硬件配置 参数设置 设备属性 通道信息 设备命令 通讯状态 附录

#莫迪康ModbusRTU

1、 设备简介

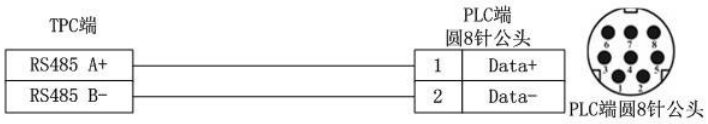
本驱动构件用于本软件通过ModbusRTU协议读写Modicon PLC设备的各种寄存器的数据；同时也可用于对支持ModbusRTU标准协议的各类PLC、仪表、控制器数据的读写。本驱动支持0x01、0x02、0x03、0x04、0x05、0x06、0x0F、0x10常用功能码，对功能码支持请参见附录1。

驱动类型	串口子设备，须挂接在“通用串口父设备”下才能工作
通讯协议	采用莫迪康modbus串口协议
通讯方式	一主一从、一主多从方式。驱动构件为主，设备为从

2、 硬件连接

本软件与设备通讯之前，必须保证通讯连接正确。

通讯连接方式：本软件驱动与设备之间采用标准的RS485或RS232通讯。Modicon TSX系列和Twido系列PLC的编程通讯口(Terminal Port)的通讯电缆图如下：



其他设备的通讯连接，具体请参考对应设备手册。

3、 设备通讯参数

“通用串口父设备”通讯参数设置如下：

其中父设备通讯参数设置应与设备的通讯参数相同，否则无法正常通讯，默认为9600、8、1、E(偶校验)。设备通讯参数的具体设置请参见对应设备手册。

4、 设备构件参数设置

“莫迪康ModbusRTU”子设备参数设置如下：

设备属性名	设备属性值
设备名称	设备0
设备注释	ModbusRTU
初始工作状态	1 - 启动
最小采集周期(ms)	100
设备地址	1
通讯等待时间	200
16位整数字节序	0 - 12
32位整数字节序	0 - 1234
32位浮点字节序	0 - 1234
字符串字节序	0 - 21
字符串编码格式	0 - ASCII
校验数据字节序	0 - LH[低字节,高字节]
功能码校验	0 - 校验功能码
分块采集方式	0 - 按最大长度分块
0区写功能码	1 - 0x0F
4区写功能码	1 - 0x10
1区最大块长	120
0区最大块长	120
3区最大块长	120
4区最大块长	120
通讯间隔字节数	4

<1>内部属性：本驱动不支持内部属性。

<2>最小采集周期(ms)：本软件对设备进行操作的时间周期，单位为ms，默认为100ms，根据采集数据量的大小，设置值可适当调整。

<3>设备地址： modbusrtu协议站号，可取值范围为0-255，默认值为1。该值必须和与组态运行环境通讯的设备的modbusrtu协议站号保持相同。

<4>通讯等待时间：通讯数据接收等待时间，默认设置为200ms，需要根据与组态运行环境通讯的设备的响应速度匹配。具体值应该参照设备的性能参数。

<5>16位整数解码顺序：调整字元件的解码顺序，默认为0-12。

	16位整数解码顺序	举例：0x0001

0-12	表示字元件高低字节不颠倒（默认值）	表示0x0001
1-21	表示字元件高低字节颠倒	表示0x0100

<6>32位整数解码顺序：调整双字元件的解码顺序，默认为0-1234，对于Modicon PLC，请设置为“2-3412”顺序解码。

	32位整数解码顺序	举例：0x0000 0001
0-1234	表示双字元件不做处理直接解码（默认值）	表示0x00000001(1)
1-2143	表示双字元件高低字不颠倒，但字内高低字节颠倒	表示0x00000100(256)
2-3412	表示双字元件高低字颠倒，但字内高低字节不颠倒	表示0x00010000(65536)
3-4321	表示双字元件内4个字节全部颠倒	表示0x01000000(16777216)

<7>32位浮点数解码顺序：调整双字元件的解码顺序，默认为0-1234，对于Modicon PLC，请设置为“2-3412”顺序解码。

	32位浮点数解码顺序	举例：0x3F80 0000
0-1234	表示双字元件不做处理直接解码（默认值）	表示1.0
1-2143	表示双字元件高低字不颠倒，但字内高低字节颠倒	表示-5.78564e-039
2-3412	表示双字元件高低字颠倒，但字内高低字节不颠倒	表示2.27795e-041
3-4321	表示双字元件内4个字节全部颠倒	表示4.60060e-041

