**软件实现说明书**

**项目名称:智能阅览室系统**

开发团队： 这是一个神奇的队伍

成　　员： 　学号1133730106 姓名 栾云腾

　　　 学号 1133730117姓名 张实唯

　　　 学号 1133730112姓名 张君凯

学号 1133730206姓名 王健宇

学号 1133730208姓名 李天宝

学号 1133730113姓名 吴添雄

学号 1133730215姓名 周韬宇

哈尔滨工业大学

软件学院

二零一六年一月

系统实现文档

## 图书管理子系统的实现

### 书籍信息展示模块的实现

iOS提供的UITableView类采用数据源绑定以及实现相关协议，需要一个数据源：在本项目中是BookItemStore类；需要实现相关协议：

在BookItemViewController中实现方法

- (UITableViewCell \*)tableView:(UITableView \*)tableView cellForRowAtIndexPath:(NSIndexPath \*)indexPath

和

- (NSInteger)tableView:(UITableView \*)tableView numberOfRowsInSection:(NSInteger)section

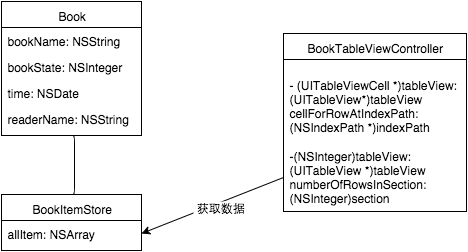


图1-1 书籍管理类图

### 书籍信息获取模块的实现

核心方法实现：

使用第三方库AFNetworking的AFHTTPSessionManager类中的GET方法向服务器请求书籍最新借阅、归还信息，将其存在BookItemStore类的静态对象中。

App打开或者在App下拉刷新时会执行上述方法，更新数据源。

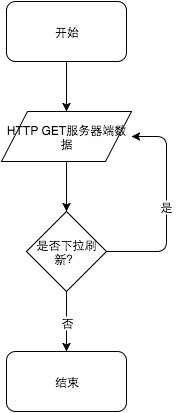


图1-2 书籍管理流程图

## 人员管理子系统的实现

### 2.1借还书信息展示模块的实现

iOS提供的UITableView类采用数据源绑定以及实现相关协议，需要一个数据源：在本项目中是ReaderItemStore类；需要实现相关协议：

在ReaderItemViewController中实现方法

- (UITableViewCell \*)tableView:(UITableView \*)tableView cellForRowAtIndexPath:(NSIndexPath \*)indexPath

和

- (NSInteger)tableView:(UITableView \*)tableView numberOfRowsInSection:(NSInteger)section

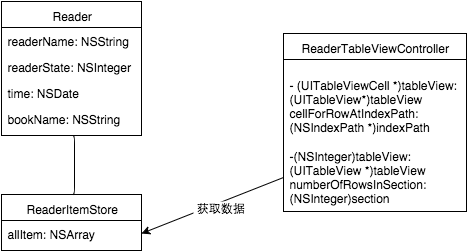


图2-1 人员管理类图

### 2.2人员信息获取模块的实现

核心方法实现：

使用第三方库AFNetworking的AFHTTPSessionManager类中的GET方法向服务器请求读者最新借阅、归还信息，将其存在ReaderItemStore类的静态对象中。

App打开或者在App下拉刷新时会执行上述方法，更新数据源。

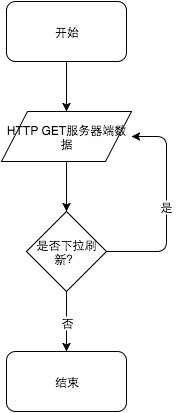


图2-2 人员管理流程图

## 3环境监测子系统的实现

### 3.1门禁模块的实现

#### 3.1.1霍尔传感器代码

#include "ioCC2530.h"

#include <string.h>

#define uint16 unsigned int

#define uchar unsigned char

#define TRUE 1

#define FALSE 0

#define RECEIVE P0\_7

#define WARN\_LED P1\_7

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

void init\_time1(void);

void initUARTtest(void);

void InitAll(void);

void delay\_ms(uint16);

void Warn(void);

uint16 value = 0;

uint16 counter = 0;

uint16 timer\_counter = 0;

uint16 receiver\_counter = 0;

uchar sound\_flag = 0;

uchar uart\_test[2] = {0x55, 0x55};

