ダイナミックスピーカー

ダイナミックスピーカーとは、磁界内にボイスコイルを入れ、
音声出力電流によりコーン紙（振動版）を振動させることで
音を発生させる電子部品のこと。
以前使用した圧電ブザーと同様に PWM 出力を用いて発音する。
圧電ブザーと比べて歪みが少ない為、高音質かつ大音量の音を
奏でることができる。

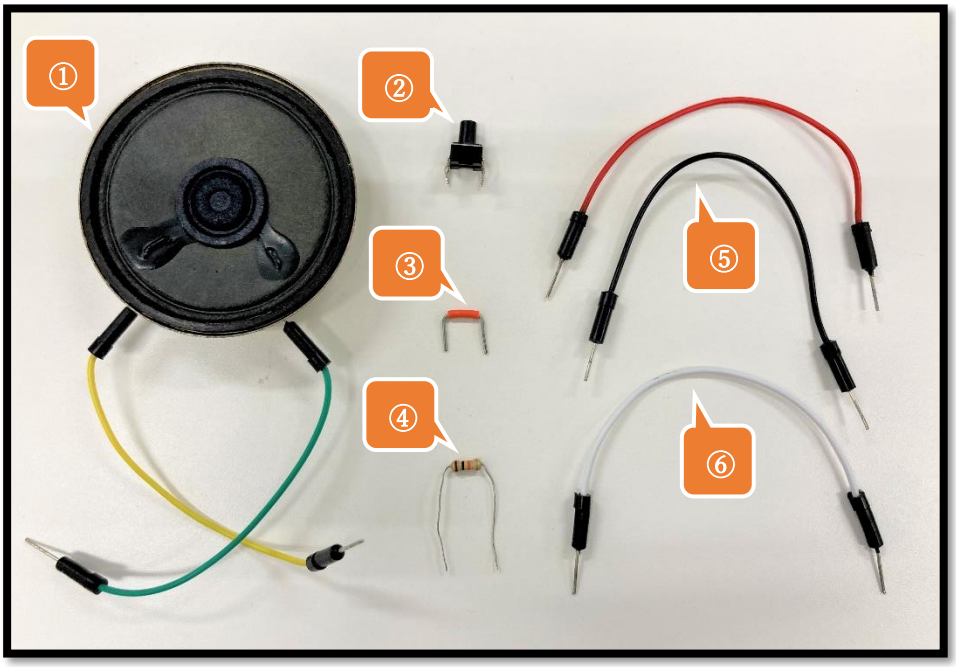
ESP32 での PWM 出力

ESP32 で PWM 出力を行う場合、まず出力したいピンに PWM チャネルを接続する。
その PWM チャネルに対して出力することで、接続された全てのピンに同時に PWM 出力が行える。
チャネルは 0~15 あり、どれを使用しても良いが基本的には先頭の 0 から割り当てを行う。

