Siglent
编程手册

SDL1000X 系列可编程直流电子负载

PG0801X-C01A

版权和声明

版权

商标信息

SIGLENT 是深圳市鼎阳科技有限公司的注册商标

声明

- 本公司产品受已获准及尚在审批的中华人民共和国专利的保护。
- 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。
- 未经本公司许可，不得以任何形式或手段复制、摘抄、翻译本手册的内容。

Content

1. 编程概述

SDL1000X 系列电子负载同时支持 USB 和 LAN 接口。用户可以通过这些接口，结合 NI-VISA 和相应的编程语言对仪器进行远程控制。本章将介绍如何构建电子负载与电脑之间的通信。本文还介绍了如何进行远程控制。

1.1 搭建编程环境

1.1.1 使用 VISA 的编程环境

1.1.1.1 安装 NI-VISA

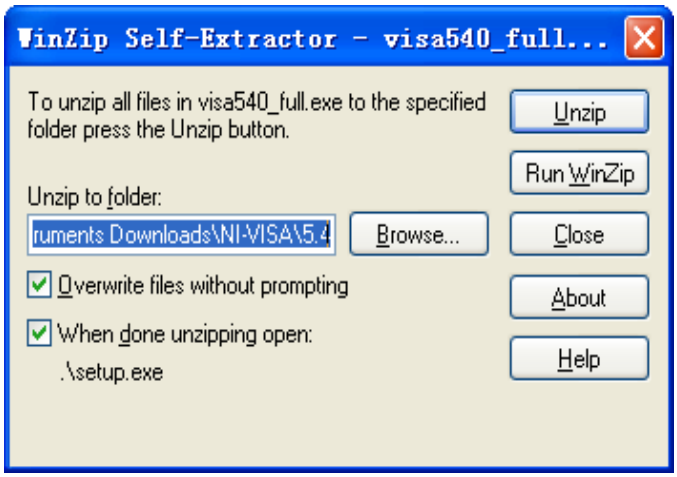
在开始编程之前，您需要安装 NI-VISA，您可以从 NI-VISA 网站下载。关于 NI-VISA，有完整版和实时版（Run-Time Engine version）。完整的版本包括 NI 设备驱动器和一个名为 NI MAX 的工具。NI MAX 是一个用户接口，用于控制该设备。实时版本比完整版小得多，它只包括 NI 设备驱动程序。

例如，你可以从 http://www.ni.com/download/ni-visa-5.4/4230/en/下载 NI-VISA 5.4 完整版。

您也可以下载 NI-VISA 实时 5.4 版到您的电脑，并安装它作为默认选择。它的安装过程与完整版相似。

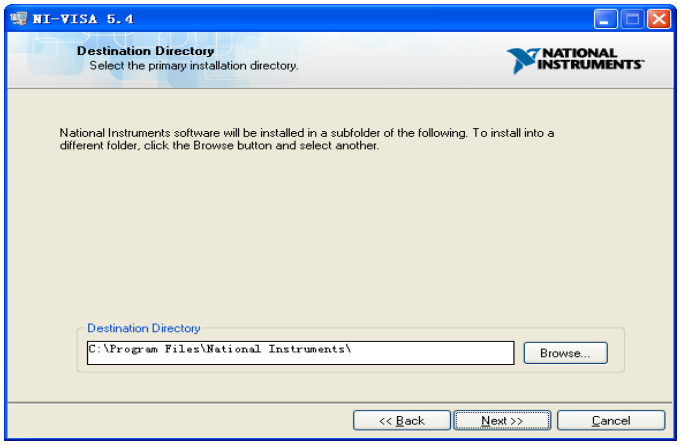
下载完文件后，您可以按照下面的步骤进行安装：

a. 双击 visa540_full.exe，对话框如下图所示：

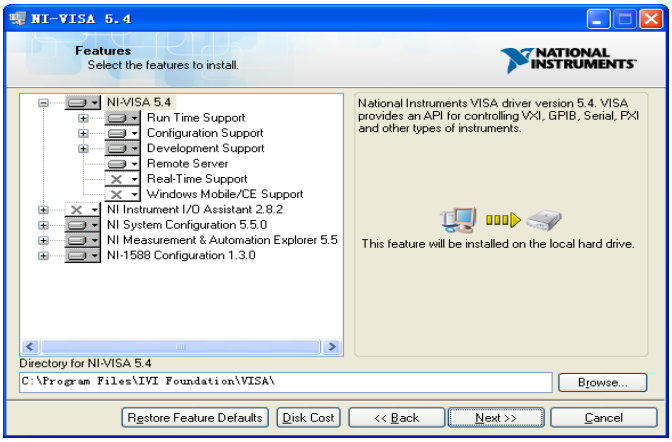


b. 单击解压缩，安装过程将在解压缩文件后自动启动。如果您的电脑需要安装.NET Framework4，它的安装过程也将自动启动。

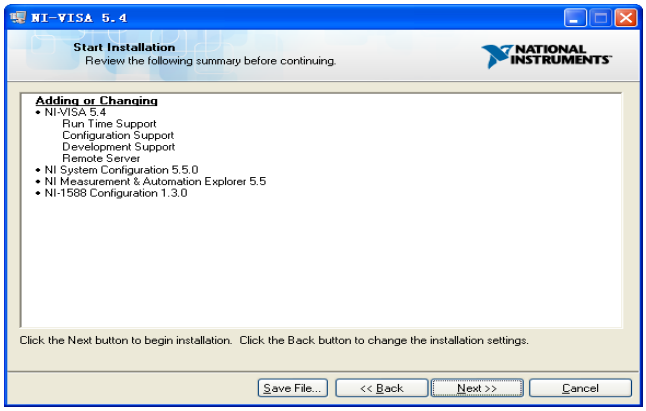
c. NI-VISA 安装对话框如上图所示，点击下一步开始安装过程。



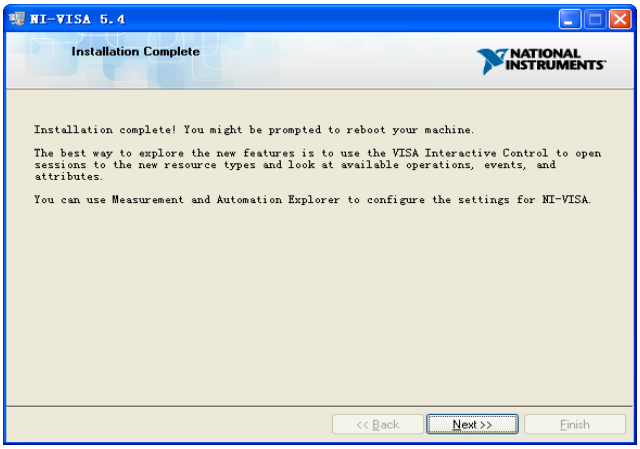
设置安装路径，默认路径为“C:\Program Files\National Instruments\”，您可以改变它。点击下一步，弹出如上图所示的对话框。



d. 单击下一步两次，在许可协议对话框中，选择“我接受上述两个许可协议（S）”，然后点击下一步，显示对话框如下图所示：



e. 点击下一步安装。



现在安装完成，请重新启动您的计算机。

1.1.1.2 连接设备

根据您电子负载设备的具体型号，能够通过 USB 或 LAN 接口与 PC 进行通信。本手册以 USB 连接作为例子。（有关通过 LAN 接口与 PC 通讯的介绍说明请参阅用户手册。）

a. 在电子负载的后面板连接 USB 设备接口，并使用 USB 电缆连接电脑的 USB 接口。若 PC 机未安装该设备的驱动程序时，它会自动安装设备驱动程序。

b. 等待安装完成，然后进入下一个步骤。

1.2 如何远程控制

1.2.1 用户自定义编程

用户可以使用 SCPI 命令编程和控制电子负载。有关的详细信息，请参阅“编程实例”。

1.2.2 通过 NI-MAX 发送 SCPI 命令

您可以通过 NI-MAX 软件直接发送 SCPI 命令远程控制电子负载。

2. SCPI 简介

2.1 命令格式

SCPI 命令为树状层次结构，包括多个子系统，每个子系统由一个根关键字和一个或数个层次关键字构成。命令行通常以冒号“**:**”开始；关键字之间用冒号“**:**”分隔，关键字后面跟随可选的参数设置，命令和参数以“空格”分开，多个参数的，参数之间用逗号“**,**”分隔。命令行后面添加问号“**?**”，表示对此功能进行查询。
例如：

:SENSe:FREQuency:CENTer <freq>
:SENSe:FREQuency:CENTer?

SENSe 是命令的根关键字，FREQuency 和 CENTer 分别是第二级、第三级关键字。命令行以冒号“**:**”开始，同时将各级关键字分开，<freq>表示可设置的参数；问号“**?**”表示查询；命令: :SENSe:FREQuency:CENTer 和参数<freq>之间用“空格”分开。

2.2 符号说明

下面四种符号不是 SCPI 命令中的内容，不随命令发送，但是通常用于辅助说明命令中的参数。

1、三角括号 < >

三角括号中的参数必须用一个有效值来替换，三角括号不随命令字符串发送。例如：以**:SOURce:BATTery:LEVel 4** 的形式发送**[:SOURce]:BATTery:LEVel <value>**命令。

2、大括号 { }

大括号中的参数是必选项，用于列举同一参数的多个选项，大括号不随命令字符串发送。例如：**[:SOURce]:INPut[:STATe] {ON | OFF | 0 | 1}**中的**{ON | OFF | 0 | 1}**。

3、**竖线 |**

竖线用于分隔多个参数选项，发送命令时必须选择其中一个参数。 例如：**[:SOURce]:INPut[:STATe] {ON | OFF | 0 | 1}**命令中，可选择的命令参数为“OFF”、

“ON”、“0”或“1”；发送指令可如下：

:SOURce:INPut:STATe ON 或 **:SOURce:INPut:STATe 0**

4、方括号 [ ]

方括号中的内容（命令关键字）是可省略的。如果省略参数，仪器将该参数设置为默认值。例如：

对于**[:SOURce]:INPut[:STATe]?**命令，发送下面四条命令的效果是一样的：

:INPut?

:INPut:STATe?

:SOURce:INPut?

:SOURce:INPut:STATe?

2.3 参数类型

本手册介绍的命令中所含的参数可以分为以下 6 种类型：布尔型、枚举型、非负整型、非负浮点型、特定格式型。

1、布尔型

参数取值为“OFF”、“ON”、“0”或“1”。例如：

[:SOURce]:SHORt[:STATe] {ON | OFF | 0 | 1}

2、枚举型

参数取值为“{}”与“|”所列举的值。例如：

[:SOURce]:FUNCtion {CURRent | VOLTage | POWer | RESistance | LED}

参数为“CURRent”、“VOLTage”、“POWer”、“RESistance”、“LED”。

3、非负整型

在命令中凡参数以<number><step>表示的皆为取非负整型的数，参数在有效值范围内可以取任意非负整数值。

注意：此时若所取的数超出范围则为无效指令；若数值为小数时，则做取整处理。例如：

[:SOURce]:LIST:LEVel? <step>

参数<step>可取任一非负整数。当取 3.988 时则取整为 3

4、非负浮点型

在命令中凡参数以<value>表示的皆为浮点型，可以任意进行取值。

注意：此时若所取的数超出范围则为无效指令；若数值为整数时，则仍做浮点数处理。例如：

[:SOURce]:CURRent:IRANGe < value >

参数< value >可任意非负浮点型。

6、特定格式型

参数必须以特定格式，指定的符号连接输入发送。例如：

LAN:IPADdress <aaa.bbb.ccc.ddd>

参数为必须以“.”进行区别连接，当缺少数值字符时，参数仍有效，但参数一递进形式设置，如：原本 IP 地址为 12.13.14.2 时，当参数为 15.6.0 设置后，IP 地址则为 15.6.0.2。

2.4 命令缩写

所有命令对大小写不敏感，可完整输入命令，包含所有大写或小写，也可以使用缩写，但是如果要缩写，必须完整且仅仅输入命令格式中的大写字母，例如：

:SOURce:CURRent:SLEW:POSitive?

可缩写成：

:SOUR:CURR:SLEW:POS?

2.5 特别说明

1、 在设置电流量程时，<value> 大于 5 时，电流量程设置为 30A；小于等于 5 时，电流量程设置为 5A。

2、 在设置电压量程时，<value> 大于 36 时，电压量程设置为 150V；小于等于 36 时，电压量程设置为 36V。

3、 在参数为{<value> | MINimum | MAXimum | DEFault}的枚举型选择中，各选项的意义如下：

<value> ：任意设定值
MINimum ：此设置值限定的最小值
MAXimum ：此设置值限定的最大值
DEFault ：此设置值限定的默认值

3. 命令系统

3.1 IEEE 公用命令子系统

命令格式：*IDN?
功能描述：获取设备信息字符串（返回串内容包括：厂商,设备型号,设备串口号,软件版本号）
举　　例：*IDN?
响应返回：Siglent\sTechnologies,SDL1020X,0123456789,1.01.01.15

命令格式：*RST
功能描述：重新恢复设备的状态为初始状态（即恢复默认设置）
举　　例：*RST

命令格式：*CLS
功能描述：将所有事件寄存器的值清零，同时清空错误列表
举　　例：*CLS

命令格式：*ESE < number >
功能描述：设置标准事件状态寄存器的使能值
举　　例：*ESE 16

命令格式：*ESE?
功能描述：查询标准事件状态寄存器的使能值
举　　例：*ESE?
响应返回：64

命令格式：*ESR?
功能描述：询问及清除标准事件状态寄存器的事件值
举　　例：*ESR?
响应返回：0

命令格式：*OPC
功能描述：所有操作结束后，在标准事件状态寄存器中设置比特位 bit0 置 1
举　　例：*OPC

命令格式：*OPC?
功能描述：查询当前操作是否完成
举　　例：*OPC?
响应返回：1

命令格式：*SRE <number>
功能描述：设置状态字节寄存器的使能值
举　　例：*SRE 24
命令格式：*SRE?
功能描述：查询状态字节寄存器的使能值
举　　例：*SRE?
响应返回：24

命令格式：*STB?
功能描述：查询状态字节寄存器的事件值
举　　例：*STB?
响应返回：72

命令格式：*TST?
功能描述：查询仪器自检结果
举　　例：*TST?
响应返回：0

命令格式：*WAI
功能描述：等待所有未完成操作完成之后，再执行任何其他命令
举　　例：*WAI

3.2　Measure 命令子系统

命令格式：MEASure:VOLTage[:DC]?
功能描述：获取实时电压测量值
举　　例：MEASure:VOLTage:DC?
响应返回：7.924678

命令格式：MEASure:CURRent[:DC]?
功能描述：获取实时电流测量值
举　　例：MEASure: CURRent:DC?
响应返回：3.986634

命令格式：MEASure:POWer[:DC]?
功能描述：获取实时功率测量值
举　　例：MEASure:POWer:DC?
响应返回：31.867329

命令格式：MEASure:RESistance[:DC]?