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

//主函数

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

uchar true\_result[3] = "1\n";

uchar false\_result[3] = "0\n";

void main(void)

{

InitAll();

while(1)

{

if(sound\_flag == 1)

{

sound\_flag = 0;

Warn();

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

}

else

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

delay\_ms(200);

}

}

void init\_time3(void)

{

T3CCTL0 |= 0x04; //配置成比较模式，

T3CC0 = 0x7D;

//T3CCTL1 = 0x00;

//T3CC1 = 0x00;

EA = 1;

T3IE = 1;

T3CTL = 0xDE; //时钟16MHz，64分频，定时器0.25MHz，定时器counter增1，步进4us。使能定时器。使能中断。清除counter。模式：modulo mode。

}

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

\*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

}

void InitAll(void)

{

EA=0;

init\_time3();

initUARTtest();

//ADCCFG |= 0x02;

P0DIR &= 0xFD; //配置P0\_1为输入

P1DIR |= 0x80; //配置P1\_7为输出

EA=1; //re-enable interrupts

}

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

\*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(uchar \*Data, int len)

{

int j;

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

{

U0DBUF = \*Data++;

while(UTX0IF == 0);

UTX0IF = 0;

}

}

#pragma vector = T3\_VECTOR

\_\_interrupt void T3\_ISR(void)

{

IRCON &= 0xFD;

T3IE = 0;

counter++;

if(counter == 50)

{

counter = 0;

timer\_counter++;

F0 = RECEIVE;

if(F0 == 1)

{

receiver\_counter++;

}

else

{

}

if(timer\_counter == 10)

{

if(receiver\_counter >= 8)

{

sound\_flag = 1;

}

else

{

}

timer\_counter = 0;

receiver\_counter = 0;

}

else

{

}

}

T3IE = 1;

}

void delay\_ms(uint16 tms)

{

uint16 ii;

while(tms)

{

tms--;

for(ii=0;ii<400;ii++)

{

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

}

}

}

//告警蜂鸣器响3次

void Warn(void)

{

uchar ii;

for(ii=0;ii<5;ii++)

{

WARN\_LED = 1;

delay\_ms(120);

WARN\_LED = 0;

delay\_ms(120);

}

}

#### 3.1.2教室门COM通信代码

import serial

import sys

import time

import json

import urllib

import urllib2

apiurldoor = "http://192.168.199.187/sensordata/halldoor"

apiurlshelf = "http://192.168.199.187/sensordata/hallshelf"

data = {}

status = 0

def post(url, data):

#data = urllib.urlencode(data)

req = urllib2.Request(url, data)

response = urllib2.urlopen(req)

res = response.read()

return res

while True:

str = ""

data = {}

try:

ser = serial.Serial('COM3', 9600)

except Exception, e:

print 'open serial failed.'

exit(1)

str = ser.readline()

temp = str.strip()

if temp != status:

status = temp

data['hallstatus'] = int(temp)

data = json.dumps(data)

print data

print post(apiurldoor, data)

sys.stdout.flush()

ser.close()

#output: 1 byte char (status)

#1:magnet in position

#0:out of range

#### 3.1.3书柜门COM通信代码

import serial

import sys

import time

import json

import urllib

import urllib2

apiurldoor = "http://192.168.199.187/sensordata/halldoor"

apiurlshelf = "http://192.168.199.187/sensordata/hallshelf"

data = {}

status = 0

def post(url, data):

#data = urllib.urlencode(data)

req = urllib2.Request(url, data)

response = urllib2.urlopen(req)

res = response.read()

return res

while True:

str = ""

data = {}

try:

ser = serial.Serial('COM3', 9600)

except Exception, e:

print 'open serial failed.'

exit(1)

str = ser.readline()

temp = str.strip()

if temp != status:

status = temp

data['hallstatus'] = int(temp)

data = json.dumps(data)

print data

print post(apiurlshelf, data)

sys.stdout.flush()

ser.close()

#output: 1 byte char (status)

