**2020年度　システムエンジニア基礎研修資料**

**加速度センサ　演習2の演習問題の模範解答**

//================================================================//

// AE-BMX055 Arduino UNO //

// VCC +5V //

// GND GND //

// SDA A4(SDA) //

// SCL A5(SCL) //

// //

// (JP6,JP4,JP5はショートした状態) //

// http://akizukidenshi.com/catalog/g/gK-13010/ //

//================================================================//

#include<Wire.h>

// BMX055　加速度センサのI2Cアドレス

#define Addr\_Accl 0x19 // (JP1,JP2,JP3 = Openの時)

// BMX055　ジャイロセンサのI2Cアドレス

#define Addr\_Gyro 0x69 // (JP1,JP2,JP3 = Openの時)

// BMX055　磁気センサのI2Cアドレス

#define Addr\_Mag 0x13 // (JP1,JP2,JP3 = Openの時)

// 加速度/ジャイロセンサの調整完了のLED出力PIN

#define PIN\_STB\_LED (16)

// 衝撃検知時のLED出力PIN

#define PIN\_SHOCK\_LED (17)

// センサーの値を保存するグローバル関数

float xAccl = 0.00;

float yAccl = 0.00;

float zAccl = 0.00;

float xGyro = 0.00;

float yGyro = 0.00;

float zGyro = 0.00;

int xMag = 0;

int yMag = 0;

int zMag = 0;

typedef enum flag{

UNSET,

SET,

}en\_flag;

//初期化フラグ

typedef enum {

INIT\_START, // 開

INIT\_END // 閉

} enInitflg;

//衝撃フラグ

typedef enum {

NO\_INPACT, //衝撃なし

INPACT //衝撃あり

} enInpactflg;

// 平均値の測定を開始するフラグ

en\_flag enMsrStartFlg = UNSET;

// LEDの点灯を開始するフラグ

en\_flag enLedOnFlg = UNSET;

int count = 0;

enInpactflg en\_inpactFlg = NO\_INPACT;

int loop\_cnt = 0;

unsigned int tskLo\_cnt;

enInitflg en\_Initflg = INIT\_START;

unsigned int i\_count = 0;

//2(加速度センサとジャイロセンサ)×3(x, y, z)分の要素を確保する

float msrSum[2][3] = {{0,0,0},{0,0,0}};

float msrAve[2][3] = {{0,0,0},{0,0,0}};

float msrResult[2][3] = {{0,0,0},{0,0,0}};

void setup()

{

// Wire(Arduino-I2C)の初期化

Wire.begin();

// デバック用シリアル通信は9600bps

Serial.begin(9600);

//BMX055 初期化

BMX055\_Init();

//LED初期化

pinMode(PIN\_SHOCK\_LED, OUTPUT);

pinMode(PIN\_STB\_LED, OUTPUT);

tskLo\_cnt = 0;

en\_Initflg = INIT\_START;

delay(300);

}

void loop(){

if (tskLo\_cnt == 20 ){

taskLo();

tskLo\_cnt = 0;

}

if (en\_Initflg != INIT\_START){

if ((xGyro > 10) || (xGyro > 10) || (xGyro > 10)){

Serial.println("The inpact Occer!!.");

en\_inpactFlg = INPACT;

}

if(en\_inpactFlg == INPACT){

loop\_cnt++;

BlinkLED(loop\_cnt);

if(loop\_cnt > 20){

en\_inpactFlg = NO\_INPACT;

en\_Initflg = INIT\_END;

i\_count = 0;

}

}else{

loop\_cnt = 0;

}

}

tskLo\_cnt++;

delay(50);

}

void taskLo()

{

Serial.println("--------------------------------------");

//BMX055 加速度の読み取り

BMX055\_Accl();

msrResult[0][0]=xAccl-msrAve[0][0];

msrResult[0][1]=yAccl-msrAve[0][1];

msrResult[0][2]=zAccl-msrAve[0][2];

Serial.print("Accl= ");

Serial.print(msrResult[0][0]);

Serial.print(",");

Serial.print(msrResult[0][1]);

Serial.print(",");

Serial.print(msrResult[0][2]);

Serial.println("");

//BMX055 ジャイロの読み取り

BMX055\_Gyro();

msrResult[1][0]=xGyro-msrAve[1][0];

msrResult[1][1]=yGyro-msrAve[1][1];

msrResult[1][2]=zGyro-msrAve[1][2];

Serial.print("Gyro= ");

Serial.print(msrResult[1][0]);