功能描述：获取实时电阻测量值
举　　例：MEASure: RESistance:DC?
响应返回：5.842789

命令格式：MEASure:EXT?
功能描述：获取（外部拉载模式下）实时外部测量值
举　　例：MEASure: EXT?
响应返回：3.863521

命令格式：MEASure:WAVEdata? {CURRent | VOLTage | POWer | RESistance}
功能描述：获取趋势图窗口显示的（CC/CV/CP/CR）波形数据（200 个 float 型数据点字符）
举　　例：MEASure:WAVEdata? VOLTage
响应返回：3.947389,3.928473,3.197493,3.573992…

3.3 Source 命令子系统

3.3.1 Source 共用命令子系统

命令格式：[:SOURce]:INPut[:STATe] {ON | OFF | 0 | 1}
功能描述：设置负载的输入状态（打开或关闭）
举　　例：:SOURce:INPut:STATe OFF

命令格式：[:SOURce]:INPut[:STATe]?
功能描述：查询负载的输入状态（打开：1 / 关闭：0）
举　　例：:SOURce:INPut:STATe?
响应返回：0

命令格式：[:SOURce]:SHORt[:STATe] {ON | OFF | 0 | 1}
功能描述：设置负载当前模式的短路状态（打开或关闭）
举　　例：:SOURce:SHORt:STATe ON

命令格式：[:SOURce]:SHORt[:STATe]?
功能描述：查询负载当前模式下的短路状态（打开：1 / 关闭：0）
举　　例：:SOURce:SHORt:STATe?
响应返回：1

命令格式：[:SOURce]:FUNCtion:TRANsient {CURRent | VOLTage | POWer | RESistance}
功能描述：设置动态操作下的模式（CC/CV/CP/CR）

举　　例：:SOURce:FUNCtion:TRANsient VOLTage

命令格式：[:SOURce]:FUNCtion:TRANsient?
功能描述：查询动态操作下的模式
举　　例：:SOURce:FUNCtion:TRANsient?
响应返回：VOLTAGE

命令格式：[:SOURce]:FUNCtion {CURRent | VOLTage | POWer | RESistance | LED}
功能描述：设置静态操作下的模式（CC/CV/CP/CR/LED）
举　　例：:SOURce:FUNCtion LED

命令格式：[:SOURce]:FUNCtion?
功能描述：查询静态操作下的模式
举　　例：:SOURce:FUNCtion?
响应返回：RESISTANCE

命令格式：[:SOURce]:FUNCtion:MODE?
功能描述：查询当前的操作模式
举　　例：:SOURce:FUNCtion:MODE?
响应返回：PROGRAM

命令格式：[:SOURce]:TEST:STEP?
功能描述：查询（LIST/PROGRAM）测试序列运行到的步数
举　　例：:SOURce:TEST:STEP?
响应返回：3

命令格式：[:SOURce]:TEST:STOP?
功能描述：查询测试序列当前运行到的步数是否停止（1：停止 / 0: 暂停）
举　　例：:SOURce:TEST:STOP?
响应返回：1

3.3.2 Source Current 命令子系统

命令格式：[:SOURce]:CURRent[:LEVel][:IMMediate] {<value> | MINimum | MAXimum | DEFault}
功能描述：设置静态操作下 CC 模式的调节电流值
举　　例：:SOURce:CURRent:LEVel:IMMediate 0.845

命令格式：[:SOURce]:CURRent[:LEVel][:IMMediate]?

功能描述：查询静态操作下 CC 模式的调节电流值
举　　例：:SOURce:CURRent:LEVel:IMMediate?
响应返回：2.674

命令格式：[:SOURce]:CURRent:IRANGe <value>
功能描述：设置静态操作下 CC 模式的电流量程
举　　例：:SOURce:CURRent:IRANGe 10

命令格式：[:SOURce]:CURRent:IRANGe?
功能描述：查询静态操作下 CC 模式的电流量程范围
举　　例：:SOURce:CURRent:IRANGe?
响应返回：30

命令格式：[:SOURce]:CURRent:VRANGe <value>
功能描述：设置静态操作下 CC 模式的电压量程
举　　例：:SOURce:CURRent:VRANGe 65

命令格式：[:SOURce]:CURRent:VRANGe?
功能描述：查询静态操作下 CC 模式的电压量程范围
举　　例：:SOURce:CURRent:VRANGe?
响应返回：36

命令格式：[:SOURce]:CURRent:SLEW[:BOTH] {<value> | MINimum | MAXimum | DEFault}
功能描述：设置静态操作下 CC 模式的斜率（上升斜率与下降斜率同步设置）
举　　例：:SOURce:CURRent:SLEW:BOTH 0.652

命令格式：[:SOURce]:CURRent:SLEW:POSitive {<value> | MINimum | MAXimum | DEFault}
功能描述：设置静态操作下 CC 模式的上升斜率
举　　例：:SOURce:CURRent:SLEW:POSitive 0.258

命令格式：[:SOURce]:CURRent:SLEW:POSitive?
功能描述：查询静态操作下 CC 模式的上升斜率
举　　例：:SOURce:CURRent:SLEW:POSitive?
响应返回：0.498

命令格式：[:SOURce]:CURRent:SLEW:NEGative {<value> | MINimum | MAXimum | DEFault}
功能描述：设置静态操作下 CC 模式的下降斜率
举　　例：:SOURce:CURRent:SLEW: NEGative 1.986

命令格式：[:SOURce]:CURRent:SLEW:NEGative?
功能描述：查询静态操作下 CC 模式的下降斜率
举　　例：:SOURce:CURRent:SLEW: NEGative?
响应返回：0.187

命令格式：[:SOURce]:CURRent:TRANsient:MODE {CONTinuous | PULSe | TOGGle}
功能描述：设置动态操作下 CC 模式的波形模式
举　　例：:SOURce:CURRent:TRANsient:MODE PULSe

命令格式：[:SOURce]:CURRent:TRANsient:MODE?
功能描述：查询动态操作下 CC 模式的波形模式
举　　例：:SOURce:CURRent:TRANsient:MODE?
响应返回：CONTINUOUS

命令格式：[:SOURce]:CURRent:TRANsient:IRANGe <value>
功能描述：设置动态操作下 CC 模式的电流量程
举　　例：:SOURce:CURRent:TRANsient:IRANGe 9

命令格式：[:SOURce]:CURRent:TRANsient:IRANGe?
功能描述：查询动态操作下 CC 模式的电流量程范围
举　　例：:SOURce:CURRent:TRANsient:IRANGe?
响应返回：30

命令格式：[:SOURce]:CURRent:TRANsient:VRANGe <value>
功能描述：设置动态操作下 CC 模式的电压量程
举　　例：:SOURce:CURRent:TRANsient:VRANGe 80

命令格式：[:SOURce]:CURRent:TRANsient:VRANGe?
功能描述：查询动态操作下 CC 模式的电压量程范围
举　　例：:SOURce:CURRent:TRANsient:VRANGe?
响应返回：150
命令格式：[:SOURce]:CURRent:TRANsient:ALEVel {<value> | MINimum | MAXimum | DEFault}
功能描述：设置动态操作下 CC 模式的 A 值
举　　例：:SOURce:CURRent:TRANsient:ALEVel 4.653

命令格式：[:SOURce]:CURRent:TRANsient:ALEVel?
功能描述：查询动态操作下 CC 模式的 A 值
举　　例：:SOURce:CURRent:TRANsient:ALEVel?

响应返回：6.000

命令格式：[:SOURce]:CURRent:TRANsient:BLEVel {<value> | MINimum | MAXimum | DEFault}
功能描述：设置动态操作下 CC 模式的 B 值
举　　例：:SOURce:CURRent:TRANsient:BLEVel 5.000

命令格式：[:SOURce]:CURRent:TRANsient:BLEVel?
功能描述：查询动态操作下 CC 模式的 B 值
举　　例：:SOURce:CURRent:TRANsient:BLEVel?
响应返回：8.000

命令格式：[:SOURce]:CURRent:TRANsient:AWIDth {<value> | MINimum | MAXimum | DEFault}
功能描述：设置动态操作下 CC 模式的 A 值的脉宽时间值（“s”为单位）
举　　例：:SOURce:CURRent:TRANsient:AWIDth 0.700

命令格式：[:SOURce]:CURRent:TRANsient:AWIDth?
功能描述：查询动态操作下 CC 模式的 A 值的脉宽时间值（“s”为单位）
举　　例：:SOURce:CURRent:TRANsient:AWIDth?
响应返回：1.000

命令格式：[:SOURce]:CURRent:TRANsient:BWIDth {<value> | MINimum | MAXimum | DEFault}
功能描述：设置动态操作下 CC 模式的 B 值的脉宽时间值（“s”为单位）
举　　例：:SOURce:CURRent:TRANsient:BWIDth 0.800

命令格式：[:SOURce]:CURRent:TRANsient:BWIDth?
功能描述：查询动态操作下 CC 模式的 B 值的脉宽时间值（“s”为单位）
举　　例：:SOURce:CURRent:TRANsient:BWIDth?
响应返回：1.500

命令格式：[:SOURce]:CURRent:TRANsient:SLEW:POSitive {<value> | MINimum | MAXimum | DEFault}
功能描述：设置动态操作下 CC 模式的上升斜率
举　　例：:SOURce:CURRent:TRANsient: SLEW:POSitive 0.400
命令格式：[:SOURce]:CURRent:TRANsient:SLEW:POSitive?
功能描述：查询动态操作下 CC 模式的上升斜率
举　　例：:SOURce:CURRent:TRANsient: SLEW:POSitive?
响应返回：0.600

命令格式：[:SOURce]:CURRent:TRANsient:SLEW:NEGative {<value> | MINimum | MAXimum | DEFault}
功能描述：设置动态操作下 CC 模式的下降斜率
举　　例：:SOURce:CURRent:TRANsient: SLEW: NEGative 0.300

命令格式：[:SOURce]:CURRent:TRANsient:SLEW:NEGative?
功能描述：查询动态操作下 CC 模式的下降斜率
举　　例：:SOURce:CURRent:TRANsient: SLEW: NEGative?
响应返回：0.900

3.3.3 Source Voltage 命令子系统

命令格式：[:SOURce]:VOLTage[:LEVel][:IMMediate] {<value> | MINimum | MAXimum | DEFault}
功能描述：设置静态操作下 CV 模式的调节电压值
举　　例：:SOURce:VOLTage:LEVel:IMMediate 3.000

命令格式：[:SOURce]:VOLTage[:LEVel][:IMMediate]?
功能描述：查询静态操作下 CV 模式的调节电压值
举　　例：:SOURce:VOLTage:LEVel:IMMediate?
响应返回：4.000

命令格式：[:SOURce]:VOLTage:IRANGe <value>
功能描述：设置静态操作下 CV 模式的电流量程
举　　例：:SOURce:VOLTage:IRANGe 10

命令格式：[:SOURce]:VOLTage:IRANGe?
功能描述：查询静态操作下 CV 模式的电流量程范围
举　　例：:SOURce:VOLTage:IRANGe?
响应返回：30
命令格式：[:SOURce]:VOLTage:VRANGe <value>
功能描述：设置静态操作下 CV 模式的电压量程
举　　例：:SOURce:VOLTage:VRANGe 60

命令格式：[:SOURce]:VOLTage:VRANGe?
功能描述：查询静态操作下 CV 模式的电压量程范围
举　　例：:SOURce:VOLTage:VRANGe?
响应返回：36

命令格式：[:SOURce]:VOLTage:TRANsient:MODE {CONTinuous | PULSe | TOGGle}

功能描述：设置动态操作下 CV 模式的波形模式
举　　例：:SOURce:VOLTage:TRANsient:MODE TOGGle

命令格式：[:SOURce]:VOLTage:TRANsient:MODE?
功能描述：查询动态操作下 CV 模式的波形模式
举　　例：:SOURce:VOLTage:TRANsient:MODE?
响应返回：PULSe

命令格式：[:SOURce]:VOLTage:TRANsient:IRANGe <value>
功能描述：设置动态操作下 CV 模式的电流量程
举　　例：:SOURce:VOLTage:TRANsient:IRANGe 5

命令格式：[:SOURce]:VOLTage:TRANsient:IRANGe?
功能描述：查询动态操作下 CV 模式的电流量程范围
举　　例：:SOURce:VOLTage:TRANsient:IRANGe?
响应返回：30

命令格式：[:SOURce]:VOLTage:TRANsient:VRANGe <value>
功能描述：设置动态操作下 CV 模式的电压量程
举　　例：:SOURce:VOLTage:TRANsient:VRANGe 36

命令格式：[:SOURce]:VOLTage:TRANsient:VRANGe?
功能描述：查询动态操作下 CV 模式的电压量程范围
举　　例：:SOURce:VOLTage:TRANsient:VRANGe?
响应返回：150

命令格式：[:SOURce]:VOLTage:TRANsient:ALEVel {<value> | MINimum | MAXimum | DEFault}
功能描述：设置动态操作下 CV 模式的 A 值
举　　例：:SOURce:VOLTage:TRANsient:ALEVel MINimum

