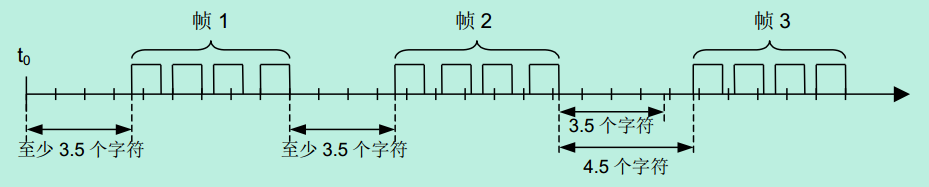
近有用到modbus主机部分，网上搜索了一圈，没找到好用的现成的开源代码。之前用过freemodbus，只有从机的源代码是免费的，其他的都需要商业授权。既然这样，那就自己动手，丰衣足食……自己编写个modbus的主机代码，并且开源出来。

modbus就不多介绍了，是工业上常用的通信协议。在物理层， Modbus 串行链路系统可以使用不同的物理接口(RS485、 RS232)。最常用的是RS485 两线制接口。

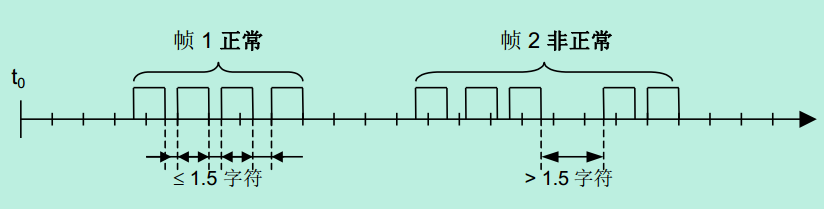
modbus又是一个主从协议：同一时刻，在总线上只能有一个主机，其他作为从机。并且从机不会主动发数据，都是主机给从机发一条，从机才会回复一条。所以主机的发送和接收也是不同步进行的。如下流程：

主机发送——》等待从机应答——》接收到从机应答——》应答处理——》回到初始状态

以上的主要流程请记好，也是编写modbus主机代码的指导方针。下面再看下modbus是如何区分一帧一帧的数据的（这里只是介绍RTU模式的,ASSCII暂不做讨论)。如下图所示，帧1与帧2之间要至少有3.5个字符的时间，我们把它简称为3.5T。意思也就是在收到数据直到总线上空闲超出了3.5t，就意味着这一帧数据接收完成了。我们就可以把刚才收到的一帧进行CRC计算解析看一帧数据接收是否正确。

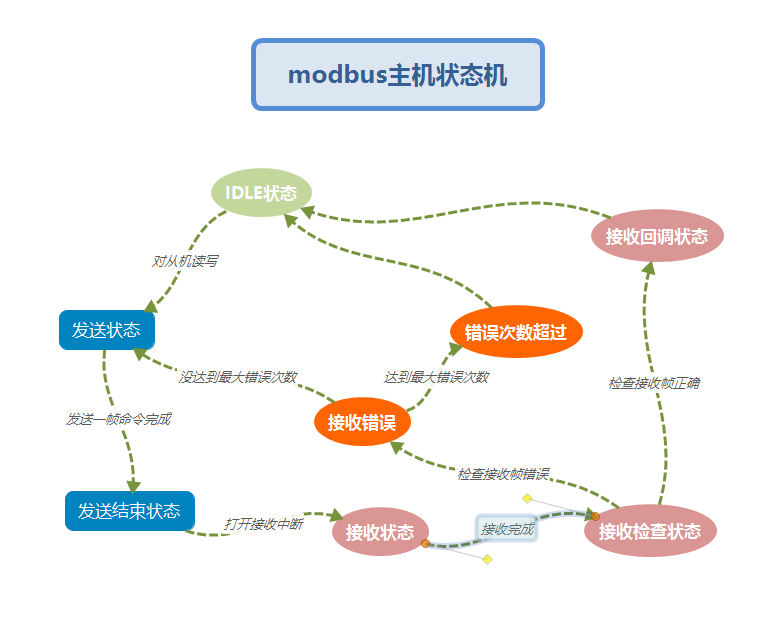


实际上除了如上规定的帧与帧之间必须间隔3.5T外，在一帧内字符字符之间要小于1.5T。如下图所示：



但是在实际运行测试中，这种字符之间超1.5T的情况概率很小，即便出现也能在CRC校验时把错误的帧过滤掉。所以我编写的modbus主机并没有进行字符之间是否超过1.5T的校验。而只做了3.5T的帧区分。

