

NS-RX231 のソフトウェアガイド

〈MPU-6050 編〉

目次

1 プロジェクトのインポート.....	2
2 概要.....	3
3 ソースコード.....	4
4 デバッグ	5
5 実行	6
6 回路図	8
7 追加	9

1 プロジェクトのインポート

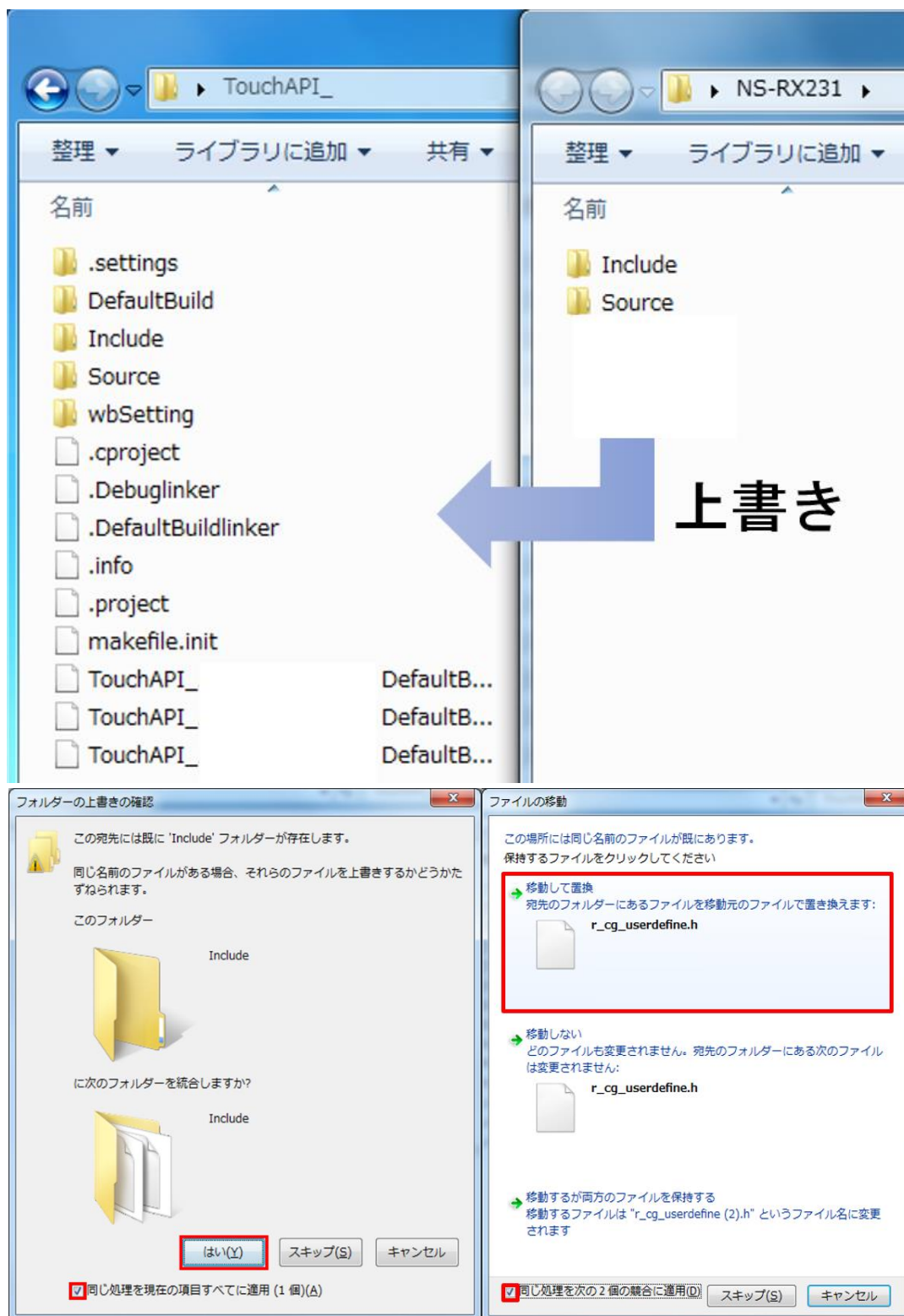


図 1-1 ソースファイルを上書き

添付されたソースファイルをWorkbench6 First step guideに従って、ウィザードで作成したプロジェクトに上書きした後、e2studioを実行します。

