir = 0;

void setup()
{
  pinMode(pinButton,INPUT);
```

```
  pinMode(pinLED,OUTPUT);
  Serial.begin(9600);
}

void loop()
{
  kondisiSekarang = digitalRead(pinButton);

  if ( kondisiSekarang != kondisiTerakhir)
  {
    if (kondisiSekarang == HIGH)
    {
      penghitung ++;
      Serial.println ("on");
      Serial.print("Banyaknya Button di Tekan : ");
      Serial.println(penghitung);
    }
    else
    {
    Serial.println("off");
    }
  }

  kondisiTerakhir = kondisiSekarang;

  if (penghitung % 4 == 0)
  {
    digitalWrite(pinLED,HIGH);
  }
  else
  {
    digitalWrite(pinLED,LOW);
  }
}
```

6. compile menggunakan verify button (tanda ceklist pada IDE arduino) untuk mengecek ada atau tidaknya error/kesalahan dalam pengetikan.

7. upload program ke arduino dengan cara, pilih File > Upload to I/O board, atau tekan tombol tanda panah pada jendela IDE arduino.

Amati hasilnya. Jika program yang anda ketikkan benar maka, hasilnya adalah LED akan menyala jika Button kita tekan sebanyak 4 kali dan sebaliknya LED akan padam jika Button belum ditekan sebanyak 4 kali.

Diskusi dan Pembahasan Sketch

Sketch atau kode-kode program kali ini, mungkin merupakan sketch yang paling kompleks pada seri praktikum dasar Arduino kita ini. Untuk itu ingat kembali bahwa salah satu cara terbaik untuk dapat memahami atau bahkan membuat sketch yang panjang, yaitu dengan membaginya menjadi beberapa bagian.
Seperti biasa, kode program atau sketch kita kali ini, kita bagi menjadi tiga bagian utama. Blok deklarasi variable, blok setup, dan blok loop.

Baiklah, sekarang kita bahas satu per satu kita mulakan dari blok pertama berikut ini.

Blok Pertama

Kita perhatikan bersama kode-kode pada blok pertama ini. Variable-variable yang kita deklarasikan dan inisialisasikan untuk pin, kita berikan qualifier constanta, agar nilainya tidak berubah-ubah. Sedangkan untuk variable yang memeriksa kondisi maupun variable penghitung kita deklarasikan tanpa qualifier. Karena, variable-variable ini akan kita ubah-ubah atau dengan kata lain kita update nilainya melalui kode program. Berikut ini variable-variable tersebut.

```
const int pinButton = 4;
const int pinLED = 13;

int penghitung = 0;
int kondisiSekarang = 0;
int kondisiTerakhir = 0;
```

Dengan membaca nama-nama dari variable-variable di atas, tentunya Anda paham apa fungsi dari tiap-tiap variable tersebut. Dua variable yang pertama berguna untuk mendefinisikan nomer pin yang digunakan, baik untuk Button maupun untuk LED. Tiga variable selanjutnya masing-masing berfungsi sebagai penghitung (counter), dan penjejak atau penanda kondisi. Inilah tiga jenis kegunaan variable yang biasa digunakan dalam aplikasi mikrokontroler. Setelah kita sudah selesai dengan deklarasi variable, sekarang kita lanjutkan ke blok program yang berikutnya.

Blok Kedua

Seperti biasanya, pada blok kedua ini kita meletakkan routine **setup**(). Pada routine ini, kita mengatur mode dari pin-pin yang kita gunakan baik itu pin untuk Button maupun pin untuk LED. Selain itu, kita juga mengatur baudrate dari komunikasi serial antara Arduino dengan Komputer. Berikut ini kode routine **setup**() tersebut.

```
void setup()
{
  pinMode(pinButton,INPUT);
  pinMode(pinLED,OUTPUT);
  Serial.begin(9600);
}
```

Sebagai catatan, saya selalu menggunakan variable untuk mendefinisikan pin yang digunakan. Hal ini saya lakukan agar lebih mudah mengingat fungsi dari tiap pin dan mempermudah dalam mengaturnya, atau bahkan memperbaiki kesalahan dalam kode-kode program yang kompleks. Saya yakin, saat nanti Anda sudah banyak bergelut dengan kode-kode pemrograman yang panjang dan kompleks, penggunaan nama variable ini

pasti akan sangat membantu. Oke, sekarang kita lanjutkan kembali pembahasan kita ke blok program yang berikutnya.

Blok Ketiga

Pada blok ke tiga ini, kita memiliki routine loop(). Routine loop() kita kali ini diawali dengan membaca kondisi dari Button pada protoboard yang terhubung ke pin 4 Arduino. Untuk melakukannya, kita menggunakan fungsi digitalRead(). Berikut ini kode sketchnya.

