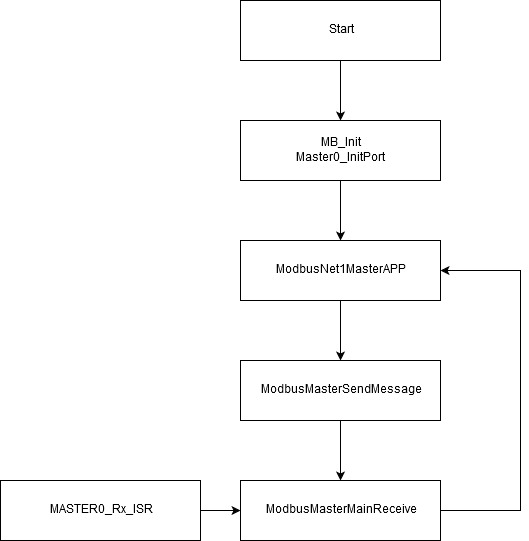
OpenModbus主站使用

1. 主站流程分析



MB\_Init(): 主要用于初始化主站状态机的相关变量

Master0\_InitPort(MASTER\_PORT0): 主要用于初始化UART口，以及挂载主站需要用到的指针函数。

ModbusNet1MasterAPP(): 应用处理函数，主要用于定义从站信息以及读写相关设备。

ModbusMasterSendMessage(MASTER\_PORT0): 用于生成发送报文，并将报文发送出去，进入发送状态后，会进行从站超时判断。

ModbusMasterMainReceive(MASTER\_PORT0)：数据接收函数，当中断接收完一帧数据后，会在中断服务程序中置标志，此函数查询该标志，并将数据根据用户的设定Copy到相应地址。

MASTER0\_Rx\_ISR：中断接收函数。

红色字体表示用户需要根据应用设定

橙色字体表示不更换平台用户只需调用，无需修改

绿色字体表示无需用户修改

1. 主站相关结构体说明

OpenModbus使用MODBUS\_PROCESStyp结构体封装了主站对从站的配置信息：

typedef struct

{

uint8\_t MasterStatus; // 1,表示开始

uint8\_t Error; // 0 OK, 1 Error

uint16\_t Offset; //用户数据区偏移

uint16\_t Number; //寄存器个数

uint16\_t RegisterAddress; //寄存器地址

uint8\_t SlaveAddress; //从站地址

uint32\_t TimeOut; //超时时间

uint32\_t SendTime; //发送时间

uint8\_t Function; //功能码

uint8\_t \*MBUF;

}

MODBUS\_PROCESStyp;

MasterStatus表示当前主站的状态，相关定义如下，0表示发送完成，1表示应用层处理完毕，底层查询到该标志后会进行发送，并设置状态为2

#define MODBUS\_MASTER\_STATUS\_END 0

#define MODBUS\_MASTER\_STATUS\_START 1

#define MODBUS\_MASTER\_STATUS\_SENDING 2

Error表示当前主站遇到的错误状态，相关定义如下：

#define MODBUS\_ERROR\_OK 0

#define MODBUS\_ERROR\_MEMORY 1

#define MODBUS\_ERROR\_OPENFAILD 2

#define MODBUS\_ERROR\_REGISTER\_ADDRESS\_ODD 3

#define MODBUS\_ERROR\_TIMEOUT\_LIMIT 4

#define MODBUS\_ERROR\_SLAVEADDR 5

#define MODBUS\_ERROR\_REGISTER\_ADDRESS\_OVERFLOW 6

#define MODBUS\_ERROR\_NUMBER 7

#define MODBUS\_ERROR\_FUNCTION 8

#define MODBUS\_ERROR\_TIMEOUT 9

#define MODBUS\_ERROR\_PROTOCOL 10

Offset表示各个从站在用户数组g\_u8MasterBuf的偏移，g\_u8MasterBuf[a][b]是一个二维数组a表示主站的网络号，b表示该网络定义的用户数据区最大值。举例说明，如果定义需要读取8个字节从站数据到用户g\_u8MasterBuf[0][10]地址，则需要设置Offset为10，协议栈会自动将从站返回的8个字节写入g\_u8MasterBuf[0][10]~[17]