<8>字符串编码：调整接收字符串数据时的编码方式，默认设置为0-ASCII。

	字符串编码方式
0-ASCII	ASCII窄字编码方式
1-UNICODE	UNICODE宽字编码方式

<9>字符串解码顺序：调整字符串元件的解码顺序，默认设置为0-21。

	字符串解码顺序	举例：AB
0-21	表示字元件存储字符时高低位顺序颠倒	BA
1-12	表示字元件存储字符时高低位顺序不颠倒	AB

<10> 校验方式：选择LRC校验值的组合方式，对于Modicon PLC及标准PLC设备，使用默认设置即可。

	校验方式
0—LH[低字节，高字节]	校验结果为2个字节，低字节在前，高字节在后
1—HL[高字节，低字节]	校验结果为2个字节，高字节在前，低字节在后

<11> 功能码校验：是否校验功能码，默认为0-校验功能码。对于Modicon PLC及标准PLC设备，使用默认设置即可。

<12> 分块采集方式：驱动采集数据分块的方式，默认为0-按最大长度分块，对于Modicon PLC及标准PLC设备，使用默认设置可以提高采集效率。

0—按最大长度分块：采集分块按最大块长处理，对地址不连续但地址相近的多个连续地址合并一次性读取，而不是多次读取，提升采集的效率。

1—按连续地址分块：采集分块按地址连续性处理，对地址不连续的地址分多次读取。该项多用于仪表类通讯。

例如：有4区寄存器地址分别为1~5、7、9~12的数据需采集，如果选择“0—按最大长度分块”，则两块可优化为地址1~12的数据打包1次完成采集；如果选择“1—按连续地址分块”，则需要采集3次。

<13> 4区写功能码选择：写4区时功能码的选择，默认为1-0x10，通常用于仪表等内存比较小的嵌入式设备，这类设备4区写可能只支持0x06功能码，而不支持0x10功能码。

	4区写功能码
0-0x06	写功能码使用0x06
1-0x10	写功能码使用0x10

<14> 0区写功能码选择：写0区时功能码的选择，默认为1-0x0F，通常用于仪表等内存比较小的嵌入式设备，这类设备0区写可能只支持0x05功能码，而不支持0x0F功能码。

	0区写功能码
0-0x05	写功能码使用0x05
1-0x0F	写功能码使用0x0F

<15> 1区读最大块长设置：1区分块的最大块长，默认值为120。

<16> 0区读最大块长设置：0区分块的最大块长，默认值为120。

<17> 3区读最大块长设置：3区分块的最大块长，默认值为120。

<18> 4区读最大块长设置：4区分块的最大块长，默认值为120。

<19> 通讯间隔字节数：驱动发送数据包后如果接收到了数据长度不为零，则等待当前波特率发送该属性值指定字节数所需的时间，默认为4。modbus-rtu在使用485通讯时，通讯

线缆上通常会挂在多个从设备，每个从设备可能性能不同。请求包之间的间隔时间太小可能导致某些设备粘包，从而导致通讯失败。适当调节该参数可以解决触摸屏通讯性能与

从设备通讯性能匹配，注意太大会导致通讯性能下降。**等待毫秒数计算公式为t ≈ 1000×通讯间隔字节数×(串口数据位数+串口停止位数+串口奇偶校验位数)÷波特率**。

注意：

1) **“解码顺序”**设置：主要是针对非标准ModbusRTU协议的不同解码顺序。当用户通过本驱动软件与设备通讯时，如果出现解析数据值不对，可与厂家咨询后对该选项进行设置。而对于Modicon PLC及支持标准ModbusRTU的PLC及控制器等设备，一般需将“32位整数解码顺序”和“32位浮点数解码顺序”设置为“2—3412”。

另外，在使用本驱动与“ModbusRTU数据转发设备”构件通讯时，“解码顺序”需按默认值设置，否则会导致通讯失败或解析数据错误。

2) **“分块采集方式”**设置：主要是针对非标准ModbusRTU协议设备。当用户通过本驱动软件与设备通讯时，如果按默认“0—按最大长度分块”时，出现读取连续地址正常，而不连续地址不正常时，可与厂家咨询，并设置为“1—按连续地址分块方式”尝试是否可正常通讯。而对于Modicon PLC及支持标准ModbusRTU协议的PLC及控制器等设备，直接使用默认设置即可，这样可以提高采集效率。

5、 通道信息

<1>寄存器信息：

寄存器	简称	寄存器类型(位/字)	数据类型	操作方式	PLC地址表示	读功能码	写功能码	地址范围
[1区]输入继电器	1	位	BT	只读	10进制	0x02	--	1~65536
[0区]输出继电器	0	位	BT	读写	10进制	0x01	0x05、0x0F	1~65536
[3区]输入寄存器	3	字	BT00~BT15、 WUB、WB、WD、DUB、DB、DD、DF、STR	只读	10进制	0x04	--	1~65536
[4区]输出寄存器	4	字	BT00~BT15、 WUB、WB、WD、DUB、DB、DD、DF、STR	读写	10进制	0x03	0x06、0x10	1~65536

说明：

1) 本驱动构件支持0x01、0x02、0x03、0x04、0x05、0x06、0x0F、0x10等常用功能码，对于其它功能码暂不支持。
2) 以上功能码均以16进制标注。
3) “[1区]输入寄存器”和“[3区]输入寄存器”不支持写功能码。
4) “[0区]输出继电器”在写入继电器时，使用设备属性0区写功能码设置中选择的功能码。
5) “[4区]输出寄存器”在数据时，使用设备属性4区写功能码设置中选择的功能码。

