Ceyear 思仪

87230 系列

USB 连续波功率探头

程控手册



中电科思仪科技股份有限公司

该手册适用下列型号的 USB 连续波功率探头，基于固件版本 Version 1.0 及以上。

- 87230：9kHz ~ 6GHz
- 87231：10MHz ~ 18GHz
- 87232：50MHz ~ 26.5GHz
- 87233：50MHz ~ 40GHz

除标准配件外的选件如下：

- 英文选件：英文菜单、英文面板等。

版　　本：　C.2　2022年07月，中电科思仪科技股份有限公司
服务咨询：　0532-86889847　400-1684191
技术支持：　0532-86880796
质量监督：　0532-86886614
传　　真：　0532-86889056
网　　址：　www.ceyear.com
电子邮箱：　techbb@ceyear.com
地　　址：　山东省青岛市黄岛区香江路98号
邮　　编：　266555

前　言

非常感谢您选择使用中电科思仪科技股份有限公司研制、生产的 87230 系列 USB 连续波功率探头！本公司产品集高、精、尖于一体，在同类产品中有较高的性价比。

我们将以最大限度满足您的需求为己任，为您提供高品质的测量仪器，同时带给您一流的售后服务。我们的一贯宗旨是“质量优良，服务周到”，提供满意的产品和服务是我们对用户的承诺。

手册编号

AV2.984.1215SC

版本

C.2　2022.07

中电科思仪科技股份有限公司

手册授权

本手册中的内容如有变更，恕不另行通知。本手册内容及所用术语最终解释权属于中电科思仪科技股份有限公司。

本手册版权属于中电科思仪科技股份有限公司，任何单位或个人非经本公司授权，不得对本手册内容进行修改，并且不得以赢利为目的对本手册进行复制、传播，中电科思仪科技股份有限公司保留对侵权者追究法律责任的权利。

产品质保

本产品从出厂之日起保修期为 18 个月。质保期内仪器生产厂家会根据用户要求及实际情况维修或替换损坏部件。具体维修操作事宜以合同为准。

产品质量证明

本产品从出厂之日起确保满足手册中的指标。校准测量由具备国家资质的计量单位予以完成，并提供相关资料以备用户查阅。

质量/环境管理

本产品从研发、制造和测试过程中均遵守质量和环境管理体系。中电科思仪科技股份有限公司已经具备资质并通过 ISO 9001 和 ISO 14001 管理体系。

安全事项

注　意

注意标识代表重要的信息提示，但不会导致危险。它提示用户注意某一操作过程、操作方法或者类似情况。若不能遵守规则或者正确操作，则可能引起的仪器损坏或丢失重要数据。在完全理解和满足所指出的注意条件之后，才可继续下一步。

提　示

提示标识代表信息提示。它提示用户注意仪器或者某一操作过程、操作方法或者类似情况。以引导仪器操作者正确的使用仪器。

目 录

1 手册导航.....1
1.1 关于手册.....1
1.2 关联文档.....1
2 远程控制.....3
2.1 远程控制基础.....3
2.1.1 程控接口.....3
2.1.2 消息.....7
2.1.3 SCPI命令.....7
2.1.4 命令序列与同步.....17
2.1.5 状态报告系统.....19
2.1.6 编程注意事项.....26
2.2 仪器程控端口与配置.....26
2.3 I/O库.....28
2.3.1 I/O库概述.....28
2.3.2 I/O库安装与配置.....29
2.4 校零功率探头.....30
2.5 进行测量.....30
2.6 使用频响偏置表.....30
2.6.1 概述.....31
2.6.2 录入频偏表.....31
2.6.3 选择频偏表.....31
2.6.4 启用频偏表.....32
2.6.5 测量应用.....32
2.6.6 具体应用.....32
2.7 设置显示分辨率.....32
2.8 设置平均.....32
2.9 设置量程.....33
2.10 设置偏置.....33
2.11 设置测量极限.....34
2.12 状态报告.....34

目录

2.13 存储调用 34

3 程控命令 37

3.1 命令说明 37
3.2 通用命令（IEEE488.2 命令） 37
3.3 仪器子系统命令 41
3.3.1 运算子系统命令（CALCulate） 41
3.3.2 校准子系统命令（CALibration） 46
3.3.3 测量子系统命令（CONFigure/FETCh/READ/MEASure） 46
3.3.4 格式子系统命令（FORMat） 49
3.3.5 内存子系统命令（MEMory/MMEMory） 50
3.3.6 传感子系统命令（SENSe） 54
3.3.7 状态子系统命令（STATus） 58
3.3.8 系统子系统命令（SYSTem） 86
3.3.9 触发子系统命令（INITiate/TRIGger） 87
3.3.10 单位子系统命令（UNIT） 89
3.3.11 服务子系统命令（SERVice） 90

4 编程示例 93

4.1 基本操作示例 93
4.1.1 VISA库 93
4.1.2 示例运行环境 94
4.1.3 初始化和设置默认状态 95
4.1.4 发送设置命令 96
4.1.5 读取测量仪器状态 96
4.1.6 命令同步 96
4.2 高级操作示例 98
4.2.1 USBTMC程控示例 98

5 错误说明 103

5.1 错误信息 103
5.1.1 本地错误信息 103
5.1.2 程控错误信息 104
5.2 返修方法 108

5.2.1 联系我们 108
5.2.2 包装与邮寄 108
附 录 111
附录A SCPI命令按子系统分类速查表 111

1 手册导航

本章介绍了 87230 系列 USB 连续波功率探头（以下简称 87230）的程控手册功能、章节构成和主要内容，并介绍了提供给用户使用的仪器关联文档。

- 关于手册 1
- 关联文档 1

1.1 关于手册

本手册介绍了 87230 的远程控制和 SCPI 命令使用方法，同时，为方便用户快速掌握程控编程方法，列举了编程示例并介绍了 I/O 函数库的基本概念。为方便您熟练使用该仪器，请在操作仪器前，仔细阅读本手册，然后按手册指导正确操作。

SCPI（Standard Commands for Programmable Instruments）定义了如何远程控制仪器的标准和方法，也是程控电子测试和测量仪器使用的程控编程语言。SCPI标准基于IEEE-488.2 标准和形式，具体请浏览 http://www.scpiconsortium.org。

该手册详细说明了 87230 的程控命令。

程控手册的章节包括：

- **远程控制**

  概述了仪器远程控制操作方法，目的使用户可以对远程控制操作快速上手。分三部分介绍：程控基础，介绍与程控有关的概念、软件配置、程控端口、SCPI 命令等；仪器端口配置方法，介绍 87230 的程控端口的连接方法和软件配置方法；I/O 函数库，介绍仪器驱动器基本概念及 IVI-COM/IVI-C 驱动的基本安装配置说明。

- **程控命令**

  分类介绍了通用命令、仪器命令及兼容命令，逐条说明SCPI命令的功能、参数并示例。

- **编程示例**

  以文字说明和示例代码的方式给出基本编程示例和高级编程示例，并进行说明，方便用户快速掌握仪器的程控编程方法。

- **错误说明**

  包括错误信息说明及返修方法。

- **附录**

  给出了87230关于程控的必要的参考信息，包括：SCPI命令速查表。

1.2 关联文档

87230 的产品文档包括：

- 快速使用指南

1.2 关联文档

- 用户手册
- 程控手册

快速使用指南

本手册介绍了仪器的配置和启动测量的基本操作方法，目的是：使用户快速了解仪器的特点、掌握基本设置和基础的操作方法。包含的主要章节是：

- 准备使用
- 典型应用
- 获取帮助

用户手册

本手册详细介绍了仪器的功能和操作使用方法，包括：配置、测量、程控和维护等信息。目的：指导用户如何全面的理解产品功能特点及掌握常用的仪器测试方法。包含的主要章节是：

- 手册导航
- 概述
- 使用入门
- 操作指南
- 菜单
- 远程控制
- 故障诊断与返修
- 技术指标和测试方法
- 附录

程控手册

本手册详细介绍了远程编程基础、SCPI 基础、SCPI 命令、编程示例和 I/O 驱动函数库等。目的：指导用户如何快速、全面的掌握仪器的程控命令和程控方法。包含的主要章节是：

- 远程控制
- 程控命令
- 编程示例
- 错误说明
- 附录

2 远程控制

本章介绍了 87230 的程控基础、程控接口与配置方法，并简要的介绍了 I/O 仪器驱动库的概念及分类。以方便用户起步实现远程控制操作。具体内容包括:

- 远程控制基础 ……3
- 仪器程控端口与配置 ……26
- I/O库 ……30
- 校零功率探头 ……30
- 进行测量 ……30
- 使用频响偏置表 ……30
- 设置显示分辨率 ……32
- 设置平均 ……32
- 设置量程 ……32
- 设置偏置 ……33
- 设置测量极限 ……33
- 状态报告 ……34
- 存储调用 ……34

2.1 远程控制基础

- 程控接口 ……3
- 消息 ……7
- SCPI命令 ……7
- 命令序列与同步 ……17
- 状态报告系统 ……19
- 编程注意事项 ……26

2.1.1 程控接口

- LAN接口 ……4
- GPIB接口 ……6
- RS-232接口 ……6
- USB接口 ……6

具备远程控制功能的仪器一般支持两种程控接口：LAN、GPIB，具体型号仪器支持的端口类型由仪器本身功能决定。

程控接口及关联 VISA 寻址字符串说明，如下表:

表2.1　远程控制接口类型和 VISA 寻址字符串

程控接口	VISA 寻址字符串	说 明
LAN (Local Area Network)	**VXI-11协议:** TCPIP::host_address[::LAN_device_name][::INSTR] **原始套接字协议:** TCPIP::host_address::port::SOCKET	控者通过仪器后面板网络端口连接仪器实现远程控制。 具体协议请参考: 2.1.1.1 LAN 接口
GPIB (IEC/IEEE Bus Interface)	GPIB::primary address[::INSTR]	控者通过仪器后面板端口连接仪器实现远程控制。 遵守IEC 625.1/IEEE 418总线接口标准。 具体请参考: 2.1.1.2 GPIB 接口
RS-232 (RecommendedStandard-232)		仪器后面板端口。 具体请参考: 2.1.1.3 RS-232 接口
USB (Universal Serial Bus)	USB::<vendor ID>::<product_ID>::<serial_number>[::INSTR]	仪器后面板端口。 具体请参考: 2.1.1.4 USB 接口

2.1.1.1 LAN 接口

仪器可通过 10Base-T 和 100Base-T 局域网内计算机进行远程控制，各种仪器在局域网内组合成系统，并统一由网内计算机控制。仪器为实现局域网内远程控制，需事先安装端口连接器、网卡和相关网络协议，并配置相关的网络服务，同时网内控者计算机也需事先安装仪器控制软件和 VISA 库。网卡的三种工作模式是:

- 10Mbit/s 以太网 IEEE802.3;
- 100Mbit/s 以太网 IEEE802.3u;
- 1Gbit/s 以太网 IEEE802.3ab。

控者计算机和仪器需通过网口连接到共同的TCP/IP协议网络上。连接计算机和仪器之间的电缆是商用RJ45 电缆（带屏蔽或无屏蔽的 5 类双绞线）。数据传输时，采用数据分组传输方式，LAN传输速度较快。通常，计算机和仪器之间的电缆长度不应超过 100 米（100Base-T 和 10Base-T）。关于LAN通信的更多信息，请参考：http://www.ieee.org。

下面介绍 LAN 接口相关知识:

1) IP 地址

通过局域网对仪器进行远程控制时，应保证网络的物理连接畅通。通过仪器的菜单“本机IP”将地址设置到主控计算机所在的子网内即可。例如: 主控计算机的 IP 地址是 192.168.12.0，则仪器的 IP 地址应设为 192.168.12.XXX，其中 XXX 为 1 ~ 255 之间的数值。

建立网络连接时只需IP地址，VISA寻址字符串形式如下：

TCPIP::host address[::LAN device name][::INSTR] 或

TCPIP::host address::port::SOCKET

其中：

- TCPIP 表示使用的网络协议；
- host address 表示仪器的IP地址或者主机名称，用于识别和控制被控仪器；
- LAN device name 定义了协议和子设备的句柄号（该项可选）；
  - 0号设备选择VXI-11协议；
  - 0号高速LAN仪器选择较新的高速LAN仪器协议；
- INSTR 表示仪器资源类型（该项可选）；
- port 标识套接字端口号；
- SOCKET 表示原始网络套接字资源类。

举例：

- 仪器的IP地址是192.1.2.3，VXI-11协议的有效资源字符串是：
  TCPIP::192.1.2.3::INSTR
- 建立原始套接字连接时可使用：
  TCPIP::192.1.2.3::5000::SOCKET

提 示

程控系统中多仪器识别方法

若网络中连接多台仪器，采用仪器单独的IP地址和关联的资源字符串区分。主控计算机使用各自的VISA资源字符串识别仪器。

2) VXI-11 协议

VXI-11 标准基于 ONC RPC(Open Network Computing Remote Procedure Call)协议，它是 TCP/IP 协议的网络/传输层。TCP/IP 网络协议和相关的网络服务被预先配置好，通信时，这种面向连接的通讯，即遵循按序交换并能识别连接的中断，保证了不丢失信息。

3) 套接字通信

TCP/IP 协议通过局域网套接字在网络中连接信号源。套接字是计算机网络编程中使用的一个基本方法，它使得使用不同硬件和操作系统的应用程序得以在网络中进行通信。这种方法通过端口（port）使仪器与计算机实现双向通信。

套接字是专门编写的一个软件类，里面定义了 IP 地址、设备端口号等网络通信所必需的信息，整合了网络编程中的一些基本操作。在操作系统中安装了打包的库就可以使用套接字。两个常用的套接字库是 UNIX 中应用的伯克利（Berkeley）套接字库和 Windows 中应用的 Winsock 库。