命令格式：[:SOURce]:VOLTage:TRANsient:ALEVel?
功能描述：查询动态操作下 CV 模式的 A 值
举　　例：:SOURce:VOLTage:TRANsient:ALEVel?
响应返回：8.000

命令格式：[:SOURce]:VOLTage:TRANsient:BLEVel {<value> | MINimum | MAXimum | DEFault}
功能描述：设置动态操作下 CV 模式的 B 值
举　　例：:SOURce:VOLTage:TRANsient:BLEVel MAXimum

命令格式：[:SOURce]:VOLTage:TRANsient:BLEVel?
功能描述：查询动态操作下 CV 模式的 B 值
举　　例：:SOURce:VOLTage:TRANsient:BLEVel?
响应返回：10.000

命令格式：[:SOURce]:VOLTage:TRANsient:AWIDth {<value> | MINimum | MAXimum | DEFault}
功能描述：设置动态操作下 CV 模式的 A 值的脉宽时间值（“s”为单位）
举　　例：:SOURce:VOLTage:TRANsient:AWIDth DEFault

命令格式：[:SOURce]:VOLTage:TRANsient:AWIDth?
功能描述：查询动态操作下 CV 模式的 A 值的脉宽时间值（“s”为单位）
举　　例：:SOURce:VOLTage:TRANsient:AWIDth?
响应返回：2.000

命令格式：[:SOURce]:VOLTage:TRANsient:BWIDth {<value> | MINimum | MAXimum | DEFault}
功能描述：设置动态操作下 CV 模式的 B 值的脉宽时间值（“s”为单位）
举　　例：:SOURce:VOLTage:TRANsient:BWIDth 0.800

命令格式：[:SOURce]:VOLTage:TRANsient:BWIDth?
功能描述：查询动态操作下 CV 模式的 B 值的脉宽时间值（“s”为单位）
举　　例：:SOURce:VOLTage:TRANsient:BWIDth?
响应返回：1.500

3.3.4 Source Power 命令子系统

命令格式：[:SOURce]:POWer[:LEVel][:IMMediate] {<value> | MINimum | MAXimum | DEFault}
功能描述：设置静态操作下 CP 模式的调节功率值
举　　例：:SOURce:POWer:LEVel:IMMediate 3.000

命令格式：[:SOURce]:POWer[:LEVel][:IMMediate]?
功能描述：查询静态操作下 CP 模式的调节功率值
举　　例：:SOURce:POWer:LEVel:IMMediate?
响应返回：4.000

命令格式：[:SOURce]:POWer:IRANGe <value>
功能描述：设置静态操作下 CP 模式的电流量程
举　　例：:SOURce:POWer:IRANGe 10

命令格式：[:SOURce]:POWer:IRANGe?
功能描述：查询静态操作下 CP 模式的电流量程范围
举　　例：:SOURce:POWer:IRANGe?
响应返回：30

命令格式：[:SOURce]:POWer:VRANGe <value>
功能描述：设置静态操作下 CP 模式的电压量程
举　　例：:SOURce:POWer:VRANGe 60

命令格式：[:SOURce]:POWer:VRANGe?
功能描述：查询静态操作下 CP 模式的电压量程范围
举　　例：:SOURce:POWer:VRANGe?
响应返回：36

命令格式：[:SOURce]:POWer:TRANsient:MODE {CONTinuous | PULSe | TOGGle}
功能描述：设置动态操作下 CP 模式的波形模式
举　　例：:SOURce:POWer:TRANsient:MODE TOGGle

命令格式：[:SOURce]:POWer:TRANsient:MODE?
功能描述：查询动态操作下 CP 模式的波形模式
举　　例：:SOURce:POWer:TRANsient:MODE?
响应返回：PULSe

命令格式：[:SOURce]:POWer:TRANsient:IRANGe <value>
功能描述：设置动态操作下 CP 模式的电流量程
举　　例：:SOURce:POWer:TRANsient:IRANGe 5

命令格式：[:SOURce]:POWer:TRANsient:IRANGe?
功能描述：查询动态操作下 CP 模式的电流量程范围
举　　例：:SOURce:POWer:TRANsient:IRANGe?
响应返回：30

命令格式：[:SOURce]:POWer:TRANsient:VRANGe <value>
功能描述：设置动态操作下 CP 模式的电压量程
举　　例：:SOURce:POWer:TRANsient:VRANGe 36

命令格式：[:SOURce]:POWer:TRANsient:VRANGe?
功能描述：查询动态操作下 CP 模式的电压量程范围
举　　例：:SOURce:POWer:TRANsient:VRANGe?
响应返回：150

命令格式：[:SOURce]:POWer:TRANsient:ALEVel {<value> | MINimum | MAXimum | DEFault}
功能描述：设置动态操作下 CP 模式的 A 值
举　　例：:SOURce:POWer:TRANsient:ALEVel MINimum

命令格式：[:SOURce]:POWer:TRANsient:ALEVel?
功能描述：查询动态操作下 CP 模式的 A 值
举　　例：:SOURce:POWer:TRANsient:ALEVel?
响应返回：8.000

命令格式：[:SOURce]:POWer:TRANsient:BLEVel {<value> | MINimum | MAXimum | DEFault}
功能描述：设置动态操作下 CP 模式的 B 值
举　　例：:SOURce:POWer:TRANsient:BLEVel MAXimum

命令格式：[:SOURce]:POWer:TRANsient:BLEVel?
功能描述：查询动态操作下 CP 模式的 B 值
举　　例：:SOURce:POWer:TRANsient:BLEVel?
响应返回：10.000

命令格式：[:SOURce]:POWer:TRANsient:AWIDth {<value> | MINimum | MAXimum | DEFault}
功能描述：设置动态操作下 CP 模式的 A 值的脉宽时间值（“s”为单位）
举　　例：:SOURce:POWer:TRANsient:AWIDth DEFault

命令格式：[:SOURce]:POWer:TRANsient:AWIDth?
功能描述：查询动态操作下 CP 模式的 A 值的脉宽时间值（“s”为单位）
举　　例：:SOURce:POWer:TRANsient:AWIDth?
响应返回：2.000

命令格式：[:SOURce]:POWer:TRANsient:BWIDth {<value> | MINimum | MAXimum | DEFault}
功能描述：设置动态操作下 CP 模式的 B 值的脉宽时间值（“s”为单位）
举　　例：:SOURce:POWer:TRANsient:BWIDth 0.800

命令格式：[:SOURce]:POWer:TRANsient:BWIDth?
功能描述：查询动态操作下 CP 模式的 B 值的脉宽时间值（“s”为单位）
举　　例：:SOURce:POWer:TRANsient:BWIDth?
响应返回：1.500

3.3.5 Source Resistance 命令子系统

命令格式：[:SOURce]:RESistance[:LEVel][:IMMediate] {<value> | MINimum | MAXimum | DEFault}
功能描述：设置静态操作下 CR 模式的调节电阻值
举　　例：:SOURce:RESistance:LEVel:IMMediate 3.000

命令格式：[:SOURce]:RESistance[:LEVel][:IMMediate]?
功能描述：查询静态操作下 CR 模式的调节电阻值
举　　例：:SOURce:RESistance:LEVel:IMMediate?
响应返回：4.000

命令格式：[:SOURce]:RESistance:IRANGe <value>
功能描述：设置静态操作下 CR 模式的电流量程
举　　例：:SOURce:RESistance:IRANGe 10

命令格式：[:SOURce]:RESistance:IRANGe?
功能描述：查询静态操作下 CR 模式的电流量程范围
举　　例：:SOURce:RESistance:IRANGe?
响应返回：30

命令格式：[:SOURce]:RESistance:VRANGe <value>
功能描述：设置静态操作下 CR 模式的电压量程
举　　例：:SOURce:RESistance:VRANGe 60

命令格式：[:SOURce]:RESistance:VRANGe?
功能描述：查询静态操作下 CR 模式的电压量程范围
举　　例：:SOURce:RESistance:VRANGe?
响应返回：36

命令格式：[:SOURce]:RESistance:RRANGe {LOW | MIDDLE | HIGH | UPPER}
功能描述：设置静态操作下 CR 模式的电阻量程
举　　例：:SOURce:RESistance:RRANGe MIDDLE

命令格式：[:SOURce]:RESistance:RRANGe?
功能描述：查询静态操作下 CR 模式的电阻量程范围
举　　例：:SOURce:RESistance:RRANGe?
响应返回：UPPER

命令格式：[:SOURce]:RESistance:TRANsient:MODE {CONTinuous | PULSe |

TOGGle}
功能描述：设置动态操作下 CR 模式的波形模式
举　　例：:SOURce:RESistance:TRANsient:MODE TOGGle

命令格式：[:SOURce]:RESistance:TRANsient:MODE?
功能描述：查询动态操作下 CR 模式的波形模式
举　　例：:SOURce:RESistance:TRANsient:MODE?
响应返回：PULSe

命令格式：[:SOURce]:RESistance:TRANsient:IRANGe <value>
功能描述：设置动态操作下 CR 模式的电流量程
举　　例：:SOURce:RESistance:TRANsient:IRANGe 5

命令格式：[:SOURce]:RESistance:TRANsient:IRANGe?
功能描述：查询动态操作下 CR 模式的电流量程范围
举　　例：:SOURce:RESistance:TRANsient:IRANGe?
响应返回：30

命令格式：[:SOURce]:RESistance:TRANsient:VRANGe <value>
功能描述：设置动态操作下 CR 模式的电压量程
举　　例：:SOURce:RESistance:TRANsient:VRANGe 36

命令格式：[:SOURce]:RESistance:TRANsient:VRANGe?
功能描述：查询动态操作下 CR 模式的电压量程范围
举　　例：:SOURce:RESistance:TRANsient:VRANGe?
响应返回：150

命令格式：[:SOURce]:RESistance:TRANsient:RRANGe {LOW | MIDDLE | HIGH | UPPER}
功能描述：设置动态操作下 CR 模式的电阻量程
举　　例：:SOURce:RESistance:TRANsient:VRANGe HIGH

命令格式：[:SOURce]:RESistance:TRANsient:RRANGe?
功能描述：查询动态操作下 CR 模式的电阻量程范围
举　　例：:SOURce:RESistance:TRANsient:VRANGe?
响应返回：LOW

命令格式：[:SOURce]:RESistance:TRANsient:ALEVel {<value> | MINimum | MAXimum | DEFault}
功能描述：设置动态操作下 CR 模式的 A 值

举　　例：:SOURce:RESistance:TRANsient:ALEVel MINimum

命令格式：[:SOURce]:RESistance:TRANsient:ALEVel?
功能描述：查询动态操作下 CR 模式的 A 值
举　　例：:SOURce:RESistance:TRANsient:ALEVel?
响应返回：8.000

命令格式：[:SOURce]:RESistance:TRANsient:BLEVel {< value > | MINimum | MAXimum | DEFault}
功能描述：设置动态操作下 CR 模式的 B 值
举　　例：:SOURce:RESistance:TRANsient:BLEVel MAXimum

命令格式：[:SOURce]:RESistance:TRANsient:BLEVel?
功能描述：查询动态操作下 CR 模式的 B 值
举　　例：:SOURce:RESistance:TRANsient:BLEVel?
响应返回：10.000

命令格式：[:SOURce]:RESistance:TRANsient:AWIDth {<value> | MINimum | MAXimum | DEFault}
功能描述：设置动态操作下 CR 模式的 A 值的脉宽时间值（“s”为单位）
举　　例：:SOURce:RESistance:TRANsient:AWIDth DEFault

命令格式：[:SOURce]:RESistance:TRANsient:AWIDth?
功能描述：查询动态操作下 CR 模式的 A 值的脉宽时间值（“s”为单位）
举　　例：:SOURce:RESistance:TRANsient:AWIDth?
响应返回：2.000

命令格式：[:SOURce]:RESistance:TRANsient:BWIDth {< value > | MINimum | MAXimum | DEFault}
功能描述：设置动态操作下 CR 模式的 B 值的脉宽时间值（“s”为单位）
举　　例：:SOURce:RESistance:TRANsient:BWIDth 0.800

命令格式：[:SOURce]:RESistance:TRANsient:BWIDth?
功能描述：查询动态操作下 CR 模式的 B 值的脉宽时间值（“s”为单位）
举　　例：:SOURce:RESistance:TRANsient:BWIDth?
响应返回：1.500

3.3.6 Source LED 命令子系统

命令格式：[:SOURce]:LED:IRANGe <value>
功能描述：设置静态操作下 LED 的电流量程
举　　例：:SOURce:LED:IRANGe 12

命令格式：[:SOURce]:LED:IRANGe?
功能描述：查询静态操作下 LED 的电流量程范围
举　　例：:SOURce:LED:IRANGe?
响应返回：30

命令格式：[:SOURce]:LED:VRANGe <value>
功能描述：设置静态操作下 LED 的电压量程
举　　例：:SOURce:LED:VRANGe 40

命令格式：[:SOURce]:LED:VRANGe?
功能描述：查询静态操作下 LED 的电压量程范围
举　　例：:SOURce:LED:VRANGe?
响应返回：150

命令格式：[:SOURce]:LED:VOLTage {< value > | MINimum | MAXimum | DEFault}
功能描述：设置静态操作下 LED 的电压 Vo
举　　例：:SOURce:LED: VOLTage 10

命令格式：[:SOURce]:LED:VOLTage?
功能描述：查询静态操作下 LED 的电压 Vo
举　　例：:SOURce:LED: VOLTage?
响应返回：60.000

命令格式：[:SOURce]:LED:CURRent {< value > | MINimum | MAXimum | DEFault}
功能描述：设置静态操作下 LED 的电压 Io
举　　例：:SOURce:LED: CURRent 3

命令格式：[:SOURce]:LED:CURRent?
功能描述：查询静态操作下 LED 的电压 Io
举　　例：:SOURce:LED: CURRent?
响应返回：2.000

