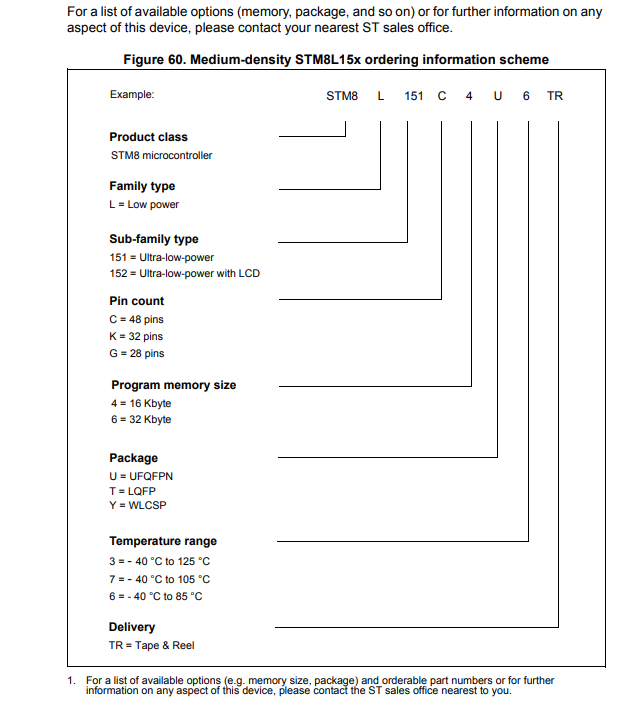
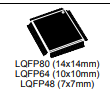
第二代包芯温度传感器设计方案

注：整个系统架构维持一代，只对传感器修改。此文档是开发前草案，后续有待更改和补充。

1. 包芯温度传感器修改动部分
2. MCU由原来的STM32L151C8T6改为STM8L151C8T6 。
3. 温度传感器由原来的温湿传感器HTU21D 为选用DS18B20温度传感器。
4. 增加470M信号放大电路。
5. 选用一节电池ER18505。
6. 传感器功能
7. 定时（定时周期可调）检测烟包包芯温度、芯片供电电压（即电池电压）及采集时间通过lora模块将数据传送给lora网关……；非工作时处于睡眠（低功耗）状态（睡眠时电流小于10uA）。
8. 数据上传：上传当前检测的温度数据及其检测时间。接收指令上传模块参数（当前频率，速率等）
9. 远程设置传感器参数：调整定时发送数据时间，通过平台下发相应指令，改变数据发送时间；测试和工作状态切换，模块参数修改速率SF的值，模块频率。
10. 簧管触发复位
11. 远程升级：当嵌入式软件有新版本需要升级时，可远程升级，（考虑到电池例用寿命，建议少用）
12. 串口打印信息打印相关状态：工作状态，模块是否入网等
13. 主要器件选型
14. **MCU**（为了满足超低功耗，2个USART，手工焊接方便经过讨论选用stm8L151C8T6），



**芯片简介：**

1. 8位超低功MCU、 64-KB Flash、4KB EEPROM 封装LQFP 48；
2. 通讯端口：3个USART（lora模块、测试各使用一个USART），一个I2C和一个Spi（flash及传感器选用）
3. 工作电压：1.65 to 3.6 V
4. 温度范围: -40 to 85
5. 功耗模式：等待，低功耗运行（5.9 µA），低功耗等待（3 µA），具有完全RTC的主动停止（1.4 µA），停止（400 nA）

进入低功耗部分参考代码：

//////////////////////////////

Radio->RFOpModeSet(RFLR\_OPMODE\_SLEEP); //进入休眠

ADC\_Cmd(ADC1,DISABLE); //ADC禁止

GPIO\_Config(DISABLE); //低功耗IO设置

CLK\_MainRegulatorCmd(DISABLE); //关闭电压调节器

PWR\_UltraLowPowerCmd(ENABLE); //超低功耗

halt();