仪器中的套接字通过应用程序接口（API）兼容 Berkeley socket 和 Winsock。此外，还兼容其他标准套接字 API。通过 SCPI 命令控制仪器时，程序中建立的套接字程序发出命令。在使用局域网套接字之前，必须先设置仪器的套接字端口号。仪器的套接字端口号为 5000。

2.1.1.2 GPIB 接口

GPIB 接口是目前仍被广泛的使用的仪器程控接口，通过 GPIB 电缆连接不同种类仪器，与主控计算机组建测试系统。为实现远程控制，主控计算机需要事先安装 GPIB 总线卡，驱动程序以及 VISA 库。通信时，主控计算机首先通过 GPIB 总线地址寻址被控仪器，用户可设置 GPIB 地址和 ID 查询字符串，GPIB 通信语言可默认为 SCPI 命令形式。

GPIB及其相关接口操作在ANSI/IEEE标准 488.1-2003 和ANSI/IEEE标准 488.2-1992 中有详细的定义和描述。具体标准细节请参考IEEE网站：http://www.ieee.org。

GPIB 以字节为单位来处理信息，数据传输速率能够达到 8MBps，因此 GPIB 的数据传输比较快。因数据传输速度受限于设备/系统与计算机之间的距离，GPIB 连接时，需注意以下几点：

- 通过 GPIB 接口最多可组建 15 台仪器；
- 传输电缆总长度不超过 15 米，或者不超过系统中仪器数量的两倍。通常，设备间传输电缆最大长度不能超过 2 米；
- 若并行连接多台仪器，需要使用“或”连接线；
- IEC 总线电缆的终端应该连接仪器或控者计算机。

2.1.1.3 RS-232 接口

RS-232 是实现程控的传统方法，因为一次只发送和接收一位数据，所以传输速率较 GPIB 或 LAN 都慢，目前已不常用。与 GPIB 和 LAN 类似，建立通信时，需设置仪器参数，比如波特率，以实现与主控计算机的参数匹配。RS-232 传输 ASCII 码形式的 SCPI 命令字符。

2.1.1.4 USB 接口

为实现 USB 程控，需要通过 USB B 型口连接计算机和仪器，并事先安装 VISA 库，VISA 自动检测和配置仪器以建立 USB 连接，而不需要输入仪器地址字符串或安装单独的驱动程序。

USB 地址：

寻址字符串格式：USB::<vendor ID>::<product ID>::<serial number>[::INSTR]

其中：

- <vendor ID>　代表生产厂家代号；
- <product ID>　代表仪器代号；
- <serial number>　代表仪器序列号。

示例：

USB::0x04b4::0x1004::100001::INSTR

0x04b4：生产厂家代号；

0x1004：仪器代号；

100001：是仪器的序列号。

2.1.2 消息

数据线上传输的消息分为以下两类：

1) **接口消息**

仪器与主控计算机间通信时，首先需要拉低attention线，然后接口消息才能通过数据线传送给仪器。只有具备GPIB总线功能的仪器才能发送接口消息。

2) **仪器消息**

有关仪器消息的结构和语法，具体请参考章节“2.1.3 SCPI命令”。根据传输方向的不同，仪器消息可分为命令和仪器响应。如不特别声明，所有程控接口使用仪器消息的方法相同。

a) **命令**：

命令（编程消息）是主控计算机发送给仪器的消息，用于远程控制仪器功能并查询状态信息。命令被划分为以下两类：

- 根据对仪器的影响：
  - 设置命令：改变仪器设置状态，例如：复位或设置频率等。
  - 查询命令：查询并返回数据，例如：识别仪器或查询参数值。查询命令以后缀问号结束。
- 根据标准中的定义：
  - 通用命令：由IEEE488.2定义功能和语法，适用所有类型仪器（若实现）用于实现：管理标准状态寄存器、复位和自检测等。
  - 仪器控制命令：仪器特性命令，用于实现仪器功能。例如：设置频率。语法同样遵循SCPI规范。

b) **仪器响应**：

仪器响应（响应消息和服务请求）是仪器发送给计算机的查询结果信息。该信息包括测量结果、仪器状态等。

2.1.3 SCPI 命令

- SCPI命令简介 ……7
- SCPI命令说明 ……8

2.1.3.1 SCPI 命令简介

SCPI(Standard Commands for Programmable Instruments——可程控设备的标准命令)是一个基于标准 IEEE488.2 建立的，适合所有仪器的命令集。其主要目的是为了使相同功能具有相同的程控命令，以实现程控命令的通用性。

SCPI 命令由命令头和一个或多个参数组成，命令头和参数之间由空格分开，命令头包含一个或多个关键字段。命令直接后缀问号即为查询命令。命令分为通用命令和仪器专用命令，它们的语法结构不同。SCPI 命令具备以下特点：

1) 程控命令面向测试功能，而不是描述仪器操作；

2) 程控命令减少了类似测试功能实现过程的重复，保证了编程的兼容性；
3) 程控消息定义在与通信物理层硬件无关的分层中；
4) 程控命令与编程方法和语言无关，SCPI 测试程序易移植；
5) 程控命令具有可伸缩性，可适应不同规模的测量控制；
6) SCPI 的可扩展性，使其成为“活”标准。

如果有兴趣了解更多关于 SCPI 的内容，可参考：

IEEE Standard 488.1-2003, IEEE Standard Digital Interface for Programmable Instrumentation. New York, NY, 1998

IEEE Standard 488.2-1992, IEEE Standard Codes, Formats, Protocols and Comment Commands for Use with ANSI/IEEE Std488.1-2003. New York, NY, 1998

Standard Commands for Programmable Instruments(SCPI) VERSION 1999.0.

87230 的程控命令集合、分类及说明，具体请参考：

1) 本手册“3 程控命令”;
2) 本手册“附录 A SCPI 命令按子系统分类速查表”。

2.1.3.2 SCPI 命令说明

1) 通用术语

下面这些术语适用本节内容。为了更好的理解章节内容，您需要了解这些术语的确切定义。

控制器

控制器是任何用来与 SCPI 设备通讯的计算机。控制器可能是个人计算机、小型计算机或者卡笼上的插卡。一些人工智能的设备也可作为控制器使用。

设备

设备是任何支持 SCPI 的装置。大部分的设备是电子测量或者激励设备，并使用 GPIB 接口通讯。

程控消息

程控消息是一个或者多个正确格式化过的 SCPI 命令的组合。程控消息告诉设备怎样去测量和输出信号。

响应消息

响应消息是指定 SCPI 格式的数据集合。响应消息总是从设备到控制器或者侦听设备。响应消息告诉控制器关于设备的内部状态或测量值。

命令

命令是指满足 SCPI 标准的指令。控制设备命令的组合形成消息。通常来说，命令包括关键字、参数和标点符号。

事件命令

事件型程控命令不能被查询。一个事件命令一般没有与之相对应的前面板按键设置，它的功能就是在某个特定的时刻触发一个事件。

查询

查询是一种特殊类型的命令。查询控制设备时，返回适合控制器语法要求的响应消息。查询语句总是以问号结束。

2) 命令类型

SCPI 命令分为两种类型：通用命令和仪器专用命令。图 2.1 显示了两种命令的差异。通用命令由 IEEE 488.2 定义，用来管理宏、状态寄存器、同步和数据存储。因通用命令均以一个星号打头，因此很容易辨认。例如*IDN？、*OPC、*RST 都是通用命令。通用命令不属于任何仪器专用命令，仪器采用同一种方法解释该类命令，而不用考虑命令的当前路径设置。

仪器专用命令因包含冒号（:），因此容易辨认。冒号用在命令表达式的开头和关键字的中间，例如：FREQuency[:CW?]。根据仪器内部功能模块，将仪器专用命令划分为对应的子系统命令子集合。例如，功率子系统（:POWer）包含功率相关命令，而状态子系统（:STATus）包含状态控制寄存器的命令。

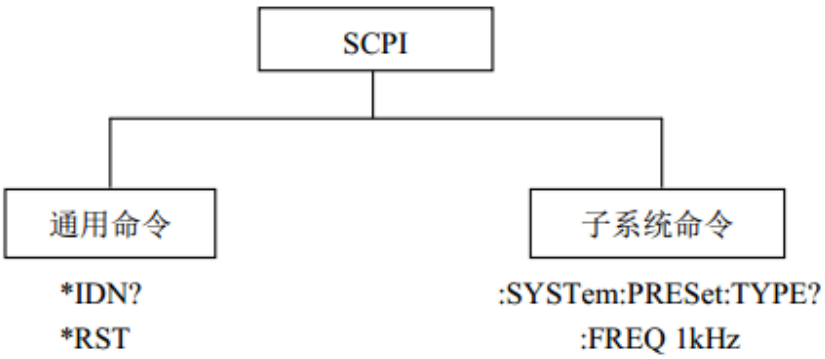


图2.1　SCPI 命令类型

3) 仪器专用命令语法

一个典型的命令是由前缀为冒号的关键字构成。关键字后面跟着参数。下面是一个语法声明的例子。

[:SENSe]:FREQuency[:CW|FIXed] MAXimum|MINimum

在上面的例子中，命令中的[:CW|FIXed]部分紧跟着:FREQuency，中间没有空格。紧跟着[:LEVel]的部分：MINimum|MAXimum 是参数部分。在命令与参数之间有一个空格。语法表达式的其它部分说明见表 2.2 和表 2.3。

2.1 远程控制基础

表2.2 命令语法中的特殊字符

符号	含义	举例
|	在关键字和参数之间的竖号代表多种选项。	[:SENSe]:BANDwidth|BWIDth HIGH|LOWer BANDwidth 和 BWIDth 是选项，HIGH 和 LOWer 是选项。
[]	方括号表示被包含的关键字或者参数在构成命令时是可选的。这些暗含的关键字或者参数甚至在它们被忽略时命令也会被执行。	[:SENSe]:BANDwidth? SENSe 是可选项。
< >	尖括号内的部分表示在命令中并不是按照字面的含义使用。它们代表必需包含的部分。	[:SENSe]:FREQency[:CW|FIXed] <val>[unit] 该命令中，<val> 必须用实际的频率替代。 [unit]是可省略的单位。 例如：FREQ 3.5GHz FREQ 3.5e+009
{ }	大括号内的部分表示其中的参数可选。	MEMory:TABLe:FREQuency <val>{,<val>} 例如：MEM:TABL:FREQ 5e7

表2.3 命令语法

字符、关键字和语法	举例
大写的字符代表执行命令所需要的最小字符集合。	[:SENSe]:FREQuency[:CW|FIXed]? FREQ 是命令的短格式部分。
命令的小写字符部分是可选择的；这种灵活性的格式被称为“灵活地听”。更多信息请参照“命令参数和响应”部分。	:FREQuency :FREQ，:FREQuency 或者:FREQUENCY， 其中任意一个都是正确的。
当一个冒号在两个命令助记符之间，它将命令树中的当前路径下移一层。更多消息请参照“命令树”的命令路径部分。	:TRIGger:MODE? TRIGger 是这个命令的最顶层关键字。
如果命令包含多个参数，相邻的参数间由逗号分隔。参数不属于命令路径部分，因此它不影响路径层。	MEMory:TABLe:FREQuency <val>{,<val>}
分号分隔相邻的 2 条命令，但不影响当前命令路径。	:FREQ 2.5GHZ;:POW 10DBM
空白字符，例如<space>或者<tab>，只要不出现在关键字之间或者关键字之中，通常是被忽略的。然而，你必须用空白字符将命令和参数分隔开来，且不影响当前路径。	:FREQ uency 或者:POWer :LEVel6.2 是不允许的。 在:LEVel 和 6.2 之间必须由空格隔开。 即 :POWer:LEVel 6.2

简化语法规范如图 2.2 所示:

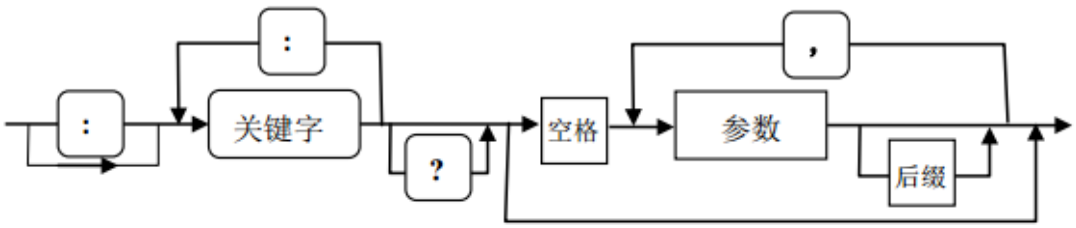


图2.2　　SCPI 命令类型

如“[:SENSe[1]:FREQuency[:CW|FIXed] <数值型参数>”的语法表达式可表示如下。

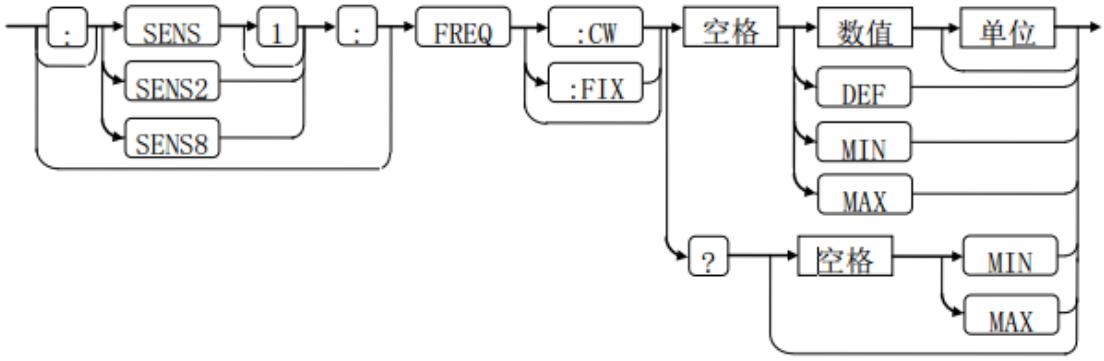


图2.3　　SCPI 命令类型

备注:

1) 上图中，未标明需要空格的地方，不得加入空格。空格可以是 1 个或多个。如果单位省略，则用频率的标准单位 Hz，时间的标准单位为秒。
2) 圆角矩形表示关键字或命令所需的实际字符，如“:”、“,”、“?”、“1”等；直角矩形表示需要用实际字符、数字等代替。如“数值”不能出现在命令中，需要用实际数值代替，如 5e+007 等。
3) 不考虑关键字的长短格式，设置频率为 50MHz，上图共有如下几种形式（只取关键字的短格式。由于频率的单位比较多，如 Hz，kHz，MHz，GHz，THz 等，限于篇幅，不再一一给出，仅需要将 5.0e+007 用相应的单位代替即可，如 50MHz，5e+007Hz，0.05GHz 等）。
   a) :SENS1:FREQ:CW　5.0e+007　　不省略关键字。
   b) SENS1:FREQ:CW　5.0e+007　　SENS1 前的“:”省略。
   c) SENS:FREQ:CW　5.0e+007　　“1”省略。
   d) FREQ:CW　5.0e+007　　SENS 省略。
   e) :SENS1:FREQ:FIX　5.0e+007　　不省略关键字。
   f) SENS1:FREQ:FIX　5.0e+007　　SENS1 前的“:”省略。
   g) SENS:FREQ:FIX　5.0e+007　　“1”省略。
   h) FREQ:FIX　5.0e+007　　SENS 省略。
   i) :SENS1:FREQ　5.0e+007　　CW 或 FIX 省略。

j) SENS1:FREQ 5.0e+007 SENS1 前的“:”和 CW 或 FIX 省略。

k) SENS:FREQ 5.0e+007 “1”和 CW 或 FIX 省略。

l) FREQ 5.0e+007 SENS 和 CW 或 FIX 省略。

4) MIN 和 MAX 可以作为设置命令的参数，也可以作为查询命令的参数。DEF 仅能作为设置命令的参数。MIN、MAX、DEF 的具体值与仪器有关。最小、最大、缺省频率分别为 1kHz、1THz、1GHz。

a) FREQ DEF 设置频率为缺省值。

b) FREQ? MAX 查询最大可设置频率，返回一个没有单位的数值。

5) 如果考虑关键字的长短形式，且不考虑带单位的命令，则上图的命令共有 1632 种形式。用户不用关心所有形式，能灵活使用就足够了。下面进行简单的计算，感兴趣的用户可以自己进行计算。

a) 先算 SENS 缺省的情况，记为 N1。

FREQ/FREQuency 有 2 种形式，记为 N11。

CW/FIX/FIXed/省略有 4 种形式，记为 N12。

设置命令的参数有数值/DEF/DEFault/MIN/MINimum/MAX/MAXimum 共 7 种。

查询命令的参数有 MIN/MINimum/MAX/MAXimum/省略共 5 种，即参数形式共 12 种，记为 N13，则

$$N1 = N11 \times N12 \times N13 = 2 \times 4 \times 12 = 96$$

b) 再计算 SENS 不缺省的情况，记为 N2。

第一个“:”有省略和不省略 2 种，记为 N21。

SENS/SENSe 有 2 种。尾缀有 1/2/6/省略共 4 种，即 SENS 关键字有 2×4 共 8 种，记为 N22。

SENS 关键字与后面的关键字形成组合关系，则

$$N2 = N21 \times N22 = 2 \times 8 \times N1 = 16 \times N1$$

c) 记总共有 N 种形式，则

$$N = N1 + N2 = 17 \times N1 = 1632$$

4) 命令树

大部分远程控制编程会使用仪器专用命令。解析该类命令时，SCPI 使用一个类似于文件系统的结构，这种命令结构被称为命令树，如图 2.4 所示:

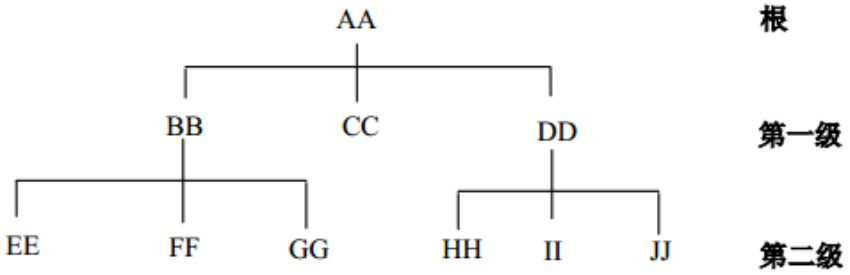


图2.4 简化的命令树示意图

顶端命令是根命令，简称“根”。命令解析时，依据树结构遵循特定的路径到达下一层命令。例如：:POWer:ALC:SOURce？，其中，:POWer 代表 AA，:ALC 代表 BB，:SOURce 代表 GG，整个命令路径是（:AA:BB:GG）。

仪器软件中的一个软件模块——**命令解释器**，专门负责解析每一条接收的 SCPI 命令。命令解释器利用一系列的分辨命令树路径的规则，将命令分成单独的命令元。解析完当前命令后，保持当前命令路径不变，这样做的好处是，因为同样的命令关键字可能出现在不同的路径中，更加快速有效的解析后续命令。开机或*RST（复位）仪器后，重置当前命令路径为根。

5) 命令参数和响应

SCPI 定义了不同的数据格式在程控和响应消息的使用中以符合“灵活地听”和“精确地讲”的原则。更多的信息请参照 IEEE488.2。“灵活地听”指的是命令和参数的格式是灵活的。

例如仪器设置频率偏移状态命令 :FREQuency:OFFSet:STATe ON|OFF|1|0，

以下命令格式都是设置频率偏移功能开:

:FREQuency:OFFSet:STATe ON，:FREQuency:OFFSet:STATe 1，

:FREQ:OFFS:STAT ON，:FREQ:OFFS:STAT 1。

不同参数类型都有一个或多个对应的响应数据类型。查询时，数值类型的参数将返回一种数据类型，响应数据是精确的，严格的，被称为“**精确地讲**”。

例如，查询功率状态（:POWer:ALC:STATe?），当其为开时，不管之前发送的设置命令是:POWer:ALC:STATe 1 或者 :POWer:ALC:STATe ON，查询时，返回的响应数据总是 1。

表2.4 SCPI 命令参数和响应类型

参数类型	响应数据类型
数值型	实数或者整数
扩展数值型	整数
离散型	离散型
布尔型	数字布尔型
字符串	字符串
块	确定长度的块
	不确定长度的块
非十进制的数值类型	十六进制
	八进制
	二进制

数值参数

仪器专用命令和通用命令中都可使用数值参数。数值参数接收所有的常用十进制计数法，包括正负号、小数点和科学记数法。如果某一设备只接收指定的数值类型，例如整数，那么它自动将接收的数值参数取整。

以下是数值类型的例子:

0 无小数点

100 可选小数点

1.23	带符号位
4.56e<space>3	指数标记符 e 后可以带空格
-7.89E-01	指数标记符 e 可以大写或小写
+256	允许前面加正号
.5	小数点可先行

扩展的数值参数

大部分与仪器专用命令有关的测量都使用扩展数值参数来指定物理量。扩展数值参数接收所有的数值参数和另外的特殊值。所有的扩展数值参数都接收 MAXimum 和 MINimum 作为参数值。其它特殊值，例如：UP 和 DOWN 是否接收由仪器解析能力决定，其 SCPI 命令表中会列出所有有效的参数。

注意：扩展数值参数不适用于通用命令或是 STATus 子系统命令。

扩展数值参数举例：

101	数值参数
1.2GHz	GHz 可以被用作指数（E009）
200MHz	MHz 可以被用作指数（E006）
-100mV	-100 毫伏
10DEG	10 度
MAXimum	最大的有效设置
MINimum	最小的有效设置
UP	增加一个步进
DOWN	减少一个步进

离散型参数

当需要设置的参数值为有限个时，使用离散参数来标识。离散参数使用助记符来表示每一个有效的设置。像程控命令助记符一样，离散参数助记符有长短两种格式，并可使用大小写混合的方式。

下面的例子，离散参数和命令一起使用。

:TRIGger[:SEQuence]:SOURce BUS|IMMediate|EXTernal

BUS	GPIB,LAN,RS-232 触发
IMMediate	立刻触发
EXTernal	外部触发

布尔型参数

布尔参数代表一个真或假的二元条件，它只能有四个可能的值。

布尔参数示例：

ON	逻辑真
OFF	逻辑假
1	逻辑真
0	逻辑假

字符串型参数

字符串型参数允许 ASCII 字符串作为参数发送。单引号和双引号被用作分隔符。

下面是字符串型参数的例子。

'This is Valid'　　"This is also Valid"　　'SO IS THIS'

实型响应数据

大部分的测试数据是实数型，其格式可以为基本的十进制计数法或科学计数法，大部分的高级程控语言均支持这两种格式。

实数响应数据示例:

1.23E+0
-1.0E+2
+1.0E+2
0.5E+0
0.23
-100.0
+100.0
0.5

整型响应数据

整数响应数据是包括符号位的整数数值的十进制表达式。当对状态寄存器进行查询时，大多返回整数型响应数据。

整数响应数据示例:

0	符号位可选
+100	允许先行正号
-100	允许先行负号
256	没有小数点

离散响应数据

离散型响应数据和离散型参数基本一样，主要区别是离散型响应数据的返回格式只为大写的短格式。

离散响应数据示例:

INTernal	稳幅方式为内部
EXTernal	稳幅方式为外部
MMHead	稳幅方式为毫米波源模块

数字布尔型响应数据

布尔型的响应数据返回一个二进制的数值 1 或者 0。

字符串型响应数据

字符串响应数据和字符串参数是同样的。主要区别是字符串响应数据的分隔符使用双引号，而不是单引号。字符串响应数据还可嵌入双引号，并且双引号间可以无字符。

下面是一些字符串型响应数据的例子:

“This is a string”

“one double quote inside brackets: (“”)”

任意数据块

<任意数据块> 见 IEEE 488.2 第 7.7.6 节。

#nNNN…Nddd……ddd<LF>

- 数据
- 换行符，表明数据块结束。
- 数据长度（即 d 的字节数）
- 数据长度的位数（即 N 的位数）
- 数据块的开始的标记。

例如#42004…..<LF>中，n = 4，N = 2004。

6) 命令中数值的进制

命令的值可以用二进制，十进制，十六进制或者八进制的格式输入。当用二进制，十六进制或者八进制时，数值前面需要一个合适的标识符。十进制（默认格式）不需要标识符，当输入一个数值前面没有表示符时，设备会确保其是十进制格式。下面的列表显示了各个格式需要的表示符:

- #B 表示这个数字是一个二进制数值。
- #H 表示这个数字是一个十六进制数值。
- #Q 表示这个数字是一个八进制数值。

下面是 SCPI 命令中十进制数 45 的各种表示:

#B101101

#H2D

#Q55

下面的例子用十六进制数值 000A 设置 RF 输出功率为 10dBm（或者当前选择单位的等数值的值，如 DBUV 或者 DBUVEMF）。

:POW #H000A

在使用非十进制格式时，一个测量单位，如 dBm 或者 mV，并没有和数值一起使用。

7) 命令行结构

一条命令行或许包含多条SCPI命令，为表示当前命令行结束，可采用下面的方法:

- 换行;
- 换行与EOI;
- EOI与最后一个数据字节。

命令行中的命令由分号隔开，属于不同子系统的命令以冒号开头。例如:

MMEM:COPY "Test1", "MeasurementXY";:HCOP:ITEM ALL

该命令行包含两条命令,第一条命令属于MMEM子系统,第二条命令属于HCOP子系统。若相邻的命令属于同一个子系统，命令路径部分重复，命令可缩写。例如:

HCOP:ITEM ALL;:HCOP:IMM

该命令行包含两条命令，两条命令均属于HCOP子系统，一级相同。所以第二条命令可

从HCOP的下级开始，并可省略命令开始的冒号。可以缩写为如下命令行：

HCOP:ITEM ALL;IMM

2.1.4 命令序列与同步

IEEE488.2 定义了交迭命令和连续命令之间的区别：

- 连续命令是指连续执行的命令序列。通常各条命令执行速度较快；
- 交迭命令是指下条命令执行前，前条命令未自动执行完成。通常交迭命令的处理时间较长并允许程序在此期间可同步处理其它事件。

即使一条命令行中的多条设置命令，也不一定按照接收的顺序依次执行。为了保证命令按照一定的顺序执行，每条命令必须以单独的命令行发送。

举例：命令行包含设置和查询命令

一条命令行的多条命令若包含查询命令，查询结果不可预知。下面的命令返回固定值：

:FREQ:STAR 1GHZ;SPAN 100;:FREQ:STAR?

