

UDP6900系列可编程数控电源

SCPI编程手册

REV 00

2023. 9

UNI-T®

版权

2023 优利德中国科技有限公司

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

软件版本

1.00.0905

软件升级可能更改或增加产品功能，请关注 **UNI-T** 网站获取最新版本手册或联系 **UNI-T** 升级软件。

声明

- 本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: 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。

第一章.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。

指令格式

编程命令由关键字、分隔符、参数域和结束符等部分构成, 以下面的命令为例:

```
:VOLTage:LEVel 25
```

VOLTage、LEVel 是命令关键字, “:” 和空格为分隔符, “25” 为参数 (部分命令具有多个参数, 参数间以 “,” 分隔), 命令后的换行符或回车符为命令结束符。

为了方便描述, 后续出现的各种符号采用如下约定:

- 方括号 “[] ” 表示可选的关键字或参数可省略。
- 花括号 “{ } ” 表示命令串中的参数选项。
- 尖括号 “< > ” 表示必须提供一个数值参数。
- 垂直线 “| ” 用于分隔多个可选参数的选项。
- 结束符: 换行符 <LF> (0x0A) 或回车符 <CR> (0x0D)

参数说明

编程参数的数据类型有数值型、字符型、布尔型等多种类型。不论哪种类型, 均以 ASCII 码串表示, 详见下表。

符号	意义	范例
<NR1>	整形	123, 0123
<NR2>	定点浮点数	123., 12.3, 0.123, .123
<NR3>	浮点数	123, 12.3, 123E+3
<NRF>	可能是 <NR1>、<NR2> 或 <NR3>	
<Boolean>	布尔数据	0 1 ON OFF

简写规则

所有命令对大小写都能识别, 可以全部采用大写或小写。如果要缩写, 必须输完命令格式中的所有大写字母。

数据返回

数据返回分为单个数据和批量数据返回, 单个数据返回相对应的参数类型, 其中实型返回用科学计数法表示, e 前部分小数点后面保留三位数据, e 部分保留三位数据; 批量数据返回必须符合 IEEE 488.2 # 格式的字符串数据, 其格式: ‘#’ + 长度所占的字符位数 [固定为一个字符] + 有效数据长度的 ASCII 值 + 有效数据 + 结束符 [‘\n’], 例如 #3123xxxxxxxxxxxxxxxxxxxx\n 表示的具有 123 个字节有效批量数据返回格式, 其中 ‘3’ 表示 “123” 占 3 个字符位。

Mult-SCPI 多机协议

Mult-SCPI 指令在标准 SCPI 协议基础上,添加地址信息实现多机 SCPI 通讯。

指令格式:

ADDR[空格]<address>:{SCPI 指令}

<address>=多机通讯地址,取值:1~32,广播地址为 0

举例说明:

ADDR 1:*IDN?

->Uni-Trend,UDP6942B,[Serial Number],1.00.0905

第二章 通信接口及通信设置

详细说明请参考《UDP6900系列数控电源说明书》的“第三章 3.15.4 RS232与RS485功能”

第三章.SCPI 指令详解

3.1 指令列表

测量指令	指令描述
:MEASure:VOLTage?	测量当前输出电压
:MEASure:CURREnt?	测量当前输出电流
:MEASure:POWER?	测量当前输出功率
:MEASure:DVM?	测量当前电压表通道的电压
:MEASure:ALL?	测量当前输出电压、电流和功率
输出设置指令	指令描述
[[:SOURce]:VOLTage[:LEVel][:IMMediate][:AMPLitude]	设置输出电压值
[[:SOURce]:CURREnt[:LEVel][:IMMediate][:AMPLitude]	设置输出电流值
[[:SOURce]:VOLTage:PROTection[:LEVel]	设置过压保护功能的保护值
[[:SOURce]:CURREnt:PROTection[:LEVel]	设置过流保护功能的保护值
[[:SOURce]:VOLTage:PROTection:STATe	打开或关闭过压保护功能
[[:SOURce:]CURREnt:PROTection:STATe	打开或关闭过流保护功能
[[:SOURce:]VOLTage:PROTection:TRIPed?	查询过压保护 OVP 的触发状态
[[:SOURce:]VOLTage:PROTection:CLEAr	清除过压保护 OVP 的触发状态
[[:SOURce:]CURREnt:PROTection:TRIPed?	查询过流保护 OCP 的触发状态
[[:SOURce:]CURREnt:PROTection:CLEAr	清除过流保护 OCP 的触发状态
[[:SOURce:]VOLTage:STEP	设置设定电压步进值
[[:SOURce:]VOLTage:UP	向上步进电压设定值

[[:SOURce:]VOLTage:DOWN	向下步进电压设定值
[[:SOURce:]CURRent:STEP	设置设定电流步进值
[[:SOURce:]CURRent:UP	向上步进电流设定值
[[:SOURce:]CURRent:DOWN	向下步进电流设定值
[[:SOURce:]VOLTage:SLEW:RISing	设置电压的上升斜率
[[:SOURce:]VOLTage:SLEW:FALLing	设置电压的下降斜率
[[:SOURce:]CURRent:SLEW:RISing	设置电流的上升斜率
[[:SOURce:]CURRent:SLEW:FALLing	设置电流的下降斜率
:OUTPut:MODE	设置电源的斜率工作模式
:OUTPut[::STATe]	打开或关闭电源输出
:OUTPut:CVCC?	查询输出恒压恒流状态
:OUTPut:OVP:VALue	设置过压保护功能的保护值
:OUTPut:OCP:VALue	设置过流保护功能的保护值
:OUTPut:OVP[::STATe]	打开或关闭过压保护功能
:OUTPut:OCP[::STATe]	打开或关闭过流保护功能
:OUTPut:OVP:TRIPed?	查询过压保护 OVP 的触发状态
:OUTPut:OVP:CLEar	清除过压保护 OVP 的触发状态
:OUTPut:OCP:TRIPed?	查询过流保护 OCP 的触发状态
:OUTPut:OCP:CLEar	清除过压保护 OCP 的触发状态
:OUTPut:TIMer	设置定时关闭的功能开关
:OUTPut:TIMer:DATA	设置定时关闭的时长
列表模式指令	指令描述
:LISTout[::STATe]	设置 List 列表模式开关与查询列表模式状态
:LISTout:BASE	设置 List 列表模式的基础参数
:LISTout:PARAMeter	设置 List 列表模式的组参数
:LISTout:TEMPlEt:CONSTRuCT	按照设定模板参数构建列表组参数构建组参数
:LISTout:TEMPlEt:SELEct	设置列表模式的模板类型
:LISTout:TEMPlEt:OBJEct	设置列表模式的模板对象
:LISTout:TEMPlEt:STARt	设置列表模式的构建起始组
:LISTout:TEMPlEt:POINtS	设置列表模式的构建组数
:LISTout:TEMPlEt:MAXValue	设置列表模式的构建对象最大值
:LISTout:TEMPlEt:MINValue	设置列表模式的构建对象最小值
:LISTout:TEMPlEt:INTERVal	设置列表模式构建模板的时间长度
:LISTout:TEMPlEt:INVERt	设置列表模式构建模板的反相模式
:LISTout:TEMPlEt:WIDTh	设置列表模式脉冲模板的脉冲宽度

