UT3510+系列直流低电阻测试仪

编程手册

V1.0

2024-07-22

UNI-T®

保证和声明

版权

商标信息

UNI-T是优利德中国科技有限公司的注册商标。

文档编号

声明

产品认证

UNI-T认证本产品符合中国国家产品标准和行业产品标准及 ISO9001：2008 标准和 ISO14001：2004 标准，并进一步认证本产品符合其它国际标准组织成员的相关标准。

1. SCPI 指令简介

SCPI（Standard Commands for Programmable Instruments，即可编程仪器标准命令集）是一种建立在现有标准 IEEE 488.1 和 IEEE 488.2 基础上，并遵循了 IEEE754 标准中浮点运算规则、ISO646 信息交换 7 位编码符号（相当于 ASCII 编程）等多种标准的标准化仪器编程语言。本节简介 SCPI 命令的格式、符号、参数和缩写规则。

1.1 简介

本章将对所有的 UT3510+系列RS232C 命令进行详细介绍。这些命令均符合 SCPI 标准命令集。每个命令的介绍将包含如下内容：
命令名称： SCPI命令的名称。
命令语法： 命令的格式包括所有必需的和可选的参数。
查询语法： 查询的格式包括所有必须的和可选的参数。
查询返回： UT3510+系列的返回数据格式。

1.2 符号约定和定义

本章 RS232C 命令的描述采用如下的符号约定和定义。
< > 尖括号中的内容用于表示命令的参数。
[ ] 方括号中的内容是可选的，可以省略。
{ } 通常花括号中包含几个可选参数，只能选择其中的一个参数。
在命令中将会用到的下列符号定义:
<NL> 换行符(十进制 10)。
空格 单 ASCII 字符(十进制 0-9, 11-32)。
例如, 回车(十进制 13) 或 空格(十进制 32)。

1.3 命令结构

UT3510+系列命令分为两种类型：公用命令和 SCPI 命令。 公用命令由 IEEE 标准定义适用于所有的仪器设备。 SCPI 命令采用三层的树状结构，最高层称为子系统命令。只有选择了子系统命令后，该子系统命令的下层命令才有效。冒号（：）用于分隔高层命令和低层命令。

树状命令基本规则如下：

- 忽略大小写。
  例如，
  LIMIT:NOMINAL <value> = limit:nominal <value> = LiMiT:NoMiNaL <value>
- 空格（␣ 表示一个空格）不能位于冒号的前后。
  例如，
  错误：LIMIT␣:␣NOMINAL <value>
  正确：LIMIT:NOMINAL <value>
- 命令可以是单词的缩写，也可以是完整拼写的单词。
  例如，
  LIMIT:NOMINAL <value> = LIM:NOM <value>
- 命令后面加一个问号（ ? ）构成该命令的查询命令。
  例如，
  LIMIT:NOMINAL_C ?

分号（ ; ）可以用于分隔同一命令行上的多个命令，多重命令的规则如下：

■ 在一个多重命令行上，可使用分号（ ; ）来分隔同一子系统命令下的同级别的多个命令。
例如，
LIMIT:NOMINAL <value>; BIN <n> <low limit>,<high limit>
■ 分号（ ; ）分隔符后面紧跟一个冒号（ : ）表示后面的命令重新从命令树的顶层开始。
例如，
LIMIT:NOMINAL <value>;:LIMIT:BIN <n> <low limit>,<high limit>

1.4 命令缩写规则

每个命令和特性参数至少拥有两种拼写形式，缩写形式和全拼形式。有些时候两种拼写方式完全相同。遵守以下规则进行缩写。

■ 如果单词的长度为四个字母或少于四个字母，则缩写形式和全拼形式相同。
■ 如果单词的长度大于四个字母，

如果第四个字母是个元音字母，那么缩写形式为该单词的前三个字母。

如果第四个字母是个辅音字母，那么缩写形式为该单词的前四个字母。例如:

LIMIT 可缩写成 **LIM**。
RANGE 可缩写成 **RANG**。
FREQUENCY 可缩写成 **FREQ**。

■ 如果要缩写的不是一个单词而是一个短语，那么全拼形式为前面单词的首个字母加上最后一个单词的完整
拼写。在全拼形式的基础上利用上述规则进行缩写，可得到其缩写形式。
例如:
短语 Source RESistor 的全拼形式为 **SRESISTOR**，根据上述规则可缩写为 **SRES**。

1.5 命令题头和参数

UT3510+系列控制命令包含命令题头和相关参数。命令题头可以是全拼或缩写形式。使用全拼方式便于理解命令的意思，而使用缩写方式可以提高计算机输入效率。参数可以为如下两种形式之一。

■ 字符数据和字符串数据
字符数据由 ASCII 字母构成。缩写规则与命令题头相同。字符串数据由加双引号 （“” ） 的 ASCII 字符构成。
■ 数值数据
整数（NR1），定点数（NR2），或浮点数（NR3），数值范围为 ±9.9E37。
NR1 举例如下：
123
+123
-123
NR2 举例如下：
12.3
+1.234
-123.4
NR3 举例如下：
12.3E+5
123.4E-56

2. SCPI 命令参考

2.1 DISPlay 显示子系统

DISPlay 子系统用来切换不同的显示页面或在页面提示栏上显示一串文本。

DISPlay 子系统树

DISPlay	:PAGE	{TEST,SETUP(MSET),COMParator,FILE,SYSTem,SYSTEMINFO(SINF)}
	:LINE	<string>

DISPlay:PAGE 命令用于设定显示方式。DISPlay:PAGE?查询返回当前的测试结果显示方式的设置。

命令语法： DISPlay:PAGE
{TEST,SETUP(MSET),COMParator,CORRection,FILE,SYStem,SYSTEMINFO(SINF)}

参数： TEST： 测量页面。
SETUP(MSET)：设置页面
COMParator：比较页面。
FILE：文件管理页面
SYSTem： 系统配置页面。
SYSTEMINFO(SINF)：系统信息页面

例如发送： DISPlay:PAGE TEST<NL> //切换到测试页面

查询语法： DISPlay:PAGE?