命令格式：[:SOURce]:LED:RCOnf {< value > | MINimum | MAXimum | DEFault}
功能描述：设置静态操作下 LED 的 Rconf
举　　例：:SOURce:LED: CURRent DEFault

命令格式：[:SOURce]:LED:RCOnf?
功能描述：查询静态操作下 LED 的 Rconf
举　　例：:SOURce:LED: RCOnf?
响应返回：0.200

3.3.7 Source Battery 命令子系统

命令格式：[:SOURce]:BATTery:FUNC
功能描述：使负载设备进入 BATTERY 电池放电测试模式
举　　例：:SOURce:BATTery:FUNC

命令格式：[:SOURce]:BATTery:FUNC?
功能描述：查询负载设备当前是否处于 BATTERY 电池放电测试模式
举　　例：:SOURce:BATTery:FUNC?
响应返回：1

命令格式：[:SOURce]:BATTery:MODE {CURRent | POWer | RESistance}
功能描述：设置 BATTERY 下的模式（CC/CP/CR）
举　　例：:SOURce:BATTery:MODE POWer

命令格式：[:SOURce]:BATTery:MODE?
功能描述：查询 BATTERY 下的模式
举　　例：:SOURce:BATTery:MODE?
响应返回：CURRENT

命令格式：[:SOURce]:BATTery:IRANGe <value>
功能描述：设置当前 BATTERY 模式的电流量程
举　　例：:SOURce:BATTery:IRANGe 6.000

命令格式：[:SOURce]:BATTery:IRANGe?
功能描述：查询当前 BATTERY 模式的电流量程范围
举　　例：:SOURce:BATTery:IRANGe?
响应返回：30

命令格式：[:SOURce]:BATTery:VRANGe <value>
功能描述：设置当前 BATTERY 模式的电压量程
举　　例：:SOURce:BATTery:IRANGe 22.000

命令格式：[:SOURce]:BATTery:VRANGe?
功能描述：查询当前 BATTERY 模式的电压量程范围
举　　例：:SOURce:BATTery: VRANGe?
响应返回：150

命令格式：[:SOURce]:BATTery:RRANGe {LOW | MIDDLE | HIGH | UPPER}
功能描述：设置当前 BATTERY 模式的电阻量程
举　　例：:SOURce:BATTery:IRANGe UPPER

命令格式：[:SOURce]:BATTery:RRANGe?
功能描述：查询当前 BATTERY 模式的电阻量程范围
举　　例：:SOURce:BATTery:IRANGe?
响应返回：MIDDLE

命令格式：[:SOURce]:BATTery:LEVel <value>
功能描述：设置当前 BATTERY 模式下的放电值（CC 为放电电流值 / CR 为放电电阻值 / CP 为放电功率值）
举　　例：:SOURce:BATTery:LEVel 5.000

命令格式：[:SOURce]:BATTery:LEVel?
功能描述：查询当前 BATTERY 模式下的放电值（CC 为放电电流值 / CR 为放电电阻值 / CP 为放电功率值）
举　　例：:SOURce:BATTery:LEVel?
响应返回：8.000

命令格式：[:SOURce]:BATTery:VOLTage {< value > | MINimum | MAXimum | DEFault}
功能描述：设置当前 BATTERY 模式下的终止放电电压值
举　　例：:SOURce:BATTery:VOLTage 9.000

命令格式：[:SOURce]:BATTery:VOLTage?
功能描述：查询当前 BATTERY 模式下的终止放电电压值
举　　例：:SOURce:BATTery:VOLTage?
响应返回：20.000

命令格式：[:SOURce]:BATTery:CAPability < value >
功能描述：设置当前 BATTERY 模式下的终止放电电容值
举　　例：:SOURce:BATTery: CAPability 100.00

命令格式：[:SOURce]:BATTery:CAPability?

功能描述：查询当前 BATTERY 模式下的终止放电电容值
举　　例：:SOURce:BATTery: CAPability?
响应返回：199.000

命令格式：[:SOURce]:BATTery:TIMer {< value > | MINimum | MAXimum | DEFault}
功能描述：设置当前 BATTERY 模式下的终止放电时间值
举　　例：:SOURce:BATTery:TIMer DEFault

命令格式：[:SOURce]:BATTery:TIMer?
功能描述：查询当前 BATTERY 模式下的终止放电时间值
举　　例：:SOURce:BATTery:TIMer?
响应返回：999.0000

命令格式：[:SOURce]:BATTery:VOLTage:STATe {ON | OFF | 0 | 1}
功能描述：设置放电电压是否作为终止判断条件
举　　例：:SOURce:BATTery:VOLTage:STATe ON

命令格式：[:SOURce]:BATTery:VOLTage:STATe?
功能描述：查询放电电压是否作为终止判断条件
举　　例：:SOURce:BATTery:VOLTage:STATe?
响应返回：0

命令格式：[:SOURce]:BATTery:CAPability:STATe {ON | OFF | 0 | 1}
功能描述：设置放电电容量是否作为终止判断条件
举　　例：:SOURce:BATTery: CAPability:STATe OFF

命令格式：[:SOURce]:BATTery:CAPability:STATe?
功能描述：查询放电电容量是否作为终止判断条件
举　　例：:SOURce:BATTery: CAPability:STATe?
响应返回：1

命令格式：[:SOURce]:BATTery:TIMer:STATe{ON | OFF | 0 | 1}
功能描述：设置放电时间是否作为终止判断条件
举　　例：:SOURce:BATTery: TIMer:STATe ON

命令格式：[:SOURce]:BATTery:TIMer:STATe?
功能描述：查询放电时间是否作为终止判断条件
举　　例：:SOURce:BATTery: TIMer:STATe?
响应返回：1

命令格式：[:SOURce]:BATTery:DISCHArg:CAPability?
功能描述：获取电池放电测试开始后的放电电容量（以“mAh”为单位）
举　　例：:SOURce:BATTery:DISCHArg:CAPability?
响应返回：13

命令格式：[:SOURce]:BATTery:DISCHArg:TIMer?
功能描述：获取电池放电测试开始后的放电时间（以“s”为单位）
举　　例：:SOURce:BATTery:DISCHArg: TIMer?
响应返回：162

3.3.8 Source List 命令子系统

命令格式：[:SOURce]:LIST:MODE {CURRent | VOLTage | POWer | RESistance}
功能描述：设置 LIST 测试下的运行模式
举　　例：:SOURce:LIST:MODE VOLTage

命令格式：[:SOURce]:LIST:MODE?
功能描述：查询当前 LIST 测试下运行模式
举　　例：:SOURce:LIST:MODE?
响应返回：CURRENT

命令格式：[:SOURce]:LIST:IRANGe <value>
功能描述：设置当前 LIST 测试下运行模式的电流量程
举　　例：:SOURce:LIST:IRANGe 5

命令格式：[:SOURce]:LIST:IRANGe?
功能描述：查询当前 LIST 测试下运行模式的电流量程范围
举　　例：:SOURce:LIST:IRANGe?
响应返回：30

命令格式：[:SOURce]:LIST:VRANGe <value>
功能描述：设置当前 LIST 测试下运行模式的电压量程
举　　例：SOURce:LIST:VRANGe 150

命令格式：[:SOURce]:LIST:VRANGe?
功能描述：查询当前 LIST 测试下运行模式的电压量程范围
举　　例：:SOURce:LIST:VRANGe?
响应返回：36

命令格式：[:SOURce]:LIST:RRANGe {LOW | MIDDLE | HIGH |UPPER}

功能描述：设置当前 LIST 测试下运行模式的电阻量程
举　　例：SOURce:LIST:RRANGe HIGH

命令格式：[:SOURce]:LIST:RRANGe?
功能描述：查询当前 LIST 测试下运行模式的电阻量程范围
举　　例：:SOURce:LIST:RRANGe?
响应返回：UPPER

命令格式：[:SOURce]:LIST:COUNt {< number > | MINimum | MAXimum | DEFault}
功能描述：设置当前 LIST 测试下运行模式的循环执行次数
举　　例：:SOURce:LIST:COUNt 10

命令格式：[:SOURce]:LIST:COUNt?
功能描述：查询当前 LIST 测试下运行模式的循环执行次数
举　　例：:SOURce:LIST:COUNt?
响应返回：255

命令格式：[:SOURce]:LIST:STEP {< number > | MINimum | MAXimum | DEFault}
功能描述：设置当前 LIST 测试下运行模式的有效执行步数
举　　例：:SOURce:LIST: STEP 5

命令格式：[:SOURce]:LIST:STEP?
功能描述：查询当前 LIST 测试下运行模式的有效执行步数
举　　例：:SOURce:LIST:STEP?
响应返回：15

命令格式：[:SOURce]:LIST:LEVel <step,value>
功能描述：设置 LIST 序列表中第 step 步的设置值（CC 为电流值 / CV 为电压值 / CP 为功率值 / CR 为电阻值）
举　　例：:SOURce:LIST:LEVel 3,4.500

命令格式：[:SOURce]:LIST:LEVel? <step>
功能描述：查询 LIST 序列表中第 step 步的设置值（CC 为电流值 / CV 为电压值 / CP 为功率值 / CR 为电阻值）
举　　例：[:SOURce]:LIST:LEVel? 5
响应返回：2.000

命令格式：[:SOURce]:LIST:SLEW[:BOTH] <step,value>
功能描述：设置 LIST 序列表中第 step 步的斜率（CC 下有效）
举　　例：:SOURce:LIST:SLEW:BOTH 4,0.500

命令格式：[:SOURce]:LIST:SLEW[:BOTH]? <step>
功能描述：查询 LIST 序列表中第 step 步的斜率（CC 下有效）
举　　例：:SOURce:LIST:SLEW:BOTH? 5
响应返回：0.100

命令格式：[:SOURce]:LIST:WIDth <step,value>
功能描述：设置 LIST 序列表中第 step 步的持续时间
举　　例：:SOURce:LIST:WIDth 3,1.000

命令格式：[:SOURce]:LIST:WIDth? <step>
功能描述：查询 LIST 序列表中第 step 步的持续时间
举　　例：:SOURce:LIST:WIDth? 5
响应返回：0.100

命令格式：[:SOURce]:LIST:STATe:ON
功能描述：使负载设备进入 LIST 测试模式
举　　例：:SOURce:LIST:STATe:ON

命令格式：[:SOURce]:LIST:STATe?
功能描述：查询负载设备当前是否在 LIST 测试模式
举　　例：:SOURce:LIST:STATe?
响应返回：1

3.3.9 Source OCPT 命令子系统

命令格式：[:SOURce]: OCP:FUNC
功能描述：使负载设备进入 OCPT 过流保护测试模式
举　　例：:SOURce: OCP:FUNC

命令格式：[:SOURce]: OCP:FUNC?
功能描述：查询负载设备当前是否处于 OCPT 过流保护测试模式
举　　例：:SOURce: OCP:FUNC?
响应返回：0

命令格式：[:SOURce]:OCP:IRANGe <value>
功能描述：设置 OCPT 模式下的电流量程
举　　例：:SOURce:OCP:IRANGe 30

命令格式：[:SOURce]:OCP:IRANGe?
功能描述：查询 OCPT 模式下的电流量程范围

举　　例：:SOURce:OCP:IRANGe?
响应返回：5

命令格式：[:SOURce]:OCP:VRANGe <value>
功能描述：设置 OCPT 模式下的电压量程
举　　例：:SOURce:OCP:VRANGe 36

命令格式：[:SOURce]:OCP:VRANGe?
功能描述：查询 OCPT 模式下的电压量程范围
举　　例：:SOURce:OCP:VRANGe?
响应返回：150

命令格式：[:SOURce]:OCP:STARt {< value > | MINimum | MAXimum | DEFault}
功能描述：设置 OCPT 模式下的初始电流值
举　　例：:SOURce:OCP:STARt 1.000

命令格式：[:SOURce]:OCP:STARt?
功能描述：查询 OCPT 模式下的初始电流值
举　　例：:SOURce:OCP:STARt?
响应返回：1.000

命令格式：[:SOURce]:OCP:STEP {< value > | MINimum | MAXimum | DEFault}
功能描述：设置 OCPT 模式下的电流步进增值
举　　例：:SOURce:OCP: STEP 0.500

命令格式：[:SOURce]:OCP:STEP?
功能描述：查询 OCPT 模式下的电流步进增值
举　　例：:SOURce:OCP: STEP?
响应返回：1.000
命令格式：[:SOURce]:OCP:STEP:DELay {< value > | MINimum | MAXimum | DEFault}
功能描述：设置 OCPT 模式下的步进延迟时间
举　　例：:SOURce:OCP:STEP:DELay 1.000

命令格式：[:SOURce]:OCP:STEP:DELay?
功能描述：查询 OCPT 模式下的步进延迟时间
举　　例：:SOURce:OCP:STEP:DELay?
响应返回：2.000

命令格式：[:SOURce]:OCP:END {< value > | MINimum | MAXimum | DEFault}
功能描述：设置 OCPT 模式下的终止电流值

举　　例：:SOURce:OCP: END 5.000

命令格式：[:SOURce]:OCP:END?
功能描述：查询 OCPT 模式下的终止电流值
举　　例：:SOURce:OCP: END?
响应返回：4.000

命令格式：[:SOURce]:OCP:MIN {< value > | MINimum | MAXimum | DEFault}
功能描述：设置 OCPT 模式下的电流下限判断值
举　　例：:SOURce:OCP: MIN 2.000

命令格式：[:SOURce]:OCP:MIN?
功能描述：查询 OCPT 模式下的电流下限判断值
举　　例：:SOURce:OCP: MIN?
响应返回：3.000

命令格式：[:SOURce]:OCP:MAX {< value > | MINimum | MAXimum | DEFault}
功能描述：设置 OCPT 模式下的电流上限判断值
举　　例：:SOURce:OCP: MAX 10.000

命令格式：[:SOURce]:OCP:MAX?
功能描述：查询 OCPT 模式下的电流上限判断值
举　　例：:SOURce:OCP: MAX?
响应返回：8.000