:LISTout:TEMPlEt:PERIod	设置列表模式脉冲模板的脉冲周期
:LISTout:TEMPlEt:SYMMetry	设置列表模式斜坡模板的对称度
:LISTout:TEMPlEt:EXPRate	设置列表模式指数模板的指数值
延时器指令	指令描述
:DELAy[:STATe]	启动/关闭延时器
:DELAy:STARt	设置延时器的起始组
:DELAy:GRouPs	设置延时器的组数
:DELAy:CYCLEs	设置延时器的循环次数
:DELAy:ENDState	设置延时器的停止状态
:DELAy:STOP	设置延时器的停止条件
:DELAy:PARAMeter	设置延时器的组参数
:DELAy:GENerate:STAT	设置延时器模板生成的状态码
:DELAy:GENerate:FIX	设置延时器模板的固定时长
:DELAy:GENerate:INC	设置延时器模板的递增时长
:DELAy:GENerate:DEC	设置延时器模板的递减时长
预设值指令	指令描述
:PRESet#[:APPLy]	将预设组参数应用
:PRESet#:SET:VOLTage	设置预设组的电压
:PRESet#:SET:CURRent	设置预设组的电流
:PRESet#:SET:OVP	设置预设组的过压保护值 OVP
:PRESet#:SET:OCP	设置预设组的过流保护值 OCP
:PRESet#:SET:TIMer	设置预设组的定时关闭功能开关
:PRESet#:SET:TIMer:DATA	设置预设组的定时关闭功能的定时时长
监测器指令	指令描述
:MONItor[:STATe]	启动或停止监测器 Monitor
:MONItor:VOLTage	设置监测器的电压条件
:MONItor:CURRent	设置监测器的电流条件
:MONItor:POWER	设置监测器的电压表条件
:MONItor:DVM	设置监测器的功率条件
:MONItor:LOGic	设置监测器的条件判断逻辑
:MONItor:STOPway	设置监测器的停止处理方式
系统指令	指令描述
:SYSTem:REMOte	设置电源为远程控制模式
:SYSTem:LOCAl	设置电源为面板控制模式
:SYSTem:BEEPer:TEST	测试蜂鸣器,响一声(蜂鸣器必须开启)
:SYSTem:BEEPer[:STATe]	打开或关闭操作蜂鸣器提示音

:SYSTem:BRIGhtness	设置电源的屏幕亮度
:SYSTem:COMMunicate:COM:BAUD	设置 RS232 与 RS485 的通讯波特率
:SYSTem:COMMunicate:COM:PROTOcol	设置 RS232 与 RS485 的通讯协议
:SYSTem:COMMunicate:COM:ADDReSS	设置 RS232 与 RS485 的多机通讯地址
:SYSTem:COMMunicate:LAN:APPLy	保存并应用已设置的网络参数
:SYSTem:COMMunicate:LAN:DHCP[:STATe]	打开/关闭网络 DHCP 自动分配地址功能
:SYSTem:COMMunicate:LAN:IPAdDress	设置网络 LAN 口的 IP 地址
:SYSTem:COMMunicate:LAN:SMASK	设置网络 LAN 口的子网掩码
:SYSTem:COMMunicate:LAN:GATEway	设置网络 LAN 口的网关地址
:SYSTem:ERRor[:NEXT]?	获取 SCPI 的错误代码和字符串
:SYSTem:ERRor:COUNT?	获取 SCPI 的错误代码和字符串
:SYSTem:VERSIon?	获取 SCPI 的版本号
SCPI 功能指令	指令描述
*IDN?	查询仪器信息
*STB?	查询状态字节事件寄存器
*SRE	设置 SCPI 状态字节事件使能寄存器
*ESR?	查询 SCPI 事件寄存器
*ESE	设置 SCPI 事件使能寄存器
:STATus:QUEStionable[:EVENT]?	查询 SCPI 的 QUSE 状态事件寄存器
:STATus:QUEStionable:CONDition?	查询 SCPI 的 QUSE 状态寄存器,返回当前状态
:STATus:QUEStionable:ENABle	设置 SCPI 的 QUSE 状态事件使能寄存器

3.2 指令解析

:MEASure:VOLTagE?

功能 测量当前输出电压

语法 :MEASure:VOLTagE?

范例 :MEASure:VOLTagE? -> <OutVoltage>

:MEASure:CURREnt?

功能 测量当前输出电流

语法 :MEASure:CURREnt?

范例 :MEASure:CURREnt? -> <OutCurrent>

:MEASure:POWEr?

功能 测量当前输出功率

语法 :MEASure:POWEr?

范例 :MEASure:POWEr? -> <OutPower>

:MEASure:DVM?

功能 测量当前电压表通道的电压

语法 :MEASure:DVM?

范例 :MEASure:DVM? -> <Dvm>

:MEASure:ALL?

功能 测量当前输出电压、电流和功率

语法 :MEASure:ALL?

范例 :MEASure:ALL? -> <OutVoltage>,<OutCurrent>,<OutPower>

[[:SOURce]:VOLTage[:LEVel]

功能 设置输出电压值

语法 [[:SOURce]:VOLTage[:LEVel] {<Value>|MINimum|MAXimum}

[[:SOURce]:VOLTage[:LEVel]?

<Value>=输出电压设定值,单位:V

范例 :VOLTage <Value>

:VOLTage? -> <Value>

[[:SOURce]:CURRent [:LEVel]

功能 设置输出电流值

语法 [[:SOURce]:CURRent [:LEVel] {<Value>|MINimum|MAXimum}

[[:SOURce]:CURRent [:LEVel]?

<Value>=输出电流设定值,单位:A

范例 :CURRent <Value>

:CURRent? -> <Value>

[[:SOURce]:VOLTage:PROTection[:LEVel]

功能 设置过压保护功能保护值

语法 [[:SOURce]:VOLTage:PROTection[:LEVel] {<Value>|MINimum|MAXimum}

[[:SOURce]:VOLTage:PROTection[:LEVel]?