#1:magnet in position

#0:out of range

### 3.2烟雾模块的实现

#### 3.2.1烟雾传感器代码

#include "ioCC2530.h"

#include <string.h>

#include <stdio.h>

#define uint unsigned int

#define uchar unsigned char

#define uint8 unsigned char

#define uint16 unsigned int

//定义控制灯的端口

#define led1 P0\_0

#define led2 P0\_1

#define HAL\_ADC\_REF\_3V3 0x10 /\* external 3.3V Reference \*/

#define HAL\_ADC\_DEC\_128 0x10 /\* Decimate by 128 : 10-bit resolution \*/

#define HAL\_ADC\_CHN\_5 0x05 /\* Input channel: channel \*/

#define HI\_UINT16(a) (((a) >> 8) & 0xFF)

#define LO\_UINT16(a) ((a) & 0xFF)

void Delay(uint);

void initUARTtest(void);

void InitialAD(void);

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

char test[20] = "this is a test!";

uint16 gas\_temp = 0;

char gas\_result[15] = {0};

float gas\_value = 0;

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

\*函数功能 ：获取气体 \*

\*入口参数 ：无 \*

\*返 回 值 ：无 \*

\*说 明 ：无 \*

\*Value = 0x0A5F，对应电压值 = 0x0A5F \* 3.3 / 8192 = 1.0695V

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

uint16 Gas\_Acquire(void)

{

uint16 value;

/\* Clear ADC interrupt flag \*/

ADCIF = 0;

//ADCCON3 = (HAL\_ADC\_REF\_3V3 | HAL\_ADC\_DEC\_128 | HAL\_ADC\_CHN\_5);

//ADCCON3 = 0xb7;

ADCCON3 = 0xb7;

ADCCON1 |= 0x30;

ADCCON1 |= 0x40;

while ((ADCCON1 & 0x80) == 0);

/\* Get the result \*/

value = ADCH;

value = value << 8;

value = value + ADCL;

value = value >> 2;

return value;

}

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

\*函数功能 ：主函数 \*

\*入口参数 ：无 \*

\*返 回 值 ：无 \*

\*说 明 ：无 \*

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

void main(void)

{

P0DIR = 0x03;

led1 = 1;

led2 = 1;

initUARTtest();

while(1)

{

gas\_temp = Gas\_Acquire();

gas\_value = gas\_temp \* 3.3 / 8192;

sprintf(gas\_result,"%s","gas\_sample: ");

UartTX\_Send\_String(gas\_result, 12);

sprintf(gas\_result, "%8.4f", gas\_value);

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

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

Delay(30000);

led1 =!led1; //完成数据处理

Delay(30000);

}

}

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

\*函数功能 ：延时 \*

\*入口参数 ：定性延时 \*

\*返 回 值 ：无 \*

\*说 明 ： \*

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

void Delay(uint n)

{

uint i;

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

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

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

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

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

}

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

\*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++;

while(UTX0IF == 0);

UTX0IF = 0;

}

}

#### 3.2.2烟雾COM通信代码

import serial

import sys

import urllib

import urllib2

import time

import json

apiurl = "http://192.168.199.187/sensordata/mq2"

data = {}

def post(url, data):

#data = urllib.urlencode(data)

req = urllib2.Request(url, data)

response = urllib2.urlopen(req)

res = response.read()

return res

while True:

data = {}

try:

ser = serial.Serial('COM4', 9600)

except Exception, e:

print 'open serial failed.'

exit(1)

str = ser.readline()

str = ser.readline()

data['mq2'] = float(str.split()[1])

data = json.dumps(data)

print data

print post(apiurl, data)

sys.stdout.flush()

ser.close()

time.sleep(1)

#output: 1 string (gas qualitative concentration)

### 3.3气压模块的实现

#### 3.3.1气压传感器代码

#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 //\*T

#define I2C\_READ 0x01 //\*R

#define MPL3115A2\_I2C\_ADDRESS 0xC0 //

//定义控制灯的端口

#define YLED P2\_0 //定义LED1为P2\_0口控制

#define RLED P0\_5 //定义LED2为P0\_5口控制

char data\_result[6] = {0x55,0x55,0x55,0x55,0x55,0x55};

long int pressure\_temp = 0;

char pressure\_result[15] = {0};

float pressure\_value = 0;

int temp\_temp = 0;

char temp\_result[15] = {0};

float temp\_value = 0;

char INT\_FLAG[1] = {0x00};

int j = 0; //调试用

void Init\_IO\_INT(void);