命令格式：[:SOURce]:OCP:VOLTage {< value > | MINimum | MAXimum | DEFault}
功能描述：设置 OCPT 模式下的保护电压值
举　　例：:SOURce:OCP: VOLTage 8.000

命令格式：[:SOURce]:OCP:VOLTage?
功能描述：查询 OCPT 模式下的保护电压值
举　　例：:SOURce:OCP: VOLTage?
响应返回：10.000

3.3.10 Source OPPT 命令子系统

命令格式：[:SOURce]: OPP:FUNC
功能描述：使负载设备进入 OPP T 过功率保护测试模式
举　　例：:SOURce: OPP:FUNC

命令格式：[:SOURce]: OPP:FUNC?
功能描述：查询负载设备当前是否处于 OPP T 过功率保护测试模式
举　　例：:SOURce: OPP:FUNC?
响应返回：1

命令格式：[:SOURce]:OPP:IRANGe <value>
功能描述：设置 OPPT 模式下的电流量程
举　　例：:SOURce:OPP:IRANGe 30

命令格式：[:SOURce]:OPP:IRANGe?
功能描述：查询 OPPT 模式下的电流量程范围
举　　例：:SOURce:OPP:IRANGe?
响应返回：5

命令格式：[:SOURce]:OPP:VRANGe <value>
功能描述：设置 OPPT 模式下的电压量程
举　　例：:SOURce:OPP:VRANGe 36

命令格式：[:SOURce]:OPP:VRANGe?
功能描述：查询 OPPT 模式下的电压量程范围
举　　例：:SOURce:OPP:VRANGe?
响应返回：150

命令格式：[:SOURce]:OPP:STARt {< value > | MINimum | MAXimum | DEFault}
功能描述：设置 OPPT 模式下的初始功率值
举　　例：:SOURce:OPP:STARt 1.000

命令格式：[:SOURce]:OPP:STARt?
功能描述：查询 OPPT 模式下的初始功率值
举　　例：:SOURce:OPP:STARt?
响应返回：1.000

命令格式：[:SOURce]:OPP:STEP {< value > | MINimum | MAXimum | DEFault}
功能描述：设置 OPPT 模式下的功率步进增值
举　　例：:SOURce:OPP: STEP 0.500

命令格式：[:SOURce]:OPP:STEP?
功能描述：查询 OPPT 模式下的功率步进增值
举　　例：:SOURce:OPP: STEP?
响应返回：1.000

命令格式：[:SOURce]:OPP:STEP:DELay {< value > | MINimum | MAXimum | DEFault}
功能描述：设置 OPPT 模式下的步进延迟时间
举　　例：:SOURce:OPP:STEP:DELay 1.000

命令格式：[:SOURce]:OPP:STEP:DELay?
功能描述：查询 OPPT 模式下的步进延迟时间
举　　例：:SOURce:OPP:STEP:DELay?
响应返回：2.000

命令格式：[:SOURce]:OPP:END {< value > | MINimum | MAXimum | DEFault}
功能描述：设置 OPPT 模式下的终止功率值
举　　例：:SOURce:OPP: END 5.000

命令格式：[:SOURce]:OPP:END?
功能描述：查询 OPPT 模式下的终止功率值
举　　例：:SOURce:OPP: END?
响应返回：4.000

命令格式：[:SOURce]:OPP:MIN {< value > | MINimum | MAXimum | DEFault}
功能描述：设置 OPPT 模式下的功率下限判断值
举　　例：:SOURce:OPP: MIN 2.000

命令格式：[:SOURce]:OPP:MIN?
功能描述：查询 OPPT 模式下的功率下限判断值
举　　例：:SOURce:OPP: MIN?
响应返回：3.000

命令格式：[:SOURce]:OPP:MAX {< value > | MINimum | MAXimum | DEFault}
功能描述：设置 OPPT 模式下的功率上限判断值
举　　例：:SOURce:OPP: MAX 10.000

命令格式：[:SOURce]:OPP:MAX?
功能描述：查询 OPPT 模式下的功率上限判断值
举　　例：:SOURce:OPP: MAX?
响应返回：8.000

命令格式：[:SOURce]:OPP:VOLTage {< value > | MINimum | MAXimum | DEFault}
功能描述：设置 OPPT 模式下的保护电压值

举　　例：:SOURce:OPP: VOLTage 8.000

命令格式：[:SOURce]:OPP:VOLTage?
功能描述：查询 OPPT 模式下的保护电压值
举　　例：:SOURce:OPP: VOLTage?
响应返回：10.000

3.3.11 Source Program 命令子系统

命令格式：[:SOURce]:PROGram:STEP {< number > | MINimum | MAXimum | DEFault}
功能描述：设置当前 PROGRAM 测试模式下的序列有效步数
举　　例：:SOURce:PROGram:STEP 8

命令格式：[:SOURce]:PROGram:STEP?
功能描述：查询当前 PROGRAM 测试模式下的序列有效步数
举　　例：:SOURce:PROGram:STEP?
响应返回：10

命令格式：[:SOURce]:PROGram:MODE <step, {CURRent | VOLTage | POWer | RESistance | LED}>
功能描述：设置 PROGRAM 测试下列表第 step 步的模式
举　　例：:SOURce:PROGram:MODE 2,VOLTage

命令格式：[:SOURce]:PROGram:MODE? <step>
功能描述：查询 PROGRAM 测试下列表第 step 步的模式
举　　例：:SOURce:PROGram:MODE? 3
响应返回：LED

命令格式：[:SOURce]:PROGram:IRANGe <step, value>
功能描述：设置 PROGRAM 测试下列表第 step 步的电流量程
举　　例：:SOURce:PROGram:IRANGe 4,5

命令格式：[:SOURce]:PROGram:IRANGe? <step>
功能描述：查询 PROGRAM 测试下列表第 step 步的电流量程范围
举　　例：:SOURce:PROGram:IRANGe? 5
响应返回：30

命令格式：[:SOURce]:PROGram:VRANGe <step, value>
功能描述：设置 PROGRAM 测试下列表第 step 步的电压量程
举　　例：:SOURce:PROGram:IRANGe 3,36

命令格式：[:SOURce]:PROGram:VRANGe? <step>
功能描述：查询 PROGRAM 测试下列表第 step 步的电压量程范围
举　　例：:SOURce:PROGram:IRANGe? 2
响应返回：150

命令格式：[:SOURce]:PROGram:RRANGe <step, {LOW | MIDDLE | HIGH | UPPER}>
功能描述：设置 PROGRAM 测试下列表第 step 步的电阻量程（此步为 CR 模式时有效）
举　　例：:SOURce:PROGram: RRANGe 3,LOW

命令格式：[:SOURce]:PROGram:RRANGe? <step>
功能描述：查询 PROGRAM 测试下列表第 step 步的电阻量程范围（此步为 CR 模式时有效）
举　　例：:SOURce:PROGram: RRANGe? 3
响应返回：MIDDLE

命令格式：[:SOURce]:PROGram:SHORt <step, {ON | OFF | 0 | 1 }>
功能描述：设置 PROGRAM 测试下列表第 step 步是否短路
举　　例：:SOURce:PROGram: SHORt 3,ON

命令格式：[:SOURce]:PROGram:SHORt? <step>
功能描述：查询 PROGRAM 测试下列表第 step 步是否短路
举　　例：:SOURce:PROGram: SHORt? 4
响应返回：0

命令格式：[:SOURce]:PROGram:PAUSE <step, {ON | OFF | 0 | 1 }>
功能描述：设置 PROGRAM 测试下列表第 step 步开始前是否暂停
举　　例：:SOURce:PROGram: PAUSE 2,1

命令格式：[:SOURce]:PROGram:PAUSE? <step>
功能描述：查询 PROGRAM 测试下列表第 step 步开始前是否暂停
举　　例：:SOURce:PROGram: PAUSE? 5
响应返回：0

命令格式：[:SOURce]:PROGram:TIME:ON <step, {< value > | MINimum | MAXimum | DEFault}>
功能描述：设置 PROGRAM 测试下列表第 step 步的带载时间（以“s”为单位）
举　　例：:SOURce:PROGram:TIME:ON 4,1.000

命令格式：[:SOURce]:PROGram:TIME:ON? <step>
功能描述：查询 PROGRAM 测试下列表第 step 步的带载时间（以“s”为单位）
举　　例：:SOURce:PROGram:TIME:ON? 5
响应返回：0.500

命令格式：[:SOURce]:PROGram:TIME:OFF <step, {< value > | MINimum | MAXimum | DEFault}>
功能描述：设置 PROGRAM 测试下列表第 step 步的关载时间（以“s”为单位）
举　　例：:SOURce:PROGram:TIME:OFF 4,0.500

命令格式：[:SOURce]:PROGram:TIME:OFF? <step>
功能描述：查询 PROGRAM 测试下列表第 step 步的关载时间（以“s”为单位）
举　　例：:SOURce:PROGram:TIME:OFF? 6
响应返回：2.000

命令格式：[:SOURce]:PROGram:TIME:DELay <step, {< value > | MINimum | MAXimum | DEFault}>
功能描述：设置 PROGRAM 测试下列表第 step 步的延迟时间（以“s”为单位，须小于带载　　时间）
举　　例：:SOURce:PROGram:TIME: DELay 3,0.500

命令格式：[:SOURce]:PROGram:TIME:DELay? <step>
功能描述：查询 PROGRAM 测试下列表第 step 步的延迟时间（以“s”为单位，须小于带载　　时间）
举　　例：:SOURce:PROGram:TIME: DELay? 1
响应返回：0.600

命令格式：[:SOURce]:PROGram:MIN <step, {< value > | MINimum | MAXimum | DEFault}>
功能描述：设置 PROGRAM 测试下列表第 step 步的最小合格判据值（CV 模式为最小电流， CC/CP/CR/LED 模式下为最小电压)）
举　　例：:SOURce:PROGram:MIN 3,1.000

命令格式：[:SOURce]:PROGram:MIN? <step>
功能描述：查询 PROGRAM 测试下列表第 step 步的最小合格判据值（CV 模式为最小电流， CC/CP/CR/LED 模式下为最小电压)）
举　　例：:SOURce:PROGram:MIN? 5
响应返回：2.000

命令格式：[:SOURce]:PROGram:MAX <step, {< value > | MINimum | MAXimum | DEFault}>

功能描述：设置 PROGRAM 测试下列表第 step 步的最大合格判据值（CV 模式为最大电流， CC/CP/CR/LED 模式下为最大电压)）
举　　例：:SOURce:PROGram: MAX 2,10.000

命令格式：[:SOURce]:PROGram:MAX? <step>
功能描述：查询 PROGRAM 测试下列表第 step 步的最大合格判据值（CV 模式为最大电流， CC/CP/CR/LED 模式下为最大电压)）
举　　例：:SOURce:PROGram:MAX? 3
响应返回：15.000

命令格式：[:SOURce]:PROGram:LEVel <step, {< value > | MINimum | MAXimum | DEFault}>
功能描述：设置 PROGRAM 测试下列表第 step 步的带载值 1
举　　例：:SOURce:PROGram:LEVel 6,7.000

命令格式：[:SOURce]:PROGram:LEVel? <step>
功能描述：查询 PROGRAM 测试下列表第 step 步的带载值 1
举　　例：:SOURce:PROGram:LEVel? 2
响应返回：6.000

命令格式：[:SOURce]:PROGram:LED:CURRent <step, {< value > | MINimum | MAXimum | DEFault}>
功能描述：设置 PROGRAM 测试下列表第 step 步的带载值 2（LED 模式下有效，即 Io）
举　　例：:SOURce:PROGram: LED:CURRent 6,7.000

命令格式：[:SOURce]:PROGram:LED:CURRent? <step>
功能描述：查询 PROGRAM 测试下列表第 step 步的带载值 2（LED 模式下有效，即 Io）
举　　例：:SOURce:PROGram: LED:CURRent? 4
响应返回：1.000

命令格式：[:SOURce]:PROGram:LED:RCOnf <step, {< value > | MINimum | MAXimum | DEFault}>
功能描述：设置 PROGRAM 测试下列表第 step 步的带载值 3（LED 模式下有效，即 Rconf）
举　　例：:SOURce:PROGram: LED: RCOnf 6,0.300

命令格式：[:SOURce]:PROGram:LED:RCOnf? <step>
功能描述：设置 PROGRAM 测试下列表第 step 步的带载值 3（LED 模式下有效，

即 Rconf）
举　　例：:SOURce:PROGram: LED: RCOnf? 5
响应返回：0.200

命令格式：[:SOURce]:PROGram:STATe:ON
功能描述：使负载设备进入 PROGRAM 测试模式
举　　例：:SOURce:PROGram:STATe:ON

命令格式：[:SOURce]:PROGram:STATe?
功能描述：查询负载设备当前是否为 PROGRAM 测试模式
举　　例：:SOURce:PROGram:STATe?
响应返回：0

命令格式：[:SOURce]:PROGram:TEST? <step>
功能描述：查询 PROGRAM 测试序列执行后第 step 步的结果
举　　例：:SOURce:PROGram:TEST? 3
响应返回：3.584720

3.3.12 Source Wave 命令子系统

命令格式：[:SOURce]:WAVE:TIME < number >
功能描述：设置趋势图界面的波形时间（以“s”为单位）
举　　例：:SOURce:WAVE:TIME 8

命令格式：[:SOURce]:WAVE:TIME?
功能描述：查询趋势图界面的波形时间（以“s”为单位）
举　　例：:SOURce:WAVE:TIME?
响应返回：3600

命令格式：[:SOURce]:WAVE:MODE {CURRent | VOLTage | POWer | RESistance }
功能描述：设置趋势图界面的波形显示的数据内容
举　　例：:SOURce]:WAVE:MODE CURRent

命令格式：[:SOURce]:WAVE:MODE?
功能描述：查询趋势图界面的波形显示的数据内容
举　　例：:SOURce]:WAVE:MODE?
响应返回：VOLTAGE