<Value>=过压保护功能保护值,单位:V

范例 :VOLTage:PROTection <Value>

:VOLTage:PROTection? -> <Value>

[[:SOURce]:CURRent:PROTection[:LEVel]

功能 设置过流保护功能保护值

语法 [:SOURce]:CURRent:PROTection[:LEVel] {<Value>|MINimum|MAXimum}

[:SOURce]:CURRent:PROTection[:LEVel]?

<Value>=过流保护功能保护值,单位:A

范例 :CURRent:PROTection <Value>

:CURRent:PROTection? -> <Value>

[:SOURce]:VOLTag:e:PROTection:STATe

功能 设置过压保护功能开关

语法 [:SOURce]:VOLTag:e:PROTection:STATe {<Boolean>}

[:SOURce]:VOLTag:e:PROTection:STATe?

<Boolean>=ON|OFF|0|1

范例 :VOLTag:e:PROTection:STATe <Boolean>

:VOLTag:e:PROTection:STATe? -> <Boolean>

[:SOURce]:CURRent:PROTection:STATe

功能 设置过流保护功能开关

语法 [:SOURce]:CURRent:PROTection:STATe {<Boolean>}

[:SOURce]:CURRent:PROTection:STATe?

<Boolean>=ON|OFF|0|1

范例 :CURRent:PROTection:STATe <Boolean>

:CURRent:PROTection:STATe? -> <Boolean>

[:SOURce:]VOLTag:e:PROTection:TRIPed?

功能 查询过压保护状态触发状态

语法 [:SOURce:]VOLTag:e:PROTection:TRIPed? <State>

<State>=1:已触发 0:未触发

范例 :VOLTag:e:PROTection:TRIPed? -> <State>

[:SOURce:]VOLTag:e:PROTection:CLEar

功能 清除过压保护状态触发状态

语法 [:SOURce:]VOLTag:e:PROTection:CLEar

[:SOURce:]CURRent:PROTection:TRIPed?

功能 查询过流保护状态触发状态

语法 [:SOURce:]CURRent:PROTection:TRIPed? <State>

<State>=1:已触发 0:未触发

范例 :CURRent:PROTection:TRIPed? -> <State>

[:SOURce:]CURRent:PROTection:CLEar

功能 清除过流保护状态触发状态

语法 [:SOURce:]CURRent:PROTection:CLEar

[:SOURce:]VOLTage:STEP

功能 设置电压步进值

语法 [:SOURce:]VOLTage:STEP {<Value>}

[:SOURce:]VOLTage:STEP?

<Value>=电压步进值,单位:V

范例 :VOLTage:STEP <Value>

:VOLTage:STEP? -> <Value>

[:SOURce:]VOLTage:UP

功能 按照步进值提高设置电压

语法 [:SOURce:]VOLTage:UP

[:SOURce:]VOLTage:DOWN

功能 按照步进值降低设置电压

语法 [:SOURce:]VOLTage:DOWN

[:SOURce:]CURRent:STEP

功能 设置电流步进值

语法 [:SOURce:]CURRent:STEP {<Value>}

[:SOURce:]CURRent:STEP?

<Value>=电流步进值,单位:A

范例 :CURRent:STEP <Value>

:CURRent:STEP? -> <Value>

[:SOURce:]CURRent:UP

功能 按照步进值提高设置电流

语法 [:SOURce:]CURRent:UP

[:SOURce:]CURRent:DOWN

功能 按照步进值降低设置电流

语法 [:SOURce:]CURRent:DOWN

[[:SOURce:]:VOLTage:SLEW:RISing

功能 设置电压的上升斜率

语法 [[:SOURce:]:VOLTage:SLEW:RISing {<Value>}

[[:SOURce:]:VOLTage:SLEW:RISing?

<Value>=电压斜率值,单位:V/s

范例 :VOLTage:SLEW:RISing <Value>

:VOLTage:SLEW:RISing? -> <Value>

[[:SOURce:]:VOLTage:SLEW:FALLing

功能 设置电压的下降斜率

语法 [[:SOURce:]:VOLTage:SLEW:FALLing {<Value>}

[[:SOURce:]:VOLTage:SLEW:FALLing ?

<Value>=电压斜率值,单位:V/s

范例 :VOLTage:SLEW:FALLing <Value>

:VOLTage:SLEW:FALLing? -> <Value>

[[:SOURce:]:CURRent:SLEW:RISing

功能 设置电流的上升斜率

语法 [[:SOURce:]:CURRent:SLEW:RISing {<Value>}

[[:SOURce:]:CURRent:SLEW:RISing?

<Value>=电流斜率值,单位:A/s

范例 :CURRent:SLEW:RISing <Value>

:CURRent:SLEW:RISing? -> <Value>

[[:SOURce:]:CURRent:SLEW:FALLing

功能 设置电流的下降斜率

语法 [[:SOURce:]:CURRent:SLEW:FALLing {<Value>}

[[:SOURce:]:CURRent:SLEW:FALLing ?

<Value>=电流斜率值,单位:A/s

范例 :CURRent:SLEW:FALLing <Value>

:CURRent:SLEW:FALLing? -> <Value>

:OUTPut:MODE

功能 设置电源的斜率工作模式

语法 :OUTPut:MODE {<Mode>}

:OUTPut:MODE?

<Mode>=NORMAL:普通模式 VSR:电压斜率,CV 优先; ISR:电流斜率,CC 优先

范例 :OUTPut:MODE <Mode>

:OUTPut:MODE? -> <Mode>

:OUTPut[:STATe]

功能 打开或关闭电源输出

语法 :OUTPut[:STATe] {<Boolean>}

:OUTPut[:STATe]?

<Boolean>=ON|OFF|0|1

范例 :OUTPut ON

:OUTPut? -> <Boolean>

:OUTPut:CVCC?

功能 查询电源当前输出状态

语法 :OUTPut:CVCC? {CC|CV}

CC=恒流模式 CV=恒压模式

范例 :OUTPut:CVCC? -> <CC|CV>

:OUTPut:OVP:VALue

功能 设置过压保护功能保护值

语法 :OUTPut:OVP:VALue {<Value>|MINimum|MAXimum}

:OUTPut:OVP:VALue?

<Value>=过压保护功能保护值,单位:V

范例 :OUTPut:OVP:VALue <Value>

:OUTPut:OVP:VALue? -> <Value>

:OUTPut:OCP:VALue

功能 设置过流保护功能保护值

语法 :OUTPut:OCP:VALue {<Value>|MINimum|MAXimum}

:OUTPut:OCP:VALue?

<Value>=过流保护功能保护值,单位:A

范例 :OUTPut:OCP:VALue <Value>

:OUTPut:OCP:VALue? -> <Value>

:OUTPut:OVP[:STATe]

功能 设置过压保护功能开关

语法 :OUTPut:OVP[:STATe] {<Boolean>}

:OUTPut:OVP[:STATe]?

