02 - Input dan output

Pada percobaan kali ini, kita akan mempelajari bagaimana sebuah ESP32 menerima perintah (*input*) dan menjalankan perintah (*output*) sesuai dengan program yang kita buat. *Pushbutton* kita anggap sebagai komponen input yang akan dibaca oleh ESP32.

- Belajar input & output ESP32

Belajar input & output ESP32

Pada percobaan yang lalu kita telah belajar membuat LED internal ESP32 menyala berkedip (*blink*), kali ini kita akan mencoba membuat LED eksternal menyala dan mati dengan dikontrol oleh sebuah tombol (*pushbutton*). Pada percobaan kali ini, kita akan mempelajari bagaimana sebuah ESP32 menerima perintah (*input*) dan menjalankan perintah (*output*) sesuai dengan program yang kita buat. *Pushbutton* kita anggap sebagai komponen input yang akan dibaca oleh ESP32. Karena *pushbutton* adalah sebuah saklar yang hanya dapat mengalirkan dan memutuskan arus, maka ESP32 menggunakan *port digital* untuk dapat membaca nilai dari *pushbutton*. *Pushbutton* bernilai 0 jika tidak mengalirkan arus, sedangkan bernilai 1 jika meneruskan arus. LED digunakan untuk menampilkan output dari ESP32, output disini adalah output digital ya teman-teman, yaitu pada saat LED menyala berarti bernilai 1 sedangkan pada saat mati bernilai 0.