查询响应： {test,comp,syst,file,sinf,mset}<NL>

2.2 FUNCtion 子系统命令

使用 FUNCtion 子系统设置的参数，仪器将不会保存在系统中，下次开机需要重新设置。此外，参数的设置应与当前的测试模式保持一致。

FUNCtion 子系统树

FUNCtion	:RANG	{量程号,max,min}		
	:RATE	{SLOW,MEDium,FAST,HIGH }		
	:MODE	{AUTO,HOLD,NOMinal}		
	:IMP	{R,RT,T,LPR,LPRT}		
	:LPR	:RANG		{量程号,max,min}
	:LPR	:RANG	:MODE	{AUTO,HOLD,NOMinal}

FUNCtion:RANGe

FUNC:RANG 用来设置量程方式和量程号

命令语法： FUNCtion:RANGe {<量程号>,min,max}

参数： <量程号>： 0-8 (UT3516+)或者
0-6 (UT3513+)。
min： 最小量程。
max： 最大量程。

例如发送： FUNCtion:RANGe 5<NL> //切换到量程 5（2k）

查询语法： FUNCtion:RANG?

查询响应： {<量程号>： 0-8 (UT3516+)或者

0-6 (UT3513+)}<NL>

FUNCtion:RANGe:MODE 命令

FUNCtion:RANGe:MODE 命令用于用来切换量程方式。FUNCtion:RANGe:MODE? 查询返回当前量程状态。

命令语法：	FUNCtion:RANGe:MODE{AUTO,HOLD(MANual),NOMinal}
参数：	AUTO： 自动量程。 HOLD(MANual)： 手动量程。 NOMinal： 标称量程。
例如发送：	FUNCtion:RANGe:MODE AUTO<NL> //切换到自动量程模式
查询语法：	FUNCtion:RANGe:MODE?
查询响应：	{AUTO,HOLD,NOMinal}<NL>

FUNCtion:RATE 命令

FUNCtion:RATE 或 FUNC:SPEED 用来设置测试速度。FUNCtion:RATE? 查询返回当前测试速度。		FUNCtion:RATE 或 FUNC:SPEED 用来设置测试速度。FUNCtion:RATE? 查询返回当前测试速度。	
命令语法：	FUNCtion:RATEE{SLOW,MEDium,FAST,HIGH}	命令语法：	FUNCtion:RATE{SLOW,MEDium,FAST,HIGH}
参数：	SLOW：慢速。	参数：	SLOW：慢速。
	MEDium：中速。		MEDium：中速。
	FAST：快速。		FAST： 快速。
	HIGH：高速。		HIGH：高速。
例如发送：	FUNCtion:RATE SLOW<NL> //切换到慢速模式	例如发送：	FUNCtion:RATE SLOW<NL> //切换到慢速模式
查询语法：	FUNCtion:RATE?	查询语法：	FUNCtion:RATE?
查询响应：	{SLOW,MEDium,FAST,HIGH}<NL>	查询响应：	{SLOW,MEDium,FAST,HIGH}<NL>
FUNCtion:RATE 或 FUNC:SPEED 用来设置测试速度。FUNCtion:RATE? 查询返回当前测试速度。		FUNCtion:RATE 或 FUNC:SPEED 用来设置测试速度。FUNCtion:RATE? 查询返回当前测试速度。	

FUNCtion:IMP 命令

FUNCtion:IMP 用来设置测试模式。FUNCtion:IMP? 查询返回当前测试模式。

命令语法：	FUNCtion:IMP{R,RT,T,LPR,LPRT}
参数：	R： R 模式。 RT： RT 模式。

	T: T 模式。 LPR: LPR 模式。 LPRT LPRT 模式
例如发送：	FUNCtion:IMP T<NL> //切换到 T 模式
查询语法：	FUNCtion:IMP?
查询响应：	{R,RT,T,LPR,LPRT}<NL>

FUNCtion:LPR:RANGe 命令

FUNCtion:LPR:RANGe 用来设置 LPR 模式量程号。FUNCtion:LPR:RANGe? 查询返回当前 LPR 模式量程号。

命令语法：	**FUNCtion:LPR:RANGe{<量程号>,min,max}**
参数：	<量程号>： 0-3 min： 最小量程。 max： 最大量程。
例如发送：	**FUNCtion:LPR:RANGe 1<NL>** //切换到 LPR 模式量程 1
查询语法：	**FUNCtion:LPR:RANGe?**
查询响应：	**{0,1,2,3}<NL>**

FUNCtion:LPR:RANGe:MODE 命令

FUNC:LPR:RANG:MODE 用来设置 LPR 模式的量程方式。FUNC:LPR:RANG:MODE? 查询返回当前 LPR 模式的量程方式。

命令语法：	**FUNCtion:LPR:RANGe:MODE{AUTO,HOLD(MANual),NOMinal}**
参数：	AUTO: 自动量程。 HOLD(MANual): 手动量程。 NOMinal: 标称量程。
例如发送：	**FUNCtion:LPR:RANGe:MODE NOMinal<NL>** //切换 LPR 模式的标称量程
查询语法：	**FUNCtion:LPR:RANGe:MODE?**
查询响应：	**{AUTO,HOLD,NOM}<NL>**