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 initUARTtest(void);

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

void initLed(void);

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

//主函数

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

void main(void)

{

initLed();

initUARTtest();

Write\_Data(0x26, 0x39);

while(1)

{

Read\_Data(0x01, &data\_result[1]);

Read\_Data(0x02, &data\_result[2]);

Read\_Data(0x03, &data\_result[3]);

Read\_Data(0x04, &data\_result[4]);

Read\_Data(0x05, &data\_result[5]);

pressure\_temp = data\_result[1];

pressure\_temp = pressure\_temp << 8;

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

pressure\_temp = pressure\_temp << 8;

pressure\_temp = pressure\_temp + data\_result[3];

pressure\_temp = pressure\_temp >> 4;

pressure\_value = pressure\_temp \* 0.25;

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

UartTX\_Send\_String(pressure\_result, 10);

sprintf(pressure\_result,"%9.2f",pressure\_value);

pressure\_result[9] = ':';

UartTX\_Send\_String(pressure\_result, 10);

temp\_temp = data\_result[4];

temp\_temp = temp\_temp << 8;

temp\_temp = temp\_temp + data\_result[5];

temp\_temp = temp\_temp >> 4;

temp\_value = temp\_temp \* 0.0625;

if (temp\_value > 22)

{

for(uchar i=0; i < 3; i++)

{

YLED = !YLED;

RLED = !RLED;

Delay\_1u(1000);

}

}

sprintf(pressure\_result,"%s","temperature: ");

UartTX\_Send\_String(pressure\_result, 13);

sprintf(temp\_result, "%8.4f", temp\_value);

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

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

Delay\_1u(65535);

}

}

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

//io及中断初始化

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

void Init\_IO\_INT(void)

{

P1SEL &= 0xFD; //P11作为普通IO

P1DIR &= 0xFD; //P11作为输入

P1INP &= 0xFD;//P11有上拉、下拉

PICTL |= 0X02; //下降沿

EA = 1;

IEN2 |= 0X10; // P1IE = 1;

P1IEN |= 0X02; //使能P1\_1中断

P1IF = 0; //清中断标志

P1IFG = 0; //清中断标志

}

#pragma vector = P1INT\_VECTOR

\_\_interrupt void P1\_ISR(void)

{

if((P1IFG & 0x02) != 0) //P1\_1中断

{

P1IFG &= 0xFD;

Read\_Data(0x02, &INT\_FLAG[0]);

Write\_Data(0x02, INT\_FLAG[0] & 0x77);

// Read\_Data(0x08, &data\_result[1]);

// Read\_Data(0x09, &data\_result[2]);

// Read\_Data(0x0A, &data\_result[3]);

j++; //调试用

}

P1IFG = 0;//清中断标志

P1IF = 0; //清中断标志

}

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;

}

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

{

I2C\_Start();

WriteI2CByte(MPL3115A2\_I2C\_ADDRESS + I2C\_WRITE);

while(Check\_Acknowledge() == FALSE);

WriteI2CByte(reg);

while(Check\_Acknowledge() == FALSE);

I2C\_Start();

WriteI2CByte(MPL3115A2\_I2C\_ADDRESS + I2C\_READ);

while(Check\_Acknowledge() == FALSE);

\*data = ReadI2CByte();

I2C\_Stop();

}

void Write\_Data(char reg, char data)

{

I2C\_Start();

WriteI2CByte(MPL3115A2\_I2C\_ADDRESS + I2C\_WRITE);

while(Check\_Acknowledge() == FALSE);

WriteI2CByte(reg);

while(Check\_Acknowledge() == FALSE);

WriteI2CByte(data);

while(Check\_Acknowledge() == FALSE);

I2C\_Stop();

}

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

\*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++;

while(UTX0IF == 0);

UTX0IF = 0;