```
kondisiSekarang = digitalRead(pinButton);
```

Seperti telah kita ketahui bersama, fungsi digitalRead() ini akan menghasilkan nilai antara HIGH atau LOW. Nilai yang dihasilkan dari bacaan fungsi inilah yang kemudian kita simpan ke dalam sebuah variable penjejak yang bernama “kondisiSekarang”.

Setelah itu, hal selanjutnya yang kita lakukan adalah memeriksa apakah kondisi dari Button berubah dari kondisinya yang terakhir? – maksudnya, apakah button sudah ditekan (artinya kondisi terakhirnya tidak ditekan) atau apakah button sudah dilepas (artinya kondisi terakhirnya atau sebelumnya sedang ditekan)?. Nah, untuk melakukan itu, kita gunakan fungsi if-statement. Perhatikan kode sketchnya berikut ini.

```
 if ( kondisiSekarang != kondisiTerakhir)
 {
```

Pada kode sketch yang sebelumnya, kita telah menyimpan hasil bacaan dari Button kedalam variable “kondisiSekarang”. Sekarang, variable “kondisiSekarang” tersebut kita bandingkan nilainya dengan variable “kondisiTerakhir” menggunakan operator NOT (!=). Tanda (!=) ini artinya “tidak sama dengan”.

Jadi, maksud dari kode sketch di atas kira-kira berbunyi seperti ini.—“ Jika Button sekarang ditekan, dan sebelumnya tidak ditekan, maka kerjakan apa yang ada dalam kurung kurawal if. Begitu juga sebaliknya, jika Button sekarang dilepaskan, dan sebelumnya keadaannya sedang ditekan, maka kerjakan kode dalam kurung kurawal”. Untuk pemeriksaan pertama ini, variable pembandingnya yaitu “kondisiTerakhir” sudah kita inisialisasikan di awal dengan nilai 0 atau LOW. Silahkan periksa kembali di blok deklarasi variable pada blok pertama.

Nah, sekarang katakanlah program atau sketch Arduino telah berjalan beberapa saat. Kemudian, kita tekan Button pada protoboard. Maka artinya, kondisi pada fungsi if statement di atas sudah terpenuhi.—karena variable “kondisisSekarang” sudah berubah nilainya dari LOW menjadi HIGH.

Kemudian, apabila kita melepaskan Button pada protoboard, maka variable “kondisiSekarang” akan kembali berubah dari yang sebelumnya HIGH menjadi LOW. Hal ini juga menyebabkan kondisi if statement terpenuhi – dan inilah yang kita sebut dengan perubahan kondisi.

Baiklah, sekarang kita sudah memiliki sebuah penjejak perubahan kondisi. Sekarang, apa yang terjadi dalam penjejak tersebut?! Oke, sekarang kita perhatikan kembali apa yang kita miliki dalam kurung kurawal if statement sebelumnya.

Di dalam kurung kurawal if statement sebelumnya, kita memiliki if statement lagi di dalamnya. (if statement di dalam if statement). Nah, berikut ini if statement yang ada di dalam if statement kita sebelumnya.

