

Laporan Instrumentasi Geofisika Lanjut

Pengenalan Microcontroler ESP32



Avif Maulana Azis

140710180048

DEPARTEMEN GEOFISIKA
FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
UNIVERSITAS PADJADJARAN
TAHUN 2021

I. Pendahuluan

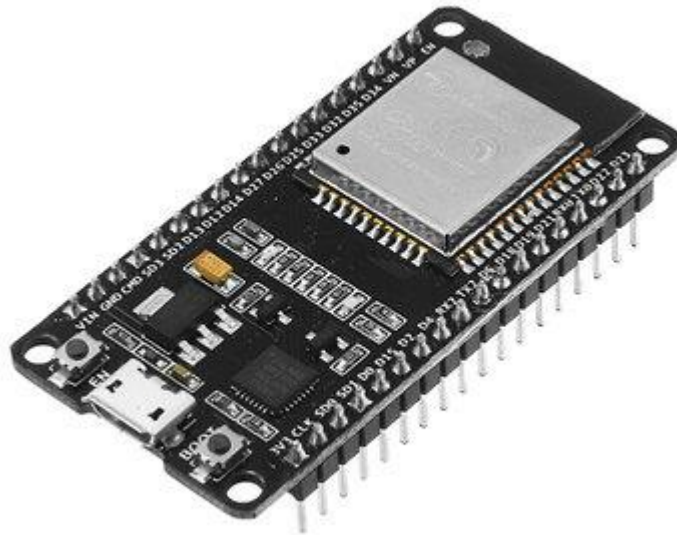
Laporan ini akan menjelaskan tentang pembuatan berbagai rangkaian example microcontroller ESP32 serta beberapa komponen elektronik lain seperti breadboard ,kabel, lampu RGB SMD LED, touch sensor TTP223 capacitive digital switch, buzzer SFM-27 DC 3-24V 95dB, dan sensor vibrasi SW1801P.

A. Microcontroler ESP32

Mikrokontroler adalah sirkuit elektronik terpadu kompak yang dirancang untuk mengatur suatu operasi tertentu dalam sebuah sistem. Dalam mikrokontroler biasanya terdapat prosesor, memori, dan periferal input/output (I/O) pada satu chip.

Microcontroler ESP 32 sendiri merupakan sebuah MCU (Microcontroller unit) yang sudah terintegrasi dengan konektivitas Wi-Fi dan Bluetooth hingga siap untuk penggunaan secara global.

MCU ESP32 ini dapat bekerja di suhu -40°C sampai dengan 125°C . MCU ini juga menunjang fitur *clock gating* dan banyak *power mode* yang dapat disesuaikan oleh keinginan dan kebutuhan. MCU ini juga dilengkapi oleh antenna yang dapat disesuaikan, RF balun, amplifier power, low-noise receive amplifier, filter dan modul manager power. Melalui interface SPI/SDIO atau I2C/UART nya, MCU ini dapat berkomunikasi dengan sistem lain melalui Wi-Fi atau Bluetooth



Gambar 1. ESP 32

II. Kode Arduino dan Maknanya

A. Touch Pin Test

```
1. // ESP32 Touch Test
2. // Just test touch pin - Touch0 is T0 which is on GPIO 4.
3.
4. void setup()
5. {
6.   Serial.begin(115200);
7.   delay(1000); // give me time to bring up serial monitor
8.   Serial.println("ESP32 Touch Sensor Test");
9. }
10.
11. void loop()
12. {
13.   Serial.println(touchRead(T0)); // get value using T0
14.   delay(1000);
15. }
```

Penjelasan

Baris	Penerjemahan
4	Mulai fungsi setup
6-8	Mulai serial dengan frekuensi 115200 kemudian delay 1 detik dan print serial "ESP 32 Touch Sensor Test"
9	Akhir dari fungsi setup
11	Mulai fungsi loop
13-14	Print hasil pembacaan digital dari GPIO 0 yang terhubung dengan sensor sentuh pada serial monitor. Kemudian delay 1 detik
15	Akhir dari fungsi loop