<Boolean>=ON|OFF|0|1

范例 :OUTPut:OVP <Boolean>

:OUTPut:OVP? -> <Boolean>

:OUTPut:OCP[:STATe]

功能 设置过流保护功能开关

语法 :OUTPut:OCP[:STATe] {<Boolean>}

:OUTPut:OCP[:STATe]?

<Boolean>=ON|OFF|0|1

范例 :OUTPut:OCP <Boolean>

:OUTPut:OCP? -> <Boolean>

:OUTPut:OVP:TRIPed?

功能 查询过压保护状态触发状态

语法 :OUTPut:OVP:TRIPed? -><State>

<State>=1:已触发 0:未触发

范例 :OUTPut:OVP:TRIPed? -> <State>

:OUTPut:OVP:CLEar

功能 清除过压保护状态触发状态

语法 :OUTPut:OVP:CLEar

:OUTPut:OCP:TRIPed?

功能 查询过流保护状态触发状态

语法 :OUTPut:OCP:TRIPed? -><State>

<State>=1:已触发 0:未触发

范例 :OUTPut:OCP:TRIPed? -> <State>

:OUTPut:OCP:CLEar

功能 清除过流保护状态触发状态

语法 :OUTPut:OCP:CLEar

:OUTPut:TIMer

功能 设置定时关闭功能开关

语法 :OUTPut:TIMer {<Boolean>}

:OUTPut:TIMer?

<Boolean>=ON|OFF|0|1

范例 :OUTPut:TIMer <Boolean>

:OUTPut:TIMer? -> <Boolean>

:OUTPut:TIMer:DATA

功能 设置定时关闭功能定时时长

语法 :OUTPut:TIMer:DATA {<Value>}

:OUTPut:TIMer:DATA?

<Value>=定时时长,单位:秒 s

范例 :OUTPut:TIMer <Value>

:OUTPut:TIMer? -> <Value>

:LISTout[:STATe]

功能 设置 List 列表模式开关与查询列表模式状态

语法 :LISTout[:STATe] {<Boolean>}

:LISTout[:STATe]?->{<Boolean>,<time>,<curGroup>,<endGroup>,<remainCycle>,<endState>}

<Boolean>=ON|OFF|0|1

<time>=当前组剩余时间,单位:秒 s

<curGroup>=当前组

<endGroup>=终止组

<remainCycle>=剩余次数

<endState>=终止状态,OFF:关闭输出;LAST:维持输出

范例 :LISTout <Boolean>

:LISTout? -> ON,0.1,000,009,00000,OFF

:LISTout:BASE

功能 设置 List 列表模式的基础参数

语法 :LISTout:BASE {<Start>,<Groups>,<Cycle>,<endState>}

:LISTout:BASE?->{<Start>,<Groups>,<Cycle>,<endState>}

<Start>=起始组

<Groups>=组数

<Cycle>=循环次数,0 为无限循环

<endState>=终止状态,OFF:关闭输出;LAST:维持输出

范例 :LISTout:BASE <Start>,<Groups>,<Cycle>,<endState>

:LISTout:BASE? -><Start>,<Groups>,<Cycle>,<endState>

:LISTout:PARAMeter

功能 设置 List 列表模式的组参数

语法 :LISTout:PARAMeter {<No>,<Volt>,<Curr>,<Time>}

:LISTout:PARAMeter?->{<No>,<Volt>,<Curr>,<Time>}

<No>=组序号

<Volt>=电压,单位:V

<Curr>=电流,单位:A

<Time>=时长,单位:s

范例 :LISTout:PARAMeter <No>,<Volt>,<Curr>,<Time>

:LISTout:PARAMeter? 0,2

->#226000,10.000,12.000, 100.0;#226001,20.000,07.539, 2.0;

#226 代表 26 占用 2 个字节,数据段有 26 个数据{000,10.000,12.000, 100.0;},
表示序号 0 电压 10V,电流 12A,100 秒,具体格式可参考第一章 数据返回小节

:LISTout:TEMPlEt:CONStRuct

功能 按照设定模板参数构建列表组参数构建组参数

语法 :LISTout:TEMPlEt:CONStRuct

:LISTout:TEMPlEt:SELEct

功能 设置列表模式的模板类型

语法 :LISTout:TEMPlEt:SELEct {SINE|PULSE|RAMP|UP|DN|UPDN|RISE|FALL}

:LISTout:TEMPlEt:SELEct?->{SINE|PULSE|RAMP|UP|DN|UPDN|RISE|FALL}

SINE: 正弦模板; **PULSE:** 脉冲模板; **RAMP:** 斜坡模板; **UP:** 阶梯上升模板;

DN: 阶梯下降模板; **UPDN:** 上下阶梯模板; **RISE:** 指数上升模板; **FALL:** 指数下降模板

范例 :LISTout:TEMPlEt:SELEct SINE

:LISTout:TEMPlEt:SELEct?->SINE

:LISTout:TEMPlEt:OBJEct

功能 设置列表模式的模板对象

语法 :LISTout:TEMPlEt:OBJEct {V|C}

:LISTout:TEMPlEt:OBJEct?->{V|C}

V=电压模板 **C=电流模板**

范例 :LISTout:TEMPlEt:OBJEct V

:LISTout:TEMPlEt:OBJEct?->V

:LISTout:TEMPlEt:STARt

功能 设置列表模式的构建起始组

语法 :LISTout:TEMPlEt:STARt {<Start>}

:LISTout:TEMPlEt:STARt?->{<Start>}

<Start>=模板构建起始组

范例 :LISTout:TEMPlEt:STARt 0

:LISTout:TEMPlEt:STARt?->0

:LISTout:TEMPlEt:POINtS

功能 设置列表模式的构建组数

语法 :LISTout:TEMPlEt:POINtS <Groups>

:LISTout:TEMPlEt:POINtS?->{<Groups>}

<Groups>=模板构建组数

范例 :LISTout:TEMPlet:POINTs 64
:LISTout:TEMPlet:POINTs?->64

:LISTout:TEMPlet:MAXValue

功能 设置列表模式的构建对象最大值

语法 :LISTout:TEMPlet:MAXValue {<Value>}
:LISTout:TEMPlet:MAXValue?->{<Value>}

<Value>=模板构建的对象最大值

范例 :LISTout:TEMPlet:MAXValue 10.000
:LISTout:TEMPlet:MAXValue?->10.000

:LISTout:TEMPlet:MINValue

功能 设置列表模式的构建对象最小值

语法 :LISTout:TEMPlet:MINValue {<Value>}
:LISTout:TEMPlet:MINValue?->{<Value>}

<Value>=模板构建的对象最小值

范例 :LISTout:TEMPlet:MINValue 1.000
:LISTout:TEMPlet:MINValue?->1.000

:LISTout:TEMPlet:INTERval

功能 设置列表模式构建模板的时间长度

语法 :LISTout:TEMPlet:INTERval {<Value>}
:LISTout:TEMPlet:INTERval?->{<Value>}

<Value>=模板构建的时间长度,单位:s,最小分辨率 0.1s

范例 :LISTout:TEMPlet:INTERval 1.0
:LISTout:TEMPlet:INTERval?->1.0

:LISTout:TEMPlet:INVErt

功能 设置列表模式构建模板的反相模式

语法 :LISTout:TEMPlet:INVErt {<Boolean>}
:LISTout:TEMPlet:INVErt?->{<Boolean>}

<Boolean>=ON 开启反相,OFF 关闭反相

(仅在模板类型为:SINE 正弦或 PULSE 脉冲或 RAMP 斜坡模板时可设置,否则无效)