```
  if (kondisiSekarang == HIGH)
  {
    penghitung ++;
    Serial.println ("on");
    Serial.print("Banyaknya Button di Tekan : ");
    Serial.println(penghitung);
  }
  else
  {
  Serial.println("off");
  }
```

Kondisi yang diperiksa dalam if statement ini yaitu,-- memeriksa apakah Button berubah kondisinya dari LOW menjadi HIGH? Atau dengan kata lain, apakah Button secara fisik telah berubah dari sebelumnya ditekan menjadi dilepas. Apabila kondisi tersebut terpenuhi, maka kita melakukan dua hal berikut.

1. Kita meng-update nilai variable “penghitung” dengan menambahnya 1 (ingat, fungsi aritmatik ++ sama artinya dengan +1). Semoga Anda masih ingat.
2. Kita mengirim beberapa informasi ke serial monitor komputer.

Di sini, informasi yang kita print ke serial monitor adalah informasi yang sedang terjadi dalam program. Seperti informasi berapa banyak Button kita tekan, dan bagaimana kondisi Button saat ini.

Apabila kondisi yang diperiksa pada if statement ini tidak terpenuhi, maka else statementlah yang akan dieksekusi. Dalam else statement ini, kita hanya menampilkan informasi teks “off” ke serial monitor komputer.

Oke, sekarang kita lanjutkan kembali ke kode program yang berikutnya pada routine loop() ini. Setelah dua fungsi if statement di atas sudah selesai di eksekusi. Maka program akan melanjutkan mengeksekusi kode selanjutnya. Berikut ini kode tersebut.

```
kondisiTerakhir = kondisiSekarang;
```

Adapun yang kita lakukan pada kode ini yaitu, meng-update nilai dari variable “kondisiTerakhir” menjadi nilai baru yang berasal dari variable “kondisiSekarang”.—ingat kembali bahwa variable “kondisiSekarang” tersebut nilainya mengambil dari kondisi pushbutton (fungsi digitalRead(), lihat kembali pada bagian awal routine loop()). Karena kita telah mengatur perubahan kondisi pada fungsi if statement sebelumnya, maka variable “kondisiSekarang” ini akan ter-update dengan nilai “kondisiTerakhir” yang sebelumnya.

Baiklah, sekarang kita berlanjut ke bagian terakhir dari routine loop() kita kali ini. Pada kode-kode berikut ini, kita akan mengatur program untuk menghasilkan output LED menyala atau padam. Aturan yang kita buat kali ini adalah, LED akan menyala apabila kita menekan pushbutton sebanyak 4 kali. Untuk melakukan penghitungan inilah maka kita gunakan sebuah variable counter yang kita berinama “penghitung”. Berikut ini kode-kode tersebut.

```
if (penghitung % 4 == 0)
 {
   digitalWrite(pinLED,HIGH);
 }
 else
```

```
 {
   digitalWrite(pinLED,LOW);
 }
```

Perhatikan kondisi yang diperiksa pada if statement kita kali ini terlihat sedikit berbeda. Di situ, terdapat tanda % (simbol persen). Simbol ini adalah operator yang disebut dengan Operator Modulo. Kita perhatikan kodenya berikut.

```
if (penghitung % 4 == 0)
```

Untuk memahami apa maksud dari kondisi dalam kurung if statement di atas, kita perlu untuk memahami terlebih dahulu tentang operator modulo. Operator modulo ini bekerja dengan cara mengambil nilai yang tersisa dari sebuah operasi pembagian dua buah bilangan bulat (integer). Dalam pembagian integer, kita tidak menggunakan angka desimal ataupun pecahan. Intinya, pembagian ini harus terbagi habis dengan merata, jika terdapat sisa pembagiannya, operator modulolah yang mengambilnya. Untuk lebih jelasnya coba perhatikan contoh berikut ini.

5 / 2 = 2 , sisanya 1 diambil operator modulo
17 / 5 = 3, sisanya 2 diambil operator modulo
10 /3 = 3, sisanya 1 diambil operator modulo
6 / 6 = 1, sisanya 0 diambil operator modulo

Nah, di atas kita bisa lihat, jika kita membagi dua buah nilai integer (bilangan bulat), maka sisa dari hasil pembagiannya itulah yang diambil nilainya oleh operator modulo.—pertanyaannya sekarang, bagaimana jika bilangan yang dibagi lebih kecil dari pembaginya?! (ingat, tidak boleh ada decimal ataupun pecahan). Kita lihat contoh berikut.

1 / 4 = 0, sisa 1
2 / 4 = 0, sisa 2
3 / 4 = 0, sisa 3
4 / 4 = 0, sisa 0

Jadi, jika bilangan yang dibagi lebih kecil dari pembaginya, maka hasilnya adalah 0 dan sisanya adalah angka yang dibagi tersebut. Semoga Anda paham maksud saya. ☺

Baiklah, sekarang kita sudah paham bahwa operator modulo ini kita gunakan untuk mengambil nilai sisa dari sebuah pembagian bilangan bulat (integer). Jadi, apa maksud dari kondisi pada kode if statement berikut?!