B. Buzzer Test

```

1. #include "pitches.h"
2. const int BUZZZER_PIN = 25; // GPIO18 pin connected to piezo buzzer
3. // notes in the melody:
4. int melody[] = {
5.     NOTE_E5, NOTE_E5, NOTE_E5,
6.     NOTE_E5, NOTE_E5, NOTE_E5,
7.     NOTE_E5, NOTE_G5, NOTE_C5, NOTE_D5,
8.     NOTE_E5,
9. };
10. // note durations: 4 = quarter note, 8 = eighth note, etc, also called tempo:
11. int noteDurations[] = {
12.     8, 8, 4,
13.     8, 8, 4,
14.     8, 8, 8, 8,
15.     2,
16. };
17. void setup() {
18.     // iterate over the notes of the melody:
19.     int size = sizeof(noteDurations) / sizeof(int);
20.     for (int thisNote = 0; thisNote < size; thisNote++) {
21.         // to calculate the note duration, take one second divided by the note type.
22.         //e.g. quarter note = 1000 / 4, eighth note = 1000/8, etc.
23.         int noteDuration = 1000 / noteDurations[thisNote];
24.         tone(BUZZZER_PIN, melody[thisNote], noteDuration);
25.         // to distinguish the notes, set a minimum time between them.
26.         // the note's duration + 30% seems to work well:
27.         int pauseBetweenNotes = noteDuration * 1.30;
28.         delay(pauseBetweenNotes);
29.         // stop the tone playing:
30.         noTone(BUZZZER_PIN);
31.     }
32. }
33.
34. void loop() {
35.     // no need to repeat the melody.
36. }

```

Penjelasan

Baris	Penerjemahan
1	Memasukan file Pitch.h yang berisi frekuensi-frekuensi nada.
2	Deklarasi pin output yang terhubung ke buzzer
4-16	Deklarasi variable melody dan durasi
17	Mulai fungsi setup
19	Deklarasi variable size = sizeof(durasi)/sizeof(int)
20	Iterasi melody. Jadi thisnote dimulai dari 0, selama thisnote < size, maka thisNote akan bertambah 1 terus.
23-24	Selama iterasi, akan melakukan fungsi Tone yang durasinya disesuaikan menjadi 1000/durasinya.
27-28	Melakukan delay pada fungsi tone di pin yang terhubung dengan buzzer selama durasi *1.30.
30	Stop outputnya dengan perintah noTone(). Fungsi for selesai.
32	Akhir dari fungsi loop

C. Wifi : Get Test

```
1. #include <WiFi.h>
2. #include <HTTPClient.h>
3. const char* ssid = "GEOFIS 18";
4. const char* password = "bapaazis";
5.
6. void setup() {
7.   Serial.begin(115200);
8.   delay(4000);
9.   WiFi.begin(ssid, password);
10.  while (WiFi.status() != WL_CONNECTED) {
11.    delay(1000);
12.    Serial.println("Connecting to WiFi..");
13.  }
14.
15.  Serial.println("Connected to the WiFi network");
16.
17. }
18.
19. void loop() {
20.
21.  if ((WiFi.status() == WL_CONNECTED)) { //Check the current connection status
22.
23.    HTTPClient http;
24.
25.    http.begin("http://jsonplaceholder.typicode.com/comments?id=10"); //Specify the
        URL
26.    int httpCode = http.GET(); //Make the request
27.
28.    if (httpCode > 0) { //Check for the returning code

        String payload = http.getString();
        Serial.println(httpCode);
        Serial.println(payload);

29.  }
30.
31.  else {
32.    Serial.println("Error on HTTP request");
33.  }
34.
35.  http.end(); //Free the resources
36. }
37.
38. delay(10000);
39.
40. }
```

Penjelasan

Baris	Penerjemahan
1-2	Include library Wifi dan HTTPClient.
3-4	Deklarasi variable untuk ssid dan password jaringan yang dituju.
6	Mulai fungsi setup
7-8	Memulai serial monitor dengan frekuensi 115200 kemudian delay 4 detik.
9	Untuk memulai koneksi WIFI ke jaringan yang dituju.
10-13	Selama wifi belum terkoneksi, maka akan mengirimkan pesan "connecting to WiFi.."
15	Print connected pada serial monitor.
17	Fungsi setup selesai.
19	Mulai fungsi loop
21-23	Saat terkoneksi wifi, maka akses http sebagai HTTP Client
25-26	Kemudian begin HTTPClient di web yang dimaksud dan buat permintaan dengan fungsi GET()