2 概要

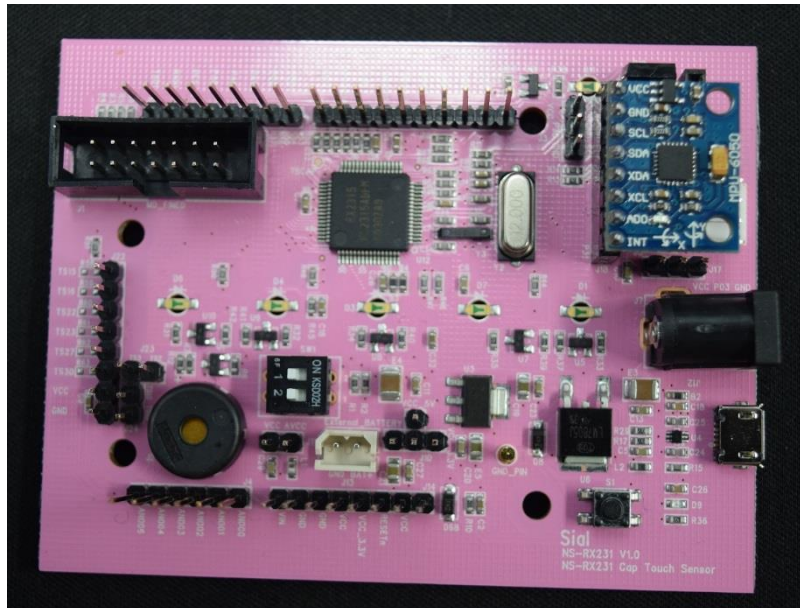


図 2-1 MPU-6050を装着したNS-RX231

MPU-6050 は 3 軸ジャイロセンサー、3 軸加速度センサー、温度センサーのデータを融合して姿勢制御システムに使用することができます。本ガイドでは、NS-RX231 ボードを利用して MPU-6050 と I²C 通信をする方法を説明します。

MPU-6050 は I²C 通信を使用してデータを送受信できます。I²C 通信では、マスタは SCL と SDA の 2 本の信号だけを利用して、複数のスレーブデバイスを制御することができます。しかし SPI などの他の通信と比較してみると、速度が遅いという欠点があります。

SCL はマスタとスレーブ間の同期のためのクロックで、そのクロックに同期して SDA ピンで通信データをやり取りします。

MPU-6050 の I²C バスでのアドレスは、AD0 (Address0) ピンによって決まります。AD0 はスレーブアドレスの最下位ビットを指定し、ハイのとき、1、ロウのとき 0 になります。アドレスが 7 ビットである理由は、R/ W ビットを含めて、8 ビットにするためです。R/ W ビットに関しては 7 追加 部分を参照してください。

