程序使用说明

1. 硬件连接方式

模块与STM32F107的引脚连接如下所示:

MAX30102: ESP8266: RC522: PULSE SENSOR: DS18B20：

SCL PB6 TX PA10 SDA PB12 DATA PA1 DATA PC11

SDA PB7 RX PA9 SCK PB13

INT PB2 MOSI PB15

MISO PB14

RST PB1

LCD: COORDINATOR: KEY:

背光 PB0 P0.3 PA3 1 PB4

CS PD3 2 PB5

RS PD6

WR PD4

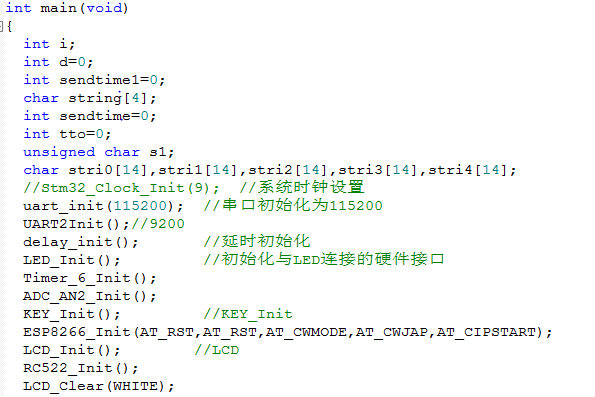
RD PD7

DATA PE0~PE15

SPI PB12~PB15

2、使用：

将所有的接线连接结束后，正式将程序烧录至STM32F107中，其烧录的过程不在赘述。

烧录后按复位键程序开始执行。由于程序总是从main函数开始执行，所以main函数中应该包括对所有外设的初始化。具体的初始化项目和初始化顺序根据使用设备的不同其过程也不同。以PULSE SENSOR为例。

程序内部定义了一些变量存储后面取得的数据，其具体数据不在此赘述。变量定义结束后进入各个外设的初始化。

(1)、uart\_init(115200);为串口1初始化的函数。因为串口1需要和ESP8266进行通信，而ESP8266的波特率固定为115200，所以串口波特率为115200。

(2)、UART2Init();为串口2的初始化函数。用于和蓝牙进行通信，波特率9200.原因同ESP8266。

(3)、delay\_init();延时初始化，其中延时用到的时钟为SYSTEMTICK即系统节拍定时器。需要注意的是其延时的最长时间约为1.85S，即如果需要的延时大于1.85，请重复使用延时函数。

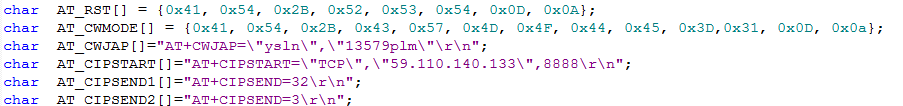
(4)、LED\_Init();LED灯初始化。其用来表明系统正在运行或者其他的一些特征。

(5)、Timer\_6\_Init();TIME6初始化，主要用于PULSE SENSOR中的延时。

(6)、ADC\_AN2\_Init();ADC初始化，主要PULSE SENSOR采集的模拟信号转换为数字信号，将其绘制成心电图，同时转换后的数字信号然后通过算法计算出心率、心跳间隔。

(7)、KEY\_Init();按键初始化。主要用于将采集后的数据通过按键这一特定的操作发送到云端。

(8)、ESP8266\_Init(AT\_RST,AT\_RST,AT\_CWMODE,AT\_CWJAP,AT\_CIPSTART);ESP8266初始化。ESP8266初始化的AT指令详见：网关板ESP8266的使用.PDF。需要注意的是，



上图中的char AT\_CWJAP[]="AT+CWJAP=\"ysln\",\"13579plm\"\r\n";必要的时候请更改为自己热点的名称、密码(例子中的热点名称为：ysln 密码为：13579plm)。

char AT\_CIPSTART[]="AT+CIPSTART=\"TCP\",\"59.110.140.133\",8888\r\n";这句话表示需要与哪台主机建立连接、端口号是多少(示例中的字符串表明与59.110.140.133这个主机的8888端口建立TCP连接)。除此以外发送到云平台的数据格式也需要注意。现给出一个示例。

捕获.2PNG

Data1[]={'3',',','I','D',':','0','0','0','0','0','0','0','0',',','B','=','0','0','0',',','Q','=','0','0','0',',','S','=','0','0','0','0'};其中字符串开始的3表示这是一个PULSE SENSOR的数据，即3表明数据类型；后面的ID:00000000将会在读到卡号后将00000000替换为当前的RFID卡号；后面的B:000在读取到心率后其中的000将会被替换;Q:000在读取到心跳间隔时数据将会填充进数组中；S:0000代表当前的心跳信号，网站端将会根据这个心率信号来绘制心电图。最后将数据上传到云端后，网站会自动根据数据类型来剥离数据。因此，千万不要将数据类型弄错。此例中，会把卡号、心率、心跳间隔、心率信号数据剥离下来，然后看卡号是否被注册，若未被注册将会丢掉整个报文；若已注册，将会将数据存储到对应用户的数据库中并将实时绘图显示。

(9)、LCD\_Init();显示屏初始化。

(10)、RC522\_Init(); 读卡器初始化。

其他程序源码与此类似。