注意：

1) 地址范围为标准modbusrtu可访问的地址范围，通常与组态运行环境通讯的设备地址范围小于在这个范围之内，例如仪表设备，它的地址范围通常比较小。总之，实际可访问的地址范围受与之通讯的设备限制，具体地址范围需要参考与组态运行环境通讯的设备的通讯手册或者直接咨询设备厂商。除此之外，与组态运行环境通讯的设备的起始地址可能不是从1开始的，那么该设备的地址与组态通道的地址存在一个固定的偏差，使用时需要注意通道地址与通讯设备地址的对应关系。

<2>数据类型表：

BTdd	位（dd范围：00－15）
BUB	8位 无符号二进制
BB	8位 有符号二进制
BD	8位 2位BCD
WUB	16位 无符号二进制
WB	16位 有符号二进制
WD	16位 4位BCD
DUB	32位 无符号二进制
DB	32位 有符号二进制
DD	32位 8位BCD
DF	32位 浮点数
STR	字符串

1) 位序号：BT00,BT01---BT15；
2) 数值类型：BB、BUB、BD、WUB、WB、WD、DUB、DB、DD、DF；
3) 第一个字母表示数据的长度，B表示是字节数据，W表示是字数据，D表示是双字数据；
4) 最后一个或两个字母表示数据类型，B表示二进制数，D表示BCD码，F表示浮点数；
5) 字符中二进制数中带U表示无符号数，不带U的表示有符号数。

6、 设备命令

本设备构件提供设备命令，用于进行相应的读写操作，设备命令的格式如下：

设备命令	命令介绍		
Read	格式1	Read(寄存器名称, 寄存器地址, 数据类型=读取值)	
	用途	按照指定数据格式读取寄存器某一地址数值。	
	参数	寄存器名称	参数可以为字符型变量、字符串常量，使用寄存器简称。
		寄存器地址	参数可以为开关型变量、数值型变量、数值常量。
		数据类型	参数可以为字符型变量、字符串常量。
		读取值	参数可以为数值型变量、开关型变量。
	例1	!SetDevice(设备0, 6, "Read(0,10,BT00=Data01)")	
		读取0区地址为10的数值，放入变量Data01。	
	例2	!SetDevice(设备0, 6, "Read(4,20,WUB=Data01)")	
		读取寄存器4区地址20的16位无符号值，放入数值型变量Data01。	
Write	格式1	Write(寄存器名称, 寄存器地址, 数据类型=写入值)	
	用途	将数值以指定数据格式写入寄存器某一地址中。	
	参数	寄存器名称	参数可以为字符型变量、字符串常量，使用寄存器简称。
		寄存器地址	参数可以为开关型变量、数值型变量、数值常量。
		数据类型	参数可以为字符型变量、字符串常量。
		写入值	参数可以为数值型变量、开关型变量、数值常量。
	例1	!SetDevice(设备0, 6, "Write(0,10,BT00=Data01)")	
		将数值型变量Data01的值写入0区地址10中。	