> Weekly project 02 | Sistem Penginderan

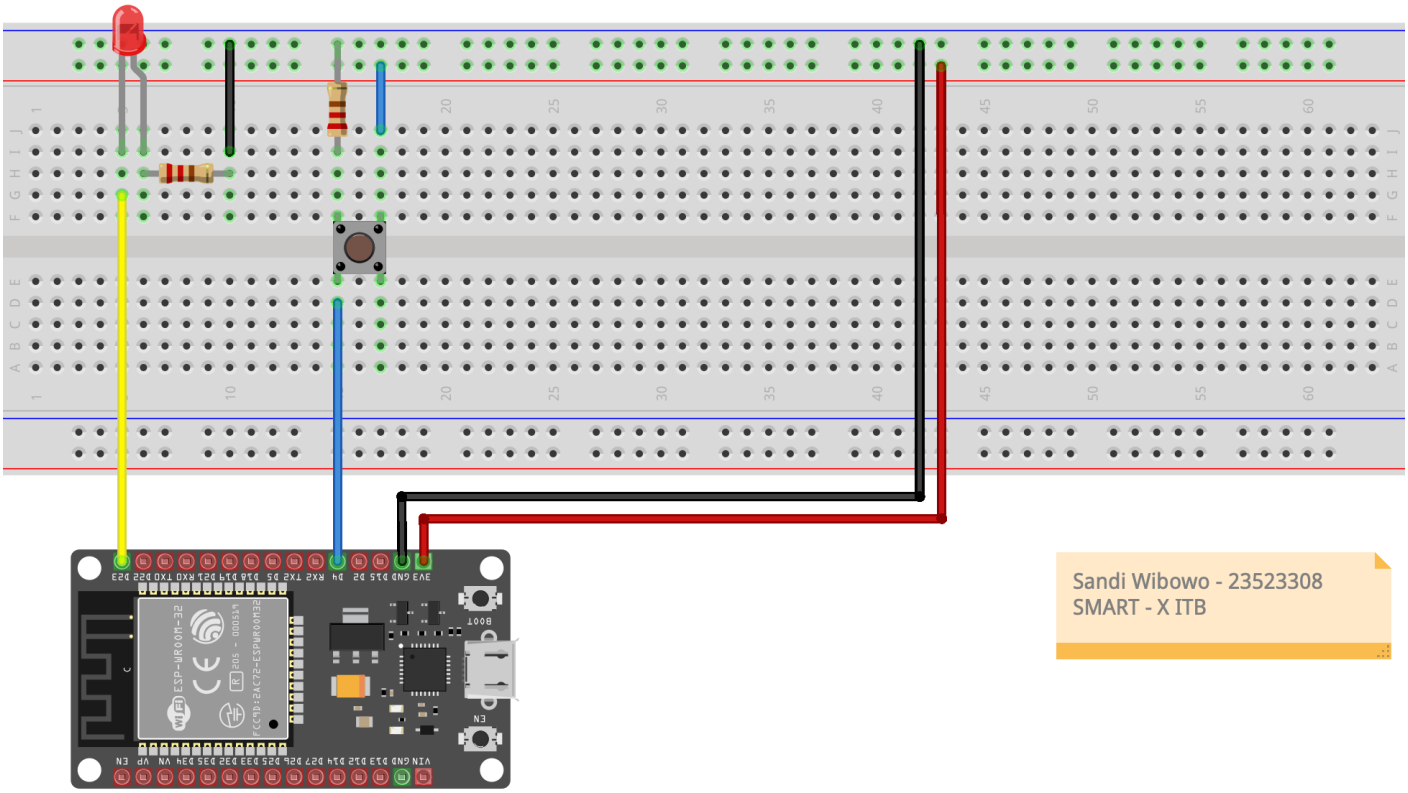
Persiapan

Sekarang kita siapkan komponen elektronika yang akan kita butuhkan. Pertama, kita membutuhkan komponen utama yaitu ESP32, versi yang saya gunakan adalah ESP32 dengan *development board* buatan Devkit versi 1. Lengkapnya dapat dilihat di tabel berikut.

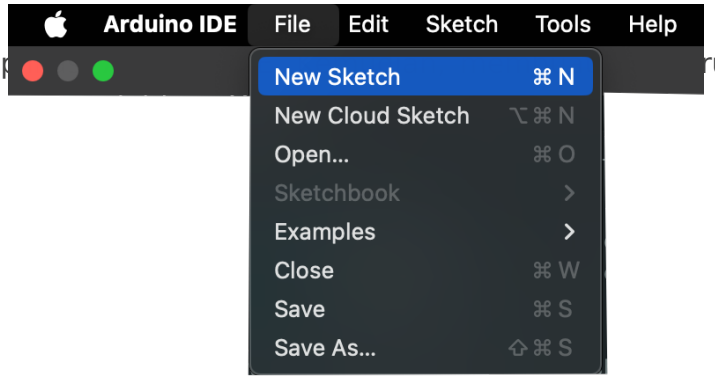
No.	Nama Komponen
1	*Development-board* ESP-32
2	5 mm LED
3	330 Ohm resistor
4	Pushbutton
5	10 K Ohm resistor
6	Breadboard
7	Jumper wires

Rangkai komponen diatas menjadi sebuah rangkaian seperti skema di bawah. Perhatikan dengan seksama terutama untuk komponen LED, karena komponen ini marupakan sebuah dioda dimana terdapat kaki kutub positif dan negatif, yang jika kita salah pasang, maka LED tidak akan menyala. Sedangkan *pushbutton* pastikan rangkaian terpasang pada sisi terbuka (NO = *Normally Open*) jika

tidak maka sinyal input yang masuk ke ESP32 akan bernilai 1 terus, yang menyebabkan LED akan menyala terus.



Membuat kode program dengan membuka ap ru (



sketch) dengan cara pilih **File > New Sketch.**

Bagi yang belum menginstal Arduino IDE, dapat mengikuti langkah-langkah instalasi di link ini. Pada kode program di bawahini terdiri dari beberapa langkah yaitu pertama mendeklarasikan sebuah variabel yang berisi nomor GPIO sebagai input dan output. Kemudian membuat variabel yang menyimpan pembacaan dari pushbutton. Setelah itu membuat operasi logika, untuk menyalakan LED jika *pushbutton* ditekan.

```
const int buttonPin = 4; // deklarasi variabel bertipe integer berisi nomor GPIO 4
const int ledPin = 23; // deklarasi variabel bertipe integer berisi nomor GPIO 23
```

```
int buttonState = 0; // deklarasi variable buttonState dengan angka 0

void setup(){
  Serial.begin(115200); // Inisiasi komunikasi serial dengan baudrate 115200
  pinMode(buttonPin, INPUT); // konfiguras pin GPIO sebagai INPUT
  pinMode(ledPin, OUTPUT); // konfigurasi pin GPIO sebagai OUTPUT
}

void loop(){
  buttonState = digitalRead(buttonPin); // membaca sinyal dari pushbutton
  if (buttonState == HIGH) { // jika pushbutton ditekan maka :
    digitalWrite(ledPin, HIGH); // memerintahkanb LED untuk menyala
  } else {
    digitalWrite(ledPin, LOW); // memerintahkan LED untuk off
  }
}
```

Penjelasan kode

Baris kode berikut, artinya kita mendeklarasikan variabel `buttonPin` dan `ledPin` bertipe integer, dengan tambahan `const` berarti varibel ini nilainya tetap atau tidak dapat diubah.

```
const int buttonPin = 4;
const int ledPin = 23;
```

Nilai 4 dan 23, mengacu pada GPIO yang digunakan pada board ESP32, untuk melihat skema pinout ESP32 dapat menggunakan link berikut. GPIO 4 dihubungkan ke pushbutton, sedangkan GPIO 23 dihubungkan ke LED.