命令格式：[:SOURce]:WAVE:PAUSE {ON | OFF | 0 | 1}
功能描述：设置趋势图界面显示的波形是否暂停
举　　例：:SOURce:WAVE:PAUSE ON

命令格式：[:SOURce]:WAVE:PAUSE?
功能描述：查询趋势图界面显示的波形是否暂停
举　　例：:SOURce:WAVE:PAUSE?
响应返回：1

命令格式：[:SOURce]:WAVE:DISPlay {ON|OFF|0|1}
功能描述：使负载设备的显示界面进入趋势图波形显示
举　　例：:SOURce:WAVE:DISPlay ON

命令格式：[:SOURce]:WAVE:DISPlay?
功能描述：查询负载设备当前是否在趋势图波形显示界面
举　　例：:SOURce:WAVE:DISPlay?
响应返回：ON

3.3.13 Source Utility 命令子系统

命令格式：[:SOURce]:VOLTage[:LEVel]:ON <value>
功能描述：设置负载设备开载电压值
举　　例：:SOURce:VOLTage:LEVel:ON 6.000

命令格式：[:SOURce]:VOLTage[:LEVel]:ON?
功能描述：查询负载设备开载电压值
举　　例：:SOURce:VOLTage:LEVel:ON?
响应返回：4.000

命令格式：[:SOURce]:VOLTage:LATCh[:STATe] {ON | OFF | 0 | 1}
功能描述：设置负载设备带载状态锁存开关状态
举　　例：:SOURce:VOLTage:LATCh:STATe OFF

命令格式：[:SOURce]:VOLTage:LATCh[:STATe]?
功能描述：查询负载设备带载状态锁存开关状态
举　　例：:SOURce:VOLTage:LATCh:STATe?
响应返回：0

命令格式：[:SOURce]:EXT:INPUT[:StATe] {ON | OFF | 0 | 1}
功能描述：设置负载设备后面板外部控制开关状态
举　　例：:SOURce: EXT:INPUT:STATe OFF

命令格式：[:SOURce]: EXT:INPUT [:STATe]?
功能描述：查询负载设备后面板外部控制开关状态

举　　例：:SOURce: EXT:INPUT:STATe?
响应返回：0

命令格式：[:SOURce]:CURRent:PROTection:STATe {ON | OFF | 0 | 1}
功能描述：设置是否打开电流保护开关
举　　例：:SOURce:CURRent:PROTection:STATe ON

命令格式：[:SOURce]:CURRent:PROTection:STATe?
功能描述：查询是否打开电流保护开关
举　　例：:SOURce:CURRent:PROTection:STATe?
响应返回：1

命令格式：[:SOURce]:CURRent:PROTection:LEVel {< value > | MINimum | MAXimum | DEFault}
功能描述：设置电流保护门限值（电流保护打开后有效）
举　　例：:SOURce:CURRent:PROTection:LEVel 7.00

命令格式：[:SOURce]:CURRent:PROTection:LEVel?
功能描述：查询电流保护门限值
举　　例：:SOURce:CURRent:PROTection:LEVel?
响应返回：8.000

命令格式：[:SOURce]:CURRent:PROTection:DELay {< value > | MINimum | MAXimum | DEFault}
功能描述：设置电流保护延时时间值
举　　例：:SOURce:CURRent:PROTection:DELay 2.00

命令格式：[:SOURce]:CURRent:PROTection:DELay?
功能描述：查询电流保护延时时间值
举　　例：:SOURce:CURRent:PROTection:DELay?
响应返回：3.000

命令格式：[:SOURce]:POWer:PROTection:STATe {ON | OFF | 0 | 1}
功能描述：设置是否打开功率保护开关
举　　例：:SOURce:POWer:PROTection:STATe ON

命令格式：[:SOURce]:POWer:PROTection:STATe?
功能描述：查询是否打开功率保护开关
举　　例：:SOURce:POWer:PROTection:STATe?
响应返回：1

命令格式：[:SOURce]:POWer:PROTection:LEVel {< value > | MINimum | MAXimum | DEFault}
功能描述：设置功率保护门限值（功率保护打开后有效）
举　　例：:SOURce:POWer:PROTection:LEVel 7.00

命令格式：[:SOURce]:POWer:PROTection:LEVel?
功能描述：查询功率保护门限值
举　　例：:SOURce:POWer:PROTection:LEVel?
响应返回：8.000

命令格式：[:SOURce]:POWer:PROTection:DELay {< value > | MINimum | MAXimum | DEFault}
功能描述：设置功率保护延时时间值
举　　例：:SOURce:POWer:PROTection:DELay 2.00

命令格式：[:SOURce]:POWer:PROTection:DELay?
功能描述：查询功率保护延时时间值
举　　例：:SOURce:POWer:PROTection:DELay?
响应返回：3.000

3.4　System 命令子系统

命令格式：SYSTem:SENSe[:STATe] {ON | OFF | 0 | 1}
功能描述：设置外部远程电压测量端使能开关状态
举　　例：SYSTem:SENSe:STATe OFF

命令格式：SYSTem:SENSe[:STATe]?
功能描述：查询外部远程电压测量端使能开关状态
举　　例：SYSTem:SENSe:STATe?
响应返回：1

命令格式：SYSTem:IMONItor[:STATe] {ON | OFF | 0 | 1}
功能描述：设置电流监控端子使能开关状态
举　　例：SYSTem:IMONItor:STATe OFF

命令格式：SYSTem: IMONItor [:STATe]?
功能描述：查询电流监控端子使能开关状态
举　　例：SYSTem: IMONItor:STATe?
响应返回：1

命令格式：SYSTem:VMONItor[:STATe] {ON | OFF | 0 | 1}

功能描述：设置电压监控端子使能开关状态
举　　例：SYSTem:VMONItor:STATe OFF

命令格式：SYSTem: VMONItor [:STATe]?
功能描述：查询电压监控端子使能开关状态
举　　例：SYSTem: VMONItor:STATe?
响应返回：1

命令格式：STOP:ON:FAIL[:STATe] {ON | OFF | 0 | 1}
功能描述：设置当测试失败时是否停止测试（ON/1:停止，OFF/0:不停止）
举　　例：STOP:ON:FAIL:STATe OFF

命令格式：STOP:ON:FAIL[:STATe]?
功能描述：查询当测试失败时是否停止开关的状态（ON/1:停止，OFF/0:不停止）
举　　例：STOP:ON:FAIL:STATe?
响应返回：0

命令格式：*TRG
功能描述：触发指令，产生一次触发动作负载运行
举　　例：*TRG

命令格式：TRIGger:SOURce {MANUal | EXTernal | BUS}
功能描述：设置负载设备的触发源
举　　例：TRIGger:SOURce BUS

命令格式：TRIGger:SOURce?
功能描述：查询负载设备的触发源
举　　例：TRIGger:SOURce?
响应返回：MANUAL

命令格式：SENSe:AVERage:COUNt {6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14}
功能描述：设置负载设备的读数平均点数（以 2 的指数为设置选择值）
举　　例：SENSe:AVERage:COUNt 10

命令格式：SENSe:AVERage:COUNt?
功能描述：查询负载设备的读数平均点数（以 2 的指数为设置选择值）
举　　例：SENSe:AVERage:COUNt?
响应返回：9

命令格式：EXT:MODE {INT | EXTI | EXTV}
功能描述：设置负载设备拉载模式（内部或外部）

举　　例：EXT:MODE INT

命令格式：EXT:MODE?
功能描述：查询负载设备拉载模式（内部或外部）
举　　例：EXT:MODE?
响应返回：EXTV

命令格式：EXT:IRANGe <value>
功能描述：设置外部拉载模式下的电流量程
举　　例：EXT:IRANGe 5

命令格式：EXT:IRANGe?
功能描述：查询外部拉载模式下的电流量程范围
举　　例：EXT:IRANGe?
响应返回：30

命令格式：EXT:VRANGe <value>
功能描述：设置外部拉载模式下的电压量程
举　　例：EXT:VRANGe 36

命令格式：EXT:VRANGe?
功能描述：查询外部拉载模式下的电压量程范围
举　　例：EXT:VRANGe?
响应返回：150

命令格式：TIME:TEST[:STATe] {ON | OFF | 0 | 1}
功能描述：设置是否开启时间测量功能
举　　例：TIME:TEST:STATe OFF

命令格式：TIME:TEST[:STATe]?
功能描述：查询是否开启时间测量功能
举　　例：TIME:TEST:STATe?
响应返回：0

命令格式：TIME:TEST:VOLTage:LOW {< value > | MINimum | MAXimum | DEFault}
功能描述：设置时间测量电压低点值
举　　例：TIME:TEST:VOLTage:LOW 1.000
命令格式：TIME:TEST:VOLTage:LOW?
功能描述：查询时间测量电压低点值

举　　例：TIME:TEST:VOLTage:LOW?
响应返回：3.00

命令格式：TIME:TEST:VOLTage:HIGH {< value > | MINimum | MAXimum | DEFault}
功能描述：设置时间测量电压高点值
举　　例：TIME:TEST:VOLTage: HIGH 10.000

命令格式：TIME:TEST:VOLTage:HIGH?
功能描述：查询时间测量电压高点值
举　　例：TIME:TEST:VOLTage: HIGH?
响应返回：150.00

命令格式：TIME:TEST:RISE?
功能描述：获取上升测量时间
举　　例：TIME:TEST:RISE?
响应返回：0.020

命令格式：TIME:TEST:FALL?
功能描述：获取下降测量时间
举　　例：TIME:TEST: FALL?
响应返回：0.030

3.5　LAN Interface 命令子系统

命令格式：LAN:LINK?
功能描述：查询负载设备网口是否接入局域网（即插入网线）
举　　例：LAN:LINK?
响应返回：0 （注： 0 为未接入网线，1 为接入网线）

命令格式：DHCP {ON | OFF | 0 | 1}
功能描述：设置 DHCP 开关状态
举　　例：DHCP ON

命令格式：DHCP?
功能描述：查询 DHCP 开关状态
举　　例：DHCP?
响应返回：1

命令格式：LAN:IPADdress <aaa.bbb.ccc.ddd>
功能描述：设置网口 IP 地址为 aaa.bbb.ccc.ddd（DHCP 开关关闭时有效）

举　　例：LAN:IPADdress 10.12.15.64

命令格式：LAN:IPADdress?
功能描述：获取网口 IP 地址
举　　例：LAN:IPADdress?
响应返回：10.11.13.76

命令格式：LAN:SMASk <aaa.bbb.ccc.ddd>
功能描述：设置网口子网掩码为 aaa.bbb.ccc.ddd（DHCP 开关关闭时有效）
举　　例：LAN: SMASk 255.255.255.255

命令格式：LAN:SMASk?
功能描述：查询网口子网掩码
举　　例：LAN: SMASk?
响应返回：255.255.255.0

命令格式：LAN:GATeway <aaa.bbb.ccc.ddd>
功能描述：设置网口网关为 aaa.bbb.ccc.ddd（DHCP 开关关闭时有效）
举　　例：LAN: GATeway 10.11.13.1

命令格式：LAN:GATeway?
功能描述：查询网口网关
举　　例：LAN: GATeway?
响应返回：10.12.16.1

命令格式：LAN:MAC?
功能描述：查询负载设备的 MAC 码
举　　例：LAN:MAC?
响应返回：00.80.e1.00.00.00

4. 编程示例

本章给出一些例子，在这些例子中你可以看到如何使用 ni-visa 库和本章之前描述的命令来控制我们的设备。通过编程示例，你可以开发更多的功能应用。这个例子是 Visual Studio 开发的项目。

4.1 使用 VISA 的编程示例

4.1.1 VC++示例

环境：Windows xp system, Visual Studio

示例内容：使用 ni-visa 控制设备和 Usbtmc TCP / IP 访问读写。

按以下步骤完成示例：

1、打开 Visual Studio,创建一个新的 vc++ win32 项目。

设置项目环境使用 ni-visa 库，有两种方法可以使用 ni-visa，静态方式和自动方式：

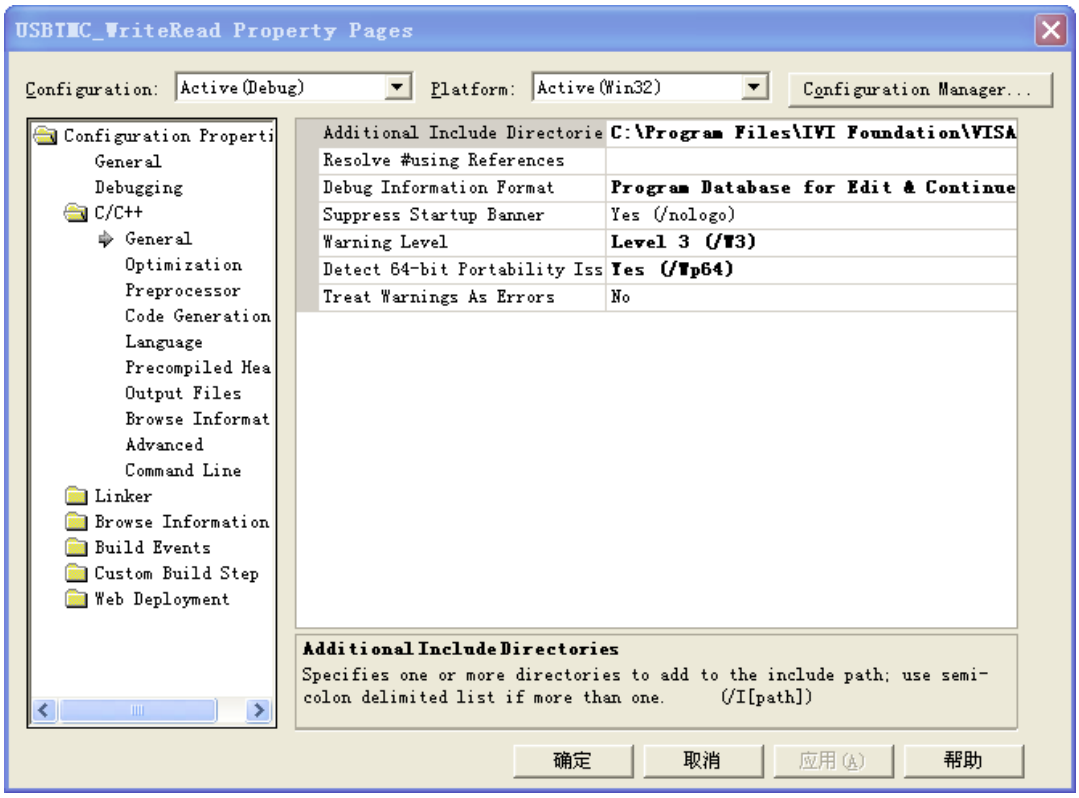
(1) 静态方式：

在 NI-VISA 安装路径查找文件：visa.h,visatype.h,visa32.lib。将它们复制到你的项目，并将它们添加到项目中。在项目.cpp 文件，添加如下两行：

```
#include "visa.h"
#pragma comment(lib,"visa32.lib")
```