機材準備

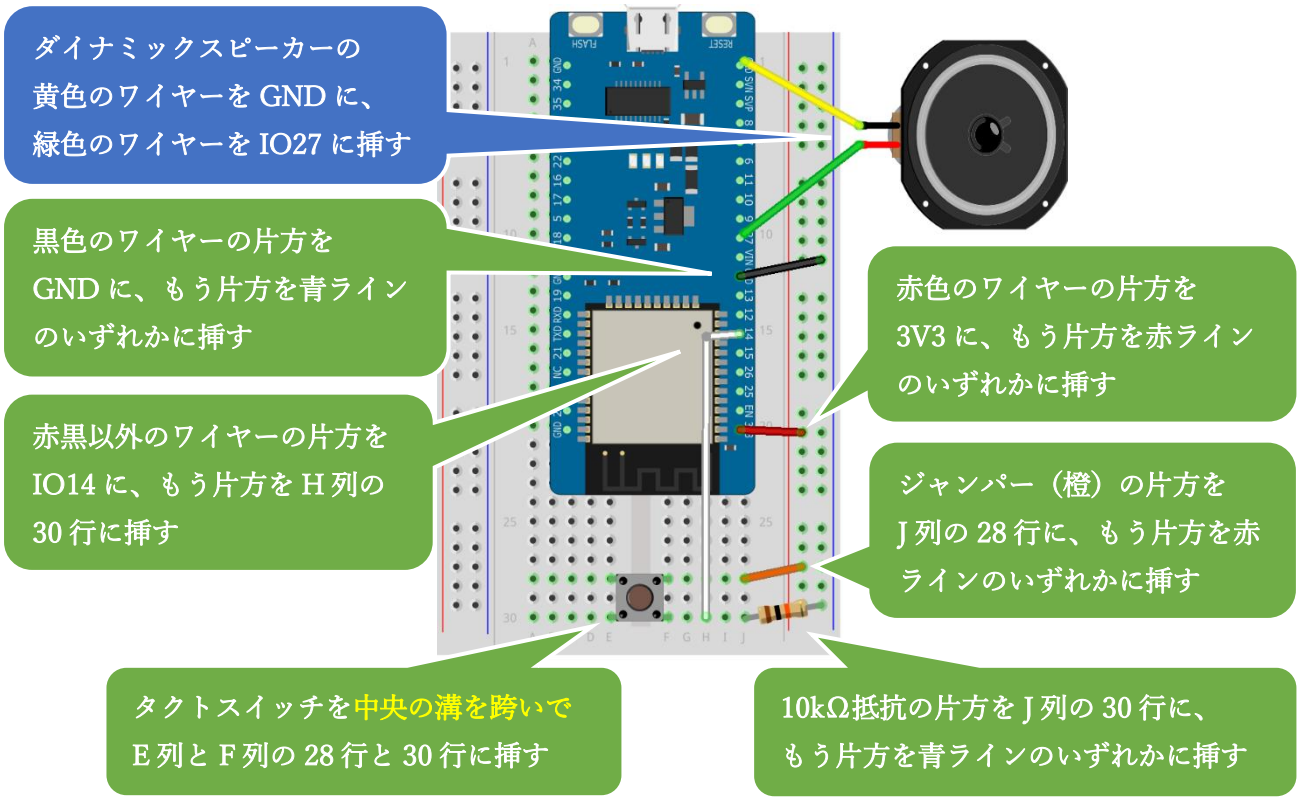
今回使用する機材を所定の位置から借りて持って来る。※授業終了時には必ず元の場所に戻すこと。

① ダイナミックスピーカー
② タクトスイッチ
③ ブレッドボード用ジャンパー（橙）
④ 10kΩ抵抗（茶・黒・橙・金）
⑤ ジャンパーワイヤー（赤と黒を 1 本ずつ）
⑥ ジャンパーワイヤー（赤・黒以外を 1 本）