返回值：1000000000（1GHz）

下面的命令返回值不固定：

:FREQ:STAR 1GHz;STAR?;SPAN 1000000

返回结果可能是当前的起始频率值，因为主机程序会推迟执行命令。若主机程序接收命令后执行，返回结果也可能是1GHz。

提 示

设置命令与查询命令分开发送

一般规则：为保证查询命令的返回结果的正确性，设置命令和查询命令应在不同的程控消息中发送。

2.1.4.1 防止命令交迭执行

为了防止命令的交迭执行，可采用多线程或者命令：*OPC、*OPC?或者*WAI，只有硬件设置完成后，才执行这三种命令。 编程时，计算机可强制等待一段时间以同步某些事件。下面分别予以说明：

- **控者程序使用多线程**

  多线程被用于实现等待命令完成和用户界面及程控的同步，即单独的线程中等待*OPC? 完成，而不会阻塞GUI 或程控线程的执行。

- **三种命令在同步执行中的用法如下表**：

表2.5 命令语法

方法	执行动作	编程方法
*OPC	命令执行完后，置位 ESR 寄存器中的操作完成位。	置位 ESE BIT0; 置位 SRE BIT5; 发送交迭命令和*OPC; 等待服务请求信号（SRQ）; 服务请求信号代表交迭命令执行完成。
*OPC?	停止执行当前命令，直到返回1。只有 ESR 寄存器中的操作完成位置位时，该命令才返回，表明前面命令处理完成。	执行其它命令前终止当前命令的处理，在当前命令后直接发送该命令。
*WAI	执行*WAI 前，等待发送完所有命令，再继续处理未完成的命令。	执行其它命令前终止当前命令的处理，在当前命令后直接发送该命令。

对于交迭命令的处理时间较短情况，可采用交迭命令后使用命令*WAI 或*OPC 以实现命令同步。为了实现计算机或仪器等待交迭命令执行完成时，同步执行其它任务，可采用下面的同步技术:

- **OPC与服务请求**

1) 设置ESE的OPC屏蔽位（bit0）：*ESE 1;
2) 设置SRE的bit5：*SRE 32 使能ESB服务请求;
3) 发送交迭命令和*OPC;
4) 等待服务请求信号。

服务请求信号代表交迭命令执行完成。

- **OPC? 与服务请求**

1) 设置SRE的bit4：*SRE 16 使能MAV服务请求;
2) 发送交迭命令和*OPC? ;
3) 等待服务请求信号。

服务请求信号代表交迭命令执行完成。

- **事件状态寄存器（ESE）**

1) 设置ESE的OPC屏蔽位（bit0）：*ESE 1;
2) 只发送交迭命令，不发送*OPC、*OPC或*WAI;
3) 在定时器中发送“*OPC;*ESR?”循环查询操作完成状态。

返回值（LSB）等于1表示交迭命令执行完成。

- ***OPC?与短超时**

1) 只发送交迭命令，不发送*OPC、*OPC或*WAI;
2) 在定时器中发送“<short timeout>; *OPC?”循环查询操作完成状态;
3) 返回值（LSB）等于1表示交迭命令执行完成。超时时，操作过程中;
4) 复位超时值到旧值;
5) 发送命令“SYSTem:ERRor?”清除错误队列，并删除“-410，查询中断”信息。

返回值（LSB）等于1表示交迭命令执行完成。

2.1.5 状态报告系统

状态报告系统存储当前仪器所有的操作状态信息，包括错误信息。它们分别存储在状态寄存器和错误队列中，并可通过程控接口查询。

- 状态寄存器组织结构 19
- SCPI状态寄存器结构 20
- 状态寄存器说明 21
- 状态报告系统的应用 24
- 复位状态报告系统 25

2.1.5.1 状态寄存器组织结构

寄存器分类说明如下:

1) STB, SRE

状态字节（STB）寄存器和与之关联的屏蔽寄存器——服务请求使能寄存器（SRE）组成了状态报告系统的最高层寄存器。STB通过收集低层寄存器信息，保存了仪器的大致工作状态。

2) ESR, SCPI状态寄存器

STB接收下列寄存器的信息:

- 事件状态寄存器（ESR）与事件状态使能（ESE）屏蔽寄存器两者相与的值;
- SCPI状态寄存器包括：STATus:OPERation 与 STATus:QUEStionable 寄存器（SCPI定义），它们包含仪器的具体操作信息。所有的SCPI状态寄存器具备相同的内部结构（具体请参考程控手册2.1.5.2“SCPI状态寄存器结构”章节部分）。

3) IST, PPE

类似SRQ，IST标志（“Individual STatus”）单独的一位，由仪器全部状态组合而成。关联的并行查询使能寄存器（PPE（parallel poll enable register））决定了STB的哪些数据位作用于IST标志。

4) 输出缓冲区

存储了仪器返回给控者的消息。它不属于状态报告系统，但是决定了STB的MAV位的值。

以上寄存器具体说明请参考程控手册“2.1.6 状态报告系统”章节部分。

提 示

SRE, ESE

服务请求使能寄存器 SRE 可被用作 STB 的使能部分。同理，ESE 可被用作 ESR 的使能部分。

2.1.5.2 SCPI 状态寄存器结构

每一个标准的 SCPI 寄存器包括 5 部分。每部分包含 16 个数据位并功能独立。例如：每个硬件状态分配一个数据位，该数据位对寄存器的所有 5 部分均有效。Bit15 若设置为 0，说明该寄存器的数值是正整型数据。

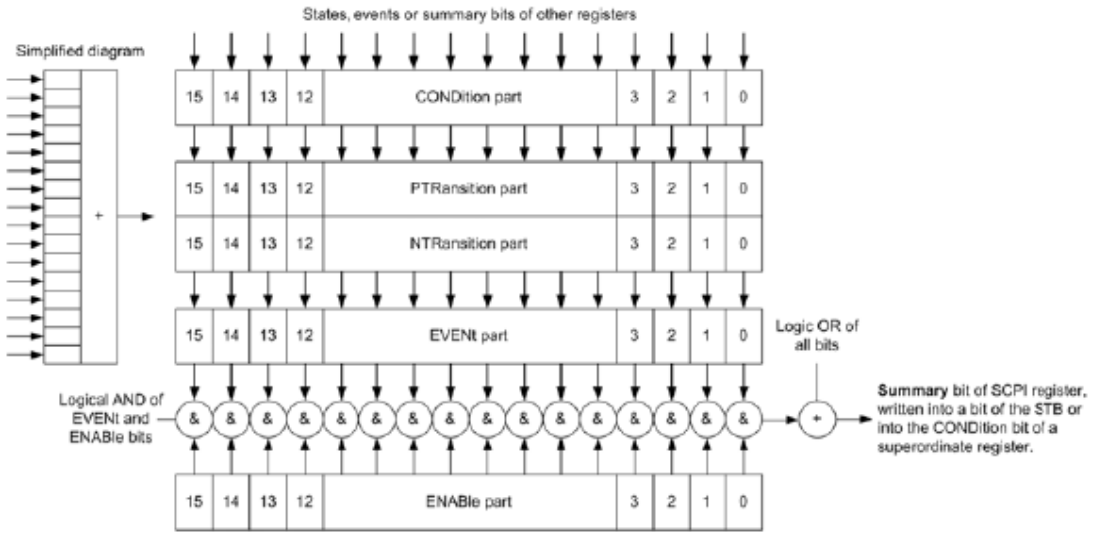


图2.5　状态寄存器结构

由上图看出状态寄存器由 5 部分组成，分别说明如下：

- **条件寄存器**

该部分由硬件或者低级寄存器的数位和直接写入，反映了当前仪器的工作状态。该寄存器只读，不能写，读取不清除数值。

- **正负跳变寄存器**

两个跳变寄存器定义了存储在事件寄存器中的条件寄存器的状态跳变位。

正跳变寄存器类似于跳变滤波器，当条件寄存器的某数据位从 0 变换到 1，相关的 PTR 位决定事件位是否设置为 1，说明如下：

—PTR 位=1：设置事件位。

—PTR 位=0：不设置事件位。

正跳变寄存器可读可写，并且读取不清除数值。

负跳变寄存器类似于跳变滤波器，当条件寄存器的某数据位从 1 变换到 0，相关的 NTR 位决定事件位是否设置为 1，说明如下：

—NTR 位=1：设置事件位。

—NTR 位=0：不设置事件位。

正跳变寄存器可读可写，并且读取不清除数值。

- **事件寄存器**

该部分表明从上次读取后该事件是否发生，存储了条件寄存器的内容。它仅代表经跳变寄存器传递的事件，只能由仪器改变，由用户读取，读取后清除数值。该部分的值常等于整个寄存器的值。

- **使能寄存器**

该部分决定了关联的事件位是否作用于最终的数据和。每一个使能部分的数据位与关联的使能位相与。该部分的逻辑操作结果与数据和位相或。

—使能位=0：关联的事件位不作用于数据和。

—使能位=1：关联的事件位作用于数据和。

该部分可读可写，并且读取不清除。

- **数据位和**

每个寄存器的数据和位由事件和使能部分组成。结果进入高位寄存器的条件部分。仪器自动产生每个寄存器的数据和位，这样事件可引起不同层次的服务请求。

2.1.5.3 状态寄存器说明

下面依次详细介绍各状态寄存器。

1) 状态字节（STB）和服务请求使能寄存器（SRE）

IEEE488.2 中定义了状态字节（STB），通过采集低位寄存器的信息反映粗略的仪器状态。其中 bit6 等于其它状态字节位的数据和。状态字节与 SCPI 寄存中的状态部分比较后的结果可假设为 SCPI 层次中的最高层。通用命令“*STB?”或者串行查询可读取状态字节数值。

状态字节连接到服务请求使能寄存器（SRE）。状态字节的每一个数据位对应 SRE 中的一位。SRE 的 bit6 忽略不用。若 SRE 其中的某数据位设置后，且关联的 STB 位从 0 变化到 1，就会产生一个服务请求（SRQ）。通用命令“*SRE”用于设置 SRE，通用命令“*SRE? ”用于读取 SRE。状态字节说明，具体如下表 2.6 状态字节说明：

表2.6　状态字节说明

数据位	意 义
0..1	未使用。
2	错误队列非空 若错误队列插入新的错误，设置该位。若关联的 SRE 位使能该位，错误队列中产生新的错误时，产生一次服务请求，这样可识别错误并查询错误信息。这种方法有效的减少了程控中的错误。
3	询问状态寄存器数据和位 若设置询问状态寄存器的事件位以及关联的使能位设置为 1，才能够设置该位。该位代表一个可查询的的仪器状态，可通过查询状态寄存器的询问状态寄存器得到具体的仪器状态信息。
4	MAV 位（message available） 若输出队列的信息可读时设置该位。控者查询仪器信息时使用该位。
5	ESB 位 事件状态寄存器的数据和位。若设置了事件状态寄存器中的其中的一位并且使能事件状态使能寄存器中的相应位，可设置该位。该位置位为 1 表明仪器出现了严重错误，并可通过查询事件状态寄存器获取具体错误信息。

数据位	意 义
6	MSS 位（master status summary bit） 若仪器触发了服务请求设置该位。
7	操作状态寄存器数据和位 若设置操作状态寄存器的事件位且对应的使能位设置为 1，则可设置该位。该位表明仪器执行了一个操作，具体操作类型可通过查询操作状态寄存器获取。

2) IST 标志和并行查询使能寄存器（PPE）

IST 以一个单独的数据位标识仪器整体状态的组合。可通过并行查询或者发送命令“*IST?”获取该标志。关联的并行查询使能寄存器（PPE（parallel poll enable register））决定了 STB 的哪些数据位作用于 IST 标志。STB 数据位与 PPE 数据位相与，且 bit6 的用法与 SRE 中的 bit6 相反，IST 标志等于所有结果的或值。设置 PPE 可通过命令“*PRE”，读取 PPE 可通过命令“*PRE?”。

3) 事件状态寄存器（ESR）和事件状态使能寄存器（ESE）

IEEE488.2 定义了 ESR。读取事件状态寄存器（ESR（event status register））可通过命令“*ESR?”。ESE 属于 SCPI 寄存器的使能部分，若其中的一位置 1 并且响应的 ESR 其中的某个数据位从 0 变化到 1，那么 STB 的 ESB 位设置为 1。设置 ESE 可通过命令“*ESE”，可通过命令“*ESE?”读取 ESE。

表2.7 事件状态字节说明

数据位	意 义
0	操作完成 当前面的命令都已执行完成且收到了命令*OPC，设置该位。
1	未使用。
2	查询错误 若控者未发送查询命令就读取仪器数据，或者还未读取查询数据就发送了了新的命令时，设置该位。表明产生了错误的查询而不能执行查询。
3	仪器产生的错误 若产生了仪器错误，设置该位。错误代码范围：-300~-399，或者正的错误代码。具体错误信息可查询错误队列中的相关信息。
4	执行错误 若收到语法正确的命令但是不能执行设置该位。同时错误队列中生成一个错误代号属于范围-200~-300 的错误。
5	命令错误 若接收的命令语法错误，设置该位。错误代号范围：-100~-200，具体错误信息可查询错误队列中的相关信息。
6	用户请求 若仪器切换到手动控制模式时，设置该位。

数据位	意 义
7	电源开 打开仪器电源时，设置该位。

4) 状态:操作寄存器

状态：操作寄存器包含当前仪器的操作信息，及之前执行的操作信息。可通过命令“STATus:OPERation:CONDition?”或者“STATus: OPERation[: EVENt]?”读取该操作寄存器数值。寄存器说明如下表2.8。

表2.8 状态：操作寄存器说明

位	值	定义
0	1	未用
1	2	校准状态
2	4	保留
3	8	保留
4-14	-	未用
15	-	始终为 0

5) 状态:疑问寄存器

该寄存器包含不符合规范要求的仪器状态。可通过命令“STAT:QUES:COND”或“STAT:QUES:EVEN”，查询该寄存器数值。寄存器说明如下表2.9。

提 示

查询寄存器

状态：疑问寄存器集合了所有低层子寄存器的信息（例如：其中的 bit2 集合了所有与时间相关的信息）。其中，因为每个通道对应单独的子寄存器，因此，若疑问寄存器的一个状态位指示有错误，那么需要回溯通道的子寄存器中，检查具体的错误源头。默认状态下，查询的子寄存器状态属于当前选择的通道。

