「みんなの M5Stack 入門」フォローサイト

株式会社リックテレコム／書籍出版部

（最終情報更新日：2021 年 8 月 30 日）

『みんなの M5Stack 入門』をご購入頂き、誠にありがとうございます。
M5Stack の内容は年々アップデートされます。ここでは本書の内容の補足やアップデート情報を提供してまいります。

<index>

●補足 1・・・・IMU（慣性計測ユニット）MPU6886 について

（2021 年 08 月 30 日更新）

●補足 1.　IMU（慣性計測ユニット）MPU6886 について

M5Stack Gray や Fire には IMU（慣性計測ユニット）が搭載されています。2019 年 8 月以前に出荷されたものには、MPU9250 というチップが使われていましたが、それ以降は MPU6886 という加速度・ジャイロセンサと BMM150 という磁気センサの組み合わせに変わりました。

書籍「みんなの M5Stack 入門」の最初の版では MPU9250 チップを使ったスケッチを紹介しましたが、MPU6886 についても補足します。

1．MPU9250 / MPU6886 どちらが搭載されているかを調べる

外見では MPU9250 / MPU6886 、どちらの IMU チップが搭載されているかは区別がつきませんが、本書の p.130 にあるスケッチ（IMUcheck.ino）を動かすことで判別できます。IMUcheck.ino は次のアドレスに公開しました。

https://github.com/AmbientData/M5Stack4everyone/tree/master/A1/IMUcheck

スケッチをビルドして動かすと、搭載されている IMU チップが LCD に表示されます。

2．IMU にアクセスする

本書を書いた時点では、IMU にアクセスするには utility/MPU9250.h というヘッダファイルをインクルードし、MPU9250 というオブジェクトを作って MPU9250 にアクセスしていました。

その後、IMU のアクセス方法が変更され、M5Stack.h をインクルードする前に、搭載されている IMU チップを定義すると、M5.IMU というオブジェクトで MPU9250 / MPU6886 どちらが搭載されていても、共通に IMU にアクセスできるようになりました。

スケッチの先頭で搭載されている IMU チップ名を宣言します。宣言は #include <M5Stack.h> よりも前に行う必要があります。MPU6886 が搭載されている場合は

M5STACK¥_MPU6886 を宣言し、MPU9250 が搭載されている場合は
M5STACK¥_MPU9250 を宣言します。

```
#define M5STACK_MPU6886
#include <M5Stack.h>
```

IMU チップの初期化は setup 関数の中で次のように行います。

```
setup() {
    M5.IMU.Init();  // IMU チップを初期化する
}
```

次のようにすると IMU チップから加速度データが取得できます。

```
float accX = 0.0F;
float accY = 0.0F;
float accZ = 0.0F;

void loop() {
    M5.IMU.getAccelData(&accX,&accY,&accZ);  // IMU から加速度データを取得する
}
```

以前は IMU チップのステータスレジスタを読み出し、データが取得可能かどうか調べてからデータを取得していましたが、新しい方法ではとても簡単にデータが取得できるようになりました。スケッチは次のアドレスに公開しました。

https://github.com/AmbientData/M5Stack4everyone/tree/master/A1/accel

加速度は x 軸、y 軸、z 軸の 3 方向のデータで表されます。M5Stack のケースと x 軸、y 軸、z 軸の位置関係は MPU9250 と MPU6886 で共通です。本書 p.118 の図 5.1 をご覧ください。

スケッチをビルドして動かし、M5Stack Gray を机の上におくと、x 軸と y 軸の加速度値としてほぼ 0G、z 軸の加速度値として重力加速度を示す 1G が表示されます。M5Stack Gray を傾けると値が変化するのが確認できます。

ジャイロデータは次のようにして取得します。

```
float gyroX = 0.0F;
float gyroY = 0.0F;
float gyroZ = 0.0F;

void loop() {
    M5.IMU.getAccelData(&accX,&accY,&accZ);
}
```

