****

**加速度传感器实验代码阅读报告**

**学 生 张君凯**

**学 号 1133730112**

**日 期 2016/1/4**

1. **代码注释**

#include "ioCC2530.h"

#include <string.h>

#include <stdio.h>

#define uint unsigned int

#define uchar unsigned char

#define TRUE 1

#define FALSE 0

#define SCL P0\_0

#define SDA P0\_1

#define I2C\_WRITE 0x00 /\* Write Direction \*/

#define I2C\_READ 0x01 /\* Read Direction \*/

#define MMA8452\_I2C\_WRITE 0x38 /\*CC2531向加速度传感器写入数据的地址\*/

#define MMA8452\_I2C\_READ 0x39 /\*CC2531从加速度传感器读取数据的地址\*/

#define CTRL\_REG1 0x2A /\*系统控制寄存器\*/

#define ACTIVE\_MASK 0x01

#define FULL\_SCALE\_2G 0x00 /\*量程为±2g\*/

#define FULL\_SCALE\_8G FS1\_MASK /\*量程为±8g\*/

#define FULL\_SCALE\_4G FS0\_MASK /\*量程为±4g\*/

#define FS1\_MASK 0x02

#define FS0\_MASK 0x01

#define XYZ\_DATA\_CFG\_REG 0x0E /\*数据配置寄存器\*/

#define DATA\_RATE\_5MS DR1\_MASK /\*数据速率\*/

#define DR1\_MASK 0x10

#define ASLP\_RATE\_20MS 0x00

//MSB 最高有效位

//LSB 最低有效位

#define MMA845x\_OUT\_X\_MSB 0x01 //加速度传感器寄存器,存放x轴加速度值最高有效位

#define MMA845x\_OUT\_X\_LSB 0x02 //加速度传感器寄存器,存放x轴加速度值最低有效位

#define MMA845x\_OUT\_Y\_MSB 0x03 //加速度传感器寄存器,存放y轴加速度值最高有效位

#define MMA845x\_OUT\_Y\_LSB 0x04 //加速度传感器寄存器,存放y轴加速度值最低有效位

#define MMA845x\_OUT\_Z\_MSB 0x05 //加速度传感器寄存器,存放z轴加速度值最高有效位

#define MMA845x\_OUT\_Z\_LSB 0x06 //加速度传感器寄存器,存放z轴加速度值最低有效位

float X\_value = 0;

int X\_temp = 0;

char X\_result[12] = {0};

float Y\_value = 0;

int Y\_temp = 0;

char Y\_result[12] = {0};

float Z\_value = 0;

int Z\_temp = 0;

char Z\_result[12] = {0};

void Delay\_1u(uint);

void WriteSDA1(void);

void WriteSDA0(void);

void WriteSCL1(void);

void WriteSCL0(void);

void ReadSDA(void);

void I2C\_Start(void);

void I2C\_Stop(void);

void SEND\_0(void);

void SEND\_1(void);

char Check\_Acknowledge(void);

void WriteI2CByte(char);

char ReadI2CByte(void);

void Read\_Data(char, char \*);

void Write\_Data(char, char);

void Initialize(void);

void MMA845x\_Active();

void MMA845x\_Standby(void);

void MMA845x\_Init (void);

void initUARTtest(void);

void UartTX\_Send\_String(char \*Data,int len);

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

//主函数

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void main(void)

{

char data\_result[7] = {0x55,0x55,0x55,0x55,0x55,0x55,0x55};

initUARTtest();

MMA845x\_Init ();

MMA845x\_Active ();//置Acitive状态

Read\_Data(XYZ\_DATA\_CFG\_REG, &data\_result[0]); //调试用

while(1)

{

/\*使用I2C读数函数从加速度传感器寄存器0x01、0x02读取x轴加速度值\*/

Read\_Data(MMA845x\_OUT\_X\_MSB, &data\_result[1]);

Read\_Data(MMA845x\_OUT\_X\_LSB, &data\_result[2]);

/\*使用I2C读数函数从加速度传感器寄存器0x03、0x04读取y轴加速度值\*/

Read\_Data(MMA845x\_OUT\_Y\_MSB, &data\_result[3]);

Read\_Data(MMA845x\_OUT\_Y\_LSB, &data\_result[4]);

/\*使用I2C读数函数从加速度传感器寄存器0x05、0x06读取z轴加速度值\*/

Read\_Data(MMA845x\_OUT\_Z\_MSB, &data\_result[5]);

Read\_Data(MMA845x\_OUT\_Z\_LSB, &data\_result[6]);

/\*拼接MSB、LSB\*/

X\_temp = data\_result[1];

X\_temp = X\_temp << 8;

X\_temp = X\_temp + data\_result[2];

X\_temp = X\_temp >> 4;

/\*判断加速度方向，相应改变数值正负\*/

if(X\_temp >= 0x800)

{

X\_temp = X\_temp - 0x1000;

}

X\_value = X\_temp \* 0.001;

sprintf(X\_result,"%s","X: ");

UartTX\_Send\_String(X\_result, 3);//发送辅助信息部分

sprintf(X\_result,"%8.3f",X\_value);