}

}

void initLed(void)

{

P0DIR |= 0x20; //P0\_5定义为输出

P2DIR |= 0x01; //P2\_0定义为输出

RLED = 1;

YLED = 1; //LED灯灭

}

#### 3.3.2气压COM通信代码

import serial

import sys

import urllib

import urllib2

import time

import json

apiurl = "http://192.168.199.187/sensordata/pressure"

data = {}

def post(url, data):

#data = urllib.urlencode(data)

req = urllib2.Request(url, data)

response = urllib2.urlopen(req)

res = response.read()

return res

while True:

data = {}

try:

ser = serial.Serial('COM4', 9600)

except Exception, e:

print 'open serial failed.'

exit(1)

str = ser.readline()

str = ser.readline()

str = str.split(':')

data['pressure'] = float(str[1])

data = json.dumps(data)

print data

print post(apiurl, data)

sys.stdout.flush()

ser.close()

time.sleep(1)

#output: 2-tuple (pressure, temperture)

### 3.4RFID模块的实现

#### 3.4.1RFID传感代码

#include "include.h"

uchar PassWd[6] = {0x00};

uchar Read\_Data[16] = {0x00};

uchar WriteData[16] = {0x00};

uchar RechargeData[4] = {0x00,0x00,0x00,0x00};

uchar ConsumeData[4] = {0x00,0x00,0x00,0x00};

uchar RevBuffer[35] = {0x00};

uchar SendBuffer[35] = {0x00,DevType,0x00,0x00,0x00,0x00,0x00,0x00};

uchar MLastSelectedSnr[4] = {0x00};

uchar uart\_count = 0;

uchar uart\_comp = 0;

uint KeyNum = 0;

uint KuaiN = 0;

uchar bWarn = 0;

uchar bPass = 0;

uchar KeyTime = 0;

uchar WaitTimes = 0;

//uchar SysTime = 0;

long int SysTime = 0;

uchar oprationcard = 0;

uchar bSendID = 0;

uchar temp = 0;

extern uchar result1;

void main(void)

{

InitAll();

while(1)

{

if(bWarn == 1)

{

bWarn = 0;

Warn();

}

if(bPass == 1)

{

bPass = 0;

Pass();

}

if(uart\_comp == 1)

{

ctrl\_uart();

uart\_comp = 0;

}

if(SysTime >= 40)

{

SysTime = 0;

YLED = !YLED;

ctrlprocess();

}

}

}

#### 3.4.2教室门检测图书COM通信代码

import serial

import sys

import random

import time

import json

import urllib

import urllib2

apiurldooruser = "http://192.168.199.187/sensordata/rfiddooruser"

apiurldoorbook = "http://192.168.199.187/sensordata/rfiddoorbook"

apiurlshelfuser = "http://192.168.199.187/sensordata/rfidshelfuser"

apiurlshelfbook = "http://192.168.199.187/sensordata/rfidshelfbook"

data = {}

status = 0

def post(url, data):

#data = urllib.urlencode(data)

req = urllib2.Request(url, data)

response = urllib2.urlopen(req)

res = response.read()

return res

while True:

data = {}

try:

ser = serial.Serial('COM4', 9600)

except Exception, e:

print 'open serial failed.'

exit(1)

code = "\x02\x02\x00\xA0\x05\x00\x00\x00\xA0\x00\x01"

n = ser.write(code)

str = ser.read(15)

str = str.strip()

str = str.split()

ID = ""

for x in str[0]:

ID += ("%02x" % ord(x)).upper() + " "

ID = ID.strip()

ID = ID.replace(" ", "")

data['id'] = ID

data = json.dumps(data)

print data

print post(apiurldoorbook, data)

ser.close()

#output: 1 string (RFID ID)

#### 3.4.3教室门检测用户COM通信代码

import serial

import sys

import random

import time

import json

import urllib

import urllib2

apiurldooruser = "http://192.168.199.187/sensordata/rfiddooruser"

apiurldoorbook = "http://192.168.199.187/sensordata/rfiddoorbook"

apiurlshelfuser = "http://192.168.199.187/sensordata/rfidshelfuser"

apiurlshelfbook = "http://192.168.199.187/sensordata/rfidshelfbook"

data = {}

status = 0

def post(url, data):

