

UNI-T

使用说明书

UT3510 微欧计系列通信协议 (SCPI&MODBUS)-V1.0

2020年 7月

UNI-T TECHNOLOGY(China) Co., Lt

保证和声明

版权

2019 优利德科技（中国）股份有限公司

商标信息

UNI-T是优利德科技（中国）股份有限公司的注册商标。

声明

- 本公司产品受中国及其它国家和地区的专利（包括已取得的和正在申请的专利）保护。
- 本公司保留改变规格及价格的权利。
- 本手册提供的信息取代以往出版的所有资料。
- 本手册提供的信息如有变更，恕不另行通知。
- 对于本手册可能包含的错误，或因手册所提供的信息及演绎的功能以及因使用本手册而导致的任何偶然或继发的损失，**UNI-T**概不负责。
- 未经 **UNI-T**事先书面许可，不得影印、复制或改编本手册的任何部分。

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 ISO9001：2015标准和 ISO14001：2015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。

联系我们

如您在使用此产品或本手册的过程中有任何问题或需求，可与 **UNI-T**联系：

电子邮箱：

网址：

SCPI 命令概述

命令串解析

主机可以发送一串命令给仪器，仪器命令解析器在捕捉到结束符（\n）或输入缓冲区溢出后开始解析。

例如：

合法的命令串：

AAA:BBB CCC;DDD EEE;:FFF

仪器命令解析器负责所有命令解析和执行，在编写程序前您必须首先对其解析规则有所了解。

命令解析规则

1. 命令解析器只对 ASCII 码数据进行解析和响应。
2. SCPI 命令串必须以 NL(‘ \n’ ASCII 0x0A)为结束符，命令解析器在收到结束符后或缓冲区溢出才开始执行命令串。
3. 如果指令握手打开，命令解析器在每接收到一个字符后，立即将该字符回送给主机，主机只有接收到这个回送字符后才能继续发送下一个字符。
4. 命令解析器在解析到错误后，立即终止解析，当前指令作废。
5. 命令解析器在解析到查询命令后，终止本次命令串解析，其后字符串被忽略。
6. 命令解析器对命令串的解析不区分大小写。
7. 命令解析器支持命令缩写形式，缩写规格参见之后章节。

符号约定和定义

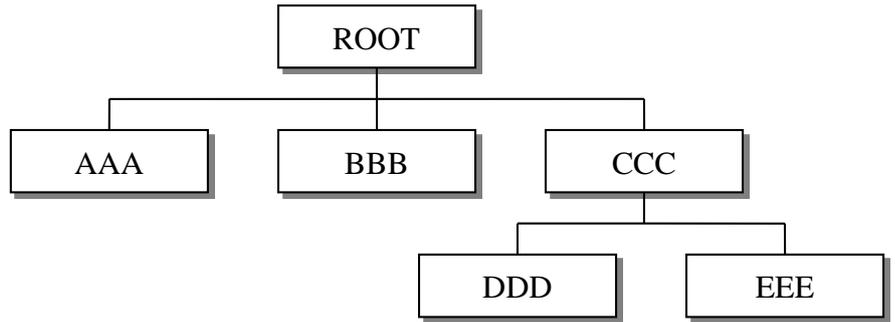
本章使用了一些符号，这些符号并不是命令树的一部分，只是为了能更好的对命令串的理解。

<>	尖括号中的文字表示该命令的参数
[]	方括号中的文字表示可选命令
{ }	当大括号包含几个参数项目时，表示只能从中选择一个项目。
()	参数的缩写形式放在小括号中。
大写字母	命令的缩写形式。

命令树结构

对 SCPI 命令采用树状结构的，可向下三级（注：此仪器的命令解析器可向下解析任意层），在这里最高级称为子系统命令。只有选择了子系统命令，该其下级命令才有效，SCPI 使用冒号 (:) 来分隔高级命令和低级命令。

图 0-1 命令树结构



举例说明

```

ROOT:CCC:DDD ppp
  ROOT  子系统命令
    CCC  第二级
      DDD  第三级
        ppp  参数
  
```

命令和参数

一条命令树由 命令和[参数] 组成，中间用 1 个空格（ASCII: 20H）分隔。

举例说明 AAA:BBB 1.234
 命令 [参数]