表2.9 状态：疑问寄存器说明

位	值	定义
0-2	-	未用
3	8	功率概要
4-7	-	未用
8	256	校准概要
9	512	开机自测试
10-14	-	未用
15	-	始终为 0

2.1.5.4 状态报告系统的应用

状态报告系统用于监测测试系统中的一个或多个仪器状态。为了正确实现状态报告系统的功能，测试系统中的控者必须接收并评估所有仪器的信息，使用的标准方法包括：

1) 仪器发起的服务请求（SRQ）；
2) 串行查询总线系统中的所有的仪器，由系统中的控者发起，目的是找到服务请求发起者及原因；
3) 并行查询所有仪器；
4) 程控命令查询特定仪器状态；
5) 查询错误队列。

1) 服务请求

某些情况下，仪器为获取控者的服务，向控者发送服务请求（SRQ），控者发起一个中断进入相应的中断处理程序。从图 2.5 得知，一个 SRQ 通常由一个或多个状态字节及关联使能寄存器（SRE）的第 2，3，4，5 或 7 位发起。这些数据位又进一步组成了高级寄存器、错误队列或输出缓冲区。为了尽可能的使用所有的服务请求，使能寄存器 SRE 和 ESE 的所有数据位需要设置为 1。

举例：使用命令*OPC 在扫描结束后，产生 SRQ 信号。

a) 调用写函数 InstrWrite 写命令“*ESE 1”，设置 ESE bit0（操作完成）。
b) 调用写函数 InstrWrite 写命令“*SRE 32”，设置 SRE bit5（ESB）。
c) 调用写函数 InstrWrite 写命令“*INIT;*OPC”，操作完成后产生 SRQ 信号。

仪器设置完成后，仪器产生一次 SRQ。

SRQ 只能由仪器本身发起，控者程序应允许在仪器产生错误时，向其发起服务请求，并由专门的中断服务程序处理。

2) 串行查询

类似于命令*STB，串行查询用来查询仪器的状态字节。串行查询采用接口消息的方式，因此查询速度较快。IEEE 488.2 定义了串行查询的具体方法。该方法主要用于快速获取测试系统中与控者连接的一个或多个仪器的状态。

3) 并行查询

测试系统中，控者通过一条命令向数据线上发送一个信息位，能够同时查询 8 个仪器。配置到仪器的数据线上的数据是逻辑“0”或“1”。除了 SRE 寄存器决定产生 SRQ 的条件，并行查询使能寄存器（PPE）与 STB 寄存器数据位相与运算。得到的结果经过或运算，再经过位取反，作为响应结果发送给并行查询的控者，也可通过命令*IST 获取该结果。

并行查询时，首先通过命令 PPC 设置仪器到并行查询状态，该命令分配仪器一根数据线并决定响应时数据位是否反转。执行并行查询时使用 PPE 寄存器。并行查询主要用于控者快速定位哪个台仪器发送了服务请求。为此寄存器 SRE 和 PPE 应设置相同的数值。

4) 查询仪器状态

可采用下面两种命令，查询状态状态寄存器的每一部分：

- 命令*ESR?, *IDN?, *IST?, *STB?查询高级寄存器；
- 状态系统命令查询 SCPI 寄存器（例如：STATus：QUEStionable…）。

被查询的寄存器的返回值通常为十进制格式，由控者程序检测使用。为获取更加详细产生 SRQ 的原因，通常 SRQ 后才进行并行查询。

响应数据位说明

STB 与 ESR 寄存器包含 8 位，SCPI 寄存器包含 16 位。查询状态寄存器的返回值采用十进制格式。该十进制数值等于各数据位与各自的权重运算，相加后的结果。

数据位与其权重的关系如下图：

数据位	7	6	5	4	3	2	1	0
权 重	128	64	32	16	8	4	2	1

图2.6　数据位与权重关系图

5) 错误队列

仪器的每个错误状态对应错误队列中的一个入口，其中包含具体的错误信息文本，可通过错误日志查看，或者通过程控命令查询：SYSTem:ERRor[:NEXT]? 或 SYSTem:ERRor:ALL?。若错误队列中无错误，查询返回 0，“无错误”。

在控者服务请求处理程序中应查询错误队列，因为可得到比状态寄存器更加精确的错误原因描述。尤其在控者程序测试阶段，应经常查询错误队列，以明确控者发送给仪器的错误命令记录。

2.1.5.5 复位状态报告系统

下面的列表列出了复位状态报告系统的命令和事件。除了命令*RST 和 SYSTem:PRESet，其它命令不改变仪器功能设置。同样，DCL 也不改变仪器设置状态。具体说明如下表：

表2.10　复位状态报告系统

事件 / 作用	电源开/关（加电状态清除）		DCL, SDC（仪器清除，选择仪器清除）	*RST 或 SYSTem:PRESet	STATus:PRESet	*CLS
	0	1				
清除 STB, ESR	—	是	—	—	—	是
清除 SRE,ESE	—	是	—	—	—	—
清除 PPE	—	是	—	—	—	—
清除寄存器的事件部分	—	是	—	—	—	是

2.2 仪器程控端口与配置

事件 / 作用	电源开/关（加电状态清除）		DCL, SDC（仪器清除，选	*RST 或 SYSTem:	STATus: PRESet	*CLS
除操作和疑问寄存器中的使能部分。其它寄存器的使能部分填充 1。	—	是	—	—	是	—
正跳变部分填充 1。清除负跳变部分。	—	是	—	—	是	—
清除错误队列	是	是	—	—	—	是
清除输出缓冲区	是	是	是	—	—	—
清除命令处理和输入缓冲区	是	是	是	—	—	—

2.1.6 编程注意事项

1) 改变设置前请初始化仪器状态

远程控制设置仪器时，首先需要初始化仪器状态（例如发送"*RST"），然后再实现需要的状态设置。

2) 命令序列

一般来说，需要分开发送设置命令和查询命令。否则，查询命令的返回值会根据当前仪器操作顺序而变化。

3) 故障反应

服务请求只能由仪器自己发起。测试系统中的控者程序应指导仪器在出现错误时主动发起服务请求，进而进入相应的中断服务程序中进行处理。

4) 错误队列

控者程序每次处理服务请求时，应查询仪器的错误队列而不是状态寄存器，来获取更加精确的错误原因。尤其在控者程序的测试阶段，应经常查询错误队列以获取控者发送给仪器的错误命令。

2.2 仪器程控端口与配置

本仪器支持的编程接口为：USB。该接口与 I/O 库和编程语言结合使用可以远程控制本仪器。

87230 无需进行前面板的远程配置。USB 接口符合 USBTMC（USB Test and Measurement Class）协议，是可枚举的设备（即通过 VISA 库的 viFindRsrc 函数可以找到它）。厂商 ID 和设备 ID 分别为 1024 和 4100，序列号见各个探头上。

如果通过 VISA 库实现对 87230 的远程控制，用户必须安装 NI 的 VISA 库或 Keysight 的 VISA 库，并正确安装了 USBTMC 设备，安装 USBTMC 设备的步骤见图 2.7 到图 2.8。

步骤 1: 如果首次使用 87230, 则系统提示用户"找到新的硬件", 并弹出图 2.7, 选择"否，暂时不", 并单击"下一步"。

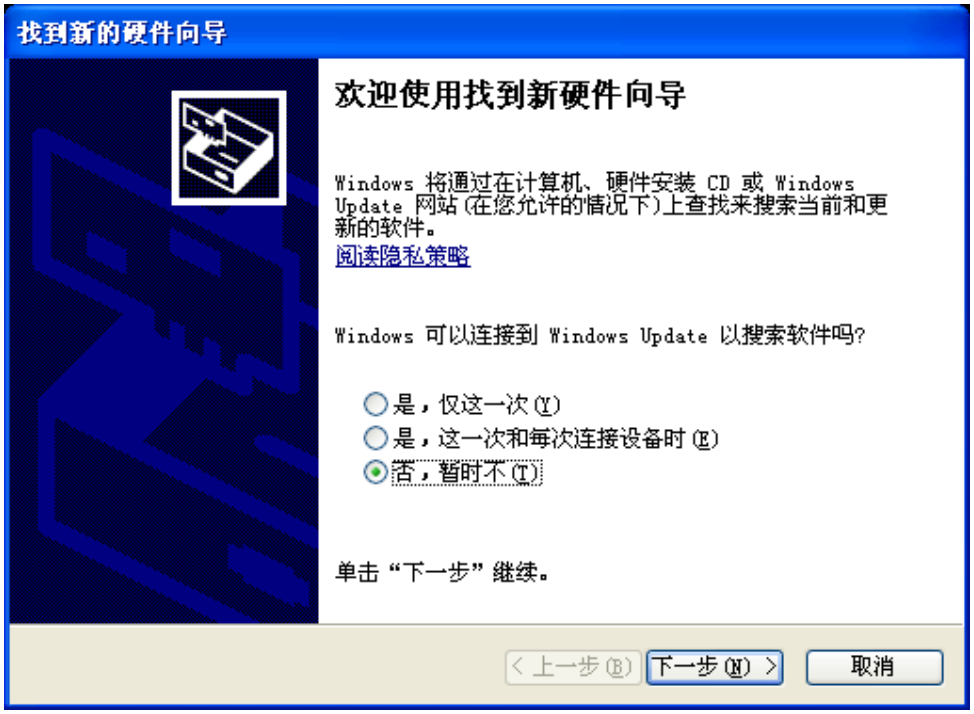


图2.7　主控计算机 USBTMC 驱动安装 1

步骤 2：系统自动识别为 USB 测试与测量设备“USB Test and Measurement Device”，选择“自动安装软件（推荐）”，并单击“下一步”。



图2.8　主控计算机 USBTMC 驱动安装 2

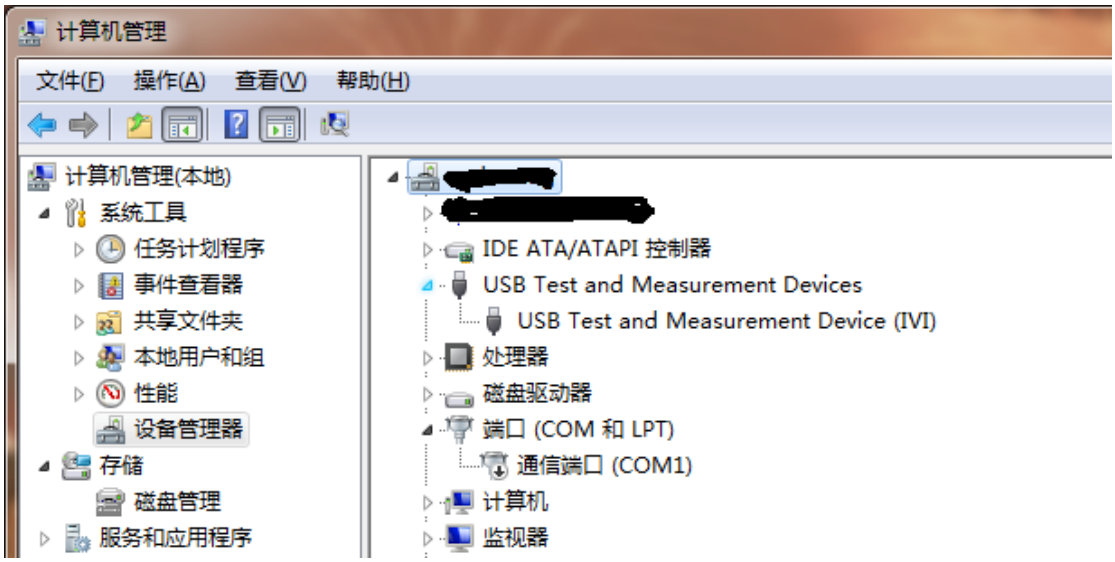


图2.9　主控计算机 USBTMC 驱动安装 3

步骤 3：其余操作系统自动完成。安装完成后，计算机的设备管理器出现“USB Test and Measurement Devices”及已经检测到 USBTMC 仪器的清单。

2.3 I/O 库

- I/O库概述 28
- I/O库安装与配置 29

2.3.1 I/O 库概述

I/O 库是为仪器预先编写的一些软件程序库被称为仪器驱动程序，即：仪器驱动器（Instrument driver），它是介于计算机与仪器硬件设备之间的软件中间层，由函数库、实用程序、工具套件等组成，是一系列软件代码模块的集合，该集合对应于一个计划的操作，如配置仪器、从仪器读取、向仪器写入和触发仪器等。它驻留在计算机中，是连接计算机和仪器的桥梁和纽带。通过提供方便编程的高层次模块化库，用户不再需要学习复杂的针对某个仪器专用的低层编程协议，采用仪器驱动器是快速开发测试测量应用的关键。

从功能上看，一个通用的仪器驱动器一般由功能体、交互式开发者接口、编程开发者接口、子程序接口和 I/O 接口五部分组成，如图 2.10 所示。

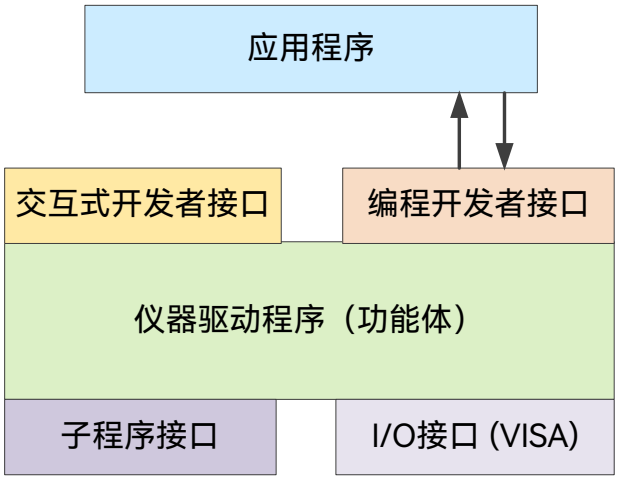


图2.10　仪器驱动器结构模型

具体说明如下:

1) 功能体。这是仪器驱动器的主功能部分，可以理解为仪器驱动器的框架程序。
2) 交互开发者接口。为方便用户使用，支持仪器驱动器开发的应用开发环境通常提供图形化的交互开发接口。例如，Labwindows/CVI 中，函数面板就是一种交互开发接口。函数面板中，仪器驱动器函数的各个参数都是以图形化的控件形式表示。
3) 编程开发者接口。它是应用程序调用仪器驱动器函数的软件接口，例如 Windows 系统下仪器驱动器的动态链接库文件.dll。
4) I/O 接口。它完成仪器驱动器与仪器间的实际通信。可以使用总线专用 I/O 软件，如 GPIB、USB；也可以使用跨多个总线使用的通用的标准 I/O 软件：VISA I/O。
5) 子程序接口。它是仪器驱动器访问其它一些支持库的软件接口，例如数据库、FFT 函数等。当仪器驱动器为完成其任务而需调用其它软件模块、操作系统、程控代码库及分析函数库时，将用到子程序接口。

2.3.2 I/O 库安装与配置

伴随着测试领域的应用经历了从传统仪器到虚拟仪器等不同的发展阶段，并且为了解决自动测试系统中仪器可互换性和测试程序的可重用性，仪器驱动程序经历了不同的发展过程。目前比较流行通用的驱动器是 IVI（Interchangeable Virtual Instruments）仪器驱动器，它基于 IVI 规范，定义了新的仪器编程接口，以及插入类驱动程序和 VPP 架构到 VISA 上，使测试应用程序与仪器硬件完全独立，并增加了独有的仪器仿真、范围检测、状态缓存等功能，提高了系统运行的效率与真正实现了仪器互换。

IVI 驱动分为两种类型：IVI-C 与 IVI-COM，IVI-COM 基于微软组件对象模型（COM）技术，采用 COM API 的形式；IVI-C 基于 ANSI C，采用 C API 的方式。这两种驱动类型都是遵照 IVI 规范定义的仪器类来设计的，它们的应用开发环境也都相同，包括 Visual Studio，Visual Basic，Keysight VEE，LabVIEW，CVI/LabWindows 等。

为满足不同用户在不同开发环境下的需求，目前需要提供两种驱动形式。仪器的 IVI 驱动利用 Nimbus Driver Studio 开发，直接生成 IVI-COM 与 IVI-C 驱动及程序安装包，具体安装配置请参阅您所选择的控制卡及 I/O 库的随机文档资料。

2.4 校零功率探头

安装后的 IVI 驱动分为：IVI 固有功能组与仪器类功能组（基本功能组和扩展功能组）。具体功能分类、函数和属性说明可参考驱动自带的帮助文档。

提 示

配置端口以及安装 IO 库

在使用计算机控制仪器之前，请确认您已正确安装且配置必要的端口和 I/O 库。

提 示

I/O 库的使用

安装随机的 IVI-COM/C 驱动程序安装包，会自动安装驱动函数面板、帮助文档及驱动函数的示例程序，方便用户开发集成程控功能。

2.4 校零功率探头

校零用于扣除通道噪声，校零前，不得加功率信号到探头。什么时候校零？建议在以下几种情况下对仪器进行校零：

- 当温度变化超过 5℃时；
- 当更换功率探头时；
- 时隔 24 小时后；
- 在测量小功率信号之前。例如，测量比功率探头所规定的最低功率高 10dB 的信号时。

相关程控命令是：

CALibration[1]:ZERO

例如校零：

CAL:ZERO

2.5 进行测量

测量可配置成绝对功率测量、相对功率测量等形式。详细情况请参考“测量子系统”。

2.6 使用频响偏置表

如何使用频响偏置（以下简称频偏）表？频偏表用于补偿测量建立过程中的频率响应。

- 概述 31
- 录入频偏表 31
- 选择频偏表 31
- 启用频偏表 32
- 测量应用 32

- 具体应用 32

2.6.1 概述

用 SENSe:CORRection:CSET:STATe 启用或禁止频偏表。当启用时，频偏表提供一个快捷的方法，用于补偿测试系统中频率响应。需要说明一点，当启用时，频偏是“额外”的探头频率响应，即探头固有的频率响应（存放在探头的 EEPROM 中）也考虑在内。探头中可以存放 1 个频偏表，每个最多 80 个频率点。

使用频偏表的方法:

a) 录入频偏表
b) 启用频偏表
c) 指定频率
d) 进行测量

2.6.2 录入频偏表

a) 录入步骤
   - 录入频率列表:MEMory:TABle:FREQuency <频率 1>{, <频率 i>}。如 50MHz, 1GHz, 10GHz, 40GHz。
   - 录入与频率列表对应的偏置因子：MEMory:TABle:GAIN <因子 1>{, <因子 i>}。如 100, 98.8, 101.2, 110.8，分别对应于 50MHz, 1GHz, 10GHz, 40GHz 的偏置因子。
b) 列举频偏表的名称：MEMory:CATalog:TABLe?。具体信息见该命令的说明。
c) 重命名频偏表:MEMory:TABLe:MOVE <原名>, <目标名>。
d) 查询频偏表中的数据。例如查询频偏表中的数据。
   - 查询频偏表的频率点数:MEMory:TABLe:FREQuency:POINTs?
   - 查询频偏表的频率列表:MEMory:TABLe:FREQuency?
   - 查询频偏表的因子点数:MEMory:TABLe:GAIN[:MAGNitude]:POINTs?
   - 查询频偏表的因子列表:MEMory:TABLe:GAIN[:MAGNitude]?
e) 修改频偏表中的数据：见 5.2 录入频偏表。
f) 说明
   - 频率列表以升序方式排列。
   - 频率列表的有效尾缀为 Hz，kHz，MHz，GHz。缺省为 Hz。
   - 确保频率列表在探头的有效频率范围之内。

2.6.3 选择频偏表

SENSe:CORRection:CSET:STATe ON

2.6.4 启用频偏表

[SENSe[1]]SENSe:CORRection:CSET:STATe ON

2.6.5 测量应用

- ABOR
- CONF:POW:AC DEF,1,(@1)
- SENS:CORR:CSET:SEL "MyFdo0"
- SENS:CORR:CSET:STAT ON
- SENS:FREQ 5GHz
- INIT1
- FETC?

2.6.6 具体应用

a) 假如频偏表的频率列表为：500MHz，1GHz，11GHz。偏置因子列表为：100，10，10。设信号频率为 Freq，计算后的偏置因子为 Gain。

b) 如果 Freq 超出频偏表的范围，则用频偏表中最高或最低频率点的频偏值。例如 18GHz，用最大频率点 10GHz 对应的偏置因子 102。例如 50MHz，用最小频率点 500MHz 对应的偏置因子 100。

c) 如果 Freq 在频率列表的有效范围之内，但介于两个频率（Freq1，Freq2）之间，如 5GHz，则利用两点线性插值得到偏置因子，设 Freq1 和 Freq2 对应的偏置因子分别为 Gain1，Gain2。Gain 可由如下公式计算得到 50。

$$Gain = Gain1 + \frac{Freq - Freq1}{Freq2 - Freq1} \times (Gain2 - Gain1)$$

d) 假如在使用频偏表之前的功率为 1.000mW（记为 Pwr0），记最终显示功率为 Pwr，则 Pwr = Pwr0/Gain/100 = 2.000mW。

2.7 设置显示分辨率

暂不支持。

2.8 设置平均

87230 量程共分高低两个量程，可以设置为自动量程和手动量程两种方式。如果对被测量功率电平不清楚，可使用自动量程。

设置自动量程状态的程控命令：

SENSe:POWer:AC:RANGe:AUTO <开关>

例如设置探头为自动量程开:
SENS:POW:AC:RANG:AUTO 1
例如设置探头为自动量程关:
SENS:POW:AC:RANG:AUTO 0
设置量程保持的程控命令:
SENSe:POWer:AC:RANGe <量程>
<量程>取值为: 0 表示低量程，1 表示高量程。
例如设置探头为高量程
SENS:POW:AC:RANG 1

2.9 设置量程

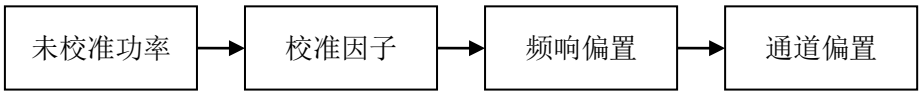
87230 内存有一个数字滤波器，用于平均功率读数。平均次数范围为 1 到 1024。如果打开平均状态，则将增加测量时间。

对于 87230，可以设置为自动平均状态，即根据不同的功率电平和显示分辨率，设置不同的平均次数，一般地，在功率越小时，平均次数越大。

相关程控命令:
SENSe:AVERage[:STATe] <开关> 设置平均开关
例如打开探头的平均开关:
SENS:AVER 1
关闭探头的平均。
AVER 0
SENSe:AVERage:COUNt:AUTO <开关> 设置自动平均开关
SENSe:AVERage:COUNt <平均次数> 设置平均次数

2.10 设置偏置

87230 可以补偿测试设备里的信号衰减或增益（例如补偿 20dB 的衰减器）。通道偏置在显示偏置之前加入到功率测量中。



相关程控命令为:
SENSe:CORRection:GAIN2 <偏置值> 设置通道偏置值
SENSe:CORRection:GAIN:STATe <开关> 设置通道偏置开关
例如设置探头的通道偏置为 3dB。
SENS:CORR:GAIN2 3

2.11 设置测量极限

通过设置测量极限，验证测量功率是否超出给定的范围。

相关程控命令为：

:CALCulate:LIMit:LOWer[:DATA] <下限> 设置下限

:CALCulate:LIMit:UPPer[:DATA] <上限> 设置上限

:CALCulate:LIMit:STATe <开关> 设置极限检测开关

:CALCulate:LIMit:FAIL? 查询是否超出极限

:CALCulate:LIMit:FCOunt? 查询是否超出极限的次数（FCO）

CALCulate:LIMit:CLEar[:IMMediate] 清零失败计数（FCO）。

2.12 状态报告

状态报告用于检测功率的错误信息、操作状态、可疑状态等。状态报告采用逐级上报的原则，如校准操作上报到操作状态，操作状态上报到状态字。

如查询探头是否正在校准，查询探头是否正在校准完毕，校零校准是否出错，是否需要校零，是否检测到测量超出范围等。以下就“查询是否正在校准探头”给出说明。

STATus:OPERation:CALibrating 的寄存器组中，第 1 位表示探头的校准状态。

设置校准操作的事件发生为：未校准（状态 0）到校准（状态 1）跳变时检测事件，即设置正跳变过滤器为 1。

STAT:OPER:CAL:PTR 2 （配置探头的校准事件检测为 0 到 1）

将校准操作事件报告到其上一级操作状态寄存器中。即 STATus:OPERation 寄存器组的校准状态位（第 0 位）。

STAT:OPER:CAL:ENAB 2 （如果需要报告探头的事件，参数可以是 4，或者 6）。

同理，如果需要再将操作状态进一步上报上“状态字”的第 7 位，需要设置操作正跳变过滤器的相应位（位 0）和使能寄存器的相应位（位 0）。

STAT:OPER:CAL:PTR 1 （配置校准事件检测为 0 到 1）

STAT:OPER:ENAB 1 （如果需要报告下限检测事件，参数可以是 2048，或者 2049）。

至此，配置完毕，接下来就可查询探头是否正在校准，

方法 1：查询校准操作条件寄存器的位 1，如果返回非 0，表示正在校准。

STAT:OPER:CAL:COND?

方法 2：查询操作条件寄存器的位 0，如果返回非 0，表示正在校准。

STAT:OPER:COND?

方法 3：查询状态字的位 7，如果返回非 0，表示正在校准。

*STB?

2.13 存储调用

为了减少重复设置过程，87230 可以存储 5 种配置数据到非易失性存储器。错误列表、频响偏置表、校零信息等不存储在该配置中。除了错误列表不存储外，其它均存储在硬配置

文件中，不随用户调用而改变。

相关程控命令为：

*SAV <NRf>

*RCL <NRf>

<NRf>的范围为 1 到 5。

3 程控命令

- 命令说明 37
- 通用命令 37
- 仪器子系统命令 41

3.1 命令说明

该章节提供了详细的命令参考信息，便于实现远程控制，具体包括：

- 完整的语法格式及参数列表；
- 对于 SCPI 非标准命令，列出语法图示；
- 详细的功能描述及关联的命令说明；
- 支持的命令格式（设置 或 查询）；
- 参数说明，包含：数据类型、取值范围及默认值（单位）；
- 按键路径；
- 命令兼容的同类仪器中的仪器型号，若不注明则说明当前命令只适用于 87230；
- 其它说明。

通用命令和仪器子系统命令章节部分，首先列出命令顺序列项，方便用户查询使用。
远程控制时，本手册相关的命令后缀说明如下表：

表3.1 命令后缀说明

后 缀	取值范围	描 述
<ch>	1	通道
<m>	1	测量
<w>	1	窗口
<t>	1	频响偏置表

3.2 通用命令（IEEE488.2 命令）

通用命令用来控制仪器状态寄存器、状态报告、同步、数据存储及其它通用功能，通用命令的用法和作用适用不同的仪器。所有的通用命令都可以通过命令字中的第一个“*”被识别，在 IEEE488.2 中详细定义了这些通用命令。
以下是 IEEE488.2 通用命令的解释和说明。

提 示

命令使用：
如不特别说明，命令可用于设置或者查询。
若一个命令仅用于设置或查询，或者启动一个事件，命令说明中会单独进行说明。

3.2 通用命令（IEEE488.2 命令）

*CLS

功能: 清空仪器状态数据结构，包括 SCPI 寄存器（如疑问状态、操作状态等）、标准事件寄存器、状态字、错误/事件队列。

查询: 不支持

设置: *CLS

举例: *CLS 清空仪器状态

错误信息: 无

复位状态: 无

*ESE

功能: 查询或设置标准事件状态使能寄存器。0 禁止，1 使能。

查询: *ESE?