練習（Speaker）

ESP32 に Micro-B USB ケーブルを差し込まずに以下の図のとおりに配線を行いなさい。



配線を行った後、Micro-B USB ケーブルを ESP32 と実習機に繋ぎ通電する。
その後、以下のサンプルコードを記述し、ESP32 に書き込みを行う。
起動時（setup 完了後）に、ダイナミックスピーカーからスーパーマリオのテーマが奏でられ、
スイッチを押した場合に 1 度だけコインの音が奏でられるか確かめる。
更に、スイッチを押したかどうかがシリアルモニタ上に出力されているか確認すること。

```
17:14:37.452 -> clk_drv:0x00,q_drv:0x00,d_drv:0x00,cs0_drv:0x00,hd_drv:0x00,wp_drv:0x00
17:14:37.452 -> mode:DIO, clock div:1
17:14:37.452 -> load:0x3fff0030,len:1324
17:14:37.452 -> ho 0 tail 12 room 4
17:14:37.452 -> load:0x40078000,len:13508
17:14:37.452 -> load:0x40080400,len:3604
17:14:37.452 -> entry 0x400805f0
17:14:39.186 -> プログラム開始
17:14:41.189 -> 押した
17:14:41.651 -> 離した
17:14:42.739 -> 押した
17:14:46.853 -> 離した
17:14:50.625 -> 押した
17:14:51.425 -> 離した
```

```
/*
  Speaker
  Date : 2022/01/01
  Author : IE1A 99 K.Murakami
*/

// ピン番号をマクロで定義
#define SW_PIN 14
#define SPKR_PIN 27

// スピーカーのPWM出力関連を定義
#define LEDC_CHANNEL_0     0    // LEDCのPWMチャネル0から15
#define LEDC_TIMER_13_BIT  13   // LEDCタイマーの精度13ビット
#define LEDC_BASE_FREQ     5000 // LEDCのベース周波数 5000Hz

// 使用する音の周波数を定義
#define C4 261.626  // ド
#define D4 293.665  // レ
#define E4 329.628  // ミ
#define F4 349.228  // ファ
#define G4 391.995  // ソ
#define A4 440.000  // ラ
#define B4 493.883  // シ
#define C5 523.251  // ド
#define D5 587.330  // レ
#define E5 659.255  // ミ
#define F5 698.456  // ファ
#define G5 783.991  // ソ
#define A5 880.000  // ラ
#define B5 987.767  // シ
#define C6 1046.502 // ド
#define D6 1174.659 // レ
#define E6 1318.510 // ミ
#define F6 1396.913 // ファ
#define G6 1567.982 // ソ
#define NONE 0      // 無音

#define WHOLE_NOTE 1000 // 全音符(1秒)
```

```
boolean swData;              // SW の状態を保持する変数
boolean swDataOld = false;  // 前回の SW の状態を保持する変数（初期値は false）

// マリオテーマ楽譜
double mario[] = {E5, E5, NONE, E5, NONE, C5, E5, NONE, G5, NONE, NONE, NONE, G4};

// 起動時に一度だけ呼び出されるメソッド
void setup() {
  Serial.begin(115200); // シリアル通信の転送レート（bps）を設定
  // ピンの入出力設定
  pinMode(SW_PIN, INPUT);

  // スピーカーチャネル設定
  ledcSetup(LEDC_CHANNEL_0, LEDC_BASE_FREQ, LEDC_TIMER_13_BIT);
  // スピーカーのチャネルにピンを接続
  ledcAttachPin(SPKR_PIN, LEDC_CHANNEL_0);

  // setup 完了後、マリオのテーマを奏でる(八分音符)
  melody(mario, sizeof(mario) / sizeof(double), WHOLE_NOTE / 8);
  Serial.println("プログラム開始");
}

// 演奏メソッド（楽譜, 楽譜の要素数, 音符の長さ）
void melody(double *score, int len,  int note) {
  for (int i = 0; i < len; i++) {
    ledcWriteTone(LEDC_CHANNEL_0, score[i]);
    delay(note);
  }
  ledcWriteTone(LEDC_CHANNEL_0, NONE);  //消音
}

// メインループメソッド
void loop() {
  // 現在の各 SW の状態を読み取る
  swData = digitalRead(SW_PIN);
  // 現在の SW の状態が、前回の SW の状態と違う場合
  if (swData != swDataOld) {
```

スピーカーを 0 チャネルに接続する
以後 0 チャネルに周波数を流して音を奏でる

Java とは違い要素数を取得する length メソッドが無い為、割り当てられているメモリ領域から要素数を算出する

ledcWriteTone メソッドにより指定したチャネルに周波数を設定することで音を奏でられる（チャネル, 周波数）

```
    if (swData) {
      Serial.println("押した");

      // コイン音を鳴らす
      ledcWriteTone(LEDC_CHANNEL_0, B5);
      delay(WHOLE_NOTE / 8);  // 八分音符
      ledcWriteTone(LEDC_CHANNEL_0, E6);
      delay(WHOLE_NOTE / 4);  // 四分音符
      ledcWriteTone(LEDC_CHANNEL_0, NONE);
    }else{
      Serial.println("離した");
    }
  }

  // 次回の為に現在の SW の状態を保存する
  swDataOld = swData;

  delay(10); // チャタリング防止
}
```

もしプログラムを記述し間違えて音が鳴り続けてしまった場合は、Micro-B USB ケーブルを抜くか、ダイナミックスピーカーのどちらか片方のワイヤーを一時的に抜くこと。

課題（SpeakerSecurity）

前述の練習を別名保存して超音波距離センサーから 1 秒間隔で物との距離を取得した後、その距離が**5cm 以内であればスピーカーから音楽が鳴り続ける**疑似的な防犯システムのプログラムに変更しなさい。
鳴らす音は本来警報音にすべきだが、不快指数が高い為今演習ではマリオの音楽にする。
音楽は**スイッチを押すと消音**できるようにして、**再度 5cm 以内に近づいた場合音が鳴る**ようにすること。
超音波距離センサーの回路を追加する必要がある為、追加で必要な部品を考えて所定の位置から持って来ること。配線を行う際は Micro-B USB ケーブルを ESP32 に差し込まずに行うこと。
また、以下の 2 つの条件を満たすこと。
① ダイナミックスピーカー及びスイッチの回路は練習のまま変更しないこと
② 音は以下の楽譜配列を使用すること

```
// ミス楽譜
double miss[] = {B4, F5, NONE, F5, F5, E5, NONE, D5, C5, NONE};
```

※課題が完成した人は ESP32 にプログラムを書き込んだ状態で講師を呼び、チェックしてもらうこと。