```
if (penghitung % 4 == 0)
```

Variable “penghitung” di atas, adalah variable yang kita gunakan untuk menyimpan jumlah tekanan pada Button di protoboard kita. Jadi, kode di atas artinya kira-kira seperti ini, (jika sisa dari jumlah Anda menekan Button di bagi 4 sisanya adalah 0, maka eksekusi if statement!!). Perhatikan penjelasan berikut ini.

Kita asumsikan kita telah menekan Button sebanyak 1 kali, maka hasil dari kondisi if statementnya,

If (1 % 4 == 0) → kondisi tidak terpenuhi, karena 1 / 4 = 0, sisanya 1.

Jika kita tekan lagi untuk kedua kalinya, maka hasilnya:

If (2 % 4 == 0) → kondisi masih tidak terpenuhi, karena 2 /4 = 0, sisanya 2.

Kita tekan lagi Button untuk ke tiga kalinya, maka hasilnya:
If (3 % 4 == 0) → kondisi masih tidak terpenuhi, karena 3 / 4 = 0, sisanya 3.

Kita tekan lagi untuk yang ke empat kalinya, maka hasilnya:

If (4 % 4 == 0) → kondisi terpenuhi, karena 4 / 4 = 0, sisanya 0.

Nah, sekarang Anda paham maksud saya?! – jika kita coba terus menekan Button, maka Anda akan melihat bahwa setiap kita menekan button sebanyak 4 kali atau kelipatannya, maka hasil dari operator modulo di atas akan kembali lagi menjadi 0. Inilah yang kita manfaatkan untuk membuat siklus terpenuhinya kondisi if statement kita.

Kita lihat kembali kode-kodenya,

```
if (penghitung % 4 == 0)
 {
  digitalWrite(pinLED,HIGH);
 }
 else
 {
  digitalWrite(pinLED,LOW);
 }
```

Jadi, pada if statement kali ini, kita membuat program untuk mengatur LED menyala jika kita telah menekan Button sebanyak 4 kali atau kelipatannya, dan selain itu, kita padamkan LED. Jika Anda perhatikan dengan baik dan mengikuti seri-seri praktikum Arduino ini, tentunya Anda bisa memahami dengan baik kode-kode di atas. Right!!

Akhirnya, selesai sudah pembahasan kita untuk routine loop() ini. Sebelum kita akhiri, berikut ini adalah review dari program pada routine loop() ini:

1. Kita periksa kondisi dari Button (ditekan atau dilepas)
2. Jika kondisi Button berubah,
    a. Dari tidak ditekan menjadi ditekan, → tambah “penghitung” dan print info
    b. Dari ditekan menjadi tidak ditekan, → print info
3. Update “kondisiTerakhir” dengan “kondisiSekarang”.
4. Periksa, apakah button sudah di tekan 4 kali atau kelipatannya?
    a. Jika Ya, nyalakan LED.
    b. Jika Tidak, Padamkan LED.
5. Ulangi terus routine loop() ini.

Dengan memahami metode untuk mendeteksi perubahan kondisi, saya yakin akan sangat membantu Anda dalam mengembangkan berbagai jenis Aplikasi, khususnya dalam aplikasi mikrokontroler ini.—metode ini tidak hanya dapat diterapkan pada Button, ada banyak yang lainnya. Memahami operator modulo juga tentu akan menambah “peralatan” Anda dalam membuat Aplikasi mikrokontroler dan trik-trik dalam pemrograman Anda.

Nah, selesai sudah pembahasan kita pada diskusi sketch kita kali ini. Terimakasih, tetap semangat dan sampai jumpa lagi di praktikum berikutnya... ☺

Latihan Mandiri

Untuk meningkatkan kemampuan dan pemahaman Anda, coba kerjakan latihan mandiri berikut.

1. Ubahlah program ini agar dapat mendeteksi kondisi yang sebaliknya. Yaitu, memeriksa perubahan dari HIGH ke LOW!
   Ubahlah sketchnya, agar LED menyala jika Button dilepaskan sebanyak 7 kali, selain itu padamkan!


www.BELAJARBIKINROBOT.WEEBLY.com