2.3 COMParator 子系统命令

使用 COMParator 子系统设置的比较器参数。COMP 子系统用来设置比较器参数。

COMParator 子系统树

COMParator	:STATe	{OFF,1-BIN,2-BIN,3-BIN,4-BIN,5-BIN,6-BIN}
	:BEEP	{OFF,PASS(OK),FAIL(NG)}
	:MODE	{ABS,PER,SEQ}
	:NOMinal	<float>
	:BIN	<1~6>,<float lower>,<float upper>

COMParator:STATe 命令

COMParator:STATe 命令用来关闭比较器或设置档位数。COMParator:STATe? 查询返回当前比较器或档位数状态。

命令语法：	**COMParator:STATe{0,1,2,3,4,5,6}**
参数：	0： 比较器关闭。 1:打开比较器，并设置为 1 档分选 2:打开比较器，并设置为 2 档分选 3:打开比较器，并设置为 3 档分选 4:打开比较器，并设置为 4 档分选 5:打开比较器，并设置为 5 档分选 6:打开比较器，并设置为 6 档分选
例如发送：	**COMParator:STATe 1<NL>** //切换到 1 档
查询语法：	**COMParator:STATe?**
查询响应：	**{0,1,2,3,4,5,6}<NL>**

COMParator:BEEP 命令

COMParator:BEEP 命令设定讯响功能状态。COMParator:BEEP? 查询返回当前讯响状态。

命令语法：	**COMParator:BEEP {OFF,OK,NG}**
参数：	OFF： 讯响功能关闭。 OK： 测试结果合格发出讯响。 NG： 测试结果不合格发出讯响。
例如发送：	**COMParator:BEEP OFF<NL>** //讯响关闭
查询语法：	**COMParator:BEEP?**
查询响应：	**{OFF,OK,NG}<NL>**

COMParator:MODE 命令

COMParator:MODE 命令设定比较模式。COMParator:MODE? 查询返回当前比较模式。

命令语法：	**COMParator:MODE {ABS,PER,SEQ}**
参数：	ABS： 绝对偏差模式。 PER： 百分比偏差模式。 SEQ： 直读模式
例如发送：	**COMParator:MODE ABS<NL>** //设定为绝对值偏差模式
查询语法：	**COMParator:MODE?**
查询响应：	**{ABS,PER,SEQ}<NL>**

COMParator:NOMinal 命令

COMParator:NOMinal 命令设定当前标称值。比较器功能利用该标称值来计算绝对偏差及百分比偏差 COMParator:NOMinal? 查询返回当前标称值。

命令语法：	**COMParator:NOMinal <float>**
参数：	<float> 为 NR1,NR2,NR3 形式的标称值
例如发送：	**COMParator:NOMinal 1.2000<NL>** //标称值设定为 1.2Ω **COMParator:NOMinal 1E3<NL>** //标称值设定为 1k **COMParator:NOMinal 1000<NL>** //标称值设定为 1k
查询语法：	**COMParator:NOMinal?**
查询响应：	**<NR3><NL>**

COMParator:BIN<n>命令

COMParator:BIN<n> 命令设定当前 BIN<n> 上下限。COMParator:BIN<n>? 查询返回当前档上下限。

命令语法：	**COMParator:BIN<n> <low limit>,<high limit>**
参数：	<n> 为 1 to 6 (NR1), 档号 <low limit> 为 NR1,NR2 或 NR3 形式的标称值 <high limit> 为 NR1,NR2 或 NR3 形式的标称值
例如发送：	**COMP:BIN 1,-10,+10<NL>** // 如果在百分比分选方式下： BIN1 下限为-10%，上限为 10%
查询语法：	**COMParator:BIN? <1~6>**
查询响应：	**<NR3>,<NR3><NL>**

2.4 TRIGger 子系统

TRIGger 用来设置触发源和产生一次触发。

TRIGger 子系统树

TRIGger	**:IMMediate**	
	:SOURce	**{INT,EXT}**
	:DELAy	**<float>**
TRG		

TRIGger:IMMediate 命令

TRIGger:IMMediate 触发源设置为 EXT 时，产生一次触发方式，并返回触发测试的数据。

命令语法：	**TRIGger:IMMediate**
例如发送：	**TRIGger:IMMediate<NL>** // 仪器测试一次，并返回测试数据
返回响应：	**<NR3>,BIN<n><NL>**

TRIGger:SOURce 命令

TRIGger:SOURce 命令设定触发源。TRIGger:SOURce?查询返回当前的触发方式。

命令语法：	**TRIGger:SOURce{INT,EXT}**
参数：	INT： 内部触发。 EXT： 外部触发。
例如发送：	**TRIGger:SOURce INT<NL>** // 设置为内部触发模式。
查询语法：	**TRIGger:SOURce?**
查询响应：	**{INT,EXT}<NL>**

TRIGger:DELAy 命令

TRIGger:DELAy 命令设定触发延时。TRIGger:DELAy?查询返回当前的触发延时时间。

命令语法：	**TRIGger:DELAy{0,<float>}**
参数：	0： 0s。 <float>： 0.1~10.0s。
例如发送：	**TRIG:DELA 0.1<NL>** // 设置触发延时 0.1s。
查询语法：	**TRIG:DELA?**
查询响应：	**{0,<float>}<NL>**

TRG 命令

TRG 在触发源设置为 EXT 时，产生一次触发，并返回触发测试的数据。

命令语法：	**TRG**
例如发送：	**TRG<NL>** //仪器测试一次，并返回测试数据
返回响应：	**<NR3>,BIN<n><NL>**

2.5 FETCh？查询

FETCh? 查询返回最近一次的测试结果。

查询语法：	**FETCh?**
查询响应：	**<scifloat>,{BIN0~BIN3}<NL>**
例如发送：	**FETCh?<NL>** // 查询最近一次测试结果。
返回响应：	**<NR3>,BIN<n><NL>**

