

TGS_m5stack_マニュアル

作者：柴田悠仁

初めに

このマニュアルは草刈りで使用しているセンサーに関するマニュアルです。

コードは

<https://github.com/CC-Circle/WWW/tree/main/m5stack>にあります。

使用機材

- [https://www.switch-science.com/products/9009?
_srsltid=AfmBOorZJ6iw5q2tYaw5JZhP6vUZ0X3Lw_zwi5eYIkrlr-
wwXf49U4kV](https://www.switch-science.com/products/9009?_srsltid=AfmBOorZJ6iw5q2tYaw5JZhP6vUZ0X3Lw_zwi5eYIkrlr-wwXf49U4kV)
- [https://www.switch-science.com/products/4048?
_pos=1&_sid=ede1914cf&_ss=r](https://www.switch-science.com/products/4048?_pos=1&_sid=ede1914cf&_ss=r) テスト環境
 - [https://www.switch-science.com/products/6558?
_pos=13&_sid=cd23764c5&_ss=r](https://www.switch-science.com/products/6558?_pos=13&_sid=cd23764c5&_ss=r) 実際の実装

開発環境

- Arduino IDE
- Unity

Unityとの連携

1. シリアルポートの確認

```
ls -l /dev/cu.usb*
```

m5sを接続し、コマンドラインで上記のコマンドを入力してください

```
k22065kk@k22065kknoMacBook-Air ~ % ls -l /dev/cu.usb*
crw-rw-rw- 1 root wheel 0x9000007 9 10 23:58 /dev/cu.usbmodem55910045011
k22065kk@k22065kknoMacBook-Air ~ %
```

上記のような結果が出力されるので、

```
// SerialHandler 14行目  
[SerializeField] private string portName = "任意のポート番号";
```

SerialHandler 14行目に/dev以下をコピペ。また、インスペクターも同時に変更

2. 接続の確認

ここからはUnity上での操作になります。

再生ボタンを押して、ゲームを開始してください。

ここでm5の電源が一度落ち再起動すれば接続完了しているはずです。

エラーの対処法

！もしここで電源が落ちなかった場合は以下のようなことが考えられます。

1. USBが接続されていない・接続場所が違う

物理的なエラーです。USBの接続状態を確認してください。

特に下のType-C電源はバッテリー用のものです。接続しても通信はできません。

2. シリアルポートが間違っている

入力したシリアルポートが間違っている可能性があります。

```
ls -l /dev/cu.usb*
```

上記のコマンドを叩いて、確認し直してください。

特に間に挟むUSBポートなどによってもポート番号は変わってしまうため、

前回と同じだからいいやといって確認しないのはあまりお勧めしません。

3. シリアルモニタの同時開き

筆者もよくやるエラーの一つです。Arduino IDEなどでシリアルモニタを同時に開いていると、

Unityで以下の警告文が出ます。

```
Failed to open serial port: Resource busy
```

この警告文の時は大体シリアルモニタが開いています。
閉じてください。

今回のプログラムの使用方法

参考にした記事は以下のものです。

M5StackでIMUユニットを使う

今回はArduinoIDEを用いて実験をします。

M5Stack公式から出されているUiFlow-IDEを用いても良いのですが、アイコンが使いづらい上処理も遅いので好きじゃありません。

https://zenn.dev/yutapon_juice/articles/c635e3c5490e10

M5StackでIMUユニットを使う

 yutapon_juice

 Zenn

今回は色々迷走した結果、ジャイロを採用しています。

本来ならジャイロは加算されていくため、計算を内部で行い平滑化する必要があります。

上記の記事のプログラムにお任せしています。

機能紹介

今回のプログラムは、接続を開始した瞬間に5秒のキャリブレーションを行なっています。

```
void setup() {
    M5.begin();
    M5.Power.begin();

    M5.IMU.Init();

    // ボタンのピン設定
    pinMode(26, INPUT);
    pinMode(36, INPUT);

    M5.Lcd.fillRect(0, 0, 128, 64, BLACK);
    M5.Lcd.setTextColor(WHITE, BLACK);
    M5.Lcd.setTextSize(2);
    delay(1);

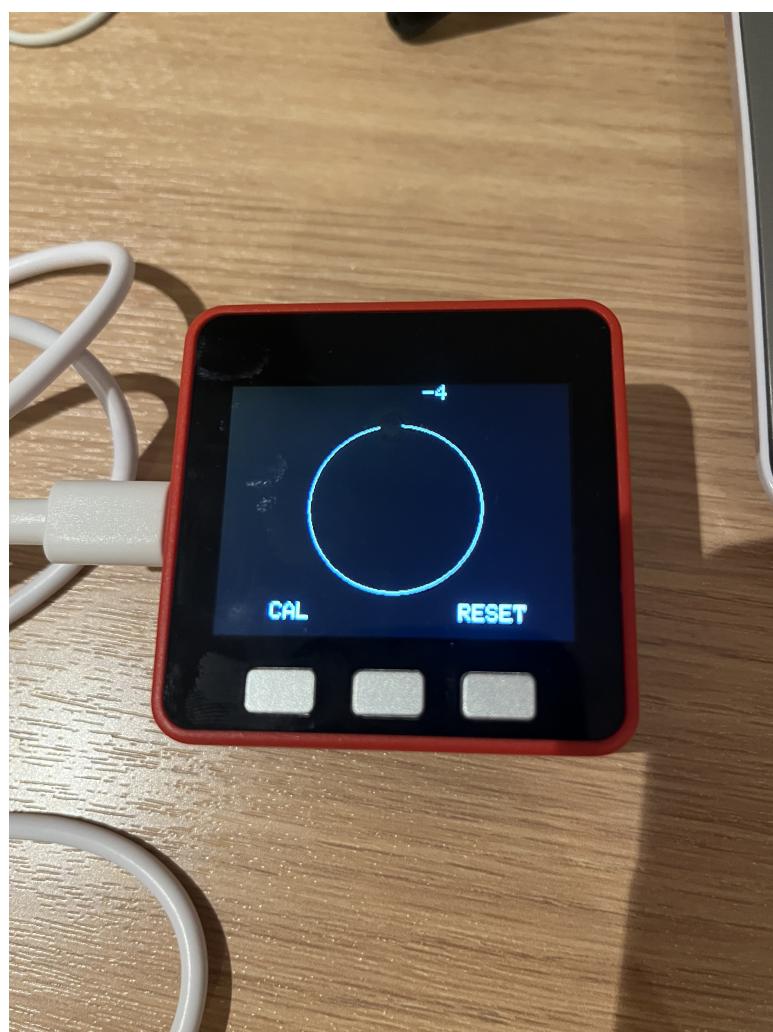
    // 自動的にキャリブレーションとMainを実行
    calibration();
```

```
Main();  
}
```

参考にしたものは少し特殊な書き方をしているため、今回はloop()での処理は行いません。

興味のある人は、記事のプログラムを読んでみてください。

ざっくり言うとMain()がloop()の代わりを行なっていると考えてください。



実際の実装画面

現在の角度を円の上に表示し、ジャイロを取得。

右のボタンが

キャリブレーション、左のボタンがリセットになります。

キャリブレーションをしっかり行えなかった場合、値が加算され続けてしまうため
キャリブレーションとリセットを押して初期化してください。

加算されてしまう理由が知りたければ、コードを読んでください。

(筆者もあんまりよくわかってません)

最後に

今回のプログラムは記事の内容をもとに実装しました。

読んでもらえれば理解できると思いますが、質問や疑問があれば柴田までお願ひします。

また、ジャイロの実装は色々な方法があるので試してみてください。