命令

命令字可以是长命令格式或缩写形式，使用长格式便于工程师更好理解命令串的含义；缩写形式适合书写。

参数

■ 单命令字命令，无参数。

例如：AAA:BBB

■ 参数可以是字符串形式，其缩写规则仍遵循上节的“命令缩写规则”。

如：AAA:BBB 1.23

■ 参数可以是数值形式

· <integer> 整数 123, +123, -123

· <float> 浮点数

· <fixfloat>: 定点浮点数: 1.23, -1.23

2. <Sciloat>: 科学计数法浮点数: 1.23E+4, +1.23e-4

3. <mpfloat>: 倍率表示的浮点数: 1.23k, 1.23M, 1.23G, 1.23u

表 0-1 倍率缩写

数值	倍率
1E18 (EXA)	EX
1E15 (PETA)	PE
1E12 (TERA)	T

1E9 (GIGA)	G
1E6 (MEGA)	MA
1E3 (KILO)	K
1E-3 (MILLI)	M
1E-6 (MICRO)	U
1E-9 (NANO)	N
1E-12 (PICO)	P
1E-15 (PEMTO)	F
1E-18 (ATTO)	A

提示：倍率不区分大小写，其写法与标准名称不同。

分隔符

仪器命令解析器只接收允许的分隔符，除此之外的分隔符命令解析器将产生“Invalid separator(非法分割符)”错误。这些分隔符包括：

- ； 分号，用于分隔两条命令。
例如：AAA:BBB 100.0 ; CCC:DDD
- ： 冒号，用于分隔命令树，或命令树重新启动。
例如：AAA : BBB : CCC 123.4 ; : DDD : EEE 567.8
- ？ 问号，用于查询。
例如：AAA ?
- 空格，用于分隔参数。
例如：AAA:BBB□1.234

错误码

对应的错误码如下：

错误码	说明	
*E00	No error	无错误
*E01	Bad command	命令错误
*E02	Parameter error	参数错误
*E03	Missing parameter	缺少参数
*E04	buffer overrun	缓冲区溢出
*E05	Syntax error	语法错误
*E06	Invalid separator	非法分隔符
*E07	Invalid multiplier	非法倍率单位
*E08	Numeric data error	数值错误
*E09	Value too long	数字太长
*E10	Invalid command	无效指令
*E11	Unknow error	未知错误

命令参考

所有命令都是按子系统命令顺序进行解释，下面列出了所有子系统

- DISPlay 显示子系统
- FUNction 功能子系统
- CORRection 校正子系统
- COMParator 比较器子系统
- SYSTem 系统子系统
- TRIGger 触发子系统
- FETCh? 获取结果子系统
- ERRor 错误信息子系统

公共命令：

- IDN? 仪器信息查询子系统
- TRG 触发并获取数据

DISPlay 显示子系统

DISPlay 子系统可以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。

图 0-1 DISPlay 子系统树

DIS Play	:PAGE	{TEST,SETUP(MSET),COMParator,CORRECTION(CSET),FILE,SYSTem,SYSTEMINFO(SINF)}
	:LINE	<string>

DISPlay:PAGE

DISP:PAGE 用来切换到指定页面。

命令语法：研发
1 楼会议室

DISPlay:PAGE <页面名称>

参数：<页面名称> 包括：

TEST	测量显示页
SETUP(MSET)	设置页
COMParator	比较器页
CORRection	校准页面
FILE	文件管理页
SYSTem	系统配置页
SYSTEMINFO(SINF)	系统信息页

例如：发送> disp:page setup<NL> //切换到设置页面

查询语法：DISP:PAGE?

查询响应：<页面名称>缩写

test
mset
comp
cset

file
syst
sinf

例如： 发送> disp:page? <NL>
返回> test<NL>

DISP:LINE

DISP:LINE 用来在页面底部的提示栏显示一串文本。文本最多可以显示 30 个字符。

命令语法： **DISPlay:LINE <string>**
参数： <string> 最多 30 个字符
例如： 发送> DISP:LINE “This is a Comment.” <NL>

FUNCTION 子系统

图 0-1 FUNCTION 子系统树

FUNCTION	:RANGe	{量程号, max, min}	
		:MODE	{AUTO,HOLD,NOMinal}
	:RATE	{SLOW,MED,FAST }	
	:TC	:COEFFicien t	<float>
:REFER		<float>	

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。

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： 其中， <量程号>
0~9 (UT3516)
0~5 (UT3513)
min 最小量程
max 最大量程

例如： 发送> FUNC:RANG 5<NL> //切换到 5 量程 (1kΩ)

查询语法： **FUNC:RANG?**

查询响应： 量程号 0~9 (UT3516)

例如： 发送> FUNC:RANGE? <NL>
返回> 5<NL>

FUNCTION:RANGe:MODE

FUNC:RANG:MODE 用来切换量程方式

命令语法:	FUNCtion:RANGe:MODE {AUTO,HOLD(MANual),NOMinal}
例如:	发送> FUNC:RANG:MODE NOM<NL> //切换到标称量程方式
查询语法:	FUNC:RANG:MODE?
查询响应:	{AUTO,HOLD,NOM}

FUNCtion:RATE

FUNC:RATE 或 FUNC:SPEED 用来设置测试速度。

命令语法:	FUNCtion:RATE {SLOW,MED,FAST}
例如:	发送> FUNC:RATE MED<NL> //设置为高速测试
查询语法:	FUNC:RATE?
查询响应:	{SLOW,MED,FAST}