27-29	Jika request berhasil, maka Mengambil tulisan “string” yang ada dalam web yang dimaksud dan print dalam serial monitor.
31-33	Namun jika request gagal, maka print “req HTTP Failed” pada serial monitor
53	Mengakhiri HTTP
36	Akhir dari fungsi jika alat terkoneksi wifi
38	Delay 10 detik
40	Akhir dari fungsi loop

D. Vibration Test

```

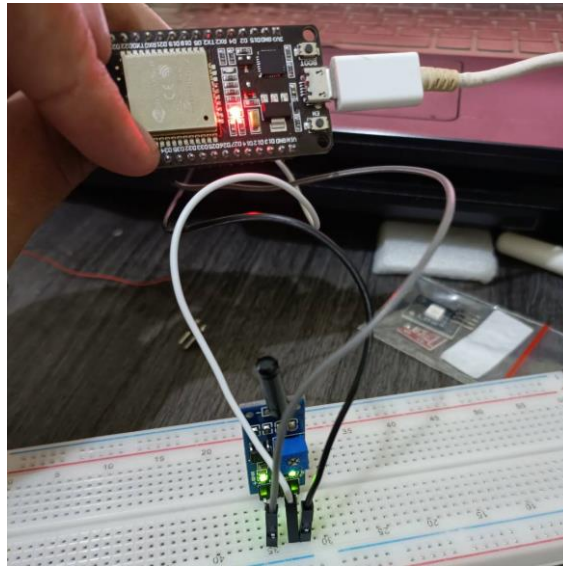
1. // the setup routine runs once when you press reset:
2. void setup() {
3.   // initialize serial communication at 9600 bits per second:
4.   Serial.begin(115200);
5. }
6.
7. // the loop routine runs over and over again forever:
8. void loop() {
9.   // read the input on analog pin 0:
10.  int sensorValue = analogRead(34);
11.  // Convert the analog reading (which goes from 0 - 1023) to a voltage (0 - 5V):
12.  float voltage = sensorValue * (5.0 / 4096.0);
13.
14.  // the lower the voltage, the brighter it is
15.  if ((voltage >= 0) && (voltage <= 0.4)) {
16.    Serial.print ("WARNING!!! - ");
17.  } else if ((voltage > 0.4) && (voltage <= 2)) {
18.    Serial.print ("Vibration - ");
19.  } else {
20.    Serial.print ("NO Vibration - ");
21.  }
22.  // print out the value you read:
23.  Serial.println(voltage); } // end of loop function