////////////////////

1. MCU具有嵌入式可编程电压检测器（PVD），可监控VDD / VDDA电源并将其与VPVD阈值进行比较。

读取电压参考代码：

////////////////////////////////////////////////////////////////////////////////////////////

void Adc\_Init(void)

{

   CLK\_PeripheralClockConfig(CLK\_Peripheral\_ADC1,ENABLE);//开启ADC1时钟

   ADC\_VrefintCmd(ENABLE); //使能内部参考电压

   ADC\_Init(ADC1,ADC\_ConversionMode\_Continuous,ADC\_Resolution\_12Bit,ADC\_Prescaler\_1);//连续转换，12位，转换时钟1分频

   ADC\_ChannelCmd(ADC1,ADC\_Channel\_Vrefint,ENABLE);//使能内部参考电压通道

   ADC\_Cmd(ADC1,ENABLE);//ADC使能

}

////////////////////////////////////////////////////////////////////////////////////////////////

ADC\_SoftwareStartConv (ADC1); //开启软件转换

while(!ADC\_GetFlagStatus (ADC1,ADC\_FLAG\_EOC)); //等待转换结束

ADC\_ClearFlag (ADC1,ADC\_FLAG\_EOC); //清除对应标志

VCC\_Buf[VCC\_cnt++] = (float)ADC\_GetConversionValue (ADC1);//获取转换值

if( VCC\_cnt >= N)

{

VCC\_cnt = 0;

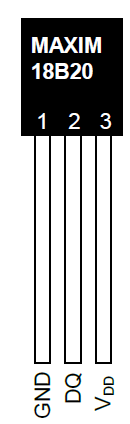
temp = filter(VCC\_Buf,N);

FD = (float)4095\*122.3/temp;

}

/////////////////////////////////////////////////////////////////////////////////////////////////

1. **温度传感器**选用DS18B20，封装选择TO92



连接方式：通过3芯护套线引出，DS18B20三个引脚之间用热缩套管隔离，再把三个管脚套在一个热缩套管固定保护好。

相关代码参考：

#define DS18B20\_DQ\_OUT GPIO\_Init(DQ\_PORT, DQ\_PINS, GPIO\_Mode\_Out\_PP\_High\_Fast)

#define DS18B20\_DQ\_IN GPIO\_Init(DQ\_PORT, DQ\_PINS, GPIO\_Mode\_In\_PU\_No\_IT)

#define DS18B20\_DQ\_High GPIO\_SetBits(DQ\_PORT,DQ\_PINS)

#define DS18B20\_DQ\_Low GPIO\_ResetBits(DQ\_PORT,DQ\_PINS)

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

\*复位18B20

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

void DS18B20\_Reset()

{

uint8\_t ErrorTime=10;

uint8\_t CY = 1;

while (CY)

{

DS18B20\_DQ\_OUT; //改变DQ引脚方向性为输出方式

DS18B20\_DQ\_Low; //送出低电平复位信号

delay\_us(240); //延时至少480us

delay\_us(240);

DS18B20\_DQ\_High; //释放数据线

delay\_us(60); //等待60us

DS18B20\_DQ\_IN; //改变DQ引脚方向性为输入方式

CY = GPIO\_ReadInputDataBit(DQ\_PORT,DQ\_PINS);

ErrorTime--;

if(ErrorTime == 0) break;

delay\_us(240); //等待设备释放数据线

delay\_us(180);

}

}

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

\*读一个字节

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

uint8\_t DS18B20\_ReadByte()

{

uint8\_t i;

uint8\_t dat = 0;

for (i=0; i<8; i++) //8位计数器

{

dat >>= 1;

DS18B20\_DQ\_OUT; //改变DQ引脚方向性为输出方式

DS18B20\_DQ\_Low; //DQ单总线拉低

delay\_us(2); //延时等待

DS18B20\_DQ\_High; //准备接收

delay\_us(2); //接收延时

DS18B20\_DQ\_IN; //改变DQ引脚方向性为输入方式

if(GPIO\_ReadInputDataBit(DQ\_PORT,DQ\_PINS)) dat |= 0x80;

delay\_us(60); //等待时间片结束

}

return dat;