	例2	!SetDevice(设备0, 6, "Write(4,10,WUB=Data01)")	
		将数值型变量Data01的值，以16位无符号格式写入寄存器4区地址10中。	
ReadP	格式1	ReadP(寄存器名称, 寄存器起始地址, 数据类型, 操作个数n, 读取值1, 读取值2,.., 读取值n)	
	格式2	ReadP(寄存器名称, 寄存器起始地址, 数据类型, 操作个数n, 读取值1, 读取值2,.., 读取值n, 返回值)	
	用途	从寄存器指定地址开始，按照指定数据类型连续读取n个数值，将读取值分别存放入变量，注读取值变量个数与操作个数值相同。	
	参数	寄存器名称	参数可以为字符型变量、字符串常量，使用寄存器简称。
		寄存器地址	参数可以为开关型变量、数值型变量、数值常量。
		数据类型	参数可以为字符型变量、字符串常量。
		操作个数	参数可以为开关型变量、数值型变量、数值常量。
		读取值	参数可以为开关型变量、数值型变量。
		返回值	参数可以为开关型变量、数值型变量。
	例1	!SetDevice(设备0,6,"ReadP(4,10,WUB,2,Data01,Data02)")	
		表示读取寄存器4区从地址10开始的两个16位无符号数值，放入数值型变量Data01、Data02。	
	例2	!SetDevice(设备0,6,"ReadP(4,10,WUB,2,Data01,Data02,nReturn)")	
		表示读取寄存器4区从地址10开始的两个16位无符号数值，放入数值型变量Data01、Data02，执行结果存入变量nReturn。	
WriteP	格式1	WriteP(寄存器名称, 寄存器起始地址, 数据类型, 操作个数n, 写入值1, 写入值2,..,写入值n)	
	格式2	WriteP(寄存器名称, 寄存器起始地址, 数据类型, 操作个数n, 写入值1, 写入值2,..,写入值n,返回值)	
	用途	从寄存器指定地址开始，按照指定数据类型连续写入n个数值，注写入值变量个数与操作个数值相同。	
	参数	寄存器名称	参数可以为字符型变量、字符串常量，使用寄存器简称。
		寄存器地址	参数可以为开关型变量、数值型变量、数值常量。
		数据类型	参数可以为字符型变量、字符串常量。
		操作个数	参数可以为开关型变量、数值型变量、数值常量。
		写入值	参数可以为开关型变量、数值型变量、数值常量。
		返回值	参数可以为开关型变量、数值型变量。
	例1	!SetDevice(设备0,6,"WriteP(4,10,WUB,2,Data01,Data02)")	
		表示将数值型变量Data01、Data02的值，以16位无符号形式写入寄存器4区从地址10起始的两个寄存器。	
	例2	!SetDevice(设备0,6,"WriteP(4,10,WUB,2,Data01,Data02,nReturn)")	
		表示将数值型变量Data01、Data02的值，以16位无符号形式写入寄存器4区从地址10起始的两个寄存器，执行结果存入变量nReturn。	
ReadPV	格式1	ReadPV (寄存器名称, 寄存器起始地址, 数据类型, 操作个数n, 读取值)	
	格式2	ReadPV (寄存器名称, 寄存器起始地址, 数据类型, 操作个数n, 读取值, 返回值)	
	用途	从寄存器指定地址开始，按照指定数据类型连续读取n个数值，将读取值分别放入以“读取值”为起始，连续n个变量中，因该命令是将读取数值分别放入“读取值”为起始的连续n个变量中，使用时注意变量对应寄存器地址的连续性。	
	参数	寄存器名称	参数可以为字符型变量、字符串常量，使用寄存器简称。
		寄存器地址	参数可以为开关型变量、数值型变量、数值常量。
		数据类型	参数可以为字符型变量、字符串常量。
		操作个数	参数可以为开关型变量、数值型变量、数值常量。
		读取值	参数可以为开关型变量、数值型变量，注该参数为变量。
		返回值	参数可以为开关型变量、数值型变量。
	例1	!SetDevice(设备0,6,"ReadPV(4,10,WUB,5,Data01)")	
		表示读取寄存器4区从地址10开始的5个16位无符号数值，放入数值型变量Data01为起始，连续5个变量（即：Data01、Data02、Data03、Data04、Data05）。	
	例2	!SetDevice(设备0,6,"ReadPV(4,10,WUB,5,Data01,nReturn)")	
		表示读取寄存器4区从地址10开始的5个16位无符号数值，放入数值型变量Data01为起始，连续5个变量（即：Data01、Data02、Data03、Data04、Data05），执行结果存入变量nReturn。	
	格式1	WritePV (寄存器名称, 寄存器起始地址, 数据类型, 操作个数n, 写入值)	
	格式2	WritePV (寄存器名称, 寄存器起始地址, 数据类型, 操作个数n, 写入值, 返回值)	
	用途	读取以“写入值”为起始，连续n个的变量数值，将读取值从寄存器指定地址开始，按照指定数据类型分别写入n个寄存器中，因该命令是读取以 “写入值”为起始的连续n个变量数值，使用时注意变量对应寄存器地址的连续性。	
	参数	寄存器名称	参数可以为字符型变量、字符串常量，使用寄存器简称。
		寄存器地址	参数可以为开关型变量、数值型变量、数值常量。
		数据类型	参数可以为字符型变量、字符串常量。