```

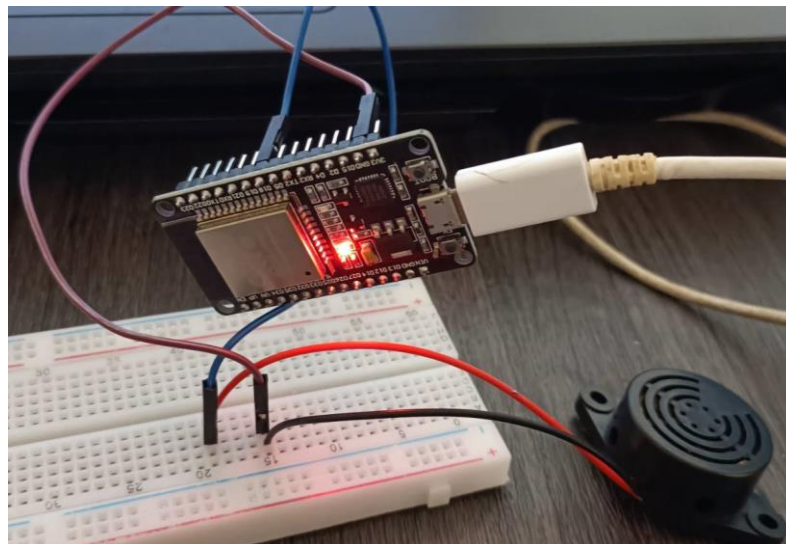
Penjelasan

2	Memulai fungsi setup
4-5	Memulai serial dengan frekuensi 115200. Akhir dari fungsi setup
8	Memulai fungsi loop
10-12	Deklarasi pin, variable dan perhitungan yang digunakan. Digunakan perhitungan tersebut karena pada saat tegangan maksimal yaitu 5V, pembacaan analognya adalah 4096. Oleh karena itu nilai yang terbaca dilalikan perbandingan tersebut.
15-16	Saat tegangannya kecil, berarti sensor sangatlah terguncang, oleh karena itu saat nilai voltagenya 0-0,4 maka alat akan mengirimkan pesan “Warning!!!-” pada serial monitor.
17-18	Saat tegangannya sedang, berarti sensor mulai terguncang, oleh karena itu saat nilai voltagenya 0.4 - 2 maka alat akan mengirimkan pesan “Vibration -” pada serial monitor.
19-20	Saat tegangannya besar, berarti sensor mengalirkan arus dengan lancar tanpa ada gangguan (getaran), oleh karena itu saat nilai voltagenya diatas 2 maka alat akan mengirimkan pesan “NO Vibration -” pada serial monitor.
23	Jika ingin melihat hasil bacaan voltage, maka print di serial monitor. Akhir dari fungsi loop.

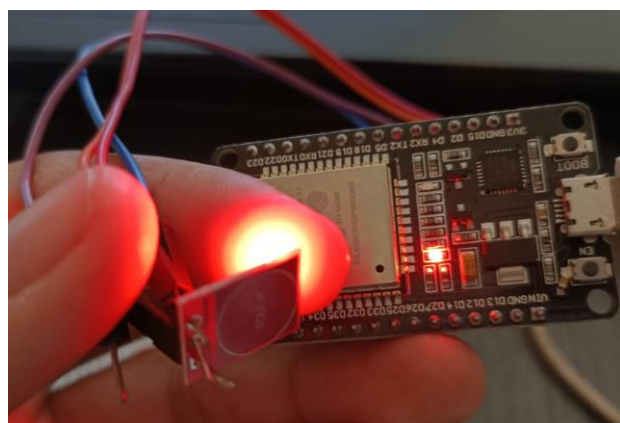
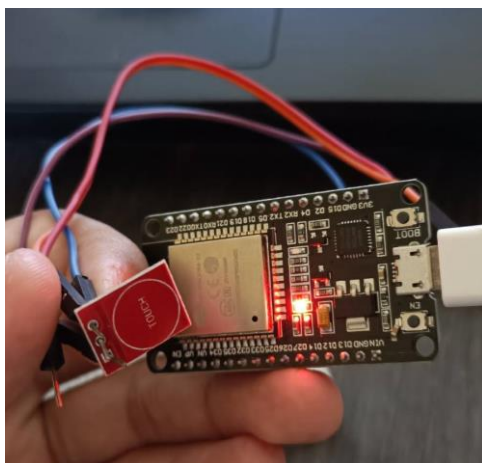
III. Foto Rangkaian



Gambar2. Rangkaian Sensor getar.