また IMU ライブラリには、加速度とジャイロデータを使って M5Stack の姿勢を計算する機能が提供されています。M5Stack の姿勢とは、次の図 1 のように x 軸、y 軸、z 軸方向の角度を示すものです。

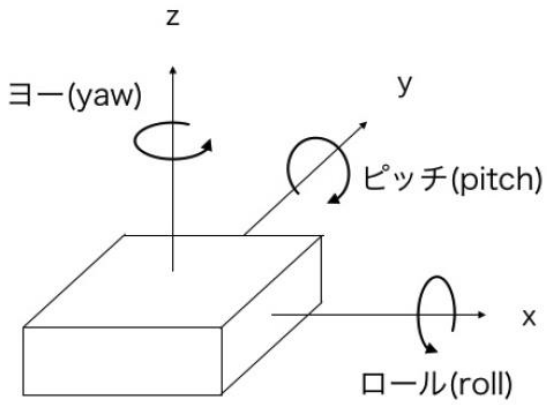

図 1　M5Stack の姿勢

```
float pitch = 0.0F;
float roll  = 0.0F;
```

```
float yaw   = 0.0F;

void loop() {
    M5.IMU.getAhrsData(&pitch,&roll,&yaw);
}
```

ちなみに、関数名の getAhrsData の Ahrs は Attitude and Heading Reference System の略で、姿勢と方位を計算するシステムのことです。

3．磁気センサ BMM150 にアクセスする

IMU チップにアクセスするライブラリが簡単で分かりやすくなったのに比べると、磁気センサ BMM150 へのアクセスはやや複雑です。

BMM150 へのアクセスは次のように行います。まず、Arduino IDE のツールメニューから「ライブラリを管理...」を選択して、ライブラリマネージャを立ち上げ、検索窓に「M5¥_BMM150」と入力し、表示された「M5¥_BMM150」ライブラリをインストールします。

ライブラリをインストールすると、スケッチ例も一緒にインストールされます。Arduino IDE のファイルメニューの「スケッチ例」→「M5¥_BMM150」→「bmm150」がスケッチ例です。「スケッチ例」→「M5Stack」→「Basics」→「bmm150」というファイルもありますが、そちらは古いようですので、「M5¥_BMM150」の方を見てください。

コメントが中国語で書かれていたりしますが、M5Stack 社が提供した公式のスケッチ例です。次のような関数が提供されています。

* bmm150¥_initialization()　　・・・BMM150 の初期化
* bmm150¥_offset¥_save()　　・・・オフセット値を記録する
* bmm150¥_offset¥_load()　　・・・オフセット値を読み出す
* bmm150¥_calibrate()　　・・・キャリブレーションを行う
* bmm150¥_read¥_mag¥_data()　　・・・磁気センサデータを読み出す

スケッチ例は LCD への表示などで少し複雑ですが、BMM150 に関連する部分を抜き出すと次のような流れです。

```
void setup() {
    …
    bmm150_initialization();  // BMM150 の初期化
    bmm150_offset_load();  // オフセット値を読み出す
}

void loop() {
    …
    bmm150_read_mag_data(&dev);  // 磁気センサデータを読み出す

    if(M5.BtnA.wasPressed()){  // A ボタンが押されたら
        bmm150_calibrate(10000);  // 10 秒間キャリブレーションを行う
    }
}
```

キャリブレーションは、引数で指定した時間（スケッチ例では 10 秒間）M5Stack を水平にしたまま回転させます。詳しくは本書 p.134 の図 5.13 をご覧ください。キャリブレーションが終わると、M5Stack を北に向けたときに LCD に表示された HEAD Angle が 0 を示すようになります。

BMM150 にアクセスするには、このスケッチ例を元にするとよいでしょう。この M5¥_BMM150 ライブラリは公式のものですが、まだバージョン番号も 0.0.1 で、M5Stack ライブラリにも統合されていません。今後、インターフェースも分かりやすく整理され、M5Stack ライブラリに統合されることを期待します。

［index に戻る］