FUNCtion:TC

FUNC:TC 用来打开或关闭温度补偿。

命令语法:	FUNCtion:TC {ON,OFF,1, 0}
例如:	发送> FUNC:TC ON <NL> //开启温度补偿功能
查询语法:	FUNC:TC?
查询响应:	{ON,OFF}

FUNCtion:TC: COEFFicient

FUNC:TC:COEFFicient 用来设置温度系数。

命令语法:	FUNCtion:TC: COEFFicient <float>
例如:	发送> FUNC:Tc:COEF 0.394<NL> //设置温度系数为 0.394%
查询语法:	FUNC:TC:COEF?
查询响应:	<fixfloat>
例如:	发送> FUNC:Tc:COEF? <NL> 响应> +0.39400<NL>

FUNCtion:TC:REFErence

FUNC:TC:REFE 用来设置参考温度

命令语法:	FUNCtion:TC:REFErence <float>
例如:	发送> FUNC:Tc:REFE 25<NL> //设置参考温度为 25 度
查询语法:	FUNC:TC:REFE?
查询响应:	<fixfloat>
例如:	发送> FUNC:Tc:REFE? <NL> 响应> +25.00<NL>

COMParator 子系统

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。

图 0-1 COMParator 子系统树

COMParator	[:STATe]	{OFF,#-BIN} (UT3516) {OFF,ON} (UT3513)
	:BEEP	{OFF,PASS(OK),FAIL(NG)}
	:MODE	{ABS,PER,SEQ}
	:NOMinal	<float>
	:BIN	<档号 1~10>, <float 下限>,<float 上限>

COMParator[:STATe]

COMP[:STATe] 用来关闭比较器或设置档位数。

命令语法: `COMParator[:STATe] {OFF,#-BIN} (UT3516)`
`COMParator[:STATe] {OFF,ON} (UT3513)`

参数: 其中,
<#-BIN> 包括: 1-BIN ~ 6-BIN

例如: 发送> COMP:STAT 6-BIN<NL> //打开比较器, 并设置为 6 档分选
发送> COMP:STAT OFF<NL> //关闭比较器

查询语法: `COMP[:STAT]?`

查询响应: `{OFF,#-BIN}`

COMParator:BEEP

COMP:BEEP 用来启用讯响。

命令语法: `COMParator:BEEP {OFF,OK,NG}`

例如: 发送> COMP:BEEP OK<NL> //合格讯响

查询语法: `COMP:BEEP?`

查询响应: `{OFF,OK,NG}`

COMParator:MODE

COMP:MODE 用来设置比较器方式。

命令语法: `COMParator:MODE {ABS,PER,SEQ}`

例如: 发送> COMP:MODE SEQ //切换到顺序比较方式

查询语法: `COMP:MODE?`

查询响应: `{ABS,PER,SEQ}`

COMParator:NOMinal

COMP:NOM 用来设置标称值。

命令语法: `COMParator:NOMinal <float>`

例如: 发送> COMP:NOM 1.0000k //标称值设定为 1k
发送> COMP:NOM 1E3 //标称值设定为 1k
发送> COMP:NOM 1000 //标称值设定为 1k

查询语法: `COMP:NOM?`

查询响应: `<scifloat>`

例如： 发送> COMP:NOM? <NL>
 返回> 1.0000E+03<NL> //标称值为 1k

COMParator:BIN

COMP:BIN 用来设置标称值。

命令语法： COMParator:BIN <档号 1~6>,<float 下限>,<float 上限> (*UT3516)
 COMParator:BIN <float 下限>,<float 上限> (*UT3513)

例如： 发送> COMP:BIN 1,-10,+10 //如果在百分比分选方式下：BIN1 下限为-10%，上限为 10%

查询语法： COMP:BIN? <1~6>

查询响应： <scifloat>,<scifloat>

例如： 发送> COMP:BIN? 1<NL>
 返回> -10.000E+00,+10.000E+00<NL> //-10,+10

TRIGger 子系统

图 0-1 TRIGger 子系统树

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

TRIGger 用来设置触发源和产生一次触发。

TRIGger[:IMMediate]

TRIG[:IMM] 在触发源设置为 EXT 时，产生一次触发，但不会返回触发测试的数据。如果要返回数据需要使用 TRG 指令。

命令语法： TRIGger[IMMediate]

例如： 发送> TRIG<NL> //仪器测试一次后停止

TRIGger:SOURce

TRIG:SOUR 用来设置触发源。

命令语法： TRIGger:SOURce {INT,EXT}

例如： 发送> TRIG:SOUR BUS<NL> //设置为总线触发模式。

查询语法： TRIG:SOUR?