X\_result[8] = '\n';

UartTX\_Send\_String(X\_result, 9);//发送数值部分

//同上

Y\_temp = data\_result[3];

Y\_temp = Y\_temp << 8;

Y\_temp = Y\_temp + data\_result[4];

Y\_temp = Y\_temp >> 4;

if(Y\_temp >= 0x800)

{

Y\_temp = Y\_temp - 0x1000;

}

Y\_value = Y\_temp \* 0.001;

sprintf(Y\_result,"%s","Y: ");

UartTX\_Send\_String(Y\_result, 3);

sprintf(Y\_result,"%8.3f",Y\_value);

Y\_result[8] = '\n';

UartTX\_Send\_String(Y\_result, 9);

//同上

Z\_temp = data\_result[5];

Z\_temp = Z\_temp << 8;

Z\_temp = Z\_temp + data\_result[6];

Z\_temp = Z\_temp >> 4;

if(Z\_temp >= 0x800)

{

Z\_temp = Z\_temp - 0x1000;

}

Z\_value = Z\_temp \* 0.001;

sprintf(Z\_result,"%s","Z: ");

UartTX\_Send\_String(Z\_result, 3);

sprintf(Z\_result,"%8.3f",Z\_value);

Z\_result[8] = '\n';

UartTX\_Send\_String(Z\_result, 9);

}

}

/\*延迟\*/

void Delay\_1u(unsigned int microSecs) {

while(microSecs--)

{

/\* 32 NOPs == 1 usecs \*/

asm("nop"); asm("nop"); asm("nop"); asm("nop"); asm("nop");

asm("nop"); asm("nop"); asm("nop"); asm("nop"); asm("nop");

asm("nop"); asm("nop"); asm("nop"); asm("nop"); asm("nop");

asm("nop"); asm("nop"); asm("nop"); asm("nop"); asm("nop");

asm("nop"); asm("nop"); asm("nop"); asm("nop"); asm("nop");

asm("nop"); asm("nop"); asm("nop"); asm("nop"); asm("nop");

asm("nop"); asm("nop");

}

}

void WriteSDA1(void)//SDA 输出1

{

P0DIR |= 0x02;

SDA = 1;

}

void WriteSDA0(void)//SDA 输出0

{

P0DIR |= 0x02;

SDA = 0;

}

void WriteSCL1(void)//SCL 输出1

{

P0DIR |= 0x01;

SCL = 1;

}

void WriteSCL0(void)//SCL 输出0

{

P0DIR |= 0x01;

SCL = 0;

}

void ReadSDA(void)//这里设置SDA对应IO口DIR可以接收数据

{

P0DIR &= 0xFD;

}

/\*启动I2C总线的函数，当SCL为高电平时使SDA产生一个负跳变\*/

void I2C\_Start(void)

{

WriteSDA1();

WriteSCL1();

Delay\_1u(50);

WriteSDA0();

Delay\_1u(50);

WriteSCL0();

Delay\_1u(50);

}

/\*终止I2C总线，当SCL为高电平时使SDA产生一个正跳变\*/

void I2C\_Stop(void)

{

WriteSDA0();

Delay\_1u(50);

WriteSCL1();

Delay\_1u(50);

WriteSDA1();

Delay\_1u(50);

WriteSCL0();

Delay\_1u(50);

}

/\*发送0，在SCL为高电平时使SDA信号为低\*/

void SEND\_0(void)

{

WriteSDA0();

WriteSCL1();

Delay\_1u(50);

WriteSCL0();

Delay\_1u(50);

}

/\*发送1，在SCL为高电平时使SDA信号为高\*/

void SEND\_1(void)

{

WriteSDA1();

WriteSCL1();

Delay\_1u(50);

WriteSCL0();

Delay\_1u(50);

}

/\*发送完一个字节后检验设备的应答信号\*/

char Check\_Acknowledge(void)

{

WriteSDA1();

WriteSCL1();

Delay\_1u(50);

F0=SDA;

Delay\_1u(50);

WriteSCL0();

Delay\_1u(50);

if(F0==1)

return FALSE;

return TRUE;

}

/\*向I2C总线写一个字节\*/

void WriteI2CByte(char b)

{

char i;

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

{

if((b<<i)&0x80)

{

SEND\_1();

}

else

{

SEND\_0();

}

}

}

/\*从I2C总线读一个字节\*/

char ReadI2CByte(void)

{

char b=0,i;

WriteSDA1();

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

{

WriteSCL0();

Delay\_1u(50);

WriteSCL1();

Delay\_1u(50);

ReadSDA();

F0=SDA;//寄存器中的一位,用于存储SDA中的一位数据

if(F0==1)

{

b=b<<1;

b=b|0x01;

}

else

b=b<<1;

}

WriteSCL0();

return b;

}

/\*从加速度传感器的某寄存器读取数据\*/

void Read\_Data(char reg, char \*data)

{

I2C\_Start();