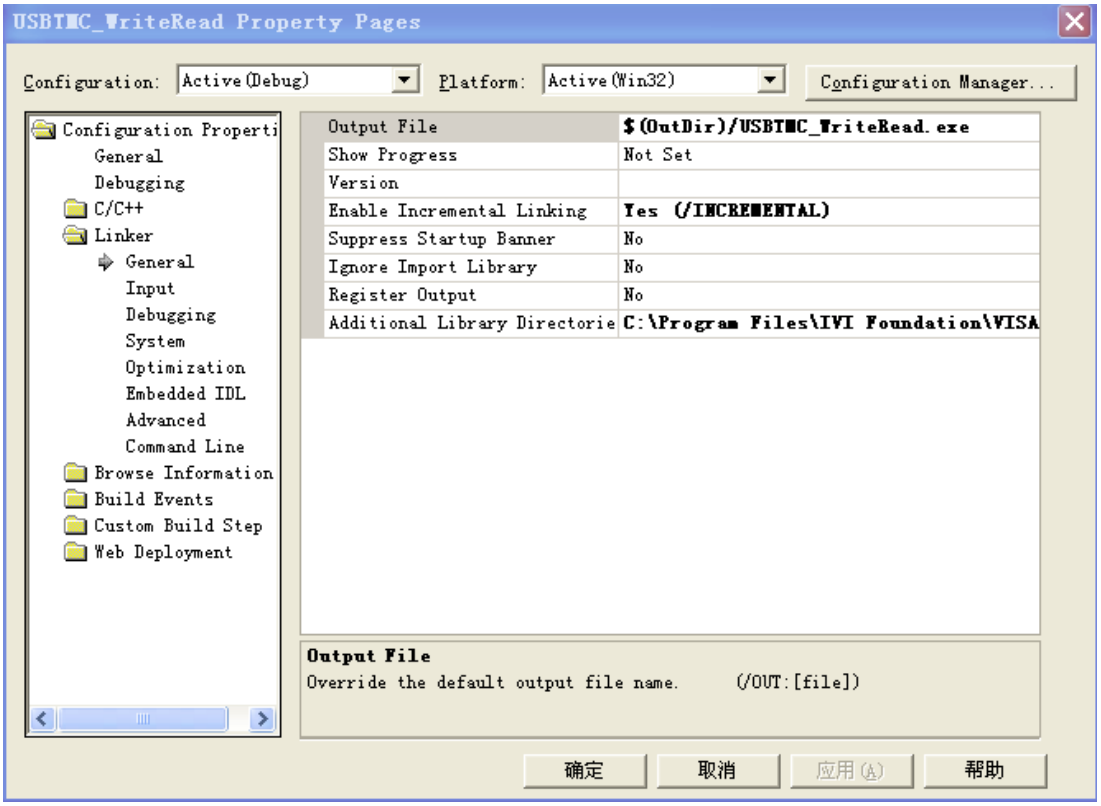
(2) 自动方式：

设置.h 文件包括目录，ni-visa 安装路径。在我们的电脑，我们设置的路径是：C：\Program Files\IVI Foundation\VISA\WINNT\include。设置这条路径到项目—属性—C/C++—通用—附加包含路径，如图：

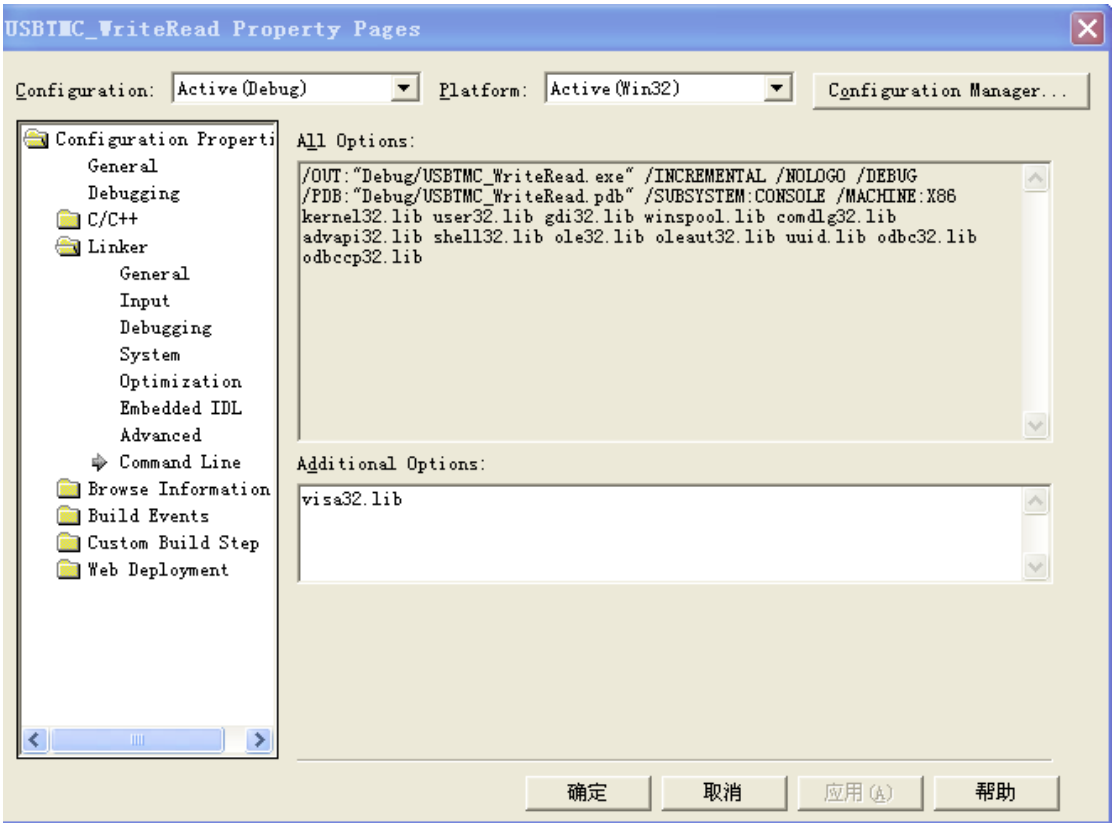


2、设置库路径设置库文件:

设置库路径：在 ni-visa 安装路径，在我们的电脑，我们设置的路径是：C：\Program Files\IVI Foundation\VISA\WINNT\LIB\MSC。设置这条路径到项目—性能—连接器—常规—附加库目录，如图：



设置库文件： project---properties---Linker---Command Line---Additional Options:visa32.lib



包括 visa.h file:在 XXX.cpp 文件里：

```
#include <visa.h>
```

3、增加代码：

(1)USBTMC 存取代码:：

写一个 Usbtmc_test 函数。

```
int Usbtmc_test()
{
/*    这段代码演示了使用 NI-VISA 发送同步读取和写入命令到        */
/*    一个 USB 测试&测量类(USBTMC)仪器。                        */
/*    这个例子写"* IDN ?\ n "字符串到所有连接到系统的 USBTMC    */
/*    设备并试图使用读写函数读回结果。      */
/*    代码的一般流程是打开资源管理器        */
/*    打开 VISA 会话到仪器                  */
/*    使用 viPrintf 写仪器标志查询          */
/*     尝试随 viScanf 读取一个响应          */
/*     关闭 VISA 会话                       */
    /************************************************************/
    ViSession defaultRM;
    ViSession instr;
    ViUInt32 numInstrs;
    ViFindList findList;
    ViUInt32 retCount;
    ViUInt32 writeCount;
    ViStatus status;
    char instrResourceString[VI_FIND_BUFLEN];
    unsigned char buffer[100];
    char stringinput[512];
    int i;
    /* 首先，我们必须调用 viOpenDefaultRM 得到管理器的句柄。 */
    /* 我们将在 defaultRM 存储此手柄。                        */
```

```
    status=viOpenDefaultRM (&defaultRM);
    if (status < VI_SUCCESS)
    {
        printf ("Could not open a session to the VISA Resource Manager!\n");
         return status;
    }
    /*  寻找我们的系统中所有的 USB TMC VISA 资源          */
    /*  然后将资源的数目存储在系统中的 numInstrs 里。      */
    status = viFindRsrc (defaultRM, "USB?*INSTR", &findList, &numInstrs,
instrResourceString);
    if (status < VI_SUCCESS)
    {
        printf ("An error occurred while finding resources.\nHit enter to
continue.");
        fflush(stdin);
        getchar();
        viClose (defaultRM);
        return status;
    }
    /**现在，我们将对所有 USB TMC 仪器打开 VISA 会话。我们必须
    *  从 viOpenDefaultRM 使用句柄，也必须使用一个字符串指示要
    *打开的仪器，这就是所谓的仪器描述符。该字符串的格式可以在功
    *能面板中右键单击参数描述中找到。打开一个会话到设备后，我们
    *将得到一个仪器使用的句柄，在之后使用 VISA 功能时用到。在这
    *个函数的 AccessMode 和超时参数是为将来的功能预留。这两个参
    *数被给予值 VI_NULL。                                          */
    for (i=0; i<numInstrs; i++)
    {
        if (i > 0)
            viFindNext (findList, instrResourceString);
```

```
        status = viOpen (defaultRM, instrResourceString, VI_NULL, VI_NULL,
&instr);
        if (status < VI_SUCCESS)
        {
            printf ("Cannot open a session to the device %d.\n", i+1);
            continue;
        }
        /* *在这一点上，我们现在有一个会话打开到 USB TMC 仪器。现在，
        *我们将使用 viPrintf 函数发送字符串“*IDN?\n”到设备，要求设备识别。
*/
        char * cmmand ="*IDN?\n";
        status = viPrintf    (instr, cmmand);
        if (status < VI_SUCCESS)
        {
            printf ("Error writing to the device %d.\n", i+1);
            status = viClose (instr);
            continue;
        }
        /**  现在我们将尝试从设备读回一个设备信息查询的响应。我们将
        *使用 viScanf 函数来获取数据。在数据被读出后，响应显示出来     */
        status =    viScanf(instr, "%t", buffer);
        if (status < VI_SUCCESS)
            printf ("Error reading a response from the device %d.\n", i+1);
        else
            printf ("\nDevice %d:%*s\n", i+1,retCount, buffer);
        status = viClose (instr);
    }
    /**现在，我们将关闭会话使用 viClose 仪器。此操作释放所有系统资源。*/
    status = viClose (defaultRM);
return 0;
```

```
}
```

(2)TCP/IP access code:

Write a function TCP_IP_Test.

```
int TCP_IP_Test(char *pIP)
{
    char outputBuffer[VI_FIND_BUFLEN];
    ViSession defaultRM, instr;
    ViStatus status;
    ViUInt32 count;
    ViUInt16 portNo;
   /* 首先，我们需要打开默认的资源管理器。          */
   status = viOpenDefaultRM (&defaultRM);
   if (status < VI_SUCCESS)
   {
      printf("Could not open a session to the VISA Resource Manager!\n");
   }
   /*现在，我们将通过 TCP / IP 设备打开一个会话*/
   char head[256] ="TCPIP0::";
   char tail[] ="::INSTR";
   char resource [256];
   strcat(head,pIP);
   strcat(head,tail);
   status = viOpen (defaultRM, head, VI_LOAD_CONFIG, VI_NULL, &instr);
   if (status < VI_SUCCESS)
   {
      printf ("An error occurred opening the session\n");
      viClose(defaultRM);
   }
   status = viPrintf(instr, "*idn?\n");
```

```
    status = viScanf(instr, "%t", outputBuffer);
    if (status < VI_SUCCESS)
    {
        printf("viRead failed with error code: %x \n",status);
        viClose(defaultRM);
    }else
        printf ("\ndata read from device: %*s\n", 0,outputBuffer);
    status = viClose (instr);
    status = viClose (defaultRM);
    return 0;
}
```

4.1.2 VB 示例

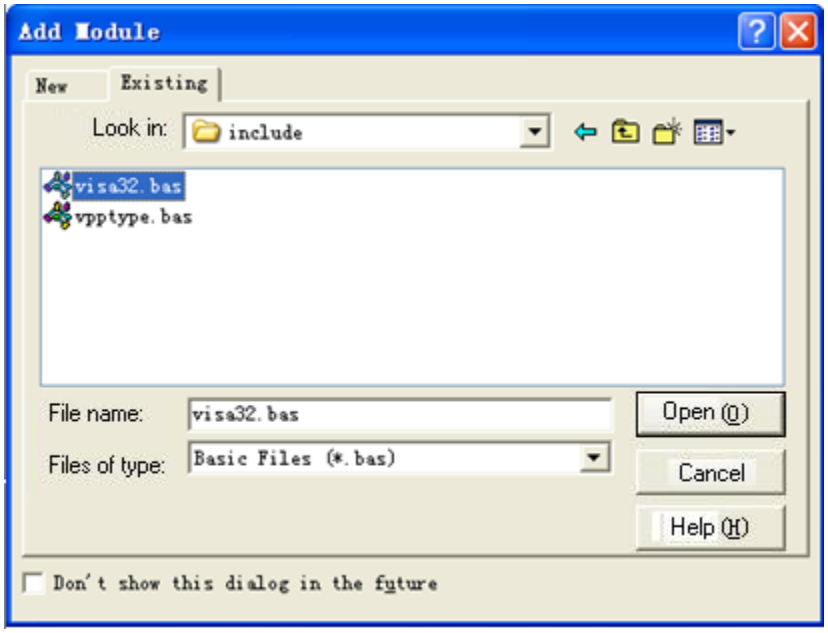
环境：Windows 7 , Microsoft Visual Basic 6.0