Kemudian, kita membuat variabel `buttonState` yang bertipe integer, variabel ini digunakan untuk menyimpan pembacaan status pushbutton. Normalnya kita set nilai 0 pada saat pushbutton tidak ditekan.

```
int buttonState = 0;
```

Dalam `setup()` kita melakukan inisiasi *pushbutton* sebagai `INPUT` dan LED sebagai `OUTPUT`. Proses inisiasi dilakukan dengan fungsi `pinMode()` untuk menetapkan sebuah GPIO sebagai `INPUT` atau `OUTPUT`.

```
pinMode(buttonPin, INPUT);
pinMode(ledPin, OUTPUT);
```

Dalam fungsi `loop()` kita menempatkan kode untuk membaca status *pushbutton* dan menyalakan LED. Kode pertama, melakukan pembacaan status *pushbutton* dengan fungsi `digitalRead()` , hasil pembacaan ini disimpan dalam variabel `buttonState`.

```
buttonState = digitalRead(buttonPin);
```

Jika `buttonState` bernilai `HIGH`, maka LED harus menyala, perintah menyalakan LED ini menggunakan fungsi `digitalWrite()`.

```
if (buttonState == HIGH) {
  digitalWrite(ledPin, HIGH);
}
```

Namun, jika `buttonState` bernilai bukan `HIGH`, maka LED harus dimatikan dengan fungsi `digitalWrite()` diatur dengan nilai `LOW`.

```
else {
  digitalWrite(ledPin, LOW);
}
```

Upload kode

Setelah kode program selesai diketik, selanjutnya upload program dengan menekan tombol upload (icon anak panah ke kanan) pada aplikasi Arduino IDE.

Kemudian, tunggu sampai program selesai di compile dan diupload ke dalam ESP32. Pastikan muncul notifikasi bahwa upload kode telah berhasil. Setelah berhasil, maka akan percobaan kita akan menjadi seperti dalam video berikut.

https://www.youtube.com/embed/ULa9wtr8xBU

> Referensi :

> https://randomnerdtutorials.com/esp32-digital-inputs-outputs-arduino

---

Modifikasi program

Pada *section* ini, kita mencoba mengembangkan program kecil untuk mengontrol ESP32 melalui dua buah *pushbutton* sebagai input. Ide **"iseng-nya"** adalah kita mencoba belajar menggunakan konsep pemrograman iterasi seperti `for loop,` bermain dengan logika kondisional `if` dan `switch`.

Kita akan fungsikan dua *pushbutton* untuk mengatur ESP32 sebagai tombol aksi dan mode. Sedangkan output yang akan kita gunakan adalah satu buah LED dan satu buzzer. Agar lebih menarik, kita akan pasang OLED berukuran 128x64 untuk menampilkan informasi status mode dan jumlah beep yang akan dinyalakan.

Cara kerja program kecil ini adalah pada saat tombol aksi ditekan, maka LED dan buzzer akan menyala bersamaan sebanyak satu kali. Pada saat tombol aksi ditekan kedua kali, maka buzzer dan LED akan menyala dua kali. Jumlah penyalaan ini akan terus bertambah sampai tombol aksi ditekan ke sepuluh kali, dan counter akan mereset kembali ke satu. Canggihnya, kita bisa memilih antara buzzer atau LED yang akan dinyalakan pada eksekusi berikutnya. Pengguna dapat melihat layar OLED untuk mengetahui jumlah beep dan mode yang digunakan.