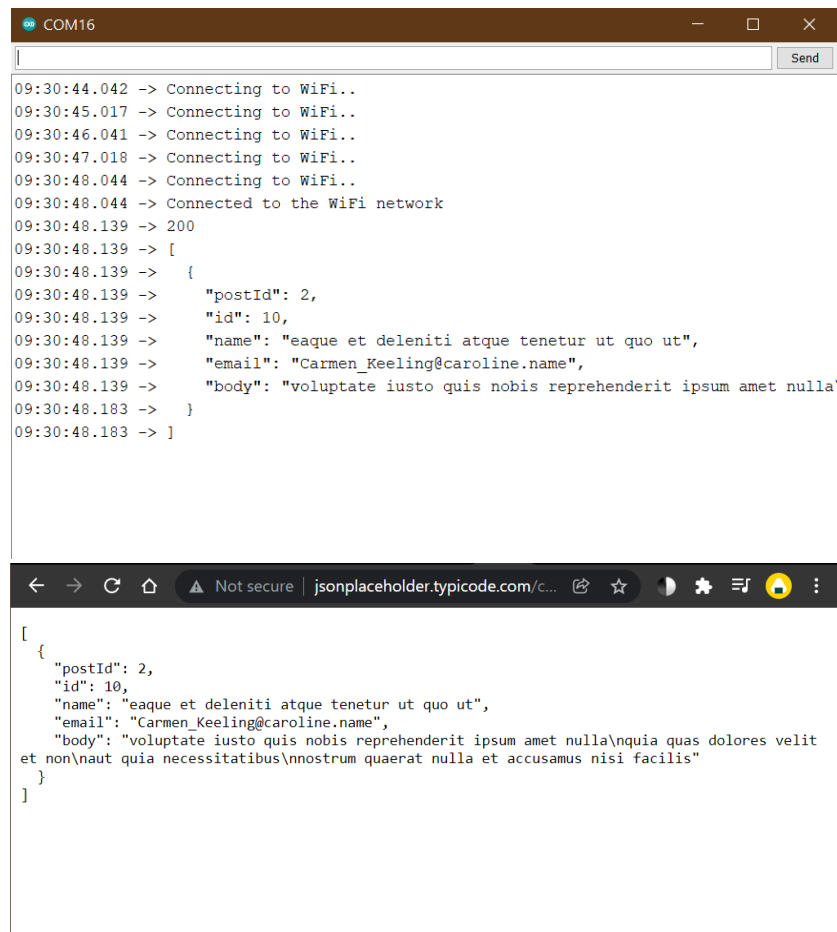


Gambar 3. Rangkaian Buzzer.



Gambar 4. Rangkaian Touch Sensor (led sensor menyala saat disentuh).