WritePV		操作个数	参数可以为开关型变量、数值型变量、数值常量
		写入值	参数可以为开关型变量、数值型变量，注该参数为变量。
		返回值	参数可以为开关型变量、数值型变量。
	例1	!SetDevice(设备0, 6,"WritePV(4,10,WUB,5,Data01)")	
		表示将以数值型变量Data01为起始，连续5个变量的值（即：Data01、Data02、Data03、Data04、Data05），以16位无符号形式写入寄存器4区从地址10起始的五个寄存器。	
	例2	!SetDevice (设备0, 6,"WritePV(4,10,WUB,5,Data01,nReturn)")	
		表示将以数值型变量Data01为起始，连续5个变量的值（即：Data01、Data02、Data03、Data04、Data05），以16位无符号形式写入寄存器4区从地址10起始的五个寄存器，执行结果存入变量nReturn。	
ReadBlock	格式1	ReadBlock (寄存器名称, 寄存器起始地址, [数据类型1][数据类型2][数据类型n], 操作个数n, 写入变量)	
	格式2	ReadBlock (寄存器名称, 寄存器起始地址, [数据类型1][数据类型2][数据类型n], 操作个数n, 写入变量,返回值)	
	用途	从寄存器指定地址开始，按照指定数据类型格式连续读取n组数据，将读取值以特定CSV[5]格式存入字符变量中。	
	参数	寄存器名称	参数可以为字符型变量、字符串常量，使用寄存器简称。
		寄存器地址	参数可以为开关型变量、数值型变量、数值常量。
		数据类型	参数可以为字符型变量、字符串常量。
		操作个数	参数可以为开关型变量、数值型变量、数值常量。
		写入值	参数可以为字符串变量，注该参数为字符型变量。
		返回值	参数可以为开关型变量、数值型变量。
	例1	!SetDevice(设备0, 6, "ReadBlock(4,10,[WUB][DF],3,strData)")	
		表示读取寄存器4区从地址10开始，按WUB,DF格式连续读取3组数据 （即：数据格式为WUB、DF、WUB、DF、WUB、DF），并以相应格式解析并以逗号间隔的CSV格式存入字符变量StrData。	
	例2	!SetDevice(设备0, 6, "ReadBlock(4,10,[WUB][DF],3,strData,nReturn)")	
		表示读取寄存器4区从地址10开始，按WUB,DF格式连续读取3组数据（即：数据格式为WUB、DF、WUB、DF、WUB、DF），并以相应格式解析并以逗号间隔的CSV格式存入字符变量StrData，执行结果存入变量nReturn。	
WriteBlock	格式1	WriteBlock (寄存器名称, 寄存器起始地址, [数据类型1][数据类型2][数据类型n], 操作个数n, 写入值)	
	格式2	WriteBlock (寄存器名称, 寄存器起始地址, [数据类型1][数据类型2][数据类型n], 操作个数n, 写入值, 返回值)	
	用途	将字符串变量中的数据，以指定格式解析并按照指定数据类型格式，写入寄存器指定地址开始的连续地址中。 注字符串变量数据格式为特定csv格式。	
	参数	寄存器名称	参数可以为字符型变量、字符串常量，使用寄存器简称。
		寄存器地址	参数可以为开关型变量、数值型变量、数值常量。
		数据类型	参数可以为字符型变量、字符串常量。
		操作个数	参数可以为开关型变量、数值型变量、数值常量。
		写入值	参数可以为字符串变量。
		返回值	参数可以为开关型变量、数值型变量。
	例1	!SetDevice(设备0, 6, "WriteBlock(4,10,[WUB][DF],3,strData)")	
		表示将strData字符变量中的CSV格式的数据，按指定格式，写入寄存器4区从地址10开始的连续地址。	
	例2	!SetDevice(设备0, 6, "WriteBlock(4,10,[WUB][DF],3,strData,nReturn)")	
		表示将strData字符变量中的CSV格式的数据，按指定格式，写入寄存器4区从地址10开始的连续地址， 执行结果存入变量nReturn。	
SetAddress	格式1	SetAddress (写入值)	
	用途	读取变量值修改设备地址	
	参数	写入值	参数可以为开关型变量、数值型变量、数值常量。
	例1	!SetDevice(设备0,6,"SetAddress(Addr)")	
		表示设置设备0的设备地址，设置地址值为Addr的值。	
GetAddress	格式1	GetAddress (读取值)	
	用途	获取当前设备地址值将其放入变量	
	参数	读取值	参数可以为开关型变量、数值型变量。
	例1	!SetDevice(设备0,6,"GetAddress(Addr)")	
		表示获取设备0的设备地址，将值赋值给Addr。	
	格式1	SetCommPara(波特率写入值,数据位写入值,停止位写入值,校验位写入值,返回值)	
	用途	读取变量值修改设备地址	
		波特率写入值	参数可以为开关型变量、数值型变量、数值常量。
		数据位写入值	参数可以为开关型变量、数值型变量、数值常量。

SetCommPara	参数	停止位写入值	参数可以为开关型变量、数值型变量、数值常量。
		校验位写入值	参数可以为开关型变量、数值型变量、数值常量。
		返回值	参数可以为开关型变量、数值型变量。
	例1	!SetDevice(设备0,6, "SetCommPara(nBaudrate,nDatabit,nStopbit,nParity)")	
		表示设定设备0所在的父设备的串口参数，nBaudrate为波特率,nDatabit为数据位,nStopbit为停止位,nParity为校验位,nReturn为返回值,设置成功返回0,设置失败返回-1。	
GetCommPara	格式1	GetCommPara(波特率读取值,数据位读取值,停止位读取值,校验位读取值)	
	用途	获取当前设备地址值将其放入变量	
	参数	波特率读取值	参数可以为开关型变量、数值型变量。
		数据位读取值	参数可以为开关型变量、数值型变量。
		停止位读取值	参数可以为开关型变量、数值型变量。
		校验位读取值	参数可以为开关型变量、数值型变量。
	例1	!SetDevice(设备0,6, "GetCommPara(nBaudrate,nDatabit,nStopbit,nParity)")	
		表示获取设备0所在的父设备的串口参数，nBaudrate为波特率,nDatabit为数据位,nStopbit为停止位,nParity为校验位	
SETRETRYCOUNT	格式1	SETRETRYCOUNT (写入值)	
	用途	设置采集重试次数。	
	参数	写入值	参数可以为开关型变量、数值型变量、字符型变量、常量。
	例1	!SetDevice(设备0, 6, " SETRETRYCOUNT (3)")	
		表示设置采集重试次数为3 次。	
GETRETRYCOUNT	格式1	GETRETRYCOUNT (读取值)	
	用途	获取采集重试次数值。	
	参数	读取值	参数可以为开关型变量、数值型变量、字符型变量。
	例1	!SetDevice(设备0, 6, " GETRETRYCOUNT (nCount)")	
		表示获取采集重试次数，将获取值放入数值变量nCount。	