Berikut adalah kodenya, ***lets compile !!!***

```
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>

// Membuat LED blink dan buzzer menyala dengan dikontrol oleh dua tombol
// Tombol pertama untuk mengeset berapa kali led atau buzzer menyala (<10)
// Tombol kedua untuk mengatur apakah hanya led, buzzer atau kedua-duanya yang menyala

#define SCREEN_WIDTH 128
#define SCREEN_HEIGHT 64
Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, -1);

const int button1Pin = 4;
const int button2Pin = 2;
const int ledPin = 23;
const int buzzerPin = 27;

int button1State = 0;
```

```
int button2State = 0;
int modeState = 0;
int actionState = 0;

void setup(){
  Serial.begin(115200);
  if(!display.begin(SSD1306_SWITCHCAPVCC, 0x3C)){
    Serial.println("OLED tidak terkoneksi");
    for(;;);
  }

  pinMode(button1Pin, INPUT);
  pinMode(button2Pin, INPUT);
  pinMode(ledPin, OUTPUT);
  pinMode(buzzerPin, OUTPUT);

  // welcome message Smart-x di OLED
  delay(2000);
  display.clearDisplay();
  display.setTextColor(WHITE);
  display.setTextSize(1);
  display.setCursor(25, 31);
  display.print("SMART-X ITB");
  display.display();
  delay(5000);
  display.clearDisplay();
  delay(500);



}
// Fungsi untuk menyalakan led dan buzzer
void ledBlink(int t, int mode){
  for (int i=1; i<=t; i++){
    switch (mode){
    case 0:
      digitalWrite(ledPin,HIGH);
      digitalWrite(buzzerPin,HIGH);
      delay(100);
      digitalWrite(ledPin,LOW);
      digitalWrite(buzzerPin, LOW);
```

```
      delay(100);
      break;
    case 1:
      digitalWrite(ledPin,HIGH);
      delay(100);
      digitalWrite(ledPin,LOW);
      delay(100);
      break;
    case 2:
      digitalWrite(buzzerPin,HIGH);
      delay(100);
      digitalWrite(buzzerPin,LOW);
      delay(100);
      break;
    }
  }
}

// Fungsi untuk membunyikan buzzer
// Bunyi notifikasi selesai menyeting beep 3x
void beepSet(int times){
  for (int i=1; i<=times; i++){
    digitalWrite(buzzerPin, HIGH);
    delay(50);
    digitalWrite(buzzerPin, LOW);
    delay(50);
  }
}

// Fungsi untuk bunyi reset (beep panjang)
void beepReset(){
  digitalWrite(buzzerPin, HIGH);
  delay(1500);
  digitalWrite(buzzerPin, LOW);
}
// Fungsi untuk menampilkan pesan di OLED
void pesanOled(int modeState, int actionState) {
  display.clearDisplay();
  display.setCursor(25, 0);
  display.print("SMART-X ITB");
```

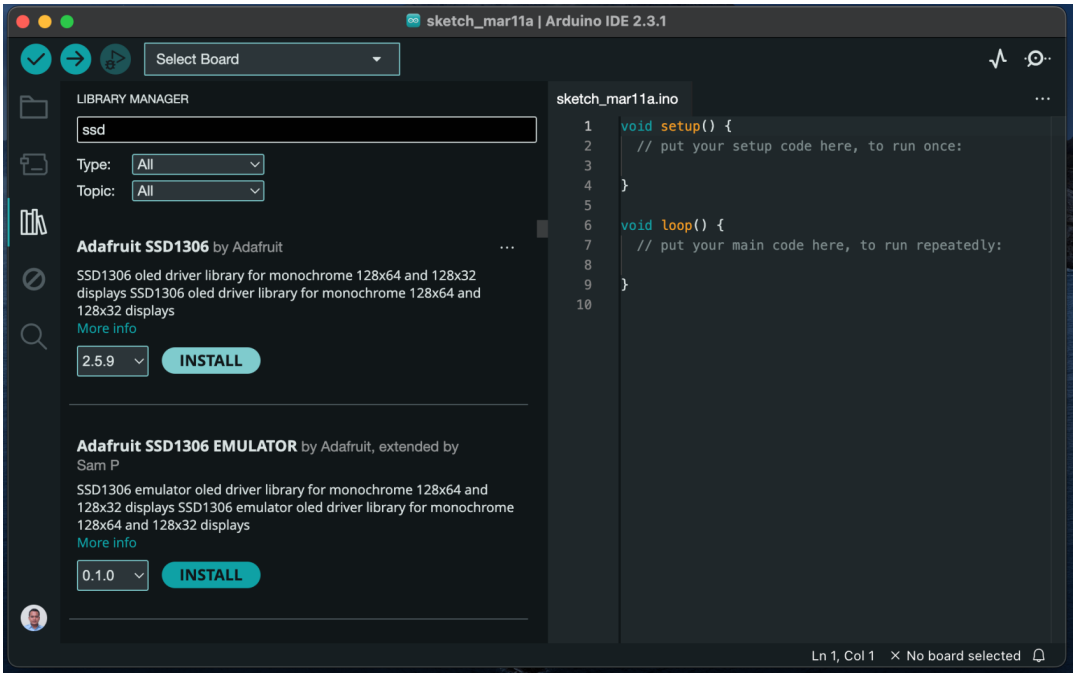
```
  display.setCursor(0, 21);
  display.print("Mode : ");
  switch (modeState) {
    case 0:
      display.print("LED & BUZ");
      break;
    case 1:
      display.print("LED");
      break;
    case 2:
      display.print("BUZ");
  }
  display.setCursor(0,31);
  display.print("Beep : ");
  display.print(actionState + 1);
  display.print("x");
  display.display();
}

// Main program yang harus dijalankan
void loop(){
  button1State = digitalRead(button1Pin);
  button2State = digitalRead(button2Pin);

  pesanOled(modeState, actionState);

  if (button1State == HIGH){
    Serial.println("Button 1: ditekan");
    actionState ++;
    delay(500);
    ledBlink(actionState, modeState);
    if (actionState >= 10){
      actionState = 0;
    }
  }

  // Mengubah value button2State untuk mengubah mode
  if (button2State == HIGH) {
    Serial.println("Button 2 : ditekan ");
    modeState ++;
```

```
    delay(500);
    if (modeState == 1) {
      beepSet(3);
    }
    if (modeState == 2) {
      beepSet(3);
    }
    if (modeState == 3) {
      modeState = 0;
      beepReset();
    }
  }
}
```