}

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

\*写数据

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

void DS18B20\_WriteByte(uchar dat)

{

uint8\_t i;

DS18B20\_DQ\_OUT; //改变DQ引脚方向性为输出方式

for (i=0; i<8; i++) //8位计数器

{

DS18B20\_DQ\_Low; //DQ单总线拉低

delay\_us(2); //延时等待

if((dat&0x01)==1) DS18B20\_DQ\_High;

else DS18B20\_DQ\_Low;

delay\_us(60); //等待时间片结束

DS18B20\_DQ\_High; //恢复数据线

dat >>= 1; //送出数据

}

}

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

\*采集

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

float DS18B20()

{

uint8\_t TL=0,TH=0;

uint16\_t temp=0;

DS18B20\_Reset();

DS18B20\_WriteByte(0xcc);

DS18B20\_WriteByte(0x44);

// while(GPIO\_ReadInputDataBit(DS18B20\_DQ\_PORT,DS18B20\_DQ\_PIN))

// {

delay\_ms(5); //延时1毫秒

// if(++TL > 50) return 65535; //数据异常

// }

DS18B20\_Reset();

DS18B20\_WriteByte(0xcc);

DS18B20\_WriteByte(0xbe);

TL = DS18B20\_ReadByte(); //读温度低字节

TH = DS18B20\_ReadByte(); //读温度高字节

1. **模组：**保留一代，使用LSD4RF-2L717M90，这里不再描述
2. **传感器天线**：使用弹簧天线，尽量设计与PCB板垂直。

****

# 或柔性FPC线路板天线IPEX

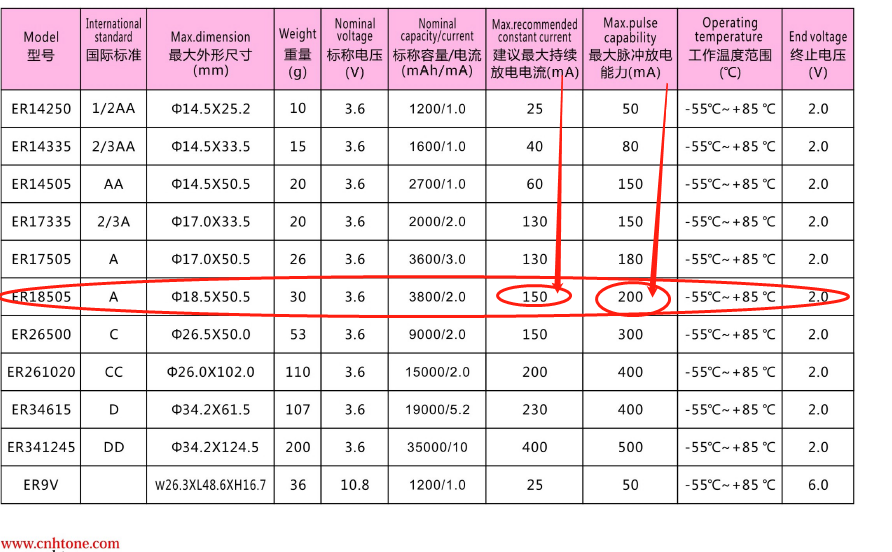


电路设计时预留IPEX。FPC天线使用时适合贴在壳外。

1. FLASH根据实际情况选型。

W25Q80BVSSIG

1. 电池，暂定使用ER 18505 容量型电池。参数如下：



1. 嵌入式软件

架构及逻辑保留一代，32位改为8位，去掉无关数据。

BootLoader 不变

主流程

开始

RTC时钟

.

sys初始化

定时时间到

采集数据

是

唤醒

发送数据

是

接收到对时

否

否

睡眠

1. PCB设计

制板要求：双面板，材质RF4，板厚1.6mm（可根据实际情况改为1.0 mm）过孔喷油，传统工艺，默认常规铜箔线路厚度1oz。

PCB外形为长方形，尽量小。

1. 外壳

根据电路板外型设计，注意天线，电池，板载干簧管位置。