#data = urllib.urlencode(data)

req = urllib2.Request(url, data)

response = urllib2.urlopen(req)

res = response.read()

return res

while True:

data = {}

try:

ser = serial.Serial('COM4', 9600)

except Exception, e:

print 'open serial failed.'

exit(1)

code = "\x02\x02\x00\xA0\x05\x00\x00\x00\xA0\x00\x01"

n = ser.write(code)

str = ser.read(15)

str = str.strip()

str = str.split()

ID = ""

for x in str[0]:

ID += ("%02x" % ord(x)).upper() + " "

ID = ID.strip()

ID = ID.replace(" ", "")

data['id'] = ID

data = json.dumps(data)

print data

print post(apiurldooruser, data)

ser.close()

#output: 1 string (RFID ID)

#### 3.4.4书柜门检测图书COM通信代码

import serial

import sys

import random

import time

import json

import urllib

import urllib2

apiurldooruser = "http://192.168.199.187/sensordata/rfiddooruser"

apiurldoorbook = "http://192.168.199.187/sensordata/rfiddoorbook"

apiurlshelfuser = "http://192.168.199.187/sensordata/rfidshelfuser"

apiurlshelfbook = "http://192.168.199.187/sensordata/rfidshelfbook"

data = {}

status = 0

def post(url, data):

#data = urllib.urlencode(data)

req = urllib2.Request(url, data)

response = urllib2.urlopen(req)

res = response.read()

return res

while True:

data = {}

try:

ser = serial.Serial('COM4', 9600)

except Exception, e:

print 'open serial failed.'

exit(1)

code = "\x02\x02\x00\xA0\x05\x00\x00\x00\xA0\x00\x01"

n = ser.write(code)

str = ser.read(15)

str = str.strip()

str = str.split()

ID = ""

for x in str[0]:

ID += ("%02x" % ord(x)).upper() + " "

ID = ID.strip()

ID = ID.replace(" ", "")

data['id'] = ID

data = json.dumps(data)

print data

print post(apiurlshelfbook, data)

ser.close()

#output: 1 string (RFID ID)

#### 3.4.5书柜门检测用户COM通信代码

import serial

import sys

import random

import time

import json

import urllib

import urllib2

apiurldooruser = "http://192.168.199.187/sensordata/rfiddooruser"

apiurldoorbook = "http://192.168.199.187/sensordata/rfiddoorbook"

apiurlshelfuser = "http://192.168.199.187/sensordata/rfidshelfuser"

apiurlshelfbook = "http://192.168.199.187/sensordata/rfidshelfbook"

data = {}

status = 0

def post(url, data):

#data = urllib.urlencode(data)

req = urllib2.Request(url, data)

response = urllib2.urlopen(req)

res = response.read()

return res

while True:

data = {}

try:

ser = serial.Serial('COM4', 9600)

except Exception, e:

print 'open serial failed.'

exit(1)

code = "\x02\x02\x00\xA0\x05\x00\x00\x00\xA0\x00\x01"

n = ser.write(code)

str = ser.read(15)

str = str.strip()

str = str.split()

ID = ""

for x in str[0]:

ID += ("%02x" % ord(x)).upper() + " "

ID = ID.strip()

ID = ID.replace(" ", "")

data['id'] = ID

data = json.dumps(data)

print data

print post(apiurlshelfuser, data)

ser.close()

#output: 1 string (RFID ID)

### 3.5继电器模块的实现

#### 3.5.1继电器传感代码

#include <ioCC2530.h>

#define uint unsigned int

#define uchar unsigned char

//定义控制灯的端口

#define RLED P0\_5 //定义LED1为P0\_5口控制

#define YLED P2\_0 //定义LED2为P2\_0口控制

#define relay P0\_7

#define K1 P1\_1

#define K2 P1\_0

#define ON 0

#define OFF 1

//函数声明

void Delay(uint); //延时函数

void Initial(void); //初始化P口

void InitKey(void);

uchar KeyScan(void);

uchar Keyvalue = 0 ;

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

//延时

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

void Delay(uint n)