I ² C ADDRESS	AD0 = 0	1101000
	AD0 = 1	1101001

図 2-2 AD0ピンの説明

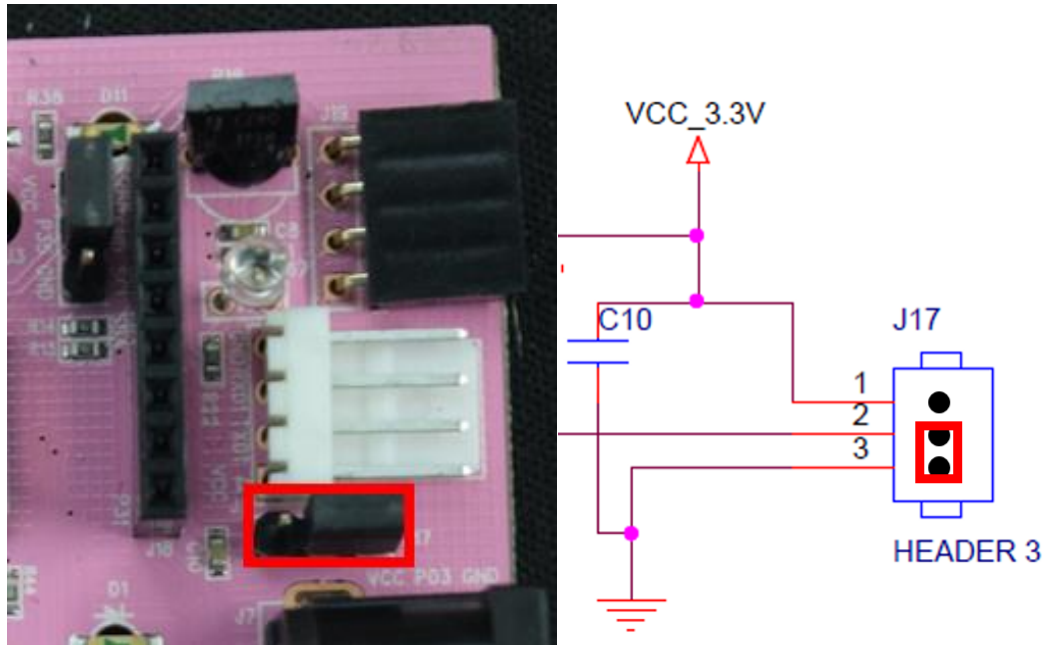


図 2-3 ジャンパー接続の設定

上の写真のように、デフォルトでは J17 のジャンパーは 2-3 を接続し、ADO を GND に接続しています。もしアドレス値を 1101001 に変更したい場合は、J17 のジャンパー接続を 1-2 に変更してください。

3 ソースコード

```
void request(){
    int i,j;
    uint8_t chrArr[7][14] =
    {"AxisX : ","AxisY : ","AxisZ : ",
    "Temperature : ","GyroX : ","GyroY : ","GyroZ : "};
    uint8_t rx_buf[14] = {0,};

    R_RIIC0_Master_Receive(0x68, 0x3B, rx_buf, 14);

    for(i = 0 ; i < 7 ; i++){
        j = i*2;
        scil_print(chrArr[i],i == 3 ? 14 : 8);
        sendAscii((rx_buf[j] << 8) | rx_buf[j+1]);
    }
    scil_println("-----", 20);
}

void sendAscii(int value){
    int count, i = 0;
    int temp = value;
    uint8_t * arr = 0;
    if(value != 0){
        while( temp > 0 )
        {
            temp /= 10;
            i++;
        }
        count = i;
        while( i > 0 )
        {
            arr[--i] = (value % 10) + 48;
            value /= 10;
        }
        scil_println(arr,count);
    }
}
```

