プロジェクトディベロップメントプロジェクト

1126100
前川大輝

発表の流れの説明

1．前回からの進展
2．プログラム概要説明
3．実機によるデモンストレーション
4．プログラム解説
5．質疑応答

１．前回からの進展

・IMU用の受信関数が正常に動作するようになり、3軸のジャイロデータを取得できるようになりました。

2.プログラム概要説明

ジャイロセンサ(IMU3000)とI2C通信し情報を取得するプログラム

LCDにZ軸の回転角度を表示

3．実機によるデモンストレーション

それではさっそくデモンストレーションします。

4.プログラム解説

今回新たに変更を加えたファイル

- _i2c.h
- _i2c.c
- main.c

今回解説するI2C通信用関数内で用いられている基本的な関数の実装方法については割愛させていただきます。

main.c

```
#include "maekawa.h"
#include "stdio.h"
//#include "stdlib.h"
//#include "string.h"
//#include "MotionCommand.h"

_FOSC(CSW_FSCM_OFF & XT_PLL8);
_FWDT(WDT_OFF);
_FBORPOR(PBOR_ON & BORV_20 & PWRT_64 & MCLR_EN);
_FGS(CODE_PROT_OFF);


int main(void){
        char test[256];

        TRISA = 0x800;
        TRISB = 0x1C7;
        TRISC = 0x6000;
        TRISD = 0x200;
        TRISF = 0x0000;

        I2C i2c = I2CInitFunc();
        Lcd lcd = LcdInitFunc();
        IMU imu = IMUInitFunc();
        machi_msec(1000);

        i2c.CalibrationIMU(&imu);

        while(1){

                i2c.UpdateIMUStatus(&imu,10);

                sprintf(test,"%f",imu.angle[Z]);
                lcd.StrPuts(test);
                machi_msec(10);
                lcd.clear();

        }
        return 0;
}
```

IMUをキャリブレーション

IMUの状態を更新

_i2c.h

```
#ifndef _I2C_H
#define _I2C_H

#define GYRO 0x68
#define INT_MAX          32767
#define ANGLE_RATE_MAX   250

typedef unsigned char I2CData;

enum Axes{
                X,
                Y,
                Z
};

typedef struct{
                double NewGyro[3];
                double OldGyro[3];
                double OffsetGyro[3];
                double angle[3];

}IMU;

typedef struct{
                void(*close)(void);
                void(*write)(I2CData address,I2CData data);
                void(*SendDataToIMU)(I2CData I2CAddress,I2CData RegisterAddress,I2CData data);
                I2CData(*read)(I2CData address);
                I2CData(*ReadDataFromIMU)(I2CData I2CAddress,I2CData RegisterAddress);
                void(*CalibrationIMU)(IMU *imu);
                void(*UpdateIMUStatus)(IMU *imu,const unsigned int time);

}I2C;

#include "i2c.h"
#include "maekawa.h"

I2C I2CInitFunc(void);
IMU IMUInitFunc(void);

#endif
```

IMU用の構造体を定義

I2C関係のメンバを追加

IMU初期化関数

_i2c.c　　I2C初期化関数

```
I2C I2CInitFunc(void){
        • • •
        if(first){
        • • •
                SendDataToIMU(GYRO,0x16,0x0);
                SendDataToIMU(GYRO,0x3D,0x28);
                first = FALSE;
        }
        • • •
        i2c.CalibrationIMU  = CalibrationIMU;
        i2c.UpdateIMUStatus = UpdateIMUStatus;
        return i2c;
}
```

IMUの初期化

関数ポインタを代入

_i2c.c(2)　　IMU3000から受信する関数

```
static I2CData ReadDataFromIMU(I2CData I2CAddress,I2CData
RegisterAddress){
        I2CData buffer;

        I2CCONbits.SEN = 1;
        IDLE_I2C;
        SendAddressI2C(I2CAddress,SEND);
        SendDataI2C(RegisterAddress);

        I2CCONbits.RSEN = 1;
        while(I2CCONbits.RSEN);

        SendAddressI2C(I2CAddress,READ);
        buffer = MasterReadI2C();
        while(I2CSTATbits.ACKSTAT);
        IDLE_I2C;

        I2CCONbits.ACKDT = 1;
        I2CCONbits.ACKEN = 1;
        IDLE_I2C;
        I2CCONbits.PEN = 1;
        IDLE_I2C;

        return buffer;

}
```