说明：

<1>寄存器名称：字符型变量，表示当前操作的寄存器，使用寄存器简称，寄存器对应简称参看**通道信息**。

<2>寄存器地址：数值型变量，表示当前操作的寄存器地址，查阅相关手册确定。

<3>数据类型： 字符型变量，表示当前操作的寄存器数据类型。

<4>返回状态： 返回批量读写设备命令的执行状态（当设备命令格式错误时无效），返回值的意义请参见**通讯状态**说明，返回状态为可选参数，用户也可通过通讯状态通道查看返回结果。

<5>指定CSV格式：每条数据以回车换行分隔，数据内部以逗号分隔：

以WriteBlock为例：!SetDevice(设备0,6, "WriteBlock(4,10,[WUB][WD],3,strData)")，strData数据内容为

213,3213

23,0

213,32

批量读写说明：

<1>批量读写操作（包括：ReadP、ReadPV、WriteP、WritePV），为对同类寄存器连续地址的一次性读写操作，使用时注意变量对应寄存器地址的连续性。

<2>批量读写操作（包括：ReadP、ReadPV、WriteP、WritePV），建议一次批量操作数据量不要过大，否则会影响正常采集效果。

获取串口参数：SetCommPara命令设置设备的串口参数(波特率、数据位、停止位、校验位)

<1>第一个参数 - 波特率为4800,9600,19200等常用波特率；

<2>第二个参数 - 数据位为7位或8位；

<3>第三个参数 - 停止位为1位或2位；校验位为0代表无校验，为1代表奇校验，为2代表偶校验。

7、 采集通道

<1>通讯状态：

通讯状态值	代表意义
0	表示当前通讯正常
-1	表示驱动加载失败
-2	表示通讯端口打开失败
1001	表示数据类型错误
1002	表示响应的协议地址或功能码错误
1003	表示协议CRC校验错误
1004	表示协议无响应
1005	表示协议响应帧长度不对，协议不完整
1006	表示通讯错误
	1008 错误码1 功能码不支持

1007-1999	表示协议拒绝错误，协议错误码为通讯值与1007的差值	1009	错误码2	读写的寄存器起始地址，起始地址加长度等超过设备所支持的最大范围
		1010	错误码3	读操作时，读取寄存器的数量超过Modbus支持最大读取数量， 写操作时，写入值异常或写入长度超过Modbus支持的最大写入长度
		1011	错误码4	读取或写入失败
2001	表示设备命令名称错误			
2002	表示设备命令寄存器名称错误			
2003	表示设备命令寄存器访问权限错误			
2004	表示设备命令指定的数据类型错误，没有该类型的名称			
2005	表示设备命令指定的数据类型存在，但是寄存器不支持该类型			
2006	表示设备命令参数个数不正确			
2007	表示设备命令参数无效，参数为非法值			
2008	表示设备命令设置分块错误			
2009	表示设备命令其他未知错误			
3001	表示驱动设备命令不完整，格式错误			
3002	表示驱动检测到通道写变量类型错误			
3003	表示驱动状态错误，主要包括内存不足或通讯接口无效、索引地址错误等错误、只读通道执行写操作			
3004	表示驱动属性流化错误			
3005	表示驱动通道、属性初始化等错误			
3006	表示数据库错误			
3007	表示字符串字节数小于等于1错误			

<2>常见故障分析：

故障现象	分析	处理建议
通讯状态为-1	驱动文件缺失	1、重新安装驱动
通讯状态为-2	通讯端口打开失败	1、检查串口父设备串口号是否正确
		2、检查父设备相关参数是否正确
		3、检查连线是否正确（硬件连接串口和父设备选择串口是否一致）
		4、检查父设备参数和PLC的设置是否相同
		5、通过三方软硬件检测客户自身设备是否正确
通讯状态为-10	地址偏移错误	1、检查存在地址偏移的通道，偏移后的地址是否超过地址范围
通讯状态为1002	响应的协议地址功能码错误	1、 检查设备是否符合标准Modbus协议
通讯状态为1003	协议CRC校验错误	1、检查现场是否存在电磁干扰等
通讯状态为1004	协议无响应	1、适当延长通讯等待时间
通讯状态为1005	协议响应帧长度不对，协议不完整	1、适当延长通讯等待时间
		2、检查是否存在电磁干扰
通讯状态为1006	表示通讯错误	1、适当延长通讯等待时间
		2、检查父设备相关参数是否正确
		3、检查连线是否正确（硬件连接串口和父设备选择串口是否一致）
		4、检查父设备参数和PLC的设置是否相同
		5、检查串口是否被占用
		6、检查通道的地址是否超过设备最大地址范围
		7、通过三方软硬件检测客户自身设备是否正确
通讯状态为1007	表示协议拒绝错误，协议错误码为	1008 错误码1 收发帧的功能码不支持，确定设备支持的具体功能码
		1009 错误码2 检查读写的寄存器起始地址，起始地址加长度等是否超过设备所支持的最大范围
		读操作时，检查读寄存器的数量是否超过