查询响应： <INT,EXT>

TRIGger:DELAy

TRIG:DELAy 用来设置触发延时。

命令语法： TRIGger:DELAy {0,<float>}

参数 0 :将关闭触发延时。

参数<float>: 0.001~9.0

例如： 发送> TRIG:DELA 0.1<NL> //设置触发延时 0.1s

发送> TRIG:DELA 10m<NL> //设置触发延时 10ms
 查询语法: TRIG:DELA?
 查询响应: 0.1

TRG

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> +9.9651e+01,BIN00 <NL>

FETCh? 子系统

FETCh? 用来获取测试数据。使用该指令前,需要将<系统配置>页面下的【结果发送】字段设置为【FETCH】。
 FETCh? 指令将返回测试数据。

图 0-1 FETCh? 子系统树

	FETCh?	
查询语法:	FETCh?	
查询响应:	<scifloat>,{BIN0~BIN6 }	
	其中 BIN0 代表不合格	
例如:	发送> FETC? <NL>	
	返回>+9.9651e+01,BIN0<NL>	

SYSTem 子系统

SYSTem 子系统用来设置与系统相关的参数。
 SYSTem 子系统设置的数据将不会保存在仪器内部。

图 0-1 SYSTem 子系统树

SYSTem	:LANGUage	{ENGLISH,CHINESE,EN,CN}
	:TIME	<YEAR>,<MONTH>,<DAY>,<HOUR>,<MINUTE>,<SECOND>
	:KEYLock(KLOCK)	{ON(1),OFF(0)}
	:BEEP	{ON(1),OFF(0)}
	:SHAKEHAND(SHAK)	{ON(1),OFF(0)}
:UPLOAD(UPLD)	{FETCh,AUTO}	

SYSTem:LANGUage 系统语言

仪器语言设置。

命令语法: SYSTem:LANGUage {ENGLISH,CHINESE,EN,CN}
 例如: 发送> SYST:LANG EN //设置为英文显示

查询语法: SYST:LANG?
查询响应: {ENGLISH,CHINESE}

SYSTem:TIME 系统时间设置

命令语法: SYSTem:TIME <YEAR>,<MONTH>,<DAY>,<HOUR>,<MINUTE>,<SECOND>
例如: 发送> SYST:TIME 2016,12,30,11,18,31 //2016-12-30 11:18:31
查询语法: SYSTem:TIME?
查询响应: <YEAR>-<MONTH>-<DAY> <HOUR>:<MINUTE>:<SECOND>
例如: 发送> SYST:TIME?
接收> 2016-12-30 11:18:31

SYSTem:KEYLock 或 SYSTem:KLOCK 键盘锁设置

命令语法: SYSTem:KEYLock {ON,OFF,0,1}
SYSTem:KLOCK {ON,OFF,0,1}
例如: 发送> SYST:KEYL OFF //键盘解锁
查询语法: SYSTem:KEYLock?
SYSTem:KLOCK?
查询响应: {on,off}

SYSTem:BEEPer 按键音

此指令不影响比较器讯响。

命令语法: SYSTem:BEEPer {OFF,ON,0,1}
参数: {OFF,ON,0,1}
OFF/0: 按键音关闭
ON/1: 按键音关闭
例如: 发送> SYST:BEEP OFF
查询语法: SYSTem:BEEPer?
查询响应: {OFF,ON}

SYSTem:SHAKhand 通讯握手指令（数据头返回）

通讯握手开启后，仪器会将接收到的指令原样返回给主机，之后再返回数据。

命令语法: SYSTem:SHAKhand {ON,OFF,0,1}
SYSTem:HEADer {ON,OFF,0,1}
例如: 发送> SYST:SHAK ON
发送> SYST:HEAD ON
查询语法: SYSTem:SHAKhand?
SYSTem:HEADer?
查询响应: {on,off}

SYSTem:UPLOAD(UPLD) 测试结果发送

SYSTem:UPLOAD(UPLD)可以设置数据发送方式：自动或

是 FETCH 指令。

命令语法: **SYSTem:UPLOAD {FETCH,AUTO}**

参数: **{FETCH,AUTO}**

FETCH: 数据需要通过指令 fetch?才能返回到主机, 仪器被动发送。

AUTO: 数据在每次测试完成后, 自动发送测试结果给主机, 仪器主动发送。

例如: 发送> SYST:UPLD AUTO //设置为自动发送

查询语法: **SYST:UPLD?**

查询响应: **{FETCH,AUTO}**

CORRect 子系统

CORR 子系统用来完成一次短路校准。

图 0-1 CORRect 子系统树

CORRect	:STATe	{ON,OFF,0,1}
	:SHORT	

CORRect:STATe

命令语法: **SYSTem:STATe {OFF,ON,0,1}**

参数: **{OFF,ON,0,1}**

OFF/0: 短路清零值关闭

ON/1: 短路清零值有效

例如: 发送> SYST:STAT OFF

查询语法: **SYSTem:STAT?**

查询响应: **{OFF,ON}**

CORRect:SHORT

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>

返回> Short Clear Zero Start. <NL> //提示清零开始

返回> PASS<NL> //提示: 清零通过, (失败: FAIL)