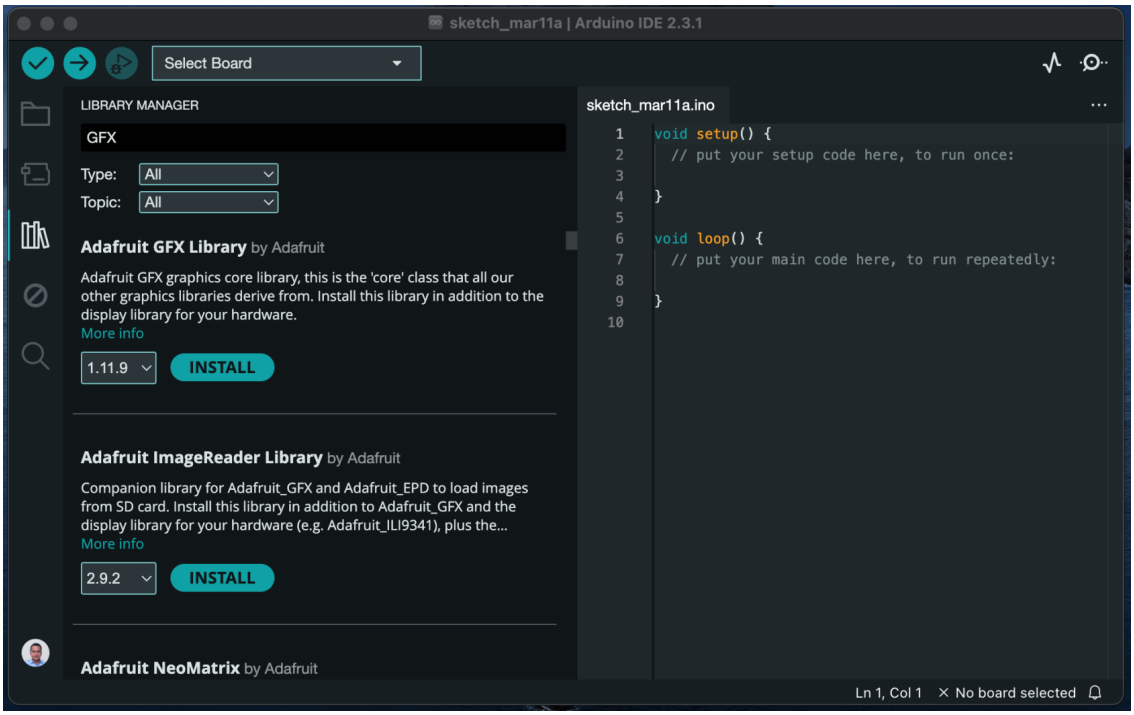
Dari *sketch* diatas, pada line satu sampai dengan tiga, kita menyertakan tiga buah library yaitu: `Wire.h`, `Adafruit_GFX.h` dan `Adafruit_SSD1306.h`. Library `Wire.h` adalah library yang digunakan oleh ESP32 untuk menghubungkan ESP32 dengan OLED melalui komunikasi I2C (*inter-integrated circuit*). Sedangkan library `Adafruit_SSD1306.h` dan `Adafruit_GFX.h` digunakan sebagai driver agar kita dapat menggunakan OLED.

Secara *default* dua library `Adafruit` tidak terinstal secara otomatis, sehingga kita perlu menginstallnya secara manual. Langkah-langkah instalasinya adalah sebagai berikut:

Langkah Pertama, pilih menu icon library pada aplikasi Arduino IDE, kemduian cari **Adafruit SSD1306** dari Adafruit.