Number表示需要读取从站的寄存器个数

RegisterAddress表示需要读取从站的寄存器地址

SlaveAddress表示从站地址

TimeOut表示给每个从站设定的超时时间

SendTime保留未用

Function表示对从站访问的功能码

1. 主站工作流程说明

主站示例代码通过MODBUS\_TASK\_STRUC\_T结构体封装了Modbus主站的状态机信息：

typedef struct

{

uint8\_t status;

uint8\_t pending\_status;

uint8\_t next\_status;

uint32\_t time;

uint32\_t delay;

}MODBUS\_TASK\_STRUC\_T;

Status：表示主站的状态，主站访问从站设备，往往需要访问多个设备或者同一个设备的不同地址，而且对设备同时存在读写两种操作，所以通过该标志来定义主站的状态。

pending\_status：表示当前主站的状态，用于从站访问异常后能重复当前状态。

next\_status：表示当前访问操作成功后，下一个操作状态。

Time: 表示当前的处理时间，主要用于判断从站超时

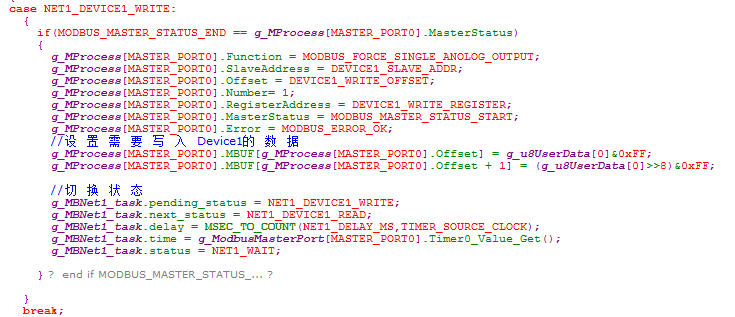
Delay: 表示主站发送请求报文后，延时一段时间在处理接收到的数据，单位ms

1. 主站示例分析：

ModbusNet1MasterAPP()函数是处理主站的用户示例函数，假设有两个从站设备（地址分别为0x01， 0x02），首先通过0x06功能码写数据0x1234到设备地址为1的从站0寄存器地址，然后读取设备1的1寄存器地址到用户g\_u8MasterBuf[00][02]偏移，之后写0x5678到设备2的0寄存器地址，之后读取设备2的1寄存器地址。这样模拟了两个设备的读写操作这4种状态，用户可以根据自己的应用在此例基础上扩展。本示例如果当前状态设备没有响应或响应帧错误，则继续本状态，不会进入下一个状态，这部分代码是在NET1\_WAIT状态中判断的，用户可以根据自身需要来修改：



ModbusNet1Checking()函数是用于判断当前从站回复数据，如果当前状态机是读从站数据，可以在这里直接处理用户数据。



写从站状态机需要完成如下内容：

g\_MProcess[MASTER\_PORT0].Function = 访问从站需要使用的功能码

g\_MProcess[MASTER\_PORT0].SlaveAddress = 从站地址

g\_MProcess[MASTER\_PORT0].Offset = 用户数据偏移

g\_MProcess[MASTER\_PORT0].Number = 寄存器访问个数

g\_MProcess[MASTER\_PORT0].RegisterAddress = 从站寄存器地址

g\_MProcess[MASTER\_PORT0].MasterStatus = MODBUS\_MASTER\_STATUS\_START;

g\_MProcess[MASTER\_PORT0].Error = MODBUS\_ERROR\_OK;

g\_MProcess[MASTER\_PORT0].MBUF[g\_MProcess[MASTER\_PORT0].Offset]

g\_MProcess[MASTER\_PORT0].MBUF[g\_MProcess[MASTER\_PORT0].Offset+1]

将需要写入的变量放到MBUF对应的偏移

//切换状态

g\_MBNet1\_task.pending\_status = NET1\_DEVICE1\_WRITE;

g\_MBNet1\_task.next\_status = NET1\_DEVICE1\_READ;

g\_MBNet1\_task.delay = MSEC\_TO\_COUNT(NET1\_DELAY\_MS,TIMER\_SOURCE\_CLOCK);

g\_MBNet1\_task.time = g\_ModbusMasterPort[MASTER\_PORT0].Timer0\_Value\_Get();

g\_MBNet1\_task.status = NET1\_WAIT;