{

uint tt;

for(tt = 0;tt < n;tt++);

for(tt = 0;tt < n;tt++);

for(tt = 0;tt < n;tt++);

for(tt = 0;tt < n;tt++);

for(tt = 0;tt < n;tt++);

}

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

//按键初始化

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

void InitKey(void)

{

P1SEL &= ~0X03;

P1DIR &= ~0X03; //按键在P12 P13

P1INP |= 0x03; //上、下拉

}

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

\*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

SLEEPCMD |= 0x04; //关闭不用的RC振荡器

PERCFG = 0x01; //位置1 P0口

P1SEL = 0x30; //P0用作串口

U0CSR |= 0x80; //UART方式

U0GCR |= 8; //baud\_e

U0BAUD |= 59; //波特率设为57600

UTX0IF = 1;

U0CSR |= 0X40; //允许接收

IEN0 |= 0x84; //开总中断，接收中断

}

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

//初始化程序

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

void Initial(void)

{

P0DIR |= 0x20; //P0\_5定义为输出

P2DIR |= 0x01; //P2\_0定义为输出

RLED = OFF;

YLED = OFF; //LED

}

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

//读键值

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

uchar KeyScan(void)

{

if(K1 == 0)

{

Delay(100);

if(K1 == 0)

{

while(!K1);

return(1);

}

};

if(K2 == 0)

{

Delay(100);

if(K2 == 0)

{

while(!K2);

return(2);

}

};

return(0);

}

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

{

int j;

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

{

U0DBUF = \*Data++;

while(UTX0IF == 0);

UTX0IF = 0;

}

}

void write\_relay\_1(void)//继电器控制管脚 输出1

{

relay = 1;

}

void write\_relay\_0(void)//继电器控制管脚 输出0

{

relay = 0;

}

uchar RTflag = 1;

uchar temp;

uint datanumber = 0;

uint stringlen;

uchar true\_result[3] = "1\n";

uchar false\_result[3] = "0\n";

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

//主函数

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

void main(void)

{

Initial(); //调用初始化函数

InitKey();

initUARTtest();

Delay(4000);

Delay(4000);

P0DIR |= 0x80;

YLED = ON;

U0CSR |= 0x40;

while(1)

{

if(temp != 0)

{

if(temp=='0')

{

RLED = ON; //亮灯

YLED = OFF;

write\_relay\_1();

//UartTX\_Send\_String(false\_result,3);

}

else

{

YLED = ON; //亮灯

RLED = OFF;

write\_relay\_0();

//UartTX\_Send\_String(true\_result,3);

}

}

Delay(4000);

}

}

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

\*函数功能 ：串口接收一个字符

\*入口参数 : 无

\*返 回 值 ：无

\*说 明 ：接收完成后打开接收

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

#pragma vector = URX0\_VECTOR

\_\_interrupt void UART0\_ISR(void)

{

URX0IF = 0; //清中断标志

temp = U0DBUF;

}

#### 3.5.2教室门继电器COM通信代码

import serial

import sys

import random

import time

import json

import urllib

import urllib2

apiurldoor = "http://192.168.199.187/command/relaydoor"

apiurlshelf = "http://192.168.199.187/command/relayshelf"

data = {}

status = 0

def get(url):

#data = urllib.urlencode(data)

req = urllib2.Request(url)

response = urllib2.urlopen(req)

res = response.read()

return res

while True:

try:

ser = serial.Serial('COM4', 9600)

except Exception, e:

print 'open serial failed.'

exit(1)

command = get(apiurldoor)

#ser.write("%d" % random.randint(0,1))

if command is not None or command is not "":

command = command.split(':')

ser.write(command[1][0])

sys.stdout.flush()

ser.close()

#input: random number of {0, 1}

#0:break

#1:access

#### 3.5.3书柜门继电器COM通信代码

import serial

import sys

import random

import time

import json

import urllib

import urllib2

apiurldoor = "http://192.168.199.187/command/relaydoor"

apiurlshelf = "http://192.168.199.187/command/relayshelf"

data = {}

status = 0

def get(url):