2.6 SYSTem 子系统

SYSTem:LANGuage 命令

SYSTem:LANGuage 命令用于设定仪器语言。SYSTem:LANGuage? 查询返回当前的仪器语言。

命令语法：	**SYSTem:LANGuage {ENGLISH,CHINESE,EN,CN}**
参数：	ENGLISH：设定为英文。 CHINESE：设定为中文。 EN： 设定为英文。 CN：设定为中文。
例如发送：	**SYSTem:LANGuage ENGLISH<NL>** // 设定仪器语言为英文。
查询语法：	**SYST:LANG?**
查询响应：	**{ENGLISH,CHINESE}<NL>**

SYSTem:BEEPer 命令

SYSTem:BEEPer 命令用于设定按键音。SYSTem:BEEPer? 查询返回按键声音的状态。

命令语法：	SYSTem:BEEPer {OFF,ON,0,1}
参数：	OFF（0）：设定为按键音关闭。 ON（1）：设定为按键音打开。
例如发送：	SYSTem:BEEPer **OFF<NL>** // 设定仪器按键音关闭。
查询语法：	**SYST:BEEP?**
查询响应：	**{OFF,ON}<NL>**

SYSTem:SETZero 命令

SYSTem:SETZero 命令用于设定 0 ADJ(清零功能)。SYSTem:SETZero? 查询返回 0 ADJ(清零功能)的状态。

命令语法：	SYSTem:SETZero {OFF,ON,0,1}
参数：	OFF（0）：设定为 0 ADJ(清零功能)关闭。 ON（1）：设定为 0 ADJ(清零功能)打开。
例如发送：	SYSTem:SETZero **OFF<NL>** // 设定仪器 0 ADJ(清零功能)关闭。
查询语法：	**SYST: SETZ?**
查询响应：	**{OFF,ON}<NL>**

SYSTem:RESET 命令

SYSTem:RESET 命令用于设定开始恢复出厂设置。

命令语法：	SYSTem:RESET { ON,1}
参数：	ON（1）：设定开始恢复出厂设置。
例如发送：	SYSTem:RESET **ON<NL>** // 设定仪器开始恢复出厂设置。

2.7 CORRect 子系统

CORR 子系统用来完成一次短路校准。

CORRect:SHORt 命令

CORR:SHOR 完成一次短路校准，在发送指令前必须将测试端短路。

命令语法：	**CORRect:SHORt**
例如发送：	**CORRect:SHORt<NL> //短路清零。**
返回响应：	**Clear Zero Start<NL>** //提示短路清零开始。
	{PASS, FAIL}<NL> //提示清零通过。（失败：FAIL）

2.8 *IDN？查询

FETCh? 查询返回仪器的版本号。

查询语法：	**IDN?**
查询响应：	**< Manufacturer>,<MODEL>,<Revision><NL>**
例如发送：	***IDN?<NL>** // 查询最近一次测试结果。
返回响应：	**UNI-T,UT3516+,CRM1224170004,REV V3.37<NL>**

2.9 ERRor？查询

ERRor? 查询返回最近一次发生错误的信息。

查询语法：	**ERRor?**
查询响应：	**Error string**
例如发送：	**ERR?<NL>** // 查询最近一次错误的信息。
返回响应：	**No error.<NL><NL>**

对应的错误码如下：

错误码	说明
*E00	No error
*E01	Bad command
*E02	Parameter error
*E03	Missing parameter
*E04	buffer overrun
*E05	Syntax error
*E06	Invalid separator
*E07	Invalid multiplier
*E08	Numeric data error
*E09	Value too long
*E10	Invalid command
*E11	Unknow error

3. Modbus(RTU)通讯协议

3.1 数据格式

我们遵循 Modbus（RTU）通讯协议，仪器将响应上位机的指令，并返回标准响应帧。

指令帧

指令帧说明

	至少需要 3.5 字符时间的静噪间隔
从站地址	1 字节 Modbus 可以支持 00~0x63 个从站 统一广播时指定为 00
功能码	1 字节 0x03：读出多个寄存器 0x04：=03H，不使用 0x06：写入单个寄存器，可以用 10H 替代 0x08:回波测试（仅用于调试时使用） 0x10：写入多个寄存器
数据	指定寄存器地址、数量和内容
CRC-16	2 字节，低位在前 CyclicRedundancy Check 将从站地址到数据末尾的所有数据进行计算，得到 CRC16 校验码
	至少需要 3.5 字符时间的静噪间隔

CRC-16 计算方法

1. 将 CRC-16 寄存器的初始值设为 0xFFFF。
2. 对 CRC-16 寄存器和信息的第 1 个字节数据进行 XOR 运算，并将计算结果返回 CRC 寄存器。
3. 用 0 填入 MSB，同时使 CRC 寄存器右移 1 位。
4. 从 LSB 移动的位如果为“0”，则重复执行步骤(3)(处理下 1 个移位)。从 LSB 移动的位如果为“1”，则对 CRC 寄存器和 0xA001 进行 XOR 运算，并将结果返回 CRC 寄存器。
5. 重复执行步骤(3) 和(4)，直到移动 8 位。
6. 如果信息处理尚未结束，则对 CRC 寄存器和信息的下 1 个字节进行 XOR 运算，并返回 CRC 寄存器，从第(3) 步起重复执行。
7. 将计算的结果(CRC 寄存器的值) 从低位字节附加到信息上。