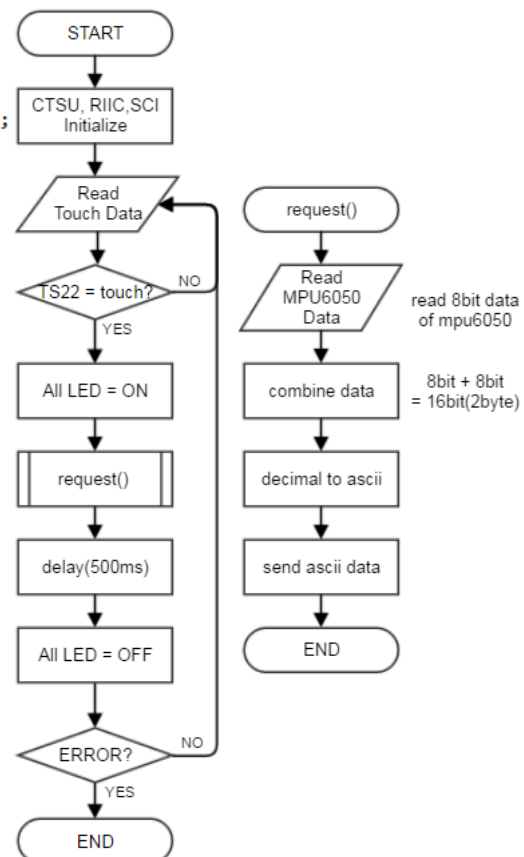


図 3-1 ソースコードとフローチャート

4 デバッグ

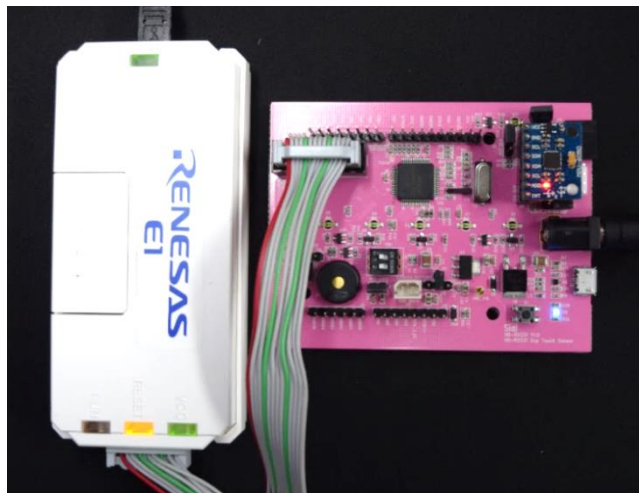


図 4-1 NS-RX231に電源アダプタとE1デバッガを接続した様子。

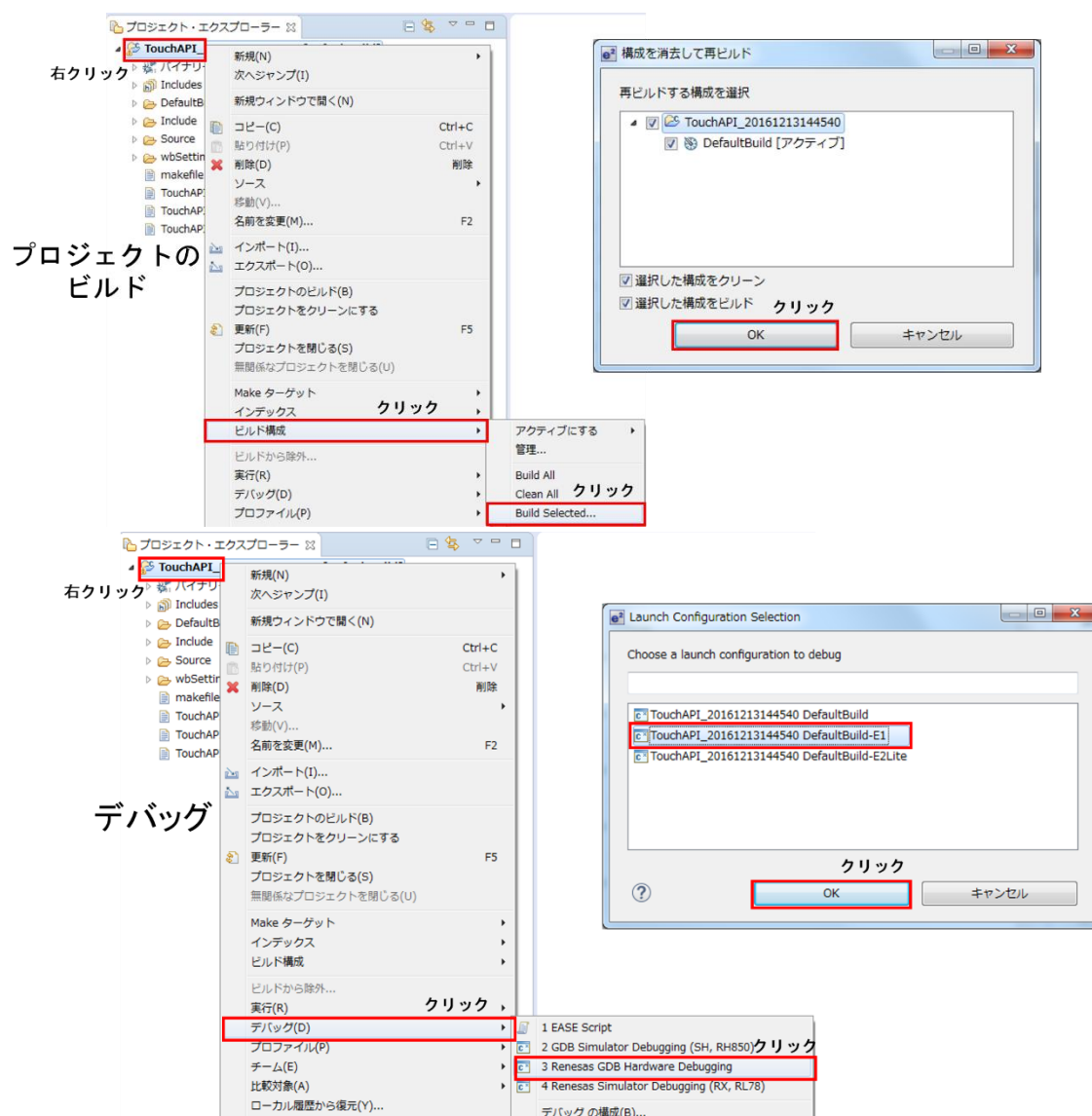


図 4-2 プロジェクトのビルドとデバッグ

5 実行

データを受信するためには、MPU-6050の内部レジスタアドレス107（0x6B）番地のSLEEPビットを0にしてMPU-6050を有効にする必要があります。

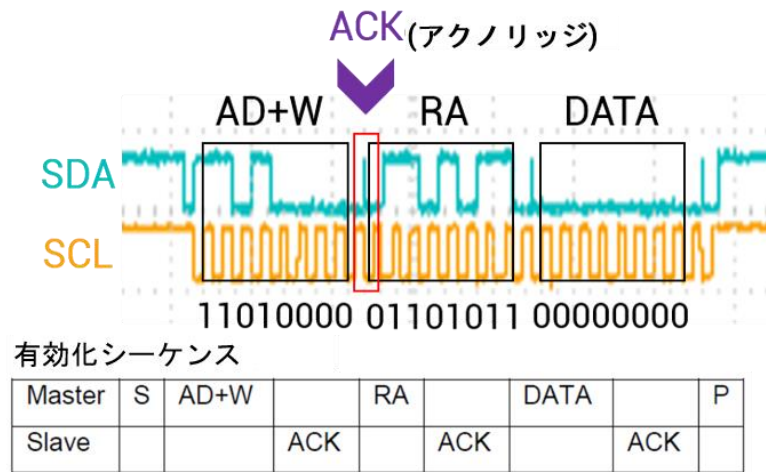


図 5-1 MPU-6050の有効化シーケンス

ACK は確認応答信号として、デバイスからのデータを正しく受信したことを示します。
信号の大きさは異なる場合があります。

MPU-6050を有効にした後に、データの受信を開始します。

3B	59	ACCEL_XOUT_H
3C	60	ACCEL_XOUT_L

図 5-2 MPU-6050の内部レジスタアドレス例

MPU-6050から加速度X、Y、Z/温度/ジャイロX、Y、Zの合計7つのデータを読み出すことができます。それぞれのデータは、2バイト（16ビット）で、データを受信するときには、上位8ビットである_Hと下位8ビットである_Lの二つに分かれているので、バッファに受信データを入れた後、OR演算でまとめます。