FILE(MMEM) 子系统

FILE(MMEM) 子系统用来管理文件, 可以用来保存用户参数到内部闪存中, 或读取闪存文件到系统里。

图 0-1 FILE(MMEM) 子系统树

FILE	:SAVE	<无参数>或<文件号 0-9>
MMEM	:LOAD	<无参数>或<文件号 0-9>
	:DElete	<文件号 0-9>

FILE:SAVE 保存文件

FILE:SAVE 可以保存当前设置到当前文件或指定的文件

中。

命令语法: FILE:SAVE
FILE:SAVE <File No. 0-9>

例如: 发送> FILE:SAVE //保存到当前文件中
发送> FILE:SAVE 1 //保存到文件 1 中

FILE:LOAD 读取文件

FILE:LOAD 可以读取文件数据到系统中。

命令语法: FILE:LOAD
FILE:LOAD <File No. 0-9>

例如: 发送> FILE:LOAD //读取当前文件数据到系统中
发送> FILE:SAVE 1 //读取文件 1 的数据到系统中

FILE:DELeTe 删除指定文件

FILE:DELeTe 可以删除指定文件的数据。

命令语法: FILE:SAVE
FILE:SAVE <File No. 0-9>

例如: 发送> FILE:SAVE //保存到当前文件中
发送> FILE:SAVE 1 //保存到到文件 1 中

注: 删除当前文件不会影响系统的参数

SAV

SAV 可以保存当前设置到当前文件中。

命令语法: SAV = FILE:SAVE

例如: 发送> SAV //保存到当前文件中

RCL

RCL 可以读取当前文件数据到系统中。

命令语法: RCL = FILE:LOAD

例如: 发送> FILE:LOAD //读取当前文件数据到系统中

IDN? 子系统

图 0-1 IDN? 子系统树



IDN?子系统用来返回仪器的版本号。

查询语法: IDN?

查询响应: <MODEL>,<Revision>,<SN>,< Manufacturer>

例如: 发送> IDN? <NL>
返回> UT3516,REV A1.0,0000000,Applent Instruments<NL>

ERRor 子系统

错误子系统用来获取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> 返回> no error.<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

Modbus (RTU) 通讯协议

& 本章包括以下几方面的内容：

- 数据格式——了解 Modbus 通讯格式。
- 功能
- 变量区域
- 功能码

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。

参见：您可以与我公司销售部联系，获取优利德仪器通讯测试工具，里面有 Modbus 通讯调试方法。包含了 CRC-16 计算器和浮点数转成 Modbus 浮点数格式。