示例内容：使用 NI-VISA，通过 USBTMC 和 TCP/ IP 访问控制设备去写入和读取。

按照步骤完成的例子：

1、 打开 Visual Basic，建立一个标准的应用程序项目（标准 EXE）。

2、 设置项目的环境中使用 NI-VISA 库，单击项目的现有标签>>添加模块。搜索 visa32.bas 中 NI-VISA 安装路径下的 include 文件夹文件，并添加文件。



这使得 VISA 功能和 VISA 的数据类型在程序中使用。

3、添加代码：

(1)USBTMC 存取代码：

写 Usbtmc_test 函数.

```
Private Function Usbtmc_test() As Long
    '   这段代码演示了使用 NI-VISA 发送同步读取和写入命令到
    '   一个 USB 测试&测量类(USBTMC)仪器。
    '   这个例子写"* IDN ?\ n "字符串到所有连接到系统的 USBTMC
    '   设备并试图使用读写函数读回结果。
    '   代码的一般流程是打开资源管理器
    '   打开 VISA 会话到仪器
    '   使用 viPrintf 写仪器标志查询
    '   尝试随 viScanf 读取一个响应
    '   关闭 VISA 会话
    Const MAX_CNT = 200

    Dim defaultRM As Long
    Dim instrsesn As Long
    Dim numInstrs As Long
```

```
    Dim findList As Long
    Dim retCount As Long
    Dim writeCount As Long
    Dim status As Long
    Dim instrResourceString As String * VI_FIND_BUFLEN
    Dim buffer As String * MAX_CNT
    Dim i As Integer

    ' 首先，我们必须调用 viOpenDefaultRM 得到管理器的句柄。
    ' 我们将在 defaultRM 存储此手柄。
    status = viOpenDefaultRM(defaultRM)
    If (status < VI_SUCCESS) Then
        Debug.Print "Could not open a session to the VISA Resource
Manager!"
        Usbtmc_test = status
        Exit Function
    End If

    ' 寻找我们的系统中所有的 USB TMC VISA 资源
    ' 和然后将资源的数目存储在系统中的 numInstrs 里。
    status = viFindRsrc(defaultRM, "USB?*INSTR", findList, numInstrs,
instrResourceString)
    If (status < VI_SUCCESS) Then
        Debug.Print "An error occurred while finding resources."
        viClose (defaultRM)
        Usbtmc_test = status
        Exit Function
    End If

    ' 现在，我们将对所有 USB TMC 仪器打开 VISA 会话。我们必须
```

```
    ' 从 viOpenDefaultRM 使用句柄， 也必须使用一个字符串指示要
    ' 打开的仪器，这就是所谓的仪器描述符。该字符串的格式可以在功
    ' 能面板中右键单击参数描述中找到。打开一个会话到设备后，我们
    ' 将得到一个仪器使用的句柄，在之后使用 VISA 功能时用到。在这
    ' 个函数的 AccessMode 和超时参数是为将来的功能预留。这两个参
    ' 数被给予值 VI_NULL。
    For i = 0 To numInstrs
        If (i > 0) Then
            status = viFindNext(findList, instrResourceString)
        End If
        status = viOpen(defaultRM, instrResourceString, VI_NULL, VI_NULL,
instrsesn)
        If (status < VI_SUCCESS) Then
            Debug.Print "Cannot open a session to the device ", i + 1
            GoTo NextFind
        End If

        ' 在这一点上，我们现在有一个会话打开到 USB TMC 仪器。现在，
        ' 我们将使用 viPrintf 函数发送字符串“*IDN?\n”到设备，要求设备识别。
        status = viWrite(instrsesn, "*IDN?", 5, retCount)
        If (status < VI_SUCCESS) Then
            Debug.Print "Error writing to the device."
            status = viClose(instrsesn)
            GoTo NextFind
        End If

        ' 现在我们将尝试从设备读回一个设备信息查询的响应。我们将
        ' 使用 viScanf 函数来获取数据。在数据被读出后，响应显示出来
        status = viRead(instrsesn, buffer, MAX_CNT, retCount)
        If (status < VI_SUCCESS) Then
```

```
            Debug.Print "Error reading a response from the device.", i + 1
        Else
            Debug.Print i + 1, retCount, buffer
        End If
        status = viClose(instrsesn)
NextFind:
    Next i

    ' 现在，我们将关闭会话使用 viClose 仪器。此操作释放所有系统资源。
    status = viClose(defaultRM)
    Usbtmc_test = 0
End Function
```

(2)TCP/IP access code:

Write a function TCP_IP_Test.

```
Private Function TCP_IP_Test(ip As String) As Long
    Dim outputBuffer As String * VI_FIND_BUFLEN
    Dim defaultRM As Long
    Dim instrsesn As Long
    Dim status As Long
    Dim count As Long

    ' 首先，我们需要打开默认的资源管理器。
    status = viOpenDefaultRM (defaultRM)
    If (status < VI_SUCCESS) Then
        Debug.Print "Could not open a session to the VISA Resource
Manager!"
        TCP_IP_Test = status
        Exit Function
    End If
```

```
    ' 现在，我们将通过 TCP / IP 设备打开一个会话
    status = viOpen(defaultRM, "TCPIP0::" + ip + "::INSTR",
VI_LOAD_CONFIG, VI_NULL, instrsesn)
    If (status < VI_SUCCESS) Then
        Debug.Print "An error occurred opening the session"
        viClose (defaultRM)
        TCP_IP_Test = status
        Exit Function
    End If

    status = viWrite(instrsesn, "*IDN?", 5, count)
    If (status < VI_SUCCESS) Then
        Debug.Print "Error writing to the device."
    End If
    status = viRead(instrsesn, outputBuffer, VI_FIND_BUFLEN, count)
    If (status < VI_SUCCESS) Then
        Debug.Print "Error reading a response from the device.", i + 1
    Else
        Debug.Print "read from device:", outputBuffer
    End If
    status = viClose(instrsesn)
    status = viClose(defaultRM)
    TCP_IP_Test = 0
End Function
```

4.1.3 MATLAB 示例

环境：windows 7, MATLAB R2010b

示例内容：使用 NI-VISA，通过 USBTMC 和 TCP/ IP 访问控制设备去写入和读取。

按照步骤完成的例子：

1、打开 MATLAB，修改当前目录。在本演示中，将当前目录修改为 D：\ USBTMC_TCPIP_Demo。

2、点击文件>>新建>>脚本（File>>New>>Script）在 Matlab 界面来创建一个空的 M 文档

3、添加代码：

(1)USBTMC 存取代码：

写入 Usbtmc_test 函数。

```
function USBTMC_test()
% 这段代码演示了使用 NI-VISA 发送同步读取和写入命令到
% 一个 USB 测试&测量类(USBTMC)仪器。

% 创建一个 VISA-USB 对象连接到 USB 仪器上
vu = visa('ni','USB0::0xF4EC::0x1600::0123456789::INSTR');

% 打开创建的 VISA 对象
fopen(vu);

% 发送字符串“* IDN？”，查询设备信息。
fprintf(vu,'*IDN?');

% 请求数据
outputbuffer = fscanf(vu);
disp(outputbuffer);

% 关闭 VISA 对象
fclose(vu);
delete(vu);
clear vu;

end
```

(2)TCP/IP 存取代码：

写入 TCP_IP_Test 函数。

```
function TCP_IP_test( IPstr )
% 这段代码演示了使用 NI-VISA 发送同步读取和写入命令到
% 一个 TCP/IP 仪器。

%创建一个 VISA-TCPIP 对象连接到配置了 IP 地址的仪器。
vt = visa('ni',['TCPIP0::',IPstr,'::INSTR']);

% 打开创建的 VISA 对象
fopen(vt);

%发送字符串"*IDN?"，查询设备信息
fprintf(vt,'*IDN?');

% 请求数据
outputbuffer = fscanf(vt);
disp(outputbuffer);

% 关闭 VISA 对象
fclose(vt);
delete(vt);
clear vt;

end
```

4.1.4 LabVIEW 示例

环境：windows 7 system, LabVIEW 2011

示例内容：使用 NI-VISA，通过 USBTMC 和 TCP/IP 访问控制设备去写入和读

取。

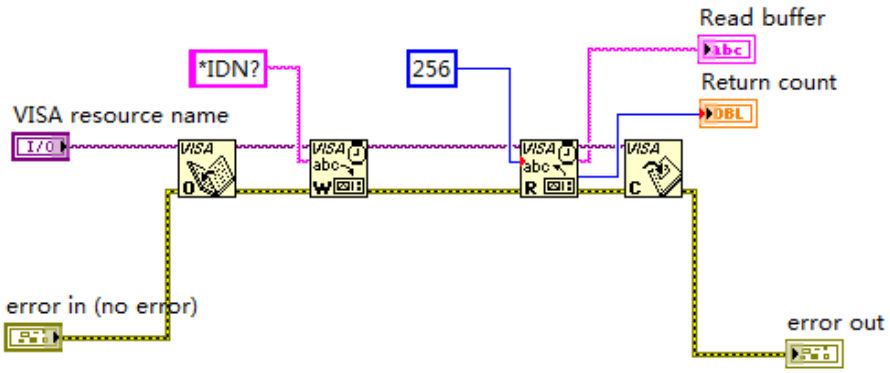
按照步骤完成的例子：

1、打开 LabVIEW，创建 VI 文件.

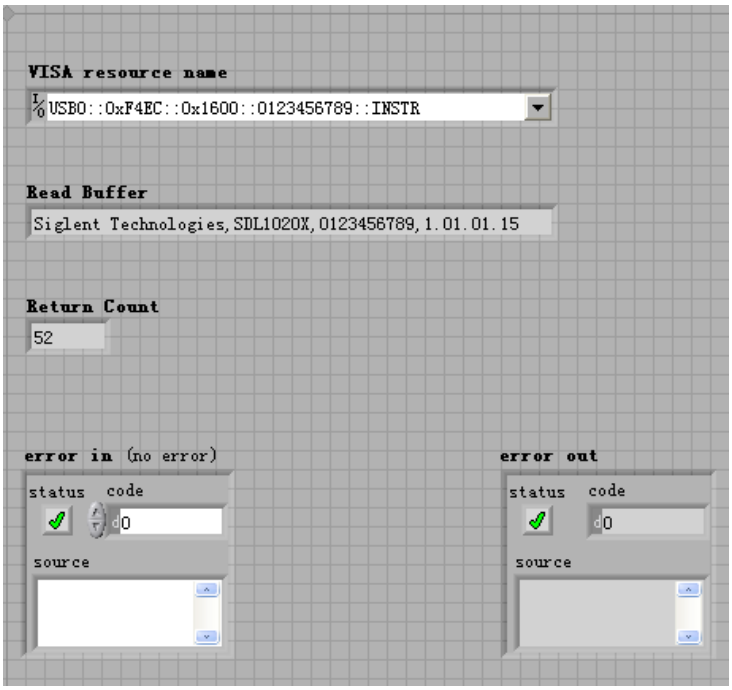
2、添加控件。右键单击前面板接口，选择并加入 VISA 资源名称，错误输入，错误输出以及控制栏的一些指标

3、打开框图接口。在 VISA 资源名称单击鼠标右键，可以从 VISA 调色板的快捷菜单中选择添加以下功能：VISA 写，VISA 读，VISA 打开和 VISA 关闭。

4、把它们连接起来，如下图所示：



5、从 VISA 资源名称列表框中选择设备资源然后运行程序。

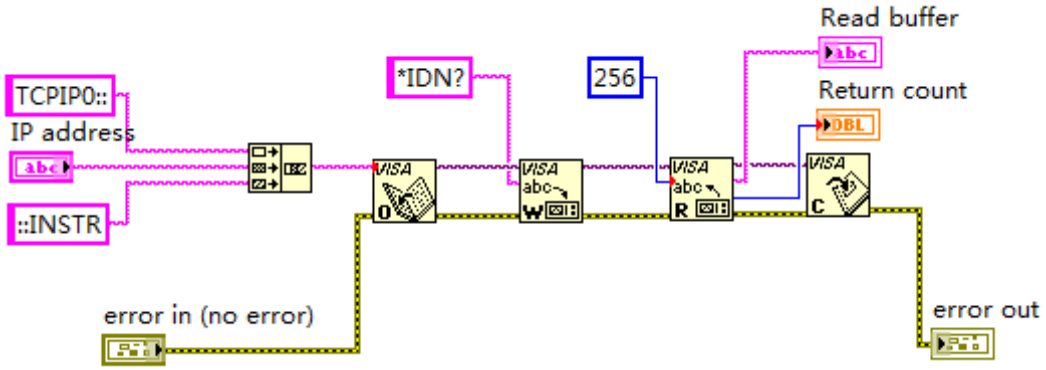


在这个例子中，VI 打开一个 VISA 会话到 USBTMC 设备，写入一个命令到设备，然后读回并响应。此例中，发送的特定的命令是设备 ID 查询。请与您的

设备制造商核对设备命令集。在所有通讯完成后，VI 关闭 VISA 会话。

通过 TCP/IP 与以太网仪器通信是类似于 USBTMC 的。但是，你需要改变 VISA 写入和 VISA 读取功能来同步 I/O。LabVIEW 的默认值是异步的 I/O。右键单击该节点，然后从快捷菜单中选择同步 I/O 模式>>同步来写入或读取同步数据。

1、把它们连接起来，如下图所示：



2、输入 IP 地址然后运行程序。