	通讯与1007差值的值	1010	错误码3	Modbus支持最大读取数量
				写操作时，检查写入值是否正常或写入长度是否超过Modbus支持的最大写入长度
		1011	错误码4	读取或写入失败
通讯状态为2001	设备命令名称错误	1、检查设备命令是否被支持		
通讯状态为2002	设备命令存名错寄器称误	1、检查设备命令是否支持该寄存器		
		2、检查设备命令中的寄存器是否存在		
通讯状态为2003	设备命令存访权错寄器问限误	1、检查Read*相关设备命令中的寄存器是否是只写寄存器		
		2、检查Write*相关设备命令中的寄存器是否是只读寄存器		
通讯状态为2004	设备命令定数类错指的据型误，没该型名称有类的	1、检查设备命令之中的寄存器是否支持设备命令之中的数据类型，检查数据类型是否存在		
通讯状态为2005	设备命令定数类存指的据型在，但寄器支该型是存不持类	1、检查设备命令之中的数据类型是否是字符串类数据类型		
		2、检查设备命令之中的数据类型是否是寄存器支持的数据类型		
通讯状态为2006	设备命令数数正参个不确	1、检查设备命令参数列表是否为空		
		2、检查设备命令参数列表的参数个数是否正确		
通讯状态为2007	设备命令数参无效，数非参为法值	1、检查设备命令是否为空字符串		
		2、检查各个参数是否正确		
通讯状态为3001	设备命令式完格不整，式格错误	1、检查设备命令格式是否完整，左右括号是否匹配		
通讯状态为3003	驱动态状错误，主要括包存内足不通或讯接口无效、索引	1、检查设备命令是否为空字符串		
		2、通道写时，检查通道是否为只读通道		

	地址错误等错误	3、检查地址偏移是否错误，超过最大地址范围
通讯状态在0与非0之间跳变	通讯不稳定或读取地址超范围	1、同通讯状态为1003的处理
		2、读取数据地址超范围（典型情况为，添加某通道后，导致通讯状态变非0）
通讯状态为0，数据不正确	组态工程错误	1、新建工程测试驱动
		2、检测通道是否连接变量
		3、检测工程是否对数据进行处理
通讯速度太慢	通讯数据过大或采集周期设置过长	1、将“采集优化”属性设置为“1-优化”
		2、减小父设备及子设备的最小采集周期（最小可设置为20ms）
		3、使用设备命令，减少实时采集的数据
	通讯次数过多	4、将数据放到连续的地址块中，提高块读效率
		5、将不同寄存器的数据放到同一寄存器连续的地址块中，减少采集块数，提高采集效率