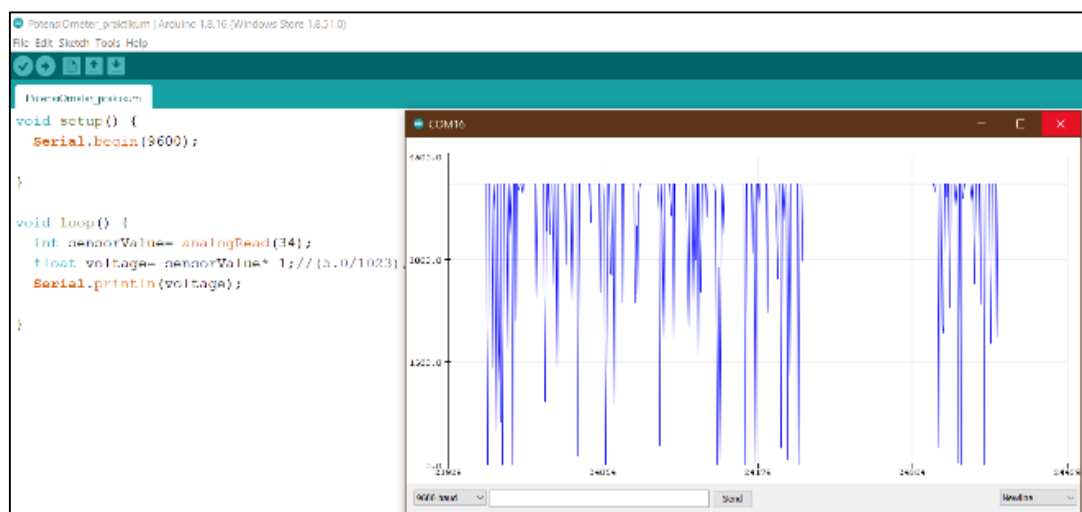
IV. Hasil serta Analisa



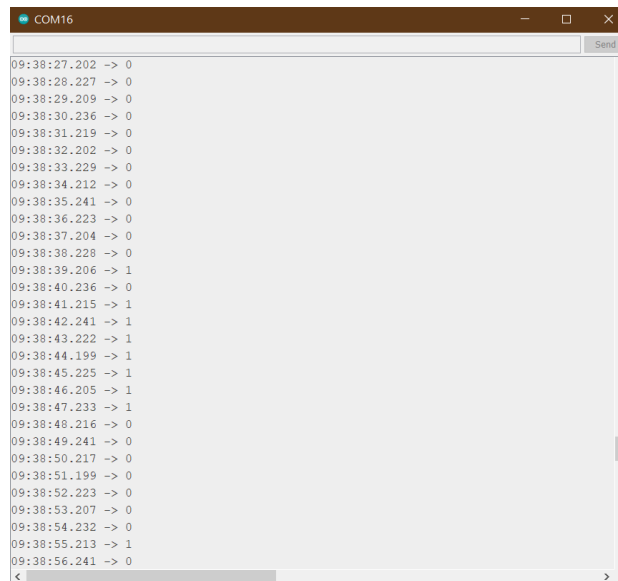
```
COM16
09:30:44.042 -> Connecting to WiFi..
09:30:45.017 -> Connecting to WiFi..
09:30:46.041 -> Connecting to WiFi..
09:30:47.018 -> Connecting to WiFi..
09:30:48.044 -> Connecting to WiFi..
09:30:48.044 -> Connected to the WiFi network
09:30:48.139 -> 200
09:30:48.139 -> [
09:30:48.139 ->   {
09:30:48.139 ->     "postId": 2,
09:30:48.139 ->     "id": 10,
09:30:48.139 ->     "name": "eaque et deleniti atque tenetur ut quo ut",
09:30:48.139 ->     "email": "Carmen_Keeling@caroline.name",
09:30:48.139 ->     "body": "voluptate iusto quis nobis reprehenderit ipsum amet nulla
09:30:48.183 ->   }
09:30:48.183 -> ]
```

```
[
  {
    "postId": 2,
    "id": 10,
    "name": "eaque et deleniti atque tenetur ut quo ut",
    "email": "Carmen_Keeling@caroline.name",
    "body": "voluptate iusto quis nobis reprehenderit ipsum amet nulla\nquia quas dolores velit
et non\naut quia necessitatibus\nnostrum quaerat nulla et accusamus nisi facilis"
  }
]
```

Gambar 4 & 5. Tampilan saat alat menjalankan program WiFi : Get Test.



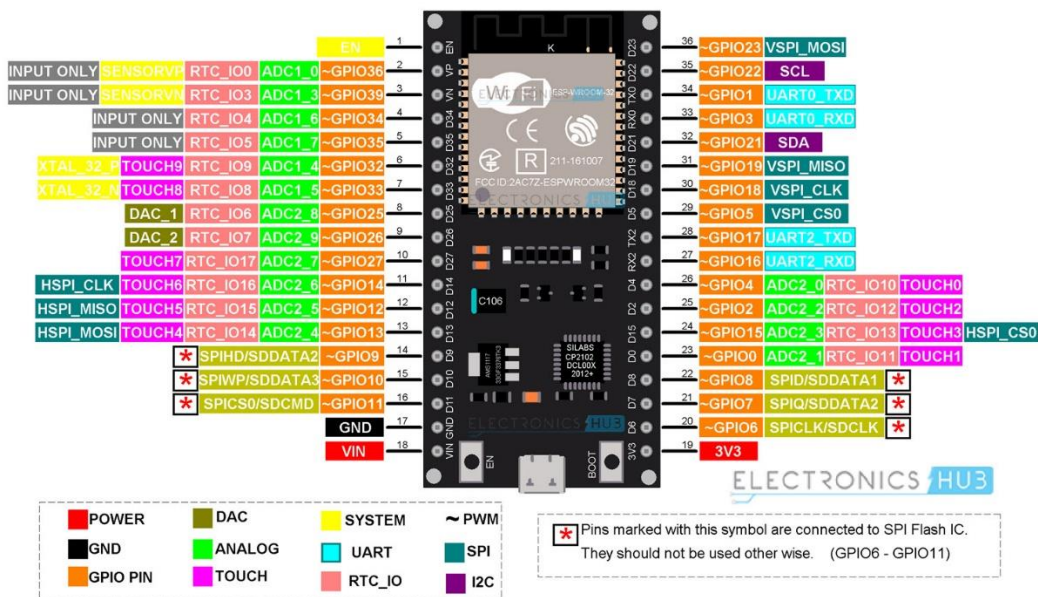
Gambar 6. Tampilan saat alat menjalankan program vibration test.



Gambar 7. Tampilan serial monitor saat alat menjalankan program touch sensor test(1=diseentuh,0=tidak disentuh).