Serial.print(",");

Serial.print(msrResult[1][1]);

Serial.print(",");

Serial.print(msrResult[1][2]);

Serial.println("");

//BMX055 磁気の読み取り

BMX055\_Mag();

Serial.print("Mag= ");

Serial.print(xMag);

Serial.print(",");

Serial.print(yMag);

Serial.print(",");

Serial.print(zMag);

Serial.println("");

msr\_jdgStartEnd();

if ( enMsrStartFlg == SET){

//加速度

msrSum[0][0] += xAccl;

msrSum[0][1] += yAccl;

msrSum[0][2] += zAccl;

// 角速度

msrSum[1][0] += xGyro;

msrSum[1][1] += yGyro;

msrSum[1][2] += zGyro;

count += 1;

}

if ( enLedOnFlg == SET){

digitalWrite(PIN\_STB\_LED, HIGH);

}

else{

digitalWrite(PIN\_STB\_LED, LOW);

}

delay(1000);

}

//=====================================================================================//

void BMX055\_Init()

{

//------------------------------------------------------------//

Wire.beginTransmission(Addr\_Accl);

Wire.write(0x0F); // Select PMU\_Range register

Wire.write(0x03); // Range = +/- 2g

Wire.endTransmission();

delay(100);

//------------------------------------------------------------//

Wire.beginTransmission(Addr\_Accl);

Wire.write(0x10); // Select PMU\_BW register

Wire.write(0x08); // Bandwidth = 7.81 Hz

Wire.endTransmission();

delay(100);

//------------------------------------------------------------//

Wire.beginTransmission(Addr\_Accl);

Wire.write(0x11); // Select PMU\_LPW register

Wire.write(0x00); // Normal mode, Sleep duration = 0.5ms

Wire.endTransmission();

delay(100);

//------------------------------------------------------------//

Wire.beginTransmission(Addr\_Gyro);

Wire.write(0x0F); // Select Range register

Wire.write(0x04); // Full scale = +/- 125 degree/s

Wire.endTransmission();

delay(100);

//------------------------------------------------------------//

Wire.beginTransmission(Addr\_Gyro);

Wire.write(0x10); // Select Bandwidth register

Wire.write(0x07); // ODR = 100 Hz

Wire.endTransmission();

delay(100);

//------------------------------------------------------------//

Wire.beginTransmission(Addr\_Gyro);

Wire.write(0x11); // Select LPM1 register

Wire.write(0x00); // Normal mode, Sleep duration = 2ms

Wire.endTransmission();

delay(100);

//------------------------------------------------------------//

Wire.beginTransmission(Addr\_Mag);

Wire.write(0x4B); // Select Mag register

Wire.write(0x83); // Soft reset

Wire.endTransmission();

delay(100);

//------------------------------------------------------------//

Wire.beginTransmission(Addr\_Mag);

Wire.write(0x4B); // Select Mag register

Wire.write(0x01); // Soft reset

Wire.endTransmission();

delay(100);

//------------------------------------------------------------//

Wire.beginTransmission(Addr\_Mag);

Wire.write(0x4C); // Select Mag register

Wire.write(0x00); // Normal Mode, ODR = 10 Hz

Wire.endTransmission();

//------------------------------------------------------------//

Wire.beginTransmission(Addr\_Mag);

Wire.write(0x4E); // Select Mag register

Wire.write(0x84); // X, Y, Z-Axis enabled

Wire.endTransmission();

//------------------------------------------------------------//

Wire.beginTransmission(Addr\_Mag);

Wire.write(0x51); // Select Mag register

Wire.write(0x04); // No. of Repetitions for X-Y Axis = 9

Wire.endTransmission();

//------------------------------------------------------------//

Wire.beginTransmission(Addr\_Mag);

Wire.write(0x52); // Select Mag register

Wire.write(0x16); // No. of Repetitions for Z-Axis = 15

Wire.endTransmission();

}

//=====================================================================================//

void BMX055\_Accl()

{

int data[6];

for (int i = 0; i < 6; i++)

{

Wire.beginTransmission(Addr\_Accl);

Wire.write((2 + i));// Select data register

Wire.endTransmission();

Wire.requestFrom(Addr\_Accl, 1);// Request 1 byte of data

// Read 6 bytes of data

// xAccl lsb, xAccl msb, yAccl lsb, yAccl msb, zAccl lsb, zAccl msb

if (Wire.available() == 1)

data[i] = Wire.read();