设置: *ESE <NRf>

NRf 表示数值，2 的倍数。位映射见表 3.2。

举例: *ESE? 查询该寄存器的当前设置。返回格式为<NR1>，0 ~ 255

*ESE 60 使能 4+8+16+32 相应位

表3.2 标准事件位映射

位	值	说明
0	1	操作完成
1	2	未用
2	4	查询错误
3	8	设备相关错误
4	16	执行错误
5	32	命令错误
6	64	未用
7	128	未用

*ESR?

功能: 查询标准事件状态寄存器的值，并清 0 该寄存器。见表 3.2

查询: *ESR?

设置: 不支持

举例: *ESR? 查询标准事件状态寄存器的值，并清零。

错误信息: 无

复位状态: 无

*IDN?

功能: 查询 87230 的标识串。

查询: *IDN?

设置: 不支持

举例: *IDN?
错误信息: 无
复位状态: 无

*OPC

功能: 当所有等待执行的操作完成时，设置标准事件状态寄存器中的操作结束位。
查询: *OPC?
设置: *OPC
举例: *OPC? 如果等待的操作完成时，返回 1，否则等待。
错误信息: 无
复位状态: 无

*RCL

功能: 调用指定的存储调用寄存器中的仪器状态。
查询: 不支持
设置: *RCL <NRf>
范围为 1 到 5
举例: *RCL 3
错误信息: 如果寄存器不位于 1 到 5，则提示“-222,"数据超范围"”。
复位状态: 无

*RST

功能: 复位仪器，请参考 SYSTem:PRESet.
查询: 不支持
设置: *RST

*SAV

功能: 存储仪器状态到指定的寄存器中。
查询: 不支持
设置: *SAV <NRf>
范围为 1 到 5
举例: *SAV 3
错误信息: 如果寄存器不位于 1 到 5，则提示“-222,"数据超范围"”。
复位状态: 无

*SRE

功能: 查询或设置服务请求寄存器。0 禁止，1 使能。
查询: *SRE?
设置: *SRE <NRf>
NRf 表示数值，2 的倍数。位映射见表 3.3。
举例: *SRE? 查询该寄存器的当前设置。返回格式为<NR1>，0 ~ 255

*SRE 316　　分别设置第 2，3，4，5，8 位（4+8+16+32+256）

表3.3　服务请求寄存器位映射

位	值	说明
0	1	未用
1	2	未用
2	4	设备相关
3	8	疑问状态
4	16	信息接收
5	32	事件状态位
6	64	必须为 0
7	128	操作状态

*STB?

功能:　查询状态字。
查询:　*STB?
设置:　不支持
举例:　*STB?
错误信息: 无
复位状态: 无
位映射如下:

表3.4　状态字

位	值	说明
0	1	未用
1	2	设备相关
2	4	错误/事件队列
3	8	疑问状态
4	16	信息接收
5	32	事件状态位
6	64	服务请求
7	128	操作状态

*TRG

功能:　触发正在等待触发的所有通道。
查询:　不支持
设置:　*TRG
举例:　*TRG
错误信息: 如果触发源不为 BUS，则提示“-211, "触发忽略"”。
复位状态: 无

*TST?

功能: 执行自测试，自测试时间较长。
查询: *TST?
设置: 不支持
举例: *TST? 返回 0 时表示通过，1 表示失败。
错误信息: 无
复位状态: 无

*WAI

功能: 使仪器处于等待状态，直到以下任意一种满足:
所有等待的操作已经完成
接收到设备清除命令
重启。
查询: 不支持
设置: *WAI
举例: *WAI
错误信息: 无
复位状态: 无

3.3 仪器子系统命令

该章节详细说明87230的子系统命令。

- 运算子系统命令（CALCulate）.....41
- 校准子系统命令（CALibration）.....46
- 测量子系统命令（CONFigure/FETCh/READ/MEASure）.....46
- 格式子系统命令（FORMat）.....49
- 内存子系统命令（MEMory/MMEMory）.....50
- 传感子系统命令（SENSe）.....54
- 状态子系统命令（STATus）.....58
- 系统子系统命令（SYSTem）.....86
- 触发子系统命令（INITiate/TRIGger）.....87
- 单位子系统命令（UNIT）.....89
- 服务子系统命令（SERVice）.....90

3.3.1 运算子系统命令（CALCulate）

运算（CALCulate）子系统用于后级数据处理，显示偏置和相对测量均在开关打开时才运算。

命令包括:

- :CALCulate[1]:LIMit:CLEar:AUTO.....42
- :CALCulate[1]:LIMit:CLEar[:IMMediate].....42
- :CALCulate[1]:LIMit:FAIL?.....42
- :CALCulate[1]:LIMit:FCOunt?.....42

- :CALCulate[1]:LIMit:LOWer[:DATA]....43
- :CALCulate[1]:LIMit:STATe....44
- :CALCulate[1]:LIMit:UPPer[:DATA]....44
- :CALCulate[1]:MATH[:EXPRession]....45
- :CALCulate[1]:MATH[:EXPRession]:CATalogue?....45
- :CALCulate[1]:RELative[:MAGNitude]:AUTO....45
- :CALCulate[1]:RELative[:MAGNitude]:VALue?....45
- :CALCulate[1]:RELative:STATe....46

:CALCulate[1]:LIMit:CLEar:AUTO

功能: 控制何时清零极限 FCO（失败计数）。

查询: :CALCulate[1]:LIMit:CLEar:AUTO?
对于 ONCE 状态，如果没有开始测量，则返回 1，否则返回 0
对于 OFF 状态，始终返回 0
对于 ON 状态，如果开始测量，返回 1，否则返回 0

设置: :CALCulate[1]:LIMit:CLEar:AUTO <布尔数据> | ONCE | 2
<布尔数据>的有效形式为：0，OFF，1，ON。
对于“ON”，当进行如下操作时，FCO 设置为 0:
用 INITiate[:IMMediate]命令初始化;
用 INITiate:CONTinuous ON 命令初始化;
用 MEASure?命令测量;
用 READ?命令读测量。
对于“OFF”，FCO 不清零。
对于“ONCE”或 2，仅在第一次初始化时清零，然后在遇到极限检测失败时累加。

举例: CALC:LIM:CLE:AUTO? 查询 FCO 清零状态
CALC:LIM:CLE:AUTO ONCE 设置第一次初始化时清零 FCO。

复位状态: 设置为开“ON”。

:CALCulate[1]:LIMit:CLEar[:IMMediate]

功能: 清零 FCO（失败计数），FCO 可由 CALCulate[1]:LIMit:FCOunt?查询得到。

查询: 不支持

设置: :CALCulate[1]:LIMit:CLEar[:IMMediate]

举例: :CALC:LIM:CLE 清零失败计数。

:CALCulate[1]:LIMit:FAIL?

功能: 查询是否有超过极限，1 表示有，0 表示没有。

查询: :CALCulate[1]:LIMit:FAIL?

设置: 不支持

举例: :CALC:LIM:FAIL? 查询检测检测状态。

:CALCulate[1]:LIMit:FCOunt?

功能: 查询极限检测失败计数（FCO）。

如下情形，均清零 FCO:

a) 复位

b) CALCulate[1]:LIMit:CLEar:IMMediate

c) CALCulate[1]:LIMit:CLEar:AUTO ON

查询: 查询：:CALCulate[1]:LIMit:FCOunt?

设置: 不支持

举例: :CALC:LIM:FCO? 查询检测检测失败计数。

:CALCulate[1]:LIMit:LOWer[:DATA]

功能: 查询或设置测量极限的下限。

查询: :CALCulate[1]:LIMit:LOWer[:DATA]? [MIN|MAX]

设置: :CALCulate[1]:LIMit:LOWer[:DATA] <数值数据>

<数值数据>的形式：DEF，MIN，MAX，NRf，其中，DEF 仅用于设置。

如果当前设置的下限值超过或等于上限值，则根据单位不同，自动调整上限值，如表 3.7 所示。

举例: CALC:LIM:LOW? 查询极限下限。

CALC:LIM:LOW 0.2 根据显示单位，设置极限下限为:

dBm 时，0.2dBm

W 时，200mW

dB 时，0.2dB

%时，0.2%

复位状态: 设置为-90dBm 或-90dB

表3.5 测量单位

测量模式	测量类型	CALC:REL:STAT OFF		CALC:REL:STAT ON	
		线性	对数	线性	对数
单通道	均值，峰值	瓦特	dBm	%	dB
	峰均比	%	dB	%	dB
比值	均值，峰值，峰均比	%	dB	%	dB
差值	均值，峰值	瓦特	dBm	%	dB
	峰均比	%	dB	%	dB

表3.6 极限范围

	瓦特	dBm	%	dB
DEF	1pW	-90	100p%	-120
MIN	1aW	-150	100a%	-180
MAX	1XW	200	100X%	180

表3.7　极限范围调整

下限值	上限值			
	瓦特	dBm	%	dB
$nLow$	$nLow \times 10^{0.001}$	$nLow + 0.01$	$nLow \times 10^{0.001}$	$nLow + 0.01$

:CALCulate[1]:LIMit:STATe

功能:　查询或设置测量极限检测开关。
查询:　:CALCulate[1]:LIMit:STATe?
设置:　:CALCulate[1]:LIMit:STATe <布尔数据>
<布尔数据>的有效形式为：0，OFF，1，ON
举例:　CALC:LIM:STAT?　查询极限检测开关状态。
CALC:LIM:STAT ON　打开极限检测开关。
CALC:LIM:STAT 0　禁止极限检测。
限制:
错误信息:
复位状态:　禁止极限检测。

:CALCulate[1]:LIMit:UPPer[:DATA]

功能:　查询或设置测量极限的上限。参考 CALCulate[1]:LIMit:LOWer[:DATA]
查询:　:CALCulate[1]:LIMit:UPPer[:DATA]?　[MIN|MAX]
设置:　:CALCulate[1]:LIMit:UPPer[:DATA]　<数值数据>
<数值数据>的形式：DEF，MIN，MAX，NRf，其中，DEF 仅用于设置。
如果当前设置的上限值小于或等于下限值，则根据单位不同，自动调整下限值，如表3.9 所示。
举例:　CALC:LIM:UPP?　查询极限上限。
CALC:LIM:UPP 8　根据显示单位，设置极限上限为：
dBm 时，8dBm
W 时，8W
dB 时，8dB
%时，8%
复位状态:　设置为 90dBm 或 90dB

表3.8　极限范围

	瓦特	dBm	%	dB
DEF	1MW	90	100M%	60
MIN	1aW	-150	100a%	-180
MAX	1XW	200	100X%	180

表3.9 极限范围调整

上限值	下限值			
	瓦特	dBm	%	dB
$nUpp$	$nUpp/10^{0.001}$	$nUpp-0.01$	$nUpp/10^{0.001}$	$nUpp-0.01$

:CALCulate[1]:MATH[:EXPRession]

功能: 查询或设置指定测量的表达式：单通道
查询: :CALCulate[1]:MATH[:EXPRession]?
设置: :CALCulate[1]:MATH[:EXPRession] <字符串>
字符串的形式如下:
"(SENS1)"
举例: CALC1:MATH? 查询测量表达式。
CALC:MATH "(SENS1)" 设置表达式为单通道绝对功率测量。
复位状态: 无

:CALCulate[1]:MATH[:EXPRession]:CATalogue?

功能: 列举所有测量表达式，各个表达式之间由逗号隔开。
字符串是:
"(SENS1)"
查询: :CALCulate[1]:MATH[:EXPRession]:CATalogue?
设置: 不支持
举例: CALC:MATH:CAT? 列出了所有定义的数学表达式。

:CALCulate[1]:RELative[:MAGNitude]:AUTO

功能: 设置用于相对测量的参考值。在 CALCulate 块里，在任意数学计算和显示偏置计算完毕后，相对值才被用于测量信号。该值应设置为 ONCE 以设置参考值用于相对测量。在设置参考值后，指令返回 OFF。设置指令为 ONCE 将指令 CALCulate[1]:RELative:STATe 转变为 ON。
0|OFF 不进行任何操作。
1|ON 无效，仪器返回错误：无效参数。
2|ONCE，有效参数，意义见上。
查询: :CALCulate[1]:RELative[:MAGNitude]:AUTO?
设置: :CALCulate[1]:RELative[:MAGNitude]:AUTO ONCE
举例: CALC:REL:AUTO? 始终返回 0。
CALC:REL:AUTO ONCE 设置相对测量的参考值。
错误信息: 如果参数选择 1 或 ON 时，提示“-224,"无效参数值"”。

:CALCulate[1]:RELative[:MAGNitude]:VALue?

功能: 查询用于相对测量的参考值。
查询: :CALCulate[1]:RELative[:MAGNitude]:VALue?

设置: 不支持

举例: CALC:REL:VAL? 查询相对测量的参考值。

:CALCulate[1]:RELative:STATe

功能: 查询或设置相对测量开关状态。
当打开时，由:CALCulate[1]:RELative[:MAGNitude]:AUTO 设置的相对测量值应用于测量。

查询: :CALCulate[1]:RELative:STATe?