范例 :LISTout:TEMPlet:INVErt ON
:LISTout:TEMPlet:INVErt?->ON

:LISTout:TEMPlet:WIDTh

功能 设置列表模式脉冲模板的脉冲宽度

语法 :LISTout:TEMPlet:WIDTh {<Width>}

:LISTout:TEMPlet:WIDTh?->{<Width>}

**<Width>=脉冲宽度,单位:s,取值范围 0.1~(Period-0.1),Period 为脉冲周期
(仅在模板类型为:PULSE 脉冲有效;否则无效)**

范例 :LISTout:TEMPlet:WIDTh 5.0

:LISTout:TEMPlet:WIDTh?->5.0

:LISTout:TEMPlet:PERIod

功能 设置列表模式脉冲模板的脉冲周期

语法 :LISTout:TEMPlet:PERIod {<Period>}

:LISTout:TEMPlet:PERIod?->{<Period>}

**<Period>=脉冲周期,单位:s,取值范围(Width+0.1),99999.9,Width 为脉冲宽度
(仅在模板类型为:PULSE 脉冲有效;否则无效)**

范例 :LISTout:TEMPlet:PERIod 5.0

:LISTout:TEMPlet:PERIod?->5.0

:LISTout:TEMPlet:SYMMetry

功能 设置列表模式斜坡模板的对称度

语法 :LISTout:TEMPlet:SYMMetry {<Value>}

:LISTout:TEMPlet:SYMMetry?->{<Value>}

**<Value>=斜坡模板对称度,取值范围 0~100
(仅在模板类型为:RAMP 斜坡有效;否则无效)**

范例 :LISTout:TEMPlet:SYMMetry 50.0

:LISTout:TEMPlet:SYMMetry?->50.0

:LISTout:TEMPlet:EXPRate

功能 设置列表模式指数模板的指数值

语法 :LISTout:TEMPlet:EXPRate {<Value>}

:LISTout:TEMPlet:EXPRate?->{<Value>}

**<Value>=指数模板的指数值,取值范围 0~10
(仅在模板类型为:RISE 指数上升/FALL 指数下降有效;否则无效)**

范例 :LISTout:TEMPlet:EXPRate 5.0

:LISTout:TEMPlet:EXPRate?->5.0

:DELAY[:STATe]

功能 设置延时器开关与查询延时器状态

语法 :DELAY[:STATe] {<Boolean>}

:DELAY[:STATe]?->{<Boolean>,<time>,<curGroup>,<endGroup>,<remainCycle>,<endState>}

<Boolean>=ON|OFF|0|1

<time>=当前组剩余时间,单位:秒 s

<curGroup>=当前组

<endGroup>=终止组

<remainCycle>=剩余次数

<endState>=终止状态,OFF:关闭输出;LAST:维持输出;ON:打开输出

范例 :DELAY <Boolean>

:DELAY? -> ON, 0.4,010,016,99999,OFF

:DELAY:START

功能 设置延时器运行起始组

语法 :DELAY:START {<Start>}

:DELAY:START?->{<Start>}

<Start>=延时器运行起始组

范例 :DELAY:START 0

:DELAY:START?->0

:DELAY:GROUPs

功能 设置延时器运行组数

语法 :DELAY:GROUPs {<Groups>}

:DELAY:GROUPs?->{<Groups>}

<Groups>=延时器运行组数

范例 :DELAY:GROUPs 64

:DELAY:GROUPs?->64

:DELAY:CYCLEs

功能 设置延时器循环次数

语法 :DELAY:CYCLEs {<Cycles>}

:DELAY:CYCLEs?->{<Cycles>}

<Cycles>=延时器运行组数,0 代表无限循环

范例 :DELAY:CYCLEs 0

:DELAY:CYCLEs?->0

:DELAY:ENDState

功能 设置延时器的停止状态

语法 :DELAY:ENDState {<endState>}

:DELAY:ENDState?->{<endState>}

<endState>=终止状态,OFF:关闭输出;LAST:维持输出 ON:打开输出

范例 :DELAY:ENDState OFF

:DELAY:ENDState?->OFF

:DELAY:STOP

功能 设置延时器的停止条件

语法 :DELAY:STOP {NONE|<V|>V|<C|>C|<P|>P [,<Value>]}

:DELAY:STOP?->{NONE|<V|>V|<C|>C|<P|>P [,<Value>]}

<Value>=条件参数,可省略,省略时不改变数值

范例 :DELAY:STOP >V,15.000

:DELAY:STOP?->V,15.000

:DELAY:PARAMeter

功能 设置延时器的组参数

语法 :DELAY:PARAMeter {<No>,<Boolean>,<Time>}

:DELAY:PARAMeter?->{<No>,<Boolean>,<Time>}

<No>=组序号

<Boolean>=输出状态,ON:打开输出;OFF:关闭输出

<Time>=时长,单位 s,最小分辨率 0.1s

范例 :DELAY:PARAMeter 0,ON,1.0

:DELAY:PARAMeter? 0,1

->#215000,OFF 1.0;

#215 代表 15 占用 2 个字节,数据段有 15 个数据{000,OFF 1.0;},

表示序号 0 打开输出,1.0 秒,具体格式可参考第一章 数据返回小节

:DELAY:GENerate:STAT

功能 以指定状态码生成延时器的组参数输出状态

语法 :DELAY:GENerate:STAT {<Start>,<Groups>, 01P|10P}

:DELAY:GENerate?->STAT,{<Start>,<Groups>, 01P|10P}

<Start>=生成起始组

<Groups>=生成组数

<01P|10P>= 01P:先关后开;10P:先开后关

范例 :DELAY:GENerate:STAT 0,64,01P

:DELAY:GENerate?->STAT,0,64,01P

:DELAY:GENerate:FIX

功能 以固定开关时长生成延时器的组参数时长

语法 :DELAY:GENerate:FIX {<Start>,<Groups>, <Time_on>,<Time_off>}
:DELAY:GENerate?->FIX,{<Start>,<Groups>, <Time_on>,<Time_off>}