ANALISA

❖ Pin ESP 32



Gambar 8. Pinout ESP32

Dengan rincian kemampuan setiap pinnya adalah sebagai berikut :

GPIO Pin	Pin on ESP32	Information
0	–	Pulled HIGH. Connected to BOOT Button
1	TX0	Do not use while TXing

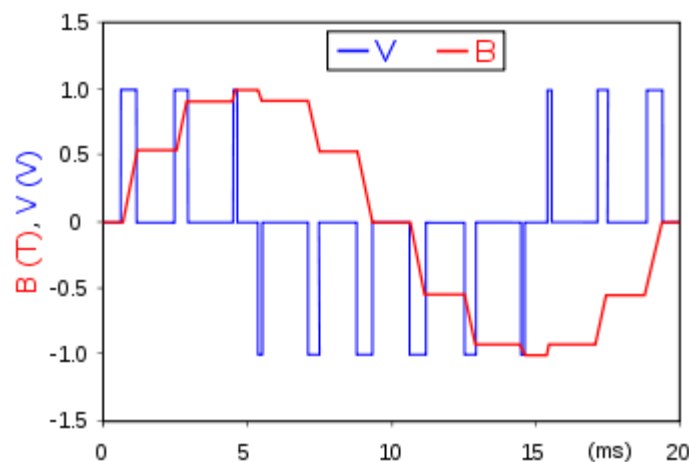
2	YES	Pulled LOW
3	RX0	Do not use while RXing
4	D4	Pulled LOW
5	D5	Pulled HIGH
12	D12	Pulled LOW. Boot fails if pulled HIGH as it sets voltage of internal voltage regulator.
13	D13	
14	D14	
15	D15	Pulled HIGH
16	RX2	UART2 RX
17	TX2	UART2 TX
18	D18	
19	D19	
21	D21	I2C SDA
22	D22	I2C SCL
23	D23	
25	D25	
26	D26	
27	D27	
32	D32	
33	D33	
34	D34	Digital Input Only. No Digital Output.
35	D35	Digital Input Only. No Digital Output.
36	YES	Digital Input Only. No Digital Output.
39	YES	Digital Input Only. No Digital Output.

❖ **analogRead(), digitalRead() dan ledcSetup()**

Untuk mengaktifkan peripheral GPIO maka kita membutuhkan argument-argumen diatas. `analogRead()` digunakan untuk membaca sebuah sinyal dari alat yang terhubung dengan GPIO tersebut secara analog. Sementara `digitalRead()` digunakan untuk membaca sebuah sinyal dari alat yang terhubung dengan GPIO tersebut secara digital (High / Low atau 1/0).

Kemudian untuk fungsi outputnya pada umumnya kita hanya bisa mengeluarkan sinyal digital, caranya dengan `digitalWrite()`.

Namun seiring berkembangnya zaman, maka saat ini kita juga bisa menggunakan `ledcSetup` atau `analogWrite` untuk mengeluarkan sebuah sinyal analog. Namun sinyal analog ini hanya dapat dikeluarkan pada Pin yang mendukung fungsi PWM (Pulse-width modulation) saja atau sebuah metode untuk mengurangi daya rata-rata yang dikirimkan oleh sinyal listrik, dengan secara efektif memotongnya menjadi bagian-bagian terpisah.



Gambar 9. Respon sinyal PWM (merah) dan sinyal sumber digital (biru). (Sumber : Wikipedia).

❖ **Sensor Getar SW1801P**

Sensor ini cukup bagus untuk menangkap getaran. Kelebihan sensor ini adalah ia dapat merekam sinyal getar baik secara digital maupun analog. Namun tetap saja untuk sensitivitasnya alat ini masih kurang teliti dibanding geophone.

Cara kerja sensor ini yaitu sensor inti harus dialiri oleh VCC dan ujung satunya dihubungkan ke ground. Kemudian aliran arus dalam sensor inti akan terganggu apabila terjadi getaran di sensor. Oleh karena itu nilai baca analog /digital saat sensor tidak bergetar adalah nilai maksimal atau HIGH, sementara saat sensor terganggu (bergetar), maka nilai bacanya menunjukkan nilai minimal atau LOW.

Referensi

<http://easycoding.tn/esp32/demos/code/>

<http://easycoding.tn/index.php/esp32/esp32-code/>

<https://www.electronicshub.org/esp32-pinout/>

https://en.wikipedia.org/wiki/Pulse-width_modulation