至此基本概念介绍完了，先看下我编写的modbus主机的主要状态流程图：



如上图所示，对应代码中定义的如下状态,状态的切换也是modbus主机的主要核心机制：

typedef enum  
 {  
 MBH\_STATE\_IDLE=0X00,  
 MBH\_STATE\_TX,  
 MBH\_STATE\_TX\_END,  
 MBH\_STATE\_RX,  
 MBH\_STATE\_RX\_CHECK,  
 MBH\_STATE\_EXEC,  
 MBH\_STATE\_REC\_ERR, //接收错误状态  
 MBH\_STATE\_TIMES\_ERR, //传输  
 }mb\_host\_state;

这篇文章主要针对代码的核心思想进行分析，处理细节在这里不做陈述，请自己参考源码，下面就针对核心的一个个状态进行讨论：

**MODBUS主机空闲状态（MBH\_STATE\_IDLE）**

这个很好理解，在一上电后就会进入到空闲状态。当对接收数据处理完或者读写从机错误次数达到最大值得时候也会回到这个初始状态。只有状态处于IDLE时才能发起新的一轮对从机的读写。

**MODBUS主机发送状态（MBH\_STATE\_TX）**

从状态切换图可以看得出来，当主机需要对从机进行读写的时候就会首先进入TX状态，该状态下回打开串口TX中断，然后通过发送中断把一帧数据发送完成。

**MODBUS发送完成（MBH\_STATE\_TX\_END）**

当一帧数据发送完成时就会进入到TX\_END状态。这时候主机发给从机的命令已经完成了，就等待从机的应答了。打开串口接收中断切换到下一个状态去

**MODBUS接收（MBH\_STATE\_RX）**

进入到RX状态下，会完成对从机回应的一帧数据的接收，都是在串口接收中断中处理。如果3.5T超时了就证明接收一帧完成，切换到下一个状态进行校验处理

**MODBUS接收检查（MBH\_STATE\_RX\_CHECK）**

在rx check状态下主要完成对接收一帧数据的合法性和CRC进行计算，看接收是否正确。如果接收正确了，就进入到回调处理。如果校验错误就进入到接收错误状态。

**MODBUS回调处理（MBH\_STATE\_EXEC）**

不管是对主机的读还是写，从机都会回复。在该状态下根据从机回复的不同功能码就会执行不同的回调函数。执行完回调再次回到初始的IDLE状态，这样主机对从机的一次读或者一次写就完成了。

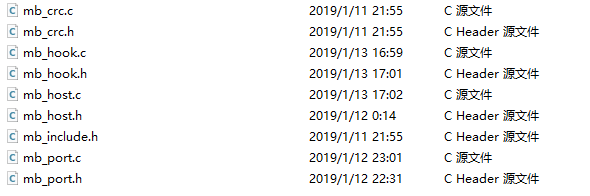
**MODBUS接收错误（MBH\_STATE\_REC\_ERR）**

当接收校验错误的时候就会进入到该状态，在该状态下会对错误的次数进行计数。如果连续错误次数超过了最大值，就进入到超过错误次数状态。如果还没达到最大错误次数，就重新回到tx状态，重头再执行一遍对从机的读写动作。

**MODBUS超错误次数（MBH\_STATE\_REC\_ERR）**

经过几次对从机的读写操作都失败以后就进入到该状态，在该状态下执行超错误次数回调，之后回到最初的IDLE状态等待下一次的读写操作。

最后是对源码文件目录进行一个大概的说明，整个modbus主机代码如下构成：



* mb\_crc.c/h 这个没啥解释的，主要放CRC计算部分的代码
* mb\_hook.c/h 该文件中存放回调函数接口，所有用户的回调处理都在这里添加。这个需要用户自己添加
* mb\_port.c/h 这个也是需要用户自己修改的文件，针对使用的不同的MCU平台进行移植，主要移植工作就是串口初始化、串口tx、rx中断开启，定时器初始化、定时器开始停止接口
* mb\_host.c/h 核心处理部分代码，不建议用户修改，上面所述的状态机切换处理全部在该文件中完成。用户层最终调用的API接口也在该文件中定义。如下：

void mbh\_init(uint32\_t baud,uint8\_t parity);  
 int8\_t mbh\_send(uint8\_t add,uint8\_t cmd,uint8\_t \*data,uint8\_t data\_len);  
 void mbh\_poll(void);   
 *void mbh\_timer3T5Isr(void);*   
 *void mbh\_uartRxIsr(void);*   
 void mbh\_uartTxIsr(void);