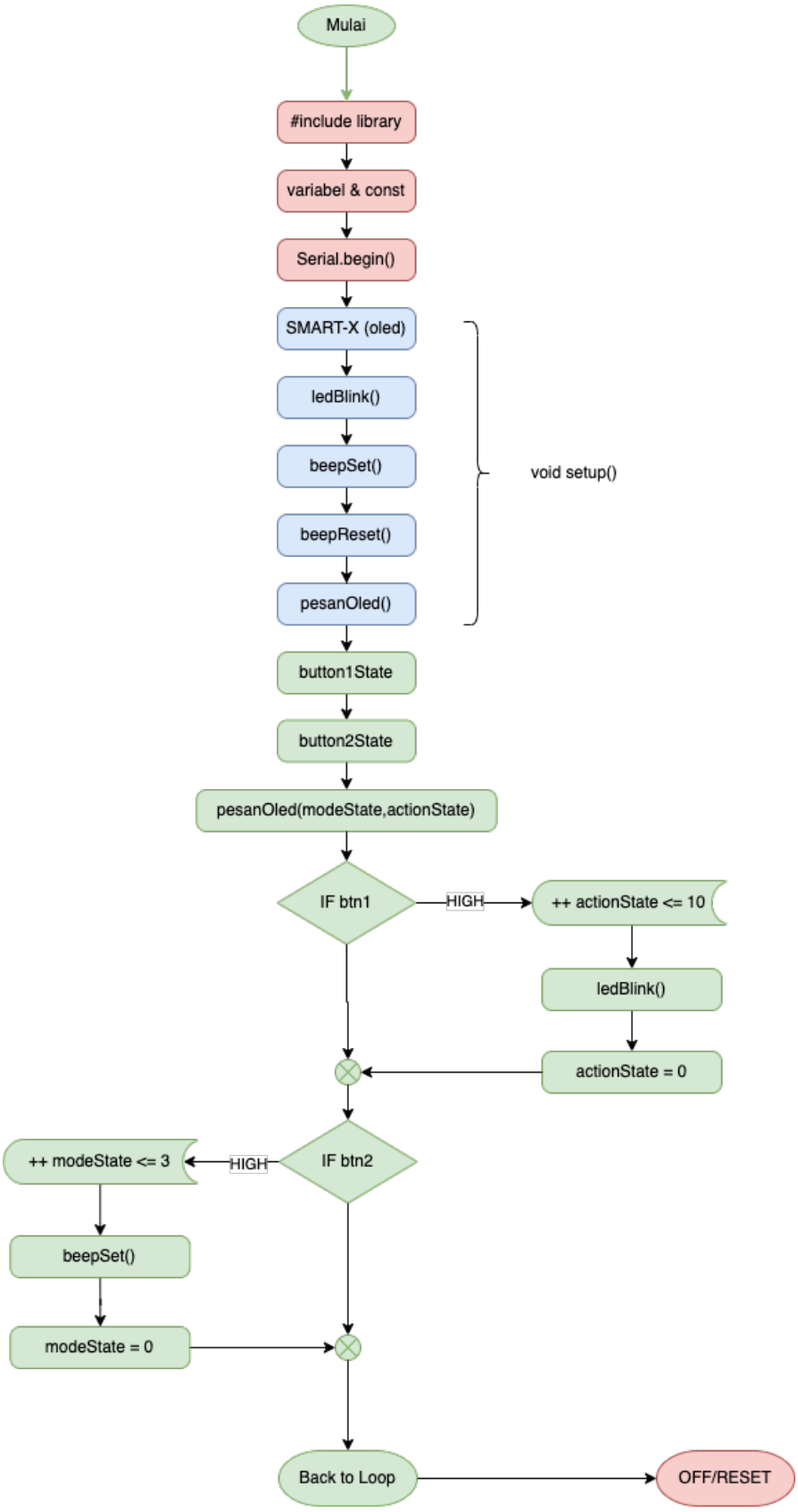


Langkah kedua, install library Adafruit GFX. Cari dan install library GFX dengan mengetikan kata kunci Adafruit GFX di kolom pencarian.



Setelah berhasil menginstal dua library diatas, maka kita sudah bisa menggunakan OLED 128x64. Kita akan menggunakan skema I2C standar di ESP32, sehingga kita tidak perlu mendeklarasikan GPIO yang akan digunakan.