WriteI2CByte(MMA8452\_I2C\_WRITE);

while(Check\_Acknowledge() == FALSE);//等待应答

WriteI2CByte(reg);

while(Check\_Acknowledge() == FALSE);//等待应答

I2C\_Start();

WriteI2CByte(MMA8452\_I2C\_READ);

while(Check\_Acknowledge() == FALSE);//等待应答

\*data = ReadI2CByte();

I2C\_Stop();

}

/\*向加速度传感器的某寄存器写入具体数据\*/

void Write\_Data(char reg, char data)

{

I2C\_Start();

WriteI2CByte(MMA8452\_I2C\_WRITE); //向IIC总线发送

while(Check\_Acknowledge() == FALSE);//等待应答

WriteI2CByte(reg);

while(Check\_Acknowledge() == FALSE);//等待应答

WriteI2CByte(data);

while(Check\_Acknowledge() == FALSE);//等待应答

I2C\_Stop();

}

//初始化

void Initialize(void)

{

Write\_Data(0x01, 0x84);

}

void MMA845x\_Active ()

{

char a[1] = {0};

/\*

\*\* Read current value of System Control 1 Register.

\*\* Put sensor into Standby Mode.

\*\* Return with previous value of System Control 1 Register.

\*/

Read\_Data(CTRL\_REG1,a);//从控制寄存器中读取数据到a

Write\_Data(CTRL\_REG1, a[1] | ACTIVE\_MASK);//置Active状态，可以得到加速度数据

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\

\* Put MMA845xQ into Standby Mode 待命状态

\\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void MMA845x\_Standby (void)

{

char a[1] = {0};

/\*

\*\* Read current value of System Control 1 Register.

\*\* Put sensor into Standby Mode.

\*\* Return with previous value of System Control 1 Register.

\*/

Read\_Data(CTRL\_REG1,a);//从控制寄存器中读取数据到a

Write\_Data(CTRL\_REG1, a[1] & ~ACTIVE\_MASK);//置待命状态

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\

\* Initialize MMA845xQ

\\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void MMA845x\_Init (void)

{

MMA845x\_Standby();

/\*

\*\* Configure sensor for:

\*\* - Sleep Mode Poll Rate of 50Hz (20ms)

\*\* - System Output Data Rate of 200Hz (5ms)

\*\* - Full Scale of +/-2g

\*/

Write\_Data(CTRL\_REG1, ASLP\_RATE\_20MS+DATA\_RATE\_5MS); //设置控制寄存器

Write\_Data(XYZ\_DATA\_CFG\_REG, FULL\_SCALE\_2G); //设置量程

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*函数功能 ：初始化串口1

\*入口参数 ：无

\*返 回 值 ：无

\*说 明 ：57600-8-n-1

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void initUARTtest(void)

{

CLKCONCMD &= ~0x40; //晶振

while(!(SLEEPSTA & 0x40)); //等待晶振稳定

CLKCONCMD &= ~0x47; //TICHSPD128分频，CLKSPD不分频

SLEEPSTA |= 0x04; //关闭不用的RC振荡器

PERCFG = 0x01; //位置1 串口0

P1SEL |= 0x30; //P1用作串口

U0CSR |= 0x80; //UART方式

U0GCR |= 8; //baud\_e

U0BAUD |= 59; //波特率设为9600

UTX0IF = 1;

U0CSR |= 0X40; //允许接收

IEN0 |= 0x84; //开总中断，接收中断

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*函数功能 ：串口发送字符串函数

\*入口参数 : data:数据

\* len :数据长度

\*返 回 值 ：无

\*说 明 ：

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void UartTX\_Send\_String(char \*Data,int len)

{

int j;

for(j=0;j<len;j++)

{

U0DBUF = \*Data++; //把数据写到缓存器中

while(UTX0IF == 0);//等待数据发送完成

UTX0IF = 0; //通过软件清0

}

}

1. **程序流程图**

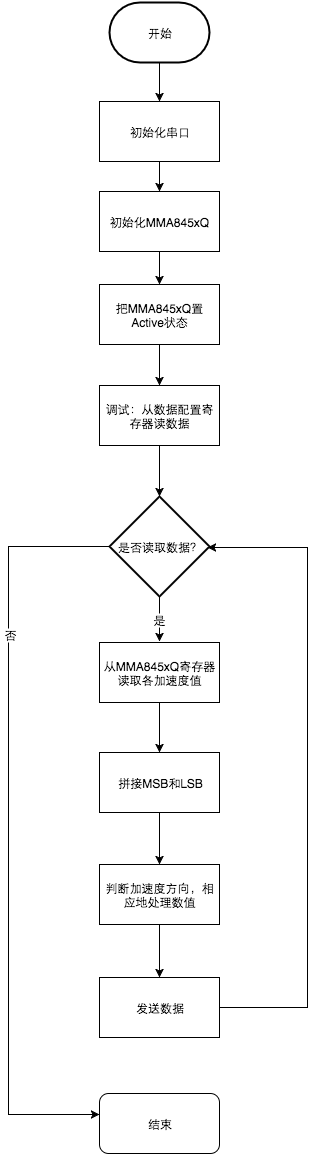
****

图 1-1