#data = urllib.urlencode(data)

req = urllib2.Request(url)

response = urllib2.urlopen(req)

res = response.read()

return res

while True:

try:

ser = serial.Serial('COM4', 9600)

except Exception, e:

print 'open serial failed.'

exit(1)

command = get(apiurlshelf)

#ser.write("%d" % random.randint(0,1))

if command is not None or command is not "":

command = command.split(':')

ser.write(command[1][0])

sys.stdout.flush()

ser.close()

#input: random number of {0, 1}

#0:break

#1:access

### 3.6温湿模块的实现

#### 3.6.1温湿传感器代码

#include "ioCC2530.h"

#include <string.h>

#include "sht10.h"

#include <stdio.h>

//#define uint unsigned int

//#define uchar unsigned char

//#define Uint16 unsigned int

//定义控制灯的端口

#define led1 P1\_0

#define led2 P1\_1

//函数声明

void Delay1(uint);

void initUARTtest(void);

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

void initLED(void);

union

{ unsigned int i;

float f;

}humi\_val,temp\_val; //定义两个共同体，一个用于湿度，一个用于温度

char temp\_result[15] = {0};

float temp\_value = 0;

char humi\_result[15] = {0};

float humi\_value = 0;

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

\*函数功能 ：主函数

\*入口参数 ：无

\*返 回 值 ：无

\*说 明 ：无

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

void main(void)

{

unsigned char error,checksum;

unsigned char HUMI,TEMP;

HUMI=0X01;

TEMP=0X02;

initUARTtest(); //初始化串口

s\_connectionreset();

while(1)

{

error=0;

error+=s\_measure((unsigned char\*) &humi\_val.i,&checksum,HUMI); //湿度测量

error+=s\_measure((unsigned char\*) &temp\_val.i,&checksum,TEMP); //温度测量

temp\_value = temp\_val.i \* 0.01 - 39.6;

sprintf(temp\_result,"%s","temperature: ");

UartTX\_Send\_String(temp\_result, 13);

sprintf(temp\_result,"%9.2f",temp\_value);

temp\_result[9] = '\n';

UartTX\_Send\_String(temp\_result, 10);

humi\_value = humi\_val.i \* 0.0367 - 2.0468;

sprintf(humi\_result,"%s","humidity: ");

UartTX\_Send\_String(humi\_result, 10);

sprintf(humi\_result,"%9.4f",humi\_value);

humi\_result[9] = '\n';

UartTX\_Send\_String(humi\_result, 10);

if(error!=0)

{

s\_connectionreset() ; //如果发生错误，系统复位

led1 = !led1;

led2 = !led2;

}

else

{

humi\_val.f=(float)humi\_val.i; //转换为浮点数

temp\_val.f=(float)temp\_val.i; //转换为浮点数

calc\_sth11(&humi\_val.f,&temp\_val.f); //修正相对湿度及温度

//dew\_point=calc\_dewpoint(humi\_val.f,temp\_val.f); //计算绝对湿度值

}

// UartTX\_Send\_String(&error,1);

Delay1(50000); //延时

}

}

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

\*函数功能 ：延时

\*入口参数 ：定性延时

\*返 回 值 ：无

\*说 明 ：

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

void Delay1(uint n)

{

uint i;

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

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

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

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

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

}

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

\*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++;

while(UTX0IF == 0);

UTX0IF = 0;

}

}

#### 温湿COM通信代码

import serial

import sys

import urllib

import urllib2

import time

import json

apiurl = "http://192.168.199.187/sensordata/th"

data = {}

def post(url, data):

#data = urllib.urlencode(data)

req = urllib2.Request(url, data)

response = urllib2.urlopen(req)

res = response.read()

return res

while True:

str = ""

data = {}

try:

ser = serial.Serial('COM4', 9600)

except Exception, e:

print 'open serial failed.'

exit(1)

str = ser.readline()

str = ser.readline()

if str.split()[0] != 'temperature':

str = ser.readline()

data['temperature'] = float(str.split()[1])

str = ser.readline()

data['humidity'] = float(str.split()[1])

data = json.dumps(data)

print data

print post(apiurl, data)

sys.stdout.flush()

ser.close()

time.sleep(1)

#output: 2-tuple (temperture, humidity)