Jika dilihat dari *sketch* diatas, akan terlihat sangat kompleks. Namun secara singkat alur program dapat digambarkan sebagai berikut:



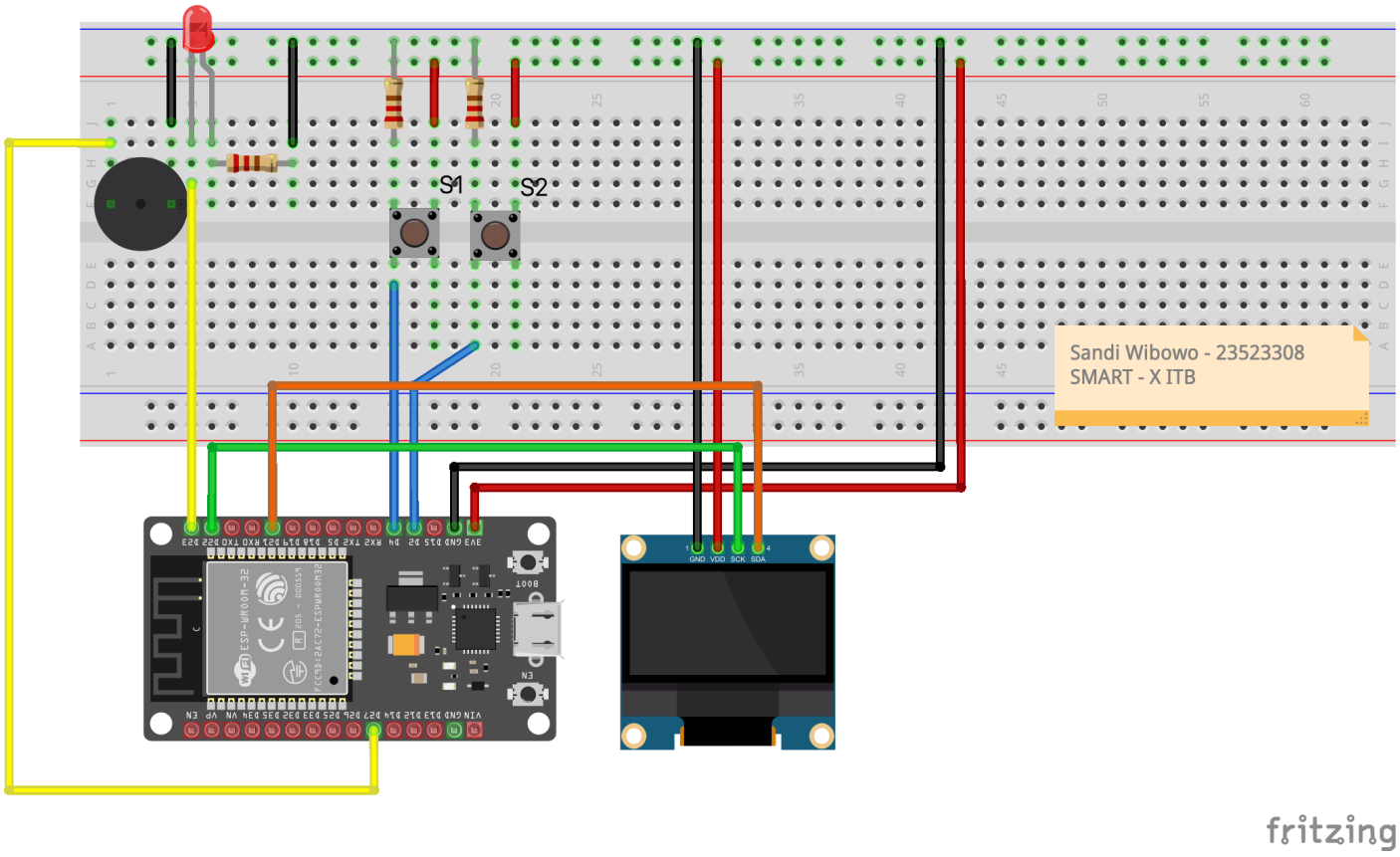
Disini kita mencoba untuk mendifinisikan fungsi-fungsi setelah `setup()`, kemudian memanggil fungsi tersebut di main porgram `loop()`. Hal ini dimaksudkan untuk membuat program ini lebih modular, dan memudahkan melokalisir jika terdapat bug dalam kode kita.

Skema rangkaian

Berdasarkan rencana kode program diatas, maka kita membutuhkan komponen sebagai berikut:

No.	Nama Komponen
1	ESP32 Dev-Board
2	5mm LED
3	Resistor 10 K Ohm
4	Resistor 330 Ohm
5	Buzzer
6	OLED 128x64
7	Push button
8	Juper wires

Skema pemsangan komponen sebagaimana ditunjukan dalam gambar berikut, yang harus diperhatikan adalah pemasangan LED dan *pushbutton,* keduanya memiliki kaki-kaki dengan polaritas positif dan negatif.