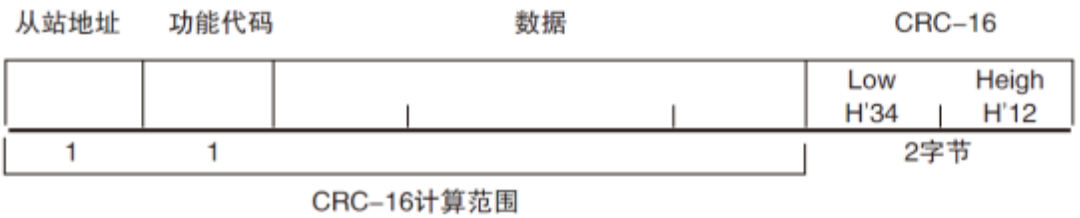
以下是一段 VB 语言的 CRC 计算函数：

```
Function CRC16(data() As Byte) As Byte()
     im CRC16Lo As Byte, CRC16Hi As Byte            'CRC 寄存器
     im CL As Byte, CH As Byte                      '多项式码
          im SaveHi As Byte, SaveLo As Byte
      im i As Integer
      im flag As Integer
      RC16Lo = &HFF
      RC16Hi = &HFF
      L = &H1
      H = &HA0
      or i = 0 To UBound(data)
            CRC16Lo = CRC16Lo Xor data(i)       '每一个数据与 CRC 寄存器进行异或
          For flag = 0 To 7
          SaveHi = CRC16Hi
          SaveLo = CRC16Lo
          CRC16Hi = CRC16Hi \ 2                 '高位右移一位
          CRC16Lo = CRC16Lo \ 2                 '低位右移一位
          If ((SaveHi And &H1) = &H1) Then      '如果高位字节最后一位为 1
              CRC16Lo = CRC16Lo Or &H80         '则低位字节右移后前面补 1
          End If                                '否则自动补 0
          If ((SaveLo And &H1) = &H1) Then      '如果 LSB 为 1，则与多项式码进行异或

        CRC16Hi = CRC16Hi Xor CH
        CRC16Lo = CRC16Lo Xor CL
      End If
   Next flag
ext i
    im ReturnData(1) As Byte
    eturnData(0) = CRC16Hi                      'CRC 高位
    eturnData(1) = CRC16Lo                      'CRC 低位
```

```
    RC16 = ReturnData
End Function
```

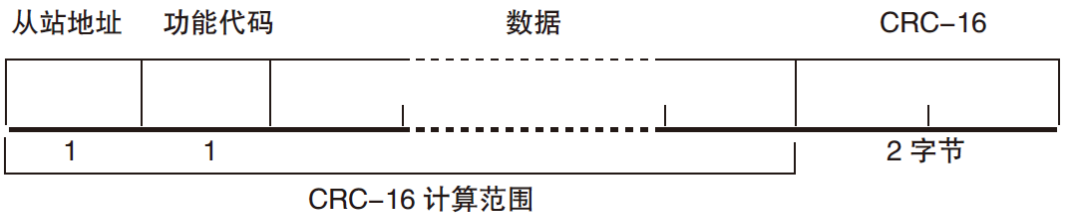
计算出 CRC-16 数据需要附加到指令帧末尾，例如：1234H



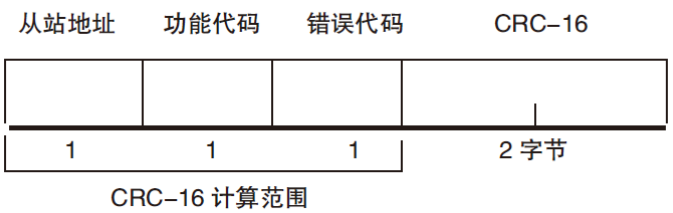
响应帧

除非是 00H 从站地址广播的指令，其它从站地址仪器都会返回响应帧。

正常响应帧



异常响应帧



异常响应帧说明

从站地址	1 字节 从站地址原样返回
功能码	1 字节 指令帧的功能码逻辑或（OR）上 BIT7（0x80），例如：0x03 OR 0x80 = 0x83
错误码	异常代码： 0x01 功能码错误（功能码不支持） 0x02 寄存器错误（寄存器不存在） 0x03 数据错误 0x04 执行错误
CRC-16	2 字节，低位在前 CyclicRedundancy Check 将从站地址到数据末尾的所有数据进行计算，得到 CRC16 校验码

无响应

以下情况，仪器将不进行任何处理，也不响应，导致通讯超时。

1. 从站地址错误
2. 传输错误
3. CRC-16 错误
4. 位数错误，例如：功能码 0x03 总位数必须为 8，而接受到的位数小于 8 或大于 8 个字节。
5. 从站地址为 0x00 时，代表广播地址，仪器不响应。

错误码错误码说明

错误码	名称	说明	优先级
0x01	功能码错误	功能码不存在	1
0x02	寄存器错误	寄存器不存在	2
0x03	数据错误	寄存器数量或字节数量错误	3
0x04	执行错误	数据非法，写入的数据不在允许范围内	4

3.2 功能码

仪器仅支持以下几个功能码，其它功能码未做支持。

功能码	名称	说明
0x03	读出多个寄存器	读出多个连续寄存器数据
0x04	与 0x03 相同	请用 0x03 代替
0x08	回波测试	接收到的数据原样返回
0x10	写入多个寄存器	写入多个连续寄存器