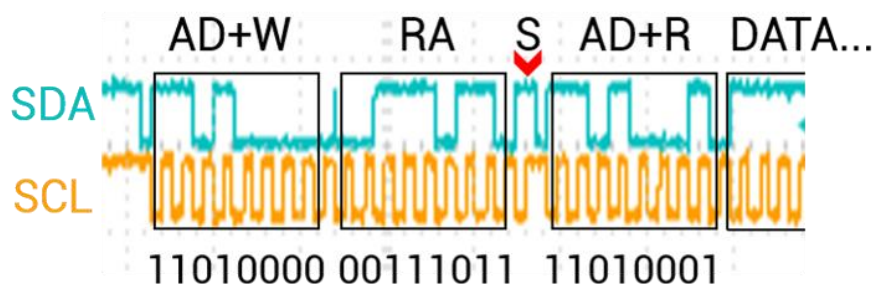


図 5-3 データの連続読み出しシーケンス

本ソースコードでは、NS-RX231でMPU-6050のデータを受信した後、UART通信でデータを送信することで、ピンヘッダーのTXD1、RXD1ピンに接続したBluetoothモジュールからデータを転送します。

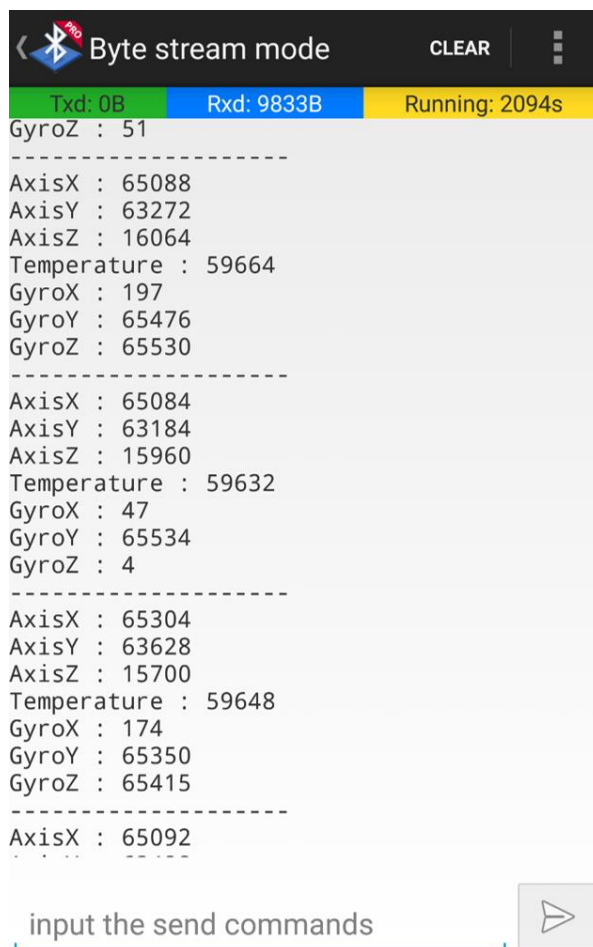
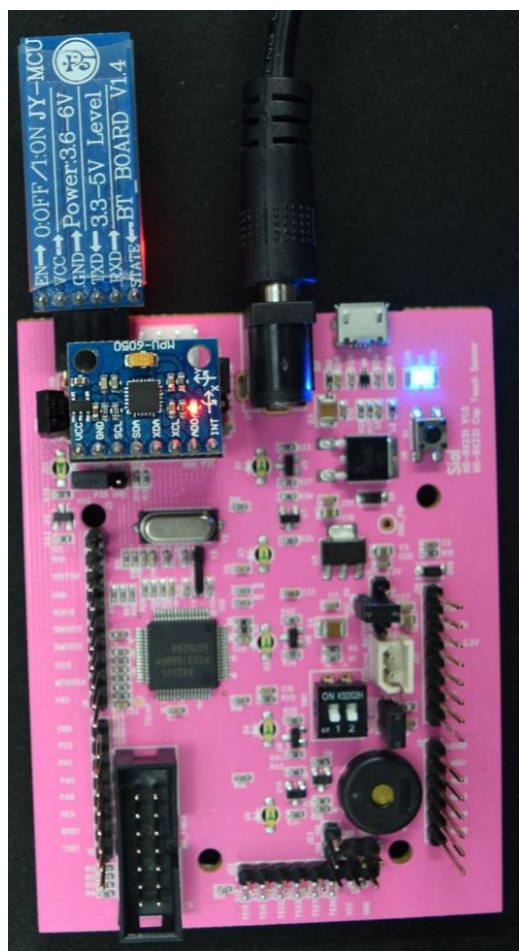