<Start>=生成起始组

<Groups>=生成组数

<Time_on>=打开时长,单位 s,最小分辨率 0.1s

<Time_off>=关闭时长,单位 s,最小分辨率 0.1s

范例 :DELAY:GENerate:FIX 0,10,5,10
:DELAY:GENerate?->FIX,0,10,5,10

:DELAY:GENerate:INC

功能 以递增时长生成延时器的组参数时长

语法 :DELAY:GENerate:INC {<Start>,<Groups>, <Time_base>,<Time_step>}
:DELAY:GENerate?->INC,{<Start>,<Groups>, <Time_base>,<Time_step>}

<Start>=生成起始组

<Groups>=生成组数

<Time_base>=时长基数,单位 s,最小分辨率 0.1s

<Time_step>=时长递增值,单位 s,最小分辨率 0.1s

范例 :DELAY:GENerate:INC 0,10,10,2
:DELAY:GENerate?->INC,0,10,10,2

:DELAY:GENerate:DEC

功能 以递减时长生成延时器的组参数时长

语法 :DELAY:GENerate:DEC {<Start>,<Groups>, <Time_base>,<Time_step>}
:DELAY:GENerate?->DEC,{<Start>,<Groups>, <Time_base>,<Time_step>}

<Start>=生成起始组

<Groups>=生成组数

<Time_base>=时长基数,单位 s,最小分辨率 0.1s

<Time_step>=时长递减值,单位 s,最小分辨率 0.1s

范例 :DELAY:GENerate:DEC 0,10,100,1
:DELAY:GENerate?->DEC,0,10,100,1

:PRESet#[:APPLy]

功能 将指定预设值参数应用到输出参数

语法 :PRESet#[:APPLy]

#取值范围 0~7

范例 :PRESet0:APPLy

:PRESet#:SET:VOLTage

功能 设定指定预设值组的电压参数

语法 :PRESet#:SET:VOLTage {<Volt>}

:PRESet#:SET:VOLTage?->{<Volt>}

<Volt>=电压参数,单位:V

#取值范围 0~7

范例 :PRESet0:SET:VOLTage 5.000

:PRESet0:SET:VOLTage?->5.000

:PRESet#:SET:CURRent

功能 设定指定预设值组的电流参数

语法 :PRESet#:SET:CURRent {<Curr>}

:PRESet#:SET:CURRent?->{<Curr>}

<Curr>=电流参数,单位:A

#取值范围 0~7

范例 :PRESet7:SET:CURRent 5.000

:PRESet7:SET:CURRent?->5.000

:PRESet#:SET:OVP

功能 设定指定预设值组的过压保护 OVP 功能

语法 :PRESet#:SET:OVP {<Boolean>,<Volt>}

:PRESet#:SET:OVP?->{<Boolean>,<Volt>}

<Boolean>=ON:打开 OVP 功能 OFF:关闭 OVP 功能

<Volt>=电压参数,单位:V

#取值范围 0~7

范例 :PRESet1:SET:OVP ON,62.000

:PRESet1:SET:OVP?->ON,62.000

:PRESet#:SET:OCP

功能 设定指定预设值组的过流保护 OCP 功能

语法 :PRESet#:SET:OCP {<Boolean>,<Curr>}

:PRESet#:SET:OCP?->{<Boolean>,<Curr>}

<Boolean>=ON:打开 OCP 功能 OFF:关闭 OCP 功能

<Curr>=电流参数,单位:A

#取值范围 0~7

范例 :PRESet1:SET:OCP ON,15.500

:PRESet1:SET:OCP?->ON,15.500

:PRESet#:SET:TIMer

功能 设定指定预设值组的定时关闭功能开关

语法 :PRESet#:SET:TIMer {<Boolean>}

:PRESet#:SET:TIMer?->{<Boolean>}

<Boolean>=ON:打开定时关闭功能 OFF:关闭定时关闭功能

#取值范围 0~7

范例 :PRESet0:SET:TIMer ON

:PRESet0:SET:TIMer?->ON

:PRESet#:SET:TIMer:DATA

功能 设定指定预设值组的定时关闭功能时间

语法 :PRESet#:SET:TIMer:DATA {<Time>}

:PRESet#:SET:TIMer:DATA?->{<Time>}

<Time>=定时关闭时长,单位:s

#取值范围 0~7

范例 :PRESet0:SET:TIMer:DATA 5.000

:PRESet0:SET:TIMer:DATA?->5.000

:PRESet#:SET:TIMer:DATA

功能 设定指定预设值组的定时关闭功能时间

语法 :PRESet#:SET:TIMer:DATA {<Time>}

:PRESet#:SET:TIMer:DATA?->{<Time>}

<Time>=定时关闭时长,单位:s

#取值范围 0~7

范例 :PRESet0:SET:TIMer:DATA 5.000

:PRESet0:SET:TIMer:DATA?->5.000

:MONItor[:STATe]

功能 启动或停止监测器

语法 :MONItor[:STATe] {<Boolean>}
:MONItor[:STATe]?->{<Boolean>}
<Boolean>=ON:打开 OFF:关闭

范例 :MONItor:STATe ON
:MONItor:STATe?->ON

:MONItor:VOLTagE

功能 设置监测器中的输出电压条件

语法 :MONItor:VOLTagE {<V|>V|NONE ,[<Volt>]}
:MONItor:VOLTagE?->{<V|>V|NONE ,[<Volt>]}
逻辑条件:

<V:当输出电压小于设定参数时为真;

>V:当输出电压大于设定参数时为真;

NONE:一直为真;

<Volt>=电压参数,可忽略,忽略则不改变设定参数;当条件为 NONE 时,不返回

范例 :MONItor:VOLTagE <V, 15.000
:MONItor:VOLTagE?-><V, 15.000

:MONItor:CURRent

功能 设置监测器中的输出电流条件

语法 :MONItor:CURRent {<C|>C|NONE ,[<Curr>]}
:MONItor:CURRent?->{<C|>C|NONE ,[<Curr>]}
逻辑条件:

<C:当输出电流小于设定参数时为真;

>C:当输出电流大于设定参数时为真;

NONE:一直为真;

<Curr>=电流参数,可忽略,忽略则不改变设定参数;当条件为 NONE 时,不返回

范例 :MONItor:CURRent <C, 1.000
:MONItor:CURRent?-><C, 1.000

:MONItor:POWER

功能 设置监测器中的输出功率条件

语法 :MONItor:POWER {<P>P|NONE ,[<Power>]}
:MONItor:POWER?->{<P>P|NONE ,[<Power>]}

逻辑条件:

<V:当输出功率小于设定参数时为真;

>V:当输出功率大于设定参数时为真;

NONE:一直为真;

<Power>=功率参数,可忽略,忽略则不改变设定参数;当条件为 NONE 时,不返回

范例 :MONItor:POWER <P, 15.000
:MONItor:POWER?-><P, 15.000

:MONItor:DVM

功能 设置监测器中的数字电压表 DVM 电压条件

语法 :MONItor:DVM {<DVM>DVM|NONE ,[<DVM>]}
:MONItor:DVM?->{<DVM>DVM|NONE ,[<DVM>]}

逻辑条件:

<DVM:当 DVM 电压小于设定参数时为真;

>DVM:当 DVM 电压大于设定参数时为真;

NONE:一直为真;

<DVM>=DVM 电压参数,可忽略,忽略则不改变设定参数;当条件为 NONE 时,不返回

范例 :MONItor:DVM >DVM, 10.000
:MONItor:DVM?->>DVM, 10.000

:MONItor:LOGic

功能 启动或停止监测器

语法 :MONItor:LOGic {<No>,<Logic>}
:MONItor:LOGic? {<No>}->{<Logic>}

<No>=逻辑符号编号,取值范围 1~3

<Logic>=逻辑,AND:与运算&;OR:或运算 |

范例 :MONItor:LOGic 1,AND
:MONItor:LOGic? 1->AND

:MONItor:STOPway

功能 设定监测器的触发处理方式

语法 :MONItor:STOPway {<Type>,<Boolean>}

:MONItor:STOPway? {<Type>}->{<Boolean>}

<Type>=OUTOFF:关闭输出 MSG:弹窗提示 BEEPER:蜂鸣器提示

<Boolean>=ON:打开 OFF:关闭

范例 :MONItor:STOPway OUTOFF, ON

:MONItor:STOPway? OUTOFF->ON

:MONItor:STOPway MSG, OFF

:MONItor:STOPway? MSG->OFF

:SYSTem:REMOte

功能 设置电源为远程控制模式,该模式下锁定按键,手动解锁后则返回面板控制模式

语法 :SYSTem:REMOte

:SYSTem:LOCAl

功能 设置电源为面板控制模式

语法 :SYSTem:LOCAl

:SYSTem:BEEPer:TEST

功能 测试蜂鸣器,发送该指令蜂鸣器响一声,仅当开启蜂鸣器后有效

语法 :SYSTem:BEEPer:TEST

:SYSTem:BEEPer[:STATe]

功能 打开或关闭蜂鸣器提示音

语法 :SYSTem:BEEPer[:STATe] {<Boolean>}

:SYSTem:BEEPer[:STATe]?->{<Boolean>}

<Boolean>=ON:打开提示音 OFF:关闭提示音

范例 :SYSTem:BEEPer:STATe ON

:SYSTem:BEEPer:STATe?->ON

:SYSTem:BRIGhtness

功能 设置显示屏的背光亮度

语法 :SYSTem:BRIGhtness {<Value>}

:SYSTem:BRIGhtness?->{<Value>}

<Value>=屏幕亮度,取值范围:20~100

范例 :SYSTem:BRIGhtness 50

:SYSTem:BRIGhtness?->50

:SYSTem:COMMunicate:COM:BAUD

功能 设置 RS232 与 RS485 接口的通讯波特率

语法 :SYSTem:COMMunicate:COM:BAUD {<BaudRate>}
:SYSTem:COMMunicate:COM:BAUD?->{<BaudRate>}

<BaudRate>=通讯波特率,取值:9600,14400,19200,38400,57600,115200

范例 :SYSTem:COMMunicate:COM:BAUD 115200
:SYSTem:COMMunicate:COM:BAUD?->115200

:SYSTem:COMMunicate:COM:PROTOcol

功能 设置 RS232 与 RS485 接口的通讯协议

语法 :SYSTem:COMMunicate:COM:PROTOcol {<Protocol>}
:SYSTem:COMMunicate:COM:PROTOcol?->{<Protocol>}

<Protocol>=通讯协议,0:SCPI 协议 1:Mult-SCPI 协议 2:Modbus 协议

范例 :SYSTem:COMMunicate:COM:PROTOcol 0
:SYSTem:COMMunicate:COM:PROTOcol?->0

:SYSTem:COMMunicate:COM:ADDRess

功能 设置 RS232 与 RS485 接口的多机通讯地址

语法 :SYSTem:COMMunicate:COM:ADDRess {<Address>}
:SYSTem:COMMunicate:COM:ADDRess?->{<Address>}

<Address>=通讯地址,取值范围:1~32,0 为广播地址

范例 :SYSTem:COMMunicate:COM:ADDRess 1
:SYSTem:COMMunicate:COM:ADDRess?->1

:SYSTem:COMMunicate:LAN:APPLY

功能 保存并应用已设置的网络参数

语法 :SYSTem:COMMunicate:LAN:APPLY

注意:网络参数设置后不会立即生效,必须执行该指令存储后生效

:SYSTem:COMMunicate:LAN:DHCP[:STATe]

功能 打开或关闭网络的 DHCP 地址分配功能

语法 :SYSTem:COMMunicate:LAN:DHCP[:STATe] {<Boolean>}
:SYSTem:COMMunicate:LAN:DHCP[:STATe]?->{<Boolean>}

<Boolean>=ON:打开 DHCP OFF:关闭 DHCP

范例 :SYSTem:COMMunicate:LAN:DHCP ON
:SYSTem:COMMunicate:LAN:APPLY
:SYSTem:COMMunicate:LAN:DHCP?->ON

:SYSTem:COMMunicate:LAN:IPAdDress

功能 设置 LAN 网络接口的 IP 地址

语法 :SYSTem:COMMunicate:LAN:IPAdDress {<Address>}
:SYSTem:COMMunicate:LAN:IPAdDress?->{<Address>}

<Address>=以"x.x.x.x"为格式的 IP 地址

范例 :SYSTem:COMMunicate:LAN:IPAdDress "192.168.1.100"
:SYSTem:COMMunicate:LAN:APPLy
:SYSTem:COMMunicate:LAN:IPAdDress?->192.168.1.100

:SYSTem:COMMunicate:LAN:SMASK

功能 设置 LAN 网络接口的子网掩码

语法 :SYSTem:COMMunicate:LAN:SMASK {<Address>}
:SYSTem:COMMunicate:LAN:SMASK?->{<Address>}

<Address>=以"x.x.x.x"为格式的子网掩码

范例 :SYSTem:COMMunicate:LAN:SMASK "255.255.255.0"
:SYSTem:COMMunicate:LAN:APPLy
:SYSTem:COMMunicate:LAN:SMASK?->255.255.255.0

:SYSTem:COMMunicate:LAN:GATEway

功能 设置 LAN 网络接口的网关地址

语法 :SYSTem:COMMunicate:LAN:GATEway {<Address>}
:SYSTem:COMMunicate:LAN:GATEway?->{<Address>}

<Address>=以"x.x.x.x"为格式的网关地址

范例 :SYSTem:COMMunicate:LAN:GATEway "192.168.1.1"
:SYSTem:COMMunicate:LAN:APPLy
:SYSTem:COMMunicate:LAN:GATEway?->192.168.1.1

:SYSTem:ERRor[:NEXT]?