3.3 寄存器

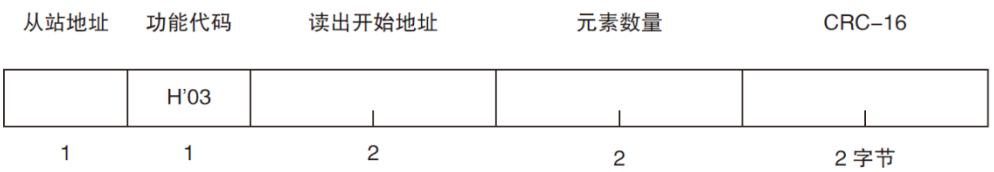
仪器的寄存器数量为 2 字节模式，即每次必须写入 2 个字节，例如：速度的寄存器为 0x3002，数据为 2 字节，数值必须写入 0x0001

数据：
仪器支持以下几种数值：

1. 1 个寄存器，双字节（16 位）整数，例如：0x64 → 00 64
2. 2 个寄存器，四字节（32 位）整数，例如：0x12345678 → 12 34 56 78
3. 2 个寄存器，四字节（32 位）单精度浮点数，3.14 → 40 48 F5 C3

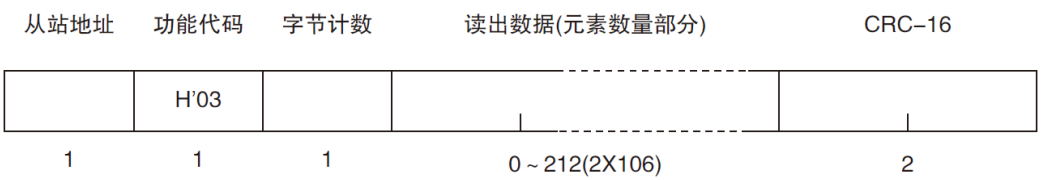
3.4 读出多个寄存器

读出多个寄存器（0x03）



名称	名称	说明
	从站地址	没有指定地址时，默认为 01
0x03	功能码	
	起始地址	寄存器起始地址，请参考 Modbus 指令集
	读取寄存器数量	连续读取的寄存器数量。请参考 Modbus 指令集，以确保这些寄存器地址都是存在的，否则将会返回错误帧。
CRC-16	校验码	

读出多个寄存器（0x03）响应帧



名称	名称	说明
	从站地址	原样返回
0x03 或 0x83	功能码	无异常：0x03 错误码：0x83
	字节数	寄存器数量 x2 例如：1 个寄存器返回 02
	数据	读取的数据
CRC-16	校验码	

3.5 写入多个寄存器

写入多个寄存器（0x10）

从站地址	功能代码	读出开始地址	元素数量	字节计数	写入数据(元素数量部分)	CRC-16
	H'10					
1	1	2	2	1	0～208(2X104)	2

名称	名称	说明
	从站地址	默认为 01
0x10	功能码	
	起始地址	寄存器起始地址，请参考 Modbus 指令集
	写入寄存器数量 0001~0068（104）	连续读取的寄存器数量。请参考 Modbus 指令集，以确保这些寄存器地址都是存在的，否则将会返回错误帧。
	字节数	=寄存器数量 x2
CRC-16	校验码	

写入多个寄存器（0x10）响应帧

从站地址	功能代码	写入开始地址	元素数量	CRC-16
	H'10			
1	1	2	2	2字节

名称	名称	说明
	从站地址	原样返回
0x10 或 0x90	功能码	无异常：0x10 错误码：0x90
	起始地址	
	寄存器数量	
	CRC-16 校验码	

3.6 回波测试

回波测试功能码 0x08，用于调试 Modbus。

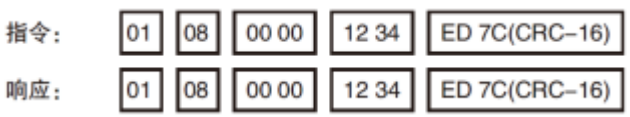
写出多个寄存器（0x03）响应帧

回波测试（0x08）

名称	名称	说明
	从站地址	原样返回
0x08	功能码	
	固定值	00 00
	测试数据	任意数值：例如 12 34
	CRC-16 校验码	

例如：

假定测试数据为 0x1234：



4. Modbus (RTU) 指令集

4.1 寄存器总览

以下列出了仪器使用的所有寄存器地址，任何不在表中的地址将返回错误码 0x02。

寄存器地址	名称	数值	说明
0200	读取测量结果	4 字节浮点数	只读，数据占用 2 个寄存器，4 字节 字节顺序 AABB CCDD，低位在前。
0202	读取通道的比较器结果	4 字节整数	只读，数据占用 2 个寄存器。
0204	读取测量结果	4 字节浮点数	只读，数据占用 2 个寄存器，4 字节 字节顺序 CCDD AABB，高位在前。
0206	触发一次并读取测量结果 AABB CCDD	4 字节单精度浮点数 字符顺序：低位在前 AABB CCDD	只读，数据占用 2 个寄存器，4 字节 收到指令会自动转入测量页面，触发方式切换为外部触发。
0208	触发一次并读取测量结果 CCDD AABB	4 字节单精度浮点数 字符顺序：高位在前 CCDD AABB	只读，数据占用 2 个寄存器，4 字节 收到指令会自动转入测量页面，触发方式切换为外部触发。
020A	量程号(R 模式)	00 00 00 00~00 00 00 08	读写寄存器，4 字节整数。