図 5-4 MPU-6050の動作

Bluetoothモジュールを差し込んで、携帯電話と接続してMPU-6050のデータを受けてみました。

6 回路図

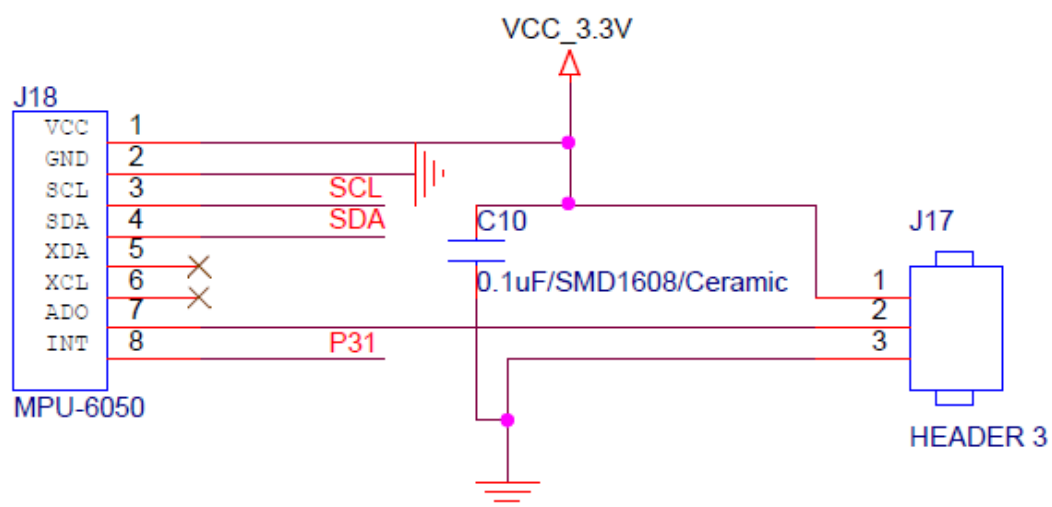


図 6-1 NS-RX231 MPU-6050接続端子部の回路図

7 追加

連続読み出しシーケンス

Master	S	AD+W		RA		S	AD+R			ACK		NACK	P
Slave			ACK		ACK			ACK	DATA		DATA		

図 7-1 データの連続読み出しシーケンス

上に示す複数データの連続読み出しシーケンスを見ると、マスタは通信開始を知らせるスタートコンディション（S）の後、デバイスのアドレス値（AD+ W）を送信し、さらに読み出したいレジスタのアドレス値（RA）を送信します。そして、通信方向を切り替えるためのリスタートコンディションとして、再びスタートコンディション（S）を出力します。

シンボル	意味
S	スタートコンディション：SCLがハイの状態ですDAが立ち下がる
AD	スレーブのI2Cバスでのアドレス
W	書き込み (0)
R	読み出し (1)
ACK	Acknowledge: SCLの9クロック目の立ち上がり時にSDAがロウ
NACK	Not-Acknowledge: SCLの9クロック目の立ち上がり時にSDAがハイ
RA	MPU-60X0内部のレジスタのアドレス
DATA	送受信するデータ
P	ストップコンディション: SCLがハイの状態ですDAが立ち上がる

図 7-2 フローチャート信号の説明

R/ Wビットは、AD（7ビット）+ R/ W（1ビット）と組み合わせられた8ビットのデータとして転送されます。

```

AD+W  RIIC0.ICDRT = (uint8_t)(g_riic0_slave_address << 1);
      g_riic0_state = _04_IIC_MASTER_SENDS_ADR_10B;

AD+R  RIIC0.ICDRT = (uint8_t)((g_riic0_slave_address << 1) | 0x0001U);
      g_riic0_state = _08_IIC_MASTER_RECEIVES_START;

```

図 7-3 AD+W & AD+R