再度スタートコンディションを発行する際はこのビットを立てる。

Single-Byte Read Sequence

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

Burst Read Sequence

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

_i2c.c(3)　IMUのオフセット値計算

```
static void CalibrationIMU(IMU *imu){
        int buffer[6],i;

        for(i=0;i < 156;i++){
                buffer[0] = (int)ReadDataFromIMU(GYRO,0x1D);
                buffer[1] = (int)ReadDataFromIMU(GYRO,0x1E);
                buffer[2] = (int)ReadDataFromIMU(GYRO,0x1F);
                buffer[3] = (int)ReadDataFromIMU(GYRO,0x20);
                buffer[4] = (int)ReadDataFromIMU(GYRO,0x21);
                buffer[5] = (int)ReadDataFromIMU(GYRO,0x22);

                imu->OffsetGyro[X] += (double)(buffer[0] << 8 | buffer[1]);
                imu->OffsetGyro[Y] += (double)(buffer[2] << 8 | buffer[3]);
                imu->OffsetGyro[Z] += (double)(buffer[4] << 8 | buffer[5]);

        }

        imu->OffsetGyro[X] /= 156.0;
        imu->OffsetGyro[Y] /= 156.0;
        imu->OffsetGyro[Z] /= 156.0;

}
```

3軸分ジャイロの値を取得

High/Lowを合成

_i2c.c(4)　　角速度、角度計算

```
static void UpdateIMUStatus(IMU *imu,const unsigned int time){
        int buf[6];
        double angle;

        buf[0] = (int)ReadDataFromIMU(GYRO,0x1D);
        buf[1] = (int)ReadDataFromIMU(GYRO,0x1E);
        buf[2] = (int)ReadDataFromIMU(GYRO,0x1F);
        buf[3] = (int)ReadDataFromIMU(GYRO,0x20);
        buf[4] = (int)ReadDataFromIMU(GYRO,0x21);
        buf[5] = (int)ReadDataFromIMU(GYRO,0x22);
        imu->NewGyro[X] = (buf[0] << 8 | buf[1]) - imu->OffsetGyro[X];
        imu->NewGyro[Y] = (buf[2] << 8 | buf[3]) - imu->OffsetGyro[Y];
        imu->NewGyro[Z] = (buf[4] << 8 | buf[5]) - imu->OffsetGyro[Z];

        angle = ((imu->NewGyro[X] + imu->OldGyro[X]) * (double)ANGLE_RATE_MAX * (double)time) / (double)INT_MAX / 1000.0;
        imu->angle[X] += angle >= -0.009 && angle <= 0.009 ? 0.0 : angle;

        angle = ((imu->NewGyro[Y] + imu->OldGyro[Y]) * (double)ANGLE_RATE_MAX * (double)time) / (double)INT_MAX / 1000.0;
        imu->angle[Y] += angle >= -0.009 && angle <= 0.009 ? 0.0 : angle;

        angle = ((imu->NewGyro[Z] + imu->OldGyro[Z]) * (double)ANGLE_RATE_MAX * (double)time) / (double)INT_MAX / 1000.0;
        imu->angle[Z] += angle >= -0.009 && angle <= 0.009 ? 0.0 : angle;

        imu->OldGyro[X] = imu->NewGyro[X];
        imu->OldGyro[Y] = imu->NewGyro[Y];
        imu->OldGyro[Z] = imu->NewGyro[Z];

}
```

回転角度を計算

_i2c.c(5)　IMUのデータを取り扱う構造体の初期化関数

```
IMU IMUInitFunc(void){
        IMU imu;
        int i;

        for(i=0;i < 3;i++){
                imu.NewGyro[i]    = 0.0;
                imu.OldGyro[i]    = 0.0;
                imu.OffsetGyro[i] = 0.0;
                imu.angle[i]      = 0.0;
        }

        return imu;
}
```

5. 質疑応答

以上にて発表を終了させていただきます。

ご清聴ありがとうございました。