020C	量程自动(R 模式)	00 00 00 00：自动量程 00 00 00 01：手动量程 00 00 00 02：标称量程	读写寄存器，4 字节整数。
020E	量程号(LPR 模式)	00 00 00 01~00 00 00 04	读写寄存器，4 字节整数。
0210	量程自动(LPR 模式)	00 00 00 00：自动量程 00 00 00 01：手动量程 00 00 00 02：标称量程	读写寄存器，4 字节整数。
0212	测试模式	00 00 00 00：R 00 00 00 01：RT 00 00 00 02：T 00 00 00 03：LPR 00 00 00 04：LPRT	读写寄存器，4 字节整数。
0214	测试速度	00 00 00 00：慢速 00 00 00 01：中速 00 00 00 02：快速 00 00 00 03: 高速	读写寄存器，4 字节整数。
0216	系统语言	00 00 00 00：英语 00 00 00 01：简体中文	读写寄存器，4 字节整数。
0218	讯响	00 00 00 00：关闭 00 00 00 01：合格讯响 00 00 00 02：不合格讯响	读写寄存器，4 字节整数。
021A	触发器设置	00 00 00 00：内部触发 00 00 00 01：外部触发	读写寄存器，4 字节整数。
021C	触发延时	0：触发延时关闭 4 字节浮点数范围（0.1~9.9s）	读写寄存器，4 字节整数。
021E	比较器档位数	00 00 00 00：比较器关闭 00 00 00 01：1-BIN 00 00 00 02：2-BIN 00 00 00 03：3-BIN 00 00 00 04：4-BIN 00 00 00 05：5-BIN 00 00 00 06：6-BIN	读写寄存器，4 字节整数。
0220	比较器模式	00 00 00 00：SEQ 00 00 00 01：ABS 00 00 00 02：PER	读写寄存器，4 字节整数。
0222	标称值	4 字节浮点数	读写寄存器，数据占用 2 个寄存器。
0224	BIN1 下限值	4 字节浮点数	读写寄存器，数据占用 2 个寄存器。
0226	BIN1 上限值	4 字节浮点数	读写寄存器，数据占用 2 个寄存器。
0228	BIN2 下限值	4 字节浮点数	读写寄存器，数据占用 2 个寄存器。
022A	BIN2 上限值	4 字节浮点数	读写寄存器，数据占用 2 个寄存器。
022C	BIN3 下限值	4 字节浮点数	读写寄存器，数据占用 2 个寄存器。
022E	BIN3 上限值	4 字节浮点数	读写寄存器，数据占用 2 个寄存器。
0230	BIN4 下限值	4 字节浮点数	读写寄存器，数据占用 2 个寄存器。

0232	BIN4 上限值	4 字节浮点数	读写寄存器，数据占用 2 个寄存器。
0234	BIN5 下限值	4 字节浮点数	读写寄存器，数据占用 2 个寄存器。
0236	BIN5 上限值	4 字节浮点数	读写寄存器，数据占用 2 个寄存器。
0238	BIN6 下限值	4 字节浮点数	读写寄存器，数据占用 2 个寄存器。
023A	BIN6 上限值	4 字节浮点数	读写寄存器，数据占用 2 个寄存器。
023C	执行清零寄存器读取清零状态	读取： 00 00 00 00 清零成功 00 00 00 01 清零失败 00 00 00 02 0 ADJ 未开启	只读寄存器，数据占用 2 个寄存器。
023E	0 ADJ	00 00 00 00 关闭 00 00 00 01 开启	读写寄存器，数据占用 2 个寄存器。