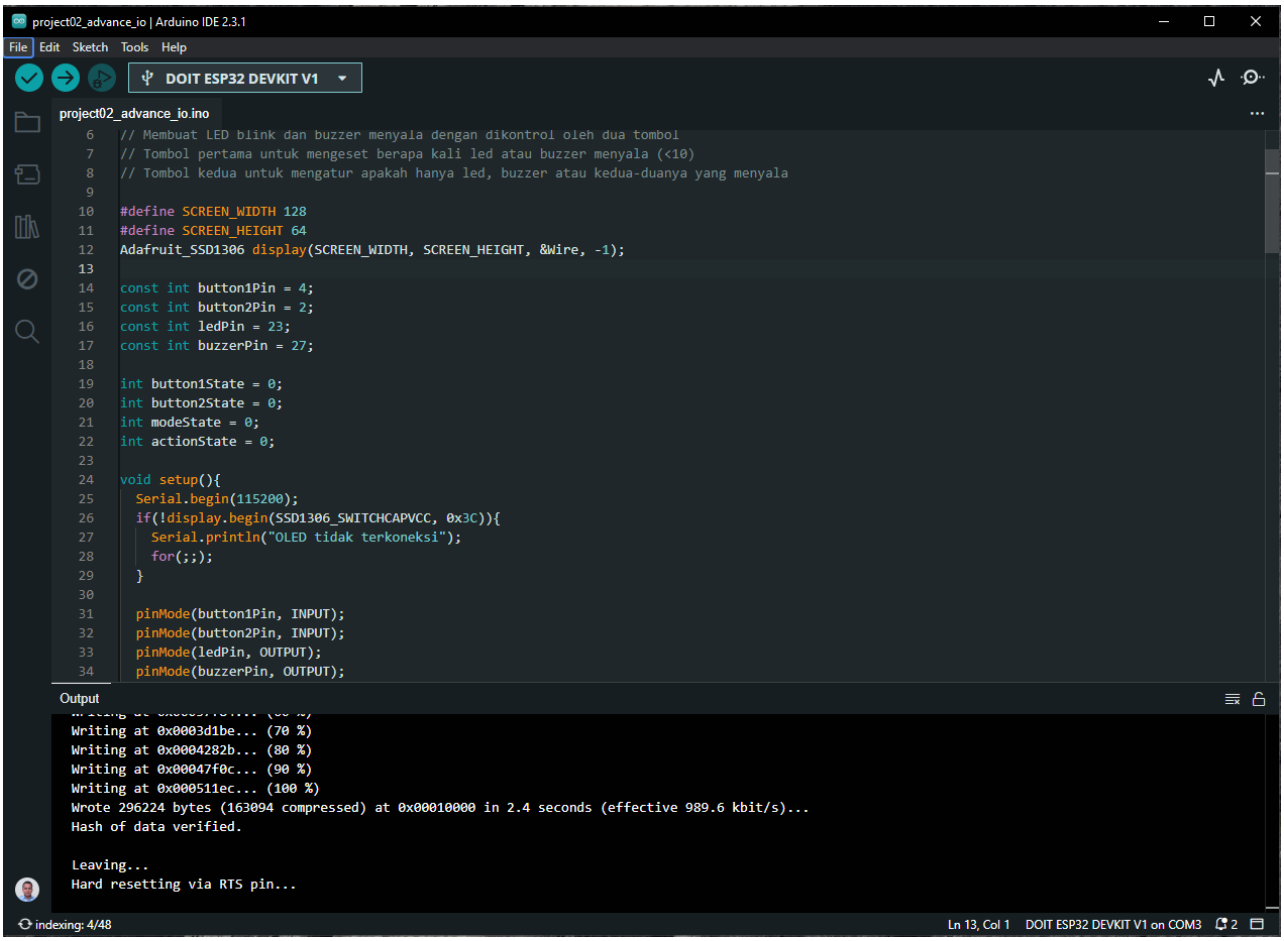
Sedangkan skema I2C dari OLED ke ESP32, seperti ditampilkan grafik berikut:

ESP32	OLED SSD1306 128x64

Vin	3.3V
GND	GND
GPIO 22	SCL
GPIO 21	SDA

Upload dan tes

Setelah rangkaian terpasang dan kode program telah dibuat, mari kita upload kode kita kedalam rangkaian ESP32 dengan menekan tombol icon panah ke kiri. Kemudian tunggu sampai notifikasi **Done uploading** muncul di bagian sisi kanan bawah aplikasi Arduino IDE.



Hasil pengetesan program yang kiat buat tadi dapat dilihat dalam tayangan video berikut.

https://www.youtube.com/embed/5HK4pyWk8zk

> Referensi:
>
> https://randomnerdtutorials.com/esp32-digital-inputs-outputs-arduino

Saat pengetikan kode, jangan lupa titik-koma ya... **;**