}

// Convert the data to 12-bits

xAccl = ((data[1] \* 256) + (data[0] & 0xF0)) / 16;

if (xAccl > 2047) xAccl -= 4096;

yAccl = ((data[3] \* 256) + (data[2] & 0xF0)) / 16;

if (yAccl > 2047) yAccl -= 4096;

zAccl = ((data[5] \* 256) + (data[4] & 0xF0)) / 16;

if (zAccl > 2047) zAccl -= 4096;

xAccl = xAccl \* 0.0098; // renge +-2g

yAccl = yAccl \* 0.0098; // renge +-2g

zAccl = zAccl \* 0.0098; // renge +-2g

}

//=====================================================================================//

void BMX055\_Gyro()

{

int data[6];

for (int i = 0; i < 6; i++)

{

Wire.beginTransmission(Addr\_Gyro);

Wire.write((2 + i)); // Select data register

Wire.endTransmission();

Wire.requestFrom(Addr\_Gyro, 1); // Request 1 byte of data

// Read 6 bytes of data

// xGyro lsb, xGyro msb, yGyro lsb, yGyro msb, zGyro lsb, zGyro msb

if (Wire.available() == 1)

data[i] = Wire.read();

}

// Convert the data

xGyro = (data[1] \* 256) + data[0];

if (xGyro > 32767) xGyro -= 65536;

yGyro = (data[3] \* 256) + data[2];

if (yGyro > 32767) yGyro -= 65536;

zGyro = (data[5] \* 256) + data[4];

if (zGyro > 32767) zGyro -= 65536;

xGyro = xGyro \* 0.0038; // Full scale = +/- 125 degree/s

yGyro = yGyro \* 0.0038; // Full scale = +/- 125 degree/s

zGyro = zGyro \* 0.0038; // Full scale = +/- 125 degree/s

}

//=====================================================================================//

void BMX055\_Mag()

{

int data[8];

for (int i = 0; i < 8; i++)

{

Wire.beginTransmission(Addr\_Mag);

Wire.write((0x42 + i)); // Select data register

Wire.endTransmission();

Wire.requestFrom(Addr\_Mag, 1); // Request 1 byte of data

// Read 6 bytes of data

// xMag lsb, xMag msb, yMag lsb, yMag msb, zMag lsb, zMag msb

if (Wire.available() == 1)

data[i] = Wire.read();

}

// Convert the data

xMag = ((data[1] <<8) | (data[0]>>3));

if (xMag > 4095) xMag -= 8192;

yMag = ((data[3] <<8) | (data[2]>>3));

if (yMag > 4095) yMag -= 8192;

zMag = ((data[5] <<8) | (data[4]>>3));

if (zMag > 16383) zMag -= 32768;

}

// 初期化

void msr\_init(){

int i,j;

count = 0;

enMsrStartFlg = UNSET;

for (i=0;i<2;i++){

for (j=0;j<3;j++){

msrSum[i][j] = 0;

}

}

}

// 基板LEDチカチカ

void BlinkLED(int call\_cnt)

{

// 基板LEDチカチカ

call\_cnt++;

if(call\_cnt % 2 == 0){

digitalWrite(PIN\_SHOCK\_LED, LOW);

}else{

digitalWrite(PIN\_SHOCK\_LED, HIGH);

}

}

void print\_result(){

int i,j;

for (i=0;i<2;i++){

for (j=0;j<3;j++){

msrAve[i][j] = msrSum[i][j]/count;

}

}

Serial.print("Accl\_Average. = ");

Serial.print( msrAve[0][0] );

Serial.print("\t||\t");

Serial.print( msrAve[0][1] );

Serial.print("\t||\t");

Serial.println( msrAve[0][2] );

Serial.print("Gyro\_Average. = ");

Serial.print( msrAve[1][0] );

Serial.print("\t||\t");

Serial.print( msrAve[1][1] );

Serial.print("\t||\t");

Serial.println( msrAve[0][2] );

}

void msr\_jdgStartEnd(){

int inputchar;

// 計測開始と終了の判断部

if ( Serial.available()>0 ){

inputchar = Serial.read();

switch (inputchar){

case 'S':

if ( enMsrStartFlg == UNSET){

enLedOnFlg = UNSET;

enMsrStartFlg = SET;

Serial.println("MEARSER\_START");

}

else {

enMsrStartFlg = UNSET;

enLedOnFlg = SET;

Serial.println("MEARSER\_END");

en\_Initflg = INIT\_END;

print\_result();

msr\_init();

}

break;

default:

break;

}

}

}