ModbusTCP协议报文解析

报文格式

交互（通信）标识：2个字节 为此次通信事务处理标识符，一般每次通信之后将被要求加1以区别不同的通信数据报文。
协议标识：2个字节 表示该条指令遵循ModbusTCP协议，一般都为00 00
报文长度：2个字节 表示后面数据的长度，有几个字节，高字节在前
（前六位Modbus/TCP协议不同功能码通用）

设备标识： 1个字节 设备地址，这个可以用于局域网里面的具体的地址，如果目标机器有固定ip，这个就不起作用，直接上写成 00
功能码：1个字节 功能码在modbus协议用于表示信息帧的功能
数据：N个字节 后面数据根据不同功能码不同。

modbus 常用功能代码

十进制	功能	数据类型
01	读取 多个线圈	位
02	读取 多个离散量输入量	位
03	读取 多个保持寄存器	16进制整型
04	读取 多个输入寄存器	16进制整型
05	写入 单个线圈	位
06	写入 单个寄存器	16进制整型
15	写入 多个线圈	位
16	写入 多个寄存器	16进制整型

功能码详解

01 读取多个线圈

示例报文：

请求：`00 01 00 00 00 06 FF 01 00 01 00 10`

第1，2位 `00 01` 交互标识
第3，4位 `00 00` 协议标识
第5，6位 `00 06` 后面报文长度 有6位
第7位 FF 设备地址，发送什么，响应什么
第8位 01 功能码

第9，10位`00 01` 起始地址

第11，12位 `00 10`查询线圈长度，查询16位线圈

响应：`00 01 00 00 00 05 FF 01 02 0A 02`

前四位和7，8位同请求发送的报文

第5，6位是后面报文长度

第9位是后面数据位的长度，

第10位开始是数据位。

01查询线圈，每一个16进制数据表示8位线圈

第10位0A --> 0000 1010 第二位是1，第四位是1

02 读取多个离散量输入

同01

03 读取 多个保持寄存器

寄存器读取与线圈的区别，响应数据，寄存器数据每两个字节表示1位，一次请求不能超过127个地址
示例报文：

请求：`00 01 00 00 00 06 01 03 00 05 00 02`
第1，2位 `00 01` 交互标识
第3，4位 `00 00` 协议标识
第5，6位 `00 06` 后面报文长度 有6位
第7位 `01` 设备地址，发送什么，响应什么
第8位 `03` 功能码

第9，10位 `00 05` 起始地址
第11，12位 `00 02`查询寄存器长度，查询2个寄存器

响应：`00 01 00 00 00 07 01 03 04 00 22 00 00`
前四位（`00 01 00 00`）和7，8位（`01 03`）同请求发送的报文
第5，6位 `00 07` 是后面报文长度
第9位 04 是后面数据位的长度
第10-13位 数据位（`00 22 00 00`）

04 读取 多个输入寄存器

同03

05 （05H）写入 单个线圈

请求：`00 01 00 00 00 06 FF 05 00 01 FF 00`

第9，10位 `00 01` 写入线圈的地址

第11，12位 写入的数据值 FF 00 表示置ON/1状态 00 00 表示置OFF/0状态

响应：`00 01 00 00 00 06 FF 05 00 01 FF 00`

15（0FH）写入 多个线圈

请求：`00 01 00 00 00 09 FF 0F 00 05 00 0A 02 CD 01`

第9，10位 `00 05` 写入的起始地址

第11，12位 `00 0A` 写入线圈数量

第13位 `02` 数据字节数量

第14位之后是数据 低字节在前

CD 01 —> 1100 1101 0000 0001

位	0C	0B	0A	09	08	07	06	05
值	1	1	0	0	1	1	0	1
位	14	13	12	11	10	0F	0E	0D
值	0	0	0	0	0	0	0	1

06（06H） 写入 单个寄存器

请求：`00 05 00 00 00 06 FF 06 00 05 00 23`

向地址为5的寄存器写入35。
第9，10位 00 05写入的起始地址
第11，12位 00 0A写入寄存器的值

响应：`00 05 00 00 00 06 FF 06 00 05 00 23`

16（10H）写入多个寄存器

请求：`00 06 00 00 00 0B FF 10 00 02 00 02 04 00 21 00 2A`

从地址2开始写入两个寄存器，2寄存器写入33，3寄存器写入42
第9，10位 00 02写入的起始地址
第11，12位 00 02写入寄存器的数量
第13位 04 后面数据的字节
第14-17位 数据

响应：`00 06 00 00 00 06 FF 10 00 02 00 02`

从地址2开始写入两个寄存器，2寄存器写入33，3寄存器写入42

第9，10位 `00 02` 写入的起始地址

第11，12位 `00 02` 写入寄存器的数量

第13位 04 后面数据的字节

第14-17位 数据

响应：`00 06 00 00 00 06 FF 10 00 02 00 02`

错误响应

当发生通讯异常时，响应前7位仍然为modbus正常协议格式，第八位响应功能码（请求功能码+0x80），第九位异常码。

异常数据即包含异常码的数据：
目前使用的异常码是：01，02，03和04。

- 响应功能码 = 请求功能码 + 0x80
- 响应报文提供异常码显示出错原因。

常见异常码含义：

异常码	名称	含义
01	非法功能	对于服务器（或从站）来说，询问中接收到的功能码是不可允许的操作，可能是因为功能码仅适用于新设备而被选单元中不可实现同时，还指出服务器（或从站）在错误状态中处理这种请求，例如：它是未配置的，且要求返回寄存器值。
02	地址非法	对于服务器（或从站）来说，询问中接收的数据地址是不可允许的地址，特别是参考号和传输长度的组合是无效的。对于带有100个寄存器的控制器来说，偏移量96和长度4的请求会成功，而偏移量96和长度5的请求将产生异常码02。
03	数据非法	对于服务器（或从站）来说，询问中包括的值是不可允许的值。该值指示了组合请求剩余结构中的故障。例如：隐含长度是不正确的。modbus协议不知道任何特殊寄存器的任何特殊值的重要意义，寄存器中被提交存储的数据项有一个应用程序期望之外的值。
04	从站设备故障	当服务器（或从站）正在设法执行请求的操作时，产生不可重新获得的差错。

特殊数据处理

float 按IEEE-754标准协议存储

C#中浮点数的二进制格式遵循IEEE754标准

IEEE-754格式标准：一个浮点数有2部分组成：底数m和指数e
IEEE-754

寄存器地址

Modbus协议定义的寄存器地址是5位十进制地址，即:
线圈(DO)地址:00001~~09999~~
~~触点(DI)地址:10001~~19999
输入寄存器(AI)地址:30001~~39999~~
~~输出寄存器(AO)地址:40001~~49999

0x代表线圈(DO)类地址，1x代表触点(DI)类地址、 3x代表输入寄存器(AI)类地址、4x代表保持寄存器(AO)类地址。
在实际编程中，前缀的区分作用（有功能码进行区分），所以只需说明后4位数，而且需转换为4位十六进制地址。
Modbus 数据地址格式是从0开始，寄存器地址对应报文中地址关系，`x0001`对应`00 00`，示例`40003`对应`00 02`地址

以上根据开发时查的资料和网上资料整理一些的有用信息,方便开发查询