指令帧

图 0-1 Modbus 指令帧



表 0-1 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 在未选配 RS485 选件的仪器里，默认的从站地址为 0x01
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 Cyclic Redundancy 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&HA001
    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

```

参见：我公司的“优利德仪器通讯测试工具“，里面有 Modbus 通讯调试方法。包含了 CRC-16 计算器。

计算出 CRC-16 数据需要附加到指令帧末尾，例如：

1234H:

图 0-2 Modbus 附加 CRC-16 值



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。

图 0-3 正常响应帧

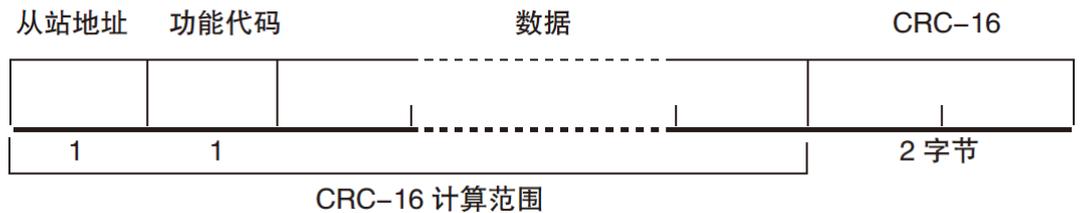


图 0-4 异常响应帧



表 0-2 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 Cyclic Redundancy 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

表 0-3 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

功能码

仪器仅支持以下几个功能码, 其它功能码, 将响应错误帧。

表 0-1 功能码

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
------	---------	-----------

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：

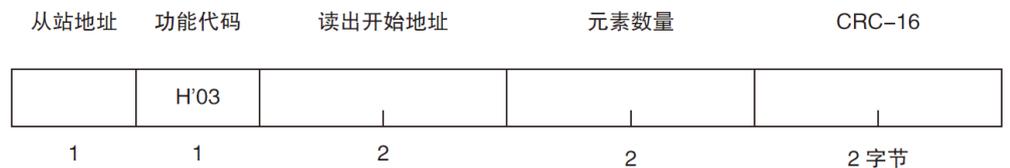
1. 1 个寄存器，双字节（16 位）整数，例如：0x64 → 00 64
2. 2 个寄存器，四字节（32 位）整数，例如：0x12345678 → 12 34 56 78
3. 2 个寄存器，四字节（32 位）单精度浮点数，3.14 → 40 48 F5 C3

参见：

我公司的“优利德仪器通讯测试工具“”，里面有 Modbus 通讯调试方法。包含了浮点数转换器。

读出多个寄存器

图 0-1 读出多个寄存器（0x03）



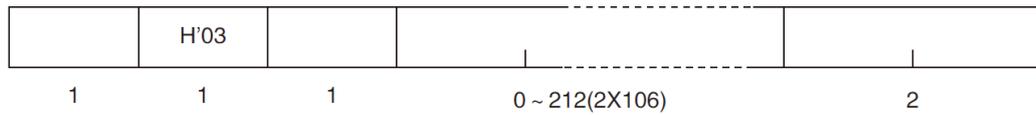
读出多个寄存器的功能码是 0x03.

表 0-1 读出多个寄存器

名称	名称	说明
	从站地址	没有指定 RS485 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 0001~006A (106)	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	

图 0-2 读出多个寄存器（0x03）响应帧

从站地址 功能代码 字节计数 读出数据(元素数量部分) CRC-16



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 x 2 例如: 1 个寄存器返回 02
	数据	读取的数据
CRC-16	校验码	

写入多个寄存器

图 0-1 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

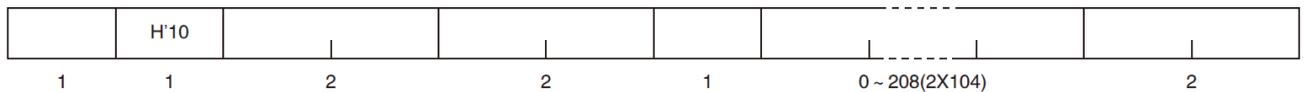
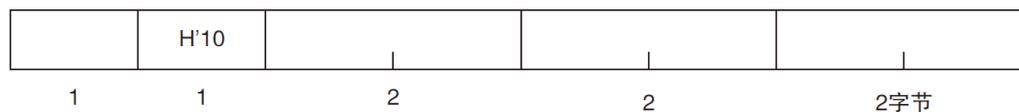


表 0-1 写入多个寄存器

名称	名称	说明
	从站地址	没有指定 RS485 地址时, 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 x 2
CRC-16	校验码	

图 0-2 写入多个寄存器 (0x03) 响应帧

从站地址 功能代码 写入开始地址 元素数量 CRC-16



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	

回波测试

回波测试功能码 0x08，用于调试 Modbus。

图 0-3 回波测试（0x08）

指令帧



响应帧



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：

指令：	01	08	00 00	12 34	ED 7C(CRC-16)
响应：	01	08	00 00	12 34	ED 7C(CRC-16)

& 本章包括以下几方面的内容:

- 寄存器地址

参见:

务必与我公司销售部联系, 获取优利德仪器通讯测试工具, 里面有 Modbus 通讯调试方法。包含了 CRC-16 计算器和浮点数转成 Modbus 浮点数格式。



注意: 除非特别说明, 以下说明中指令和响应帧的数值都是 16 进制数据。

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.

表 0-2 寄存器总览

寄存器地址	名称	数值	说明
2000	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 ABCD, 低位在前
2100	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
2200	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 CDAB
2300	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, 触发方式切换为远程触发。
2400	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, 触发方式切换为远程触发。

0000	读取仪器版本号	4 字节整数	只读，数据占用 2 个寄存器