功能 获取 SCPI 的错误代码和字符串

语法 :SYSTem:ERRor[:NEXT]?->{<errorNum>,<errorString>}

<errorNum>=错误代码

<errorString>=错误字符串

范例 :SYSTem:ERRor[:NEXT]?

->0,"No error"

:SYSTem:ERRor:COUNT?

功能 获取 SCPI 的错误代码和字符串

语法 :SYSTem:ERRor:COUNT?->{<errorCount>}

<errorCount>=错误队列个数

范例 :SYSTem:ERRor:COUNT?

->0

:SYSTem:VERSion?

功能 获取 SCPI 的版本号

语法 :SYSTem:VERSion?->{<Version>}

<Version>=SCPI 版本号

范例 :SYSTem:VERSion?

1999.0

***IDN?**

功能 查询仪器信息

语法 *IDN?

范例 *IDN?

-> Uni-Trend,UDP6942B,000000000000,1.00.0905

说明 仪器信息返回的格式为 <制造商>,<型号>,<序列号>,<软件版本>

状态字节寄存器 STB

状态位寄存器记录其他寄存器的触发状态,当寄存器&使能寄存器不为 0 时,STB 对应的位将被置位,该寄存器不会锁存,其随事件改变而改变。

Bit	十进制		Definition	描述
0		R01	Not Used	
1	2	PRO	Protection Event Flag	
2	4	QMA	Error/Event queue message available	错误事件消息队列可用
3	8	QES	Questionable status	QUES 状态寄存器触发
4	16	MAV	Message Available	
5	32	ESR	Standard Event Status Register	标准状态寄存器触发
6	64	SRQ	Service Request	
7		OPS	Not Used	

*STB?

功能 查询状态字节事件寄存器,每次读取后自动清除标志位,以十进制返回

语法 *STB?

范例 *STB? -> 4

说明 若返回值为 4,则指示状态字节寄存器被置位 Bit 2;这意味着错误队列非空,也即产生了错误;
具体请查看状态字节寄存器 STB 的描述

*SRE

功能 设置 SCPI 状态字节事件使能寄存器

若 SRE 寄存器&STB 寄存器不为 0,则 STB 寄存器的 SRQ 位为 1;

语法 *SRE <Value>

*SRE?-><Value>

<Value>=状态字节使能寄存器值

范例 *SRE 128

*SRE?->128

事件寄存器 ESR

事件寄存器记录 SCPI 及电源运行的错误事件,当事件发生后,寄存器会锁存事件,仅当查询该寄存器或发送*CLS 清除命令才会使该寄存器清零。

Bit	十进制		Definition	描述
0		OPC	Operation complete	默认为 0
1		Not Used		默认为 0
2	4	QER	Query Error	查询错误
3	8	DER	Device Dependent Error	设备发生错误,如机器自检异常
4	16	EER	Execution Error (e.g. range error)	指令执行错误
5	32	CER	Command error (e.g. syntax error)	指令错误,如语法错误
6		Not Used		默认为 0
7	128	PON	Power On	电源重新上电

*ESR?

功能 查询 SCPI 事件寄存器,每次读取后自动清除标志位,以十进制返回

语法 *ESR?

范例 *ESR? -> 128

说明 若返回值为 128, 则指示 SCPI 事件寄存器被置位 Bit 7; 这意味着发生了上电事件;
具体请查看事件寄存器 ESR 的描述

*ESE

功能 设置 SCPI 事件使能寄存器

若 ESE 寄存器&ESR 寄存器不为 0,则 STB 寄存器的 ESR 位为 1

语法 *SRE <Value>

*SRE?-><Value>

<Value>=状态字节使能寄存器值

范例 *ESE 128

*ESE?->128

查询状态寄存器 QUES

查询状态寄存器提供电源的工作状态信息,比如恒流、恒压工作模式、过温保护 OTP、过压过流保护等状态变化。

Bit	十进制		Definition	描述
0	1	CV	CV Mode	工作于恒压模式 CV
1	2	CC	CC Mode	工作于恒流模式 CC
2		Not Used	Not Used	
3		Not Used	Not Used	
4	16	OTP	Over Temperature	过温保护(OTP)
5		Not Used	Not Used	
6		Not Used	Not Used	
7		Not Used	Not Used	
8	256	OSP	Over Sense	线损补偿过大(OSP)
9	512	OVP	Over Voltage	过压保护(OVP)
10	1024	OCP	Over Current	过流保护(OCP)

:STATus:QUEStionable[:EVENT]?

功能 查询 SCPI 的 QUSE 状态事件寄存器,每次读取后自动清除标志位,以十进制返回

语法 :STATus:QUEStionable[:EVENT]?

范例 :STATus:QUES? -> 512

说明 若返回值为 512,则指示 SCPI 的 QUSE 状态事件寄存器被置位 Bit 9,这意味着发生了 OVP 过压保护事件;**具体请查看查询状态寄存器 QUES 的描述**

:STATus:QUEStionable:CONDition?

功能 查询 SCPI 的 QUSE 状态寄存器,返回当前状态

语法 :STATus:QUEStionable:CONDition?

范例 :STATus:QUES:COND? -> 1

说明 若返回值为 1,则指示 SCPI 的 QUSE 状态寄存器被置位 Bit 0;这意味着当前电源工作于 CV 恒压状态;**具体请查看查询状态寄存器 QUES 的描述**

:STATus:QUEStionable:ENABle

功能 设置 SCPI 的 QUSE 状态事件使能寄存器

若 QUES 状态事件寄存器&QUES 状态事件使能寄存器不为 0,则 STB 寄存器的 QUES 位为 1

语法 :STATus:QUEStionable:ENABle <Value>

:STATus:QUEStionable:ENABle?-><Value>

<Value>=QUES 时间使能寄存器值

范例 :STATus:QUEStionable:ENABle 512

:STATus:QUEStionable:ENABle?->512