4.2 获取测量结果

获取测量结果(AABB CCDD)【0200】

寄存器 0200~0201 用来获取仪器测量数据。

例如：获取测量数据

指令：

1	2	3	4	5	6	7	8
01	03	0200		0002		CRC-16	
从站	读	寄存器地址		寄存器数量		校验码	

响应：

1	2	3	4	5	6	7	8	9
01	03	字节	单精度浮点数				CRC-16	

获取测量数据：

发送：

1	2	3	4	5	6	7	8
01	03	02	00	00	02	C5	B3

响应：

1	2	3	4	5	6	7	8	9
01	03	04	42	C7	F9	9E	9C	4E

其中 B4~B7 为测量数据：42C7F99E 代表单精度浮点数，低位在前。

字节顺序 AA BB CC DD 换算为十进制数为 99.987564

获取比较器结果【0202】

寄存器 0202~0203 用来获取仪器测量数据。

返回的 4 字节整数代表了比较器结果：

00：不合格

01：合格档 1

02：合格档 2

03：合格档 3

04：合格档 4

05：合格档 5

06：合格档 6

发送：

1	2	3	4	5	6	7	8
01	03	02	02	00	02	64	73

响应：

1	2	3	4	5	6	7	8	9
01	03	04	00	00	00	00	FA	33

获取测量结果(CCDD AABB)【0204】

寄存器 0204~0205 用来获取仪器测量数据。

例如：获取测量数据

指令：

1	2	3	4	5	6	7	8
01	03	0204		0002		CRC-16	
从站	读	寄存器地址		寄存器数量		校验码	

响应：

1	2	3	4	5	6	7	8	9
01	03	字节	单精度浮点数				CRC-16	

获取测量数据：

发送：

1	2	3	4	5	6	7	8
01	03	02	04	00	02	84	72

响应：

1	2	3	4	5	6	7	8	9
01	03	04	F9	A2	42	C7	1A	7F

其中 B4~B7 为测量数据：F9 A2 42 C7 代表单精度浮点数，高位字在前，字节顺序 CC DD AA BB。

交换字顺序为 AABBCCDD 低位字在前 ： 42 C7 F9 A2 换算为十进制 99.987564

触发一次并返回测量结果(AABB CCDD)【0206】

寄存器 0206~0207 用来获取仪器测量数据。

例如：获取测量数据

指令：

1	2	3	4	5	6	7	8
01	03	0206		0002		CRC-16	
从站	读	寄存器地址		寄存器数量		校验码	

响应：

1	2	3	4	5	6	7	8	9
01	03	字节	单精度浮点数				CRC-16	

获取测量数据：

发送：

1	2	3	4	5	6	7	8
01	03	02	06	00	02	25	B2

响应：

1	2	3	4	5	6	7	8	9
01	03	04	42	C7	F9	A2	9C	5F

其中 B4~B7 为测量数据：42 C7 F9 A2 代表单精度浮点数，低位字在前，字节顺序 AA BB CC DD。换算为十进制 99.987572

注意：

此指令只在触发方式为外部触发时，触发一次并返回测量结果。

触发一次并返回测量结果(CCDD AABB)【0208】

寄存器 0208~0209 用来获取仪器测量数据。

例如：获取测量数据

指令：

1	2	3	4	5	6	7	8
01	03	0208		0002		CRC-16	
从站	读	寄存器地址		寄存器数量		校验码	

响应：

1	2	3	4	5	6	7	8	9
01	03	字节	单精度浮点数				CRC-16	

获取测量数据：

发送：

1	2	3	4	5	6	7	8
01	03	02	08	00	02	44	71

响应：

1	2	3	4	5	6	7	8	9
01	03	04	F9	A2	42	C7	EB	07

其中 B4~B7 为测量数据：F9 A2 42 C7 代表单精度浮点数，高位字在前，字节顺序 CC DD AA BB。

交换字顺序为 AABBCCDD 低位字在前 ： 42 C7 F9 A2 换算为十进制 99.987564

换算为十进制 99.987564

注意：

此指令只在触发方式为外部触发时，触发一次并返回测量结果。

4.3 参数设置

量程档位【020A】

写入：

1	2	3	4	5	6	7	8	9	10	11	12	13
01	10	02	0A	00	02	04	00	00	00	02	EB	71
	写	寄存器地址		寄存器数量		字节	数据				CRC	

响应：

1	2	3	4	5	6	7	8
01	10	02	0A	00	02	60	72
		寄存器地址		寄存器数量		CRC	

读取：

1	2	3	4	5	6	7	8
01	03	02	0A	00	02	E5	B1
	读	寄存器地址		寄存器数量		CRC	

响应：

1	2	3	4	5	6	7	8	9
01	03	04	00	00	00	02	7B	F2
		字节	数据				CRC	

其中，输入量程范围为 0000~0008

量程类型【020C】

写入：

1	2	3	4	5	6	7	8	9	10	11	12	13
01	10	02	0C	00	02	04	00	00	00	00	EA	9A
	写	寄存器地址		寄存器数量		字节	数据				CRC	

响应：

1	2	3	4	5	6	7	8
01	10	02	0C	00	02	80	73
		寄存器地址		寄存器数量		CRC	

读取：

1	2	3	4	5	6	7	8
01	03	02	0C	00	02	05	B0
	读	寄存器地址		寄存器数量		CRC	

响应：

1	2	3	4	5	6	7	8	9
01	03	04	00	00	00	00	FA	33
		字节	数据				CRC	

其中，量程类型范围为 0000~0002

4.4 比较器设置

标称值设置【0222-0223】

写入：

100 (单精度浮点数：0x42C80000)

1	2	3	4	5	6	7	8	9	10	11	12	13
01	10	02	22	00	02	04	42	C8	00	00	FC	88
	写	寄存器地址		寄存器数量		字节	数据				CRC	

响应：

1	2	3	4	5	6	7	8
01	10	02	22	00	02	E0	7A
		寄存器地址		寄存器数量		CRC	

读取：

1	2	3	4	5	6	7	8
01	03	02	22	00	02	65	B9
	读	寄存器地址		寄存器数量		CRC	

响应：

1	2	3	4	5	6	7	8	9
01	03	04	42	C8	00	00	FA	33
		字节	数据 100				CRC	

极限值【0224-023B】

6 档比较器的极限值从 0224 开始到 023B 结束，每个比较器档下限使用 2 个寄存器，上限使用 2 个寄存器，总共 4 个寄存器。上下限极限值可单独设置，也可同时设置。

写入:

下限：1E-5,

1	2	3	4	5	6	7	8	9	10	11	12	13
01	10	02	24	00	02	04	37	27	C5	AC	04	76
	写	寄存器地址		寄存器数量		字节	数据				CRC	

响应:

1	2	3	4	5	6	7	8
01	10	02	24	00	02	00	7B

上限：1.2E5

1	2	3	4	5	6	7	8	9	10	11	12	13
01	10	02	26	00	02	04	47	EA	60	00	75	BD
	写	寄存器地址		寄存器数量		字节	数据				CRC	

响应:

1	2	3	4	5	6	7	8
01	10	02	26	00	02	A1	BB

读取:

1	2	3	4	5	6	7	8
01	03	02	24	00	02	85	B8

响应:

1	2	3	4	5	6	7	8	9
01	03	04	37	27	C5	AC	17	61
		字节	数据 1.2E5				CRC	

读取:

1	2	3	4	5	6	7	8
01	03	02	26	00	02	24	78

响应:

1	2	3	4	5	6	7	8	9
01	03	04	47	EA	60	00	E7	73
		字节	数据 1E-5				CRC	

4.5 系统功能

清零【023C】

读取寄存器 023C，仪器将开始执行短路清零操作。执行清零前，务必将测试线短路及 0ADJ 打开，否则清零将失败。由于清零过程需要几秒钟时间。清零执行期间或清零完成后，将返回取清零状态：

0000 清零成功

FFFF 清零失败

0002 0 ADJ 未开启

读取:

1	2	3	4	5	6	7	8
01	03	02	3C	00	02	05	BF

响应:

1	2	3	4	5	6	7	8	9
01	03	04	00	00	00	00	FA	33
		字节	清零成功				CRC	