3000	量程号	0000~0009	读写寄存器，2 字节整数
3001	量程自动	0000: 自动量程 0001: 手动量程 0002: 标称量程	读写寄存器，2 字节整数
3002	测试速度	0000: 慢速 0001: 中速 0002: 快速	读写寄存器，2 字节整数
3003	文件开机调用	0000: 文件 0 0001: 当前文件	读写寄存器，2 字节整数
3004	自动保存	0000: 禁止 0001: 允许	读写寄存器，2 字节整数
3005	系统语言	0000: 英语 0001: 简体中文	读写寄存器，2 字节整数
3006	讯响	0000: 关闭 0001: 合格讯响 0002: 不合格讯响	读写寄存器，2 字节整数
3008	触发器设置	0000: 内部触发 0003: 外部触发	读写寄存器，2 字节整数
3009	触发延时	0: 触发延时关闭 4 字节浮点数范围 (0.1~9.0s)	读写寄存器，2 字节整数
3100	比较器档位数	0000: 比较器关闭 0001: 1-BIN 0002: 2-BIN 0003: 3-BIN 0004	读写寄存器，2 字节整数
3101	比较器模式	0000: ABS 0001: PER 0002: SEQ	读写寄存器，2 字节整数
3102	标称值	4 字节浮点数	读写寄存器，数据占用 2 个寄存器
3110	BIN1 下限值	4 字节浮点数	读写寄存器，数据占用 2 个寄存器

3112	BIN1 上限值	4 字节浮点数	读写寄存器，数据占用 2 个寄存器
3114	BIN2 下限值	4 字节浮点数	读写寄存器，数据占用 2 个寄存器
3116	BIN2 上限值	4 字节浮点数	读写寄存器，数据占用 2 个寄存器
3118	BIN3 下限值	4 字节浮点数	读写寄存器，数据占用 2 个寄存器
311A	BIN3 上限值	4 字节浮点数	读写寄存器，数据占用 2 个寄存器
311C	BIN4 下限值	4 字节浮点数	读写寄存器，数据占用 2 个寄存器
311E	BIN4 上限值	4 字节浮点数	读写寄存器，数据占用 2 个寄存器
3120	BIN5 下限值	4 字节浮点数	读写寄存器，数据占用 2 个寄存器
3122	BIN5 上限值	4 字节浮点数	读写寄存器，数据占用 2 个寄存器
3124	BIN6 下限值	4 字节浮点数	读写寄存器，数据占用 2 个寄存器
3126	BIN6 上限值	4 字节浮点数	读写寄存器，数据占用 2 个寄存器
4000	保存设置到当前文件	固定值：0001	只写寄存器，数据 2 字节
4001	读取当前文件数据	固定值：0001	只写寄存器，数据 2 字节
4002	保存设置到指定文件	0000~0009	只写寄存器，数据 2 字节
4003	读取指定文件数据	0000~0009	只写寄存器，数据 2 字节
5000	执行清零寄存器 读取清零状态	读取： 0001 正在清零 0000 清零成功	只读寄存器，数据占用 1 个寄存器
5001	键锁	0000：解锁 0001：上锁	只写寄存器，2 字节
5002	触发一次 = Handler Trig 引脚	固定值： 0001	只写寄存器，2 字节

获取测量数据

获取测量结果

寄存器 2000~2003 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	2000		0002		CRC-16	
从站	读	寄存器		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	20	00	00	02	CF	CB

响应：

1	2	3	4	5	6	7	8	9
01	03	04	60	AD	78	EC	56	5F

其中 B4~B6 为测量数据：60AD78EC 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 1E20

获取比较器结果【2100】

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	21	00	00	02	CE	37

响应：

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

获取测量结果(CCDD AABB) [2200]

寄存器 2200~2203 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	2200		0002		CRC-16	
从站	读	寄存器		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	22	00	00	02	CE	73

响应：

1	2	3	4	5	6	7	8	9
01	03	04	43	8D	3F	80	6F	CC

其中 B4~B6 为测量数据：43 8D 3F 80 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 AABBCDD：3F 80 8D 43 换算为十进制数为 1.0020614862442017

触发一次并返回测量结果(AABB CCDD) [2300]

寄存器 2300~2303 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	2300		0002		CRC-16	
从站	读	寄存器		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	23	00	00	02	CF	8F

响应:

1	2	3	4	5	6	7	8	9
01	03	04	3F	80	44	98	C5	65

其中 B4~B6 为测量数据: 3F80 4498 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 1.0020933151245117

注意:

此指令会自动将页面切回测量页面, 同时触发方式更改为远程触发。

触发一次并返回测量结果(CCDD AABB) [2400]

寄存器 2400~2403 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	2400		0002		CRC-16	
从站	读	寄存器		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	24	00	00	02	CF	CB