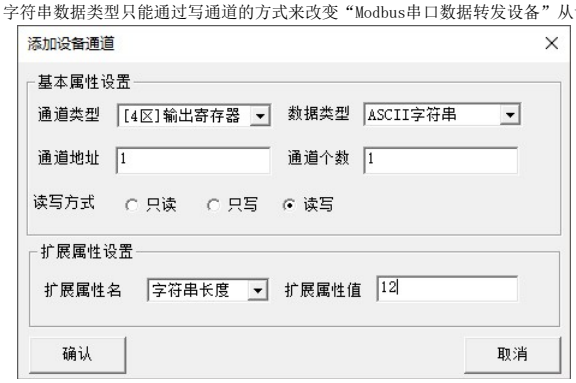
<4>“[0区]输出继电器”在批量写入多个继电器时，使用0x0F功能码。

注意：添加寄存器通道时，起始地址均为1，这是遵从modbusrtu协议，即所谓的“协议地址”，对于部分寄存器起始地址为0的设备，通道添加时，地址应加1处理

附录2

字符串通道：

<1>字符串数据类型只能通过写通道的方式来改变“Modbus串口数据转发设备”从设备的值。建立字符串通道4STR0001_12如下图所示：

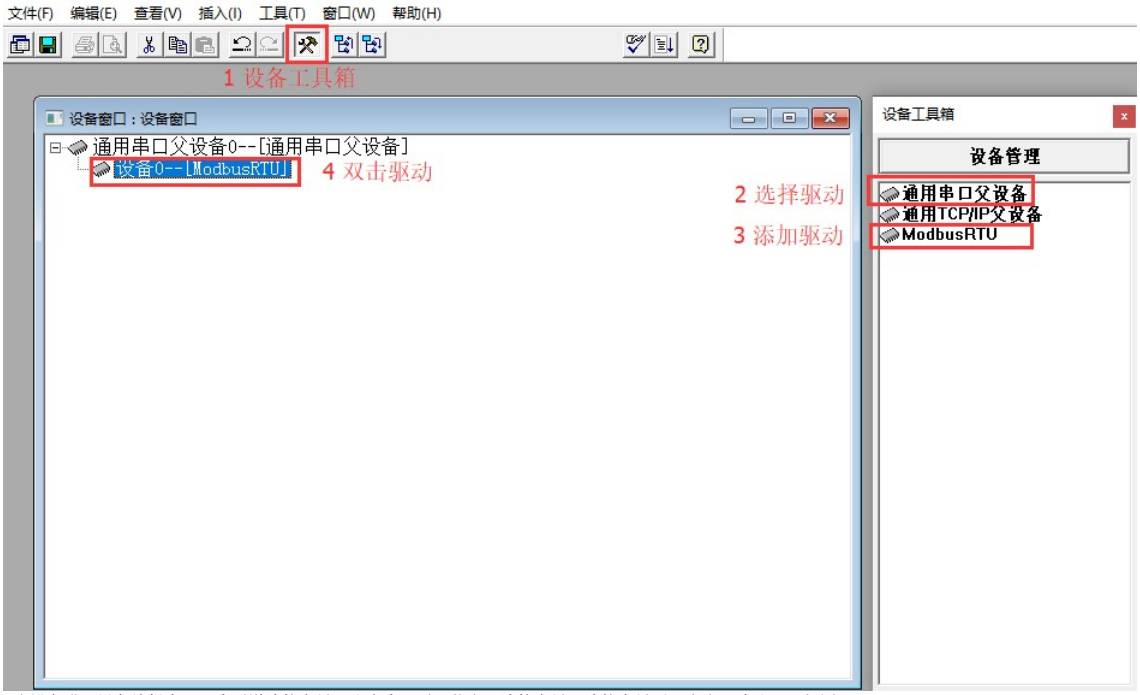


注意：在使用字符串时，写字符串的长度尽量不要超过规定的字节数（汉字占两个字节，其余字符占一个字节），字符串的字节数不得小于等于1如果字符串的字节数小于等于1，则驱动在运行时报告错误3007。特别注意是否存在字符串的字节数是否小于等于1。

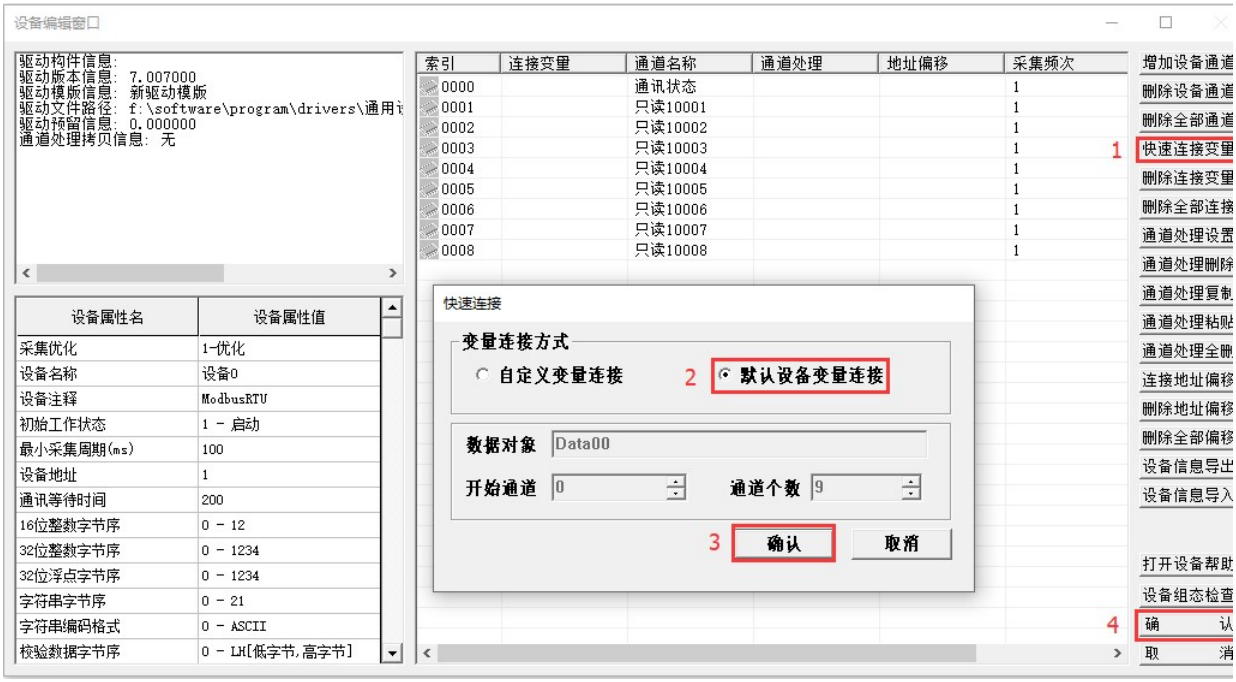
附录3

通讯状态设置

<1>新建工程，双击设备窗口，进入“设备组态：设备窗口”后，双击“工具箱”弹出设备工具箱，将驱动构建从设备管理面板添加设备窗口。如图：



<2> 双击设备进入设备编辑窗口，为通道连接变量，注意为“通讯状态”连接变量，连接变量后，点击“确认”。如图：



返回顶部