响应:

1	2	3	4	5	6	7	8	9
01	03	04	44	CE	3F	80	9F	6C

其中 B4~B6 为测量数据: 44CE 3F80 代表单精度浮点数, 高位在前。

调整字节顺序 CCDD AABB 为 AABCCDD, 即 3F8044CE 换算为十进制数为 1.0020997524261475

参数设置

速度【3002】

● 写入

1	2	3	4	5	6	7	8	9	10	11
0	1	3	0	0	0	0	0	0	5	7
1	0	0	2	0	1	2	0	1	6	1
	写	寄存器		寄存器数量		字节	数据		CRC	

响应:

1	2	3	4	5	6	7	8
01	10	30	02	00	01	AF	09
		寄存器		寄存器数量		CRC	

● 读取

1	2	3	4	5	6	7	8
01	03	30	02	00	01	2A	CA
	读	寄存器		寄存器数量		CRC	

响应:

1	2	3	4	5	6	7
01	03	02	00	00	B8	44
		字节	数据		CRC	

其中

0000 慢速

0001 中速

0002 快速

0003 高速

比较器设置

比较器参数寄存器地址从 3100 开始。

标称值【3102-3103】

标称值使用 2 个寄存器，3102 和 3103。注意！3103 无法单独读取。

● 写入

100E-3 (单精度浮点数: 0x3DCCCCD)

1	2	3	4	5	6	7	8	9	10	11	12	13
									0	1	2	3

0	1	3	0	0	0	0	3	C	C	C	7	E
1	0	1	2	0	2	4	D	C	C	D	2	1
	写	寄存器		寄存器数量		字节	数据				CRC	

响应:

1	2	3	4	5	6	7	8
01	10	31	02	00	02	EE	F4
		寄存器		寄存器数量		CRC	

● 读取

1	2	3	4	5	6	7	8
01	03	31	02	00	02	6B	37
	读	寄存器		寄存器数量		CRC	

响应:

1	2	3	4	5	6	7	8	9
01	03	04	3D	CC	CC	CD	A3	35
		字节	数据 100E-3				CRC	

极限值【3110-3126】

6 档比较器的极限值从 3110 开始到 3126 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。

下限和上限可以分别设置，也可以同时设置。

● 写入

下限：1E-3, 上限：2E3

发送：01 10 3110 0004 08 3A83126F 3B03126F 6384

响应：01 10 3110 0004 CEF3

● 读取

发送：01 03 3110 0004 4B30

响应：01 03 08 3A83126F 3B03126F C2A7

文件操作

由于仪器设置存储在文件里，如果<文件>页面里的[自动保存]字段未打开，所有的 Modbus 指令设置后，数据无法实时存储在内部 FlashRom 中，会导致下次上电开机之前的寄存器数据恢复成原文件的数值。

用户可以同文件操作寄存器来将所有设置值存储到当前或指定的文件中。同时，也可以调用指定的文件数据到设置寄存器中。

提示：可以将<文件>页面里的[自动保存]字段打开，每次设置的参数会自动保存，文件指令可以不考虑。

保存到当前文件【4000】

发送数值 0001 到 4000 寄存器，仪器将执行文件写入操作，所有设置将全部保存到当前文件中。
此寄存器无法读出。

● 写入

1	2	3	4	5	6	7	8	9	1	1
0	1	4	0	0	0	0	0	0	2	5
1	0	0	0	0	1	2	0	1	6	4
	写	寄存器	寄存器数量	字节	数据	CRC				

响应：

1	2	3	4	5	6	7	8
01	10	40	00	00	01	14	09
		寄存器	寄存器数量	CRC			

其中数据值：

数据	功能	说明
0001	允许操作	固定值

重新载入当前文件【4001】

发送固定值 0001 到 4001 寄存器，仪器将当前文件数据载入到系统中。
此寄存器无法读出。

● 写入

1	2	3	4	5	6	7	8	9	1	1
0	1	4	0	0	0	0	0	0	2	8
1	0	0	1	0	1	2	0	1	7	5
	写	寄存器	寄存器数量	字节	数据	CRC				

其中数据值：

数据	功能	说明
0001	固定值	

保存到指定文件【4002】

发送文件号到 4002 寄存器，仪器将执行文件写入操作，所有设置将全部保存到指定文件中，同时指定的文件将

作为系统当前文件使用。

此寄存器无法读出。

● 写入

1	2	3	4	5	6	7	8	9	1	1
0	1	4	0	0	0	0	0	0	2	8
1	0	0	2	0	1	2	0	1	7	5
	写	寄存器		寄存器数量		字节	数据		CRC	

其中数据值：

数据	功能	说明
0000~0009	文件 0~9	

载入指定文件【4003】

发送文件号到 4003 寄存器，仪器将载入指定文件的设置到系统中，同时指定的文件将作为系统当前文件使用。

此寄存器无法读出。

● 写入

1	2	3	4	5	6	7	8	9	1	1
0	1	4	0	0	0	0	0	0	2	6
1	0	0	3	0	1	2	0	1	6	7
	写	寄存器		寄存器数量		字节	数据		CRC	

其中数据值：

数据	功能	说明
0000~0009	文件 0~9	

系统功能

清零【5000】

读取寄存器 5000，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，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

- 读取

执行清零期间，可以通过读取寄存器数据来确定是否清零完成

发送：01 03 5000 0001 950A

响应：01 03 02 0000 #####

键锁【5001】

写入解锁指令 5001，数据 0000：

01 10 50 01 00 01 02 00 00 F7 84

触发【5002】

写入触发指令 5002，数据 0001：

01 10 50 02 00 01 02 00 01 36 77

注意：此指令仅在内部触发下返回错误码。