设置: :CALCulate[1]:RELative:STATe <布尔数据>

举例: CALC:REL:STAT? 当相对测量打开时，返回 1，否则返回 0。
CALC:REL:STAT ON 打开相对测量状态。

复位状态: 禁止相对测量。

3.3.2 校准子系统命令（CALibration）

CALibration 子系统用来控制 RF 功率探头和通道的自动零偏设置。如果没有射频信号加载到探头上，可以随时进行校零设置。

命令包括:

- :CALibration[1]:ZERO:AUTO.....46
- :CALibration[1]:ZERO:TYPE.....46

:CALibration[1]:ZERO:AUTO

功能: 校零探头。

查询: 不支持

设置: :CALibration[1]:ZERO:AUTO ONCE

举例: CAL:ZERO:AUTO ONCE 校零探头。

错误信息: 如果校零出错，则提示“-231,"数据疑问;校零错误"”。

:CALibration[1]:ZERO:TYPE

功能: 配置 87230 的校零类型。

查询: :CALibration[1]:ZERO:TYPE?

设置: :CALibration[1]:ZERO:TYPE <字符数据>
<字符数据>的取值如下:
EXTernal 或 EXT 或 0 表示外部校零。
INTernal 或 INT 或 1 表示内部校零。

举例: CAL:ZERO:TYPE INT 设置探头的校零类型为内部。
CAL:ZERO:TYPE? 查询探头的校零类型。

复位状态: 不影响（即保持复位前的状态，下同）

3.3.3 测量子系统命令（CONFigure/FETCh/READ/MEASure）

一般地，如果没有特别说明，线性功率单位为 W，对数功率单位为 dBm，线性功率比单位为%，对数功率比单位为 dB，时间单位为 s。

如果返回的值无效，则返回 NAN（9.91E37），定义见 IEEE 754。

FETCh 命令与 MEASure 命令的区别如下：

FETCh 命令是查询当前测量值。FETCh 命令的执行速度与功率的大小没有关系，一般立即返回，FETCh 命令可能返回的是功率测量过程中的的中间结果，而并非实际功率值。

MEASure 命令是启动一次测量，测量完成后才返回测量结果，一般地，功率越小，MEASure 命令所需的时间越长。

命令包括：

- :CONFigure[1][:SCALar][:POWer][:AC]……47
- :CONFigure[1][:SCALar][:POWer][:AC]:RELative……47
- :FETCh[1][:SCALar][:POWer][:AC]?……47
- :FETCh[1][:SCALar][:POWer][:AC]:RELative?……48
- :MEASure[1][:SCALar][:POWer][:AC]?……48
- :MEASure[1][:SCALar][:POWer][:AC]:RELative? ……48
- :READ[1][:SCALar][:POWer][:AC]?……49
- :READ[1][:SCALar][:POWer][:AC]:RELative?……49

:CONFigure[1][:SCALar][:POWer][:AC]

功能: 查询或设置的功率测量模式。

查询: :CONFigure[1]?

设置: :CONFigure[1][:SCALar][:POWer][:AC] [<期望值>[, <分辨率>[, <源通道列表>]]]

设置为绝对功率测量，相对测量关。其中期望值和分辨率均可以用 DEF 表示，表示不改变。分辨率的范围为 1 到 4；通道列表的形式为：(@1)，表示通道 A。

举例: CONF? 查询功率测量配置。

CONF1 DEF, 3, (@1) 设置为绝对功率测量，分辨率为 3，源通道为 A。

错误信息 如果没有连接探头，则提示“-241,"缺少硬件"”。

复位状态: 设置为测量绝对功率，分辨率为 3。

:CONFigure[1][:SCALar][:POWer][:AC]:RELative

功能: 设置绝对功率测量模式，打开相对测量。

查询: 不支持

设置: :CONFigure[1][:SCALar][:POWer][:AC]:RELative [<期望值>[, <分辨率>[, <源通道列表>]]]

设置为绝对功率测量，相对测量开。其中期望值和分辨率均可以用 DEF 表示，表示不改变。分辨率的范围为 1 到 4；通道列表的形式为：(@1)，表示通道 A。

举例: CONF:REL DEF, 3, (@2) 设置为绝对功率测量，分辨率为 3，源通道为 B，打开相对测量。

错误信息:

:FETCh[1][:SCALar][:POWer][:AC]?

功能: 设置绝对功率测量，并关闭相对测量，返回当前测量值。测量单位由 UNIT[1]:POWer 指定。该命令并不等待一次测量完成后才返回，如果要准确获取准确的测量值，请用

meas 命令。

查询: :FETCh[1][:SCALar][:POWer][:AC]? [<期望值>[, <分辨率>[, <源通道列表>]]]
设置为绝对功率测量，相对测量关。其中期望值和分辨率均可以用 DEF 表示，表示不改变。分辨率的范围为 1 到 4；通道列表的形式为：(@1)，表示通道 A。

设置: 不支持

举例: FETC? 查询绝对功率测量值。

错误信息:

:FETCh[1][:SCALar][:POWer][:AC]:RELative?

功能: 设置绝对功率测量，并打开相对测量，返回测量值。测量单位由 UNIT[1]:POWer:RATio 指定。该命令并不等待一次测量完成后才返回，如果要准确获取准确的测量值，请用 MEAS 命令。

查询: :FETCh[1][:SCALar][:POWer][:AC]:RELative? [<期望值>[, <分辨率>[, <源通道列表>]]]
设置为绝对功率测量，相对测量关。其中期望值和分辨率均可以用 DEF 表示，表示不改变。分辨率的范围为 1 到 4；通道列表的形式为：(@1)，表示通道 A。

设置: 不支持

举例: FETC:REL? 查询相对功率测量值。

错误信息:

:MEASure[1][:SCALar][:POWer][:AC]?

功能: 设置绝对功率测量，并关闭相对测量，返回测量值。测量单位由 UNIT[1]:POWer 指定。启动一次测量，测量完成后返回测量值。而相应的 FETCh 命令是返回当前测量值，而不是测量完成后返回。在组成测试系统时，请用 MEASure 命令。

查询: :MEASure[1][:SCALar][:POWer][:AC]? [<期望值>[, <分辨率>[, <源通道列表>]]]
设置为绝对功率测量，相对测量关。其中期望值和分辨率均可以用 DEF 表示，表示不改变。分辨率的范围为 1 到 4；通道列表的形式为：(@1)，表示通道 A。

设置: 不支持

举例: MEAS? 启动一次测量，测量完成后返回绝对功率测量值。

错误信息:

:MEASure[1][:SCALar][:POWer][:AC]:RELative?

功能: 设置绝对功率测量，并打开相对测量，返回测量值。测量单位由 UNIT[1]:POWer 指定。启动一次测量，测量完成后返回测量值。而相应的 FETCh 命令是返回当前测量值，而不是测量完成后返回。在组成测试系统时，请用 MEASure 命令。

查询: :MEASure[1][:SCALar][:POWer][:AC]:RELative? [<期望值>[, <分辨率>[, <源通道列表>]]]
设置为绝对功率测量，相对测量关。其中期望值和分辨率均可以用 DEF 表示，表示不改变。分辨率的范围为 1 到 4；通道列表形式为：(@1)，表示通道 A。

设置: 不支持

举例: MEAS:REL? 启动一次测量，测量完成后返回相对功率测量值。

错误信息:

:READ[1][:SCALar][:POWer][:AC]?

功能: 设置绝对功率测量，并关闭相对测量，返回测量值。测量单位由 UNIT[1]:POWer 指定。

查询: :READ[1][:SCALar][:POWer][:AC]? [<期望值>[, <分辨率>[, <源通道列表>]]]
设置为绝对功率测量，相对测量关。其中期望值和分辨率均可以用 DEF 表示，表示不改变。分辨率的范围为 1 到 4；通道列表的形式为：(@1)，表示通道 A。

设置: 不支持

举例: READ? 查询绝对功率测量值。

错误信息:

:READ[1][:SCALar][:POWer][:AC]:RELative?

功能: 设置绝对功率测量，并打开相对测量，返回测量值。测量单位由 UNIT[1]:POWer:RATio 指定。

查询: :READ[1][:SCALar][:POWer][:AC]:RELative? [<期望值>[, <分辨率>[, <源通道列表>]]]
设置为绝对功率测量，相对测量关。其中期望值和分辨率均可以用 DEF 表示，表示不改变。分辨率的范围为 1 到 4；通道列表的形式为：(@1)，表示通道 A。

设置: 不支持

举例: READ:REL? 查询相对功率测量值。

错误信息:

3.3.4 格式子系统命令（FORMat）

- :FORMat[:READings]:BORDer 49
- :FORMat[:READings][:DATA] .....
  :\DISPlay:CLEar 49

:FORMat[:READings]:BORDer

功能: 查询或设置二进制数据的传输顺序：正常还是字节交换。
仅在 FORMat[:READings][:DATA]设置为 REAL 时有效。

查询: :FORMat[:READings]:BORDer?

设置: :FORMat[:READings]:BORDer <字符数据>
<字符数据>的有效值为：0 或 NORMal，1 或 SWAPped。

举例: FORM:BORD? 查询传输顺序。
FORM:BORD SWAP 设置传输顺序为字节交换。

复位状态: 设置为正常 NORMal

:FORMat[:READings][:DATA]

功能: 查询或设置数据传输格式：ASCii 和 REAL（实数）

查询: :FORMat[:READings][:DATA]?

设置: :FORMat[:READings][:DATA] <字符数据>
<字符数据>的有效值为：0 或 ASCii，1 或 REAL。
ASCii：数值数据的传输格式为<NRf>形式。
REAL：数值数据的传输格式为 IEEE 754 64，并且每个数据 8 个字节。

举例: FORM? 查询数据传输格式。
FORM REAL 设置数据传输格式为 REAL。

复位状态: 设置为 ASCii

3.3.5 内存子系统命令（MEMory/MMEMory）

- :MEMory:CATalog[:ALL]?.....50
- :MEMory:CATalog:STATe?.....50
- :MEMory:CATalog:TABLe?.....51
- :MEMory:CLEar[:NAME].....51
- :MEMory:CLEar:TABLe[1].....52
- :MEMory:FREE[:ALL]?.....52
- :MEMory:FREE:STATe?.....52
- :MEMory:FREE:TABLe?.....52
- :MEMory:NSTates?.....52
- :MEMory:STATe:CATalog?.....52
- :MEMory:TABLe[1]:DEFine.....52
- :MEMory:TABLe[1]:FREQuency.....53
- :MEMory:TABLe[1]:FREQuency:POINts?.....53
- :MEMory:TABLe[1]:GAIN[:MAGNitude].....53
- :MEMory:TABLe[1]:GAIN[:MAGNitude]:POINts?.....53
- :MEMory:TABLe:MOVE.....54
- :MEMory:TABLe:SELect.....54

:MEMory:CATalog[:ALL]?

功能: 列举仪器中的用户配置，包括存储调用配置、频响偏置表（FDO）等。
数据格式为：<数值 1>, <数值 2>{, <字符串>}
其中，<数值 1>表示用户配置已用的字节长度，<数值 2>表示未用的字节长度。
每个<字符串>的形式如下:
<字符串 i>, <类型>, <长度>
<字符串 i>表明用户配置的名称。
<类型>指出用户配置的类型，TABL 表示频响偏置表，STAT 表示存储调用配置。
<长度>是指该配置项的字节长度。

查询: :MEMory:CATalog[:ALL]?

设置: 不支持

举例: MEM:CAT? 列举仪器中的所有用户配置。

复位状态: 无

:MEMory:CATalog:STATe?

功能: 列举仪器中存储调用配置。
数据格式为：<数值 1>, <数值 2>{, <字符串>}
其中，<数值 1>表示用户配置已用的字节长度，<数值 2>表示未用的字节长度。

每个<字符串>的形式如下:
<字符串 i>, <类型>, <长度>
<字符串 i>表明用户配置的名称。
<类型>指出用户配置的类型，STAT 表示存储调用配置。
<长度>是指该配置项的字节长度。

查询: :MEMory:CATalog:STATe?

设置: 不支持

举例: MEM:CAT:STAT? 列举仪器中的存储调用配置。

:MEMory:CATalog:TABLe?

功能: 列举仪器中频响偏置表。
数据格式为: <数值 1>, <数值 2>{, <字符串>}
其中，<数值 1>表示用户配置已用的字节长度，<数值 2>表示未用的字节长度。
每个<字符串>的形式如下:
<字符串 i>, <类型>, <长度>
<字符串 i>表明用户配置的名称。
<类型>指出用户配置的类型，TABL 表示频响偏置表。
<长度>是指该配置项的字节长度。

查询: :MEMory:CATalog:TABLe?

设置: 不支持

举例: MEM:CAT:TABL? 列举仪器中的频响偏置表。

:MEMory:CLEar[:NAME]

功能: 用于清零仪器指定的频响偏置表或者存储调用表。

查询: 不支持

设置: :MEMory:CLEar[:NAME] <字符串>
<字符串>表示频响偏置表或者存储调用表的名称。

举例: MEM:CLE "fdo0" 清空 fdo0 频响偏置表。

说明: 1）如果指定的频响偏置表处于启用状态，则自动关闭。
2）存储调用状态的英文名称为“State1”,“State2”……“State5”，名称不能改变。存储调用状态的中文名称为“状态 1”,“状态 2”……“状态 5”，名称不能改变。
3）如果指定的名称在频响偏置列表中能找到，则清空该频响偏置表;
4）如果指定的名称在存储调用状态列表中能找到，则清空该状态表。
5）如果存在“State1”的频响偏置，则调用 MEM:CLE “State1”，将清空频响偏置表和状态表。
6）程控设置中，存储调用的状态名与界面的语言切换没有关系，即如下两条命令均清空状态 1（推荐使用英文名称）:
MEM:CLE "状态 1"
MEM:CLE "State1"

错误信息: 如果指定的名称不存在，则提示“-224,"无效参数值"”。

复位状态: 无

3.3 仪器子系统命令

:MEMory:CLEar:TABLe[1]

功能: 清空指定的频响偏置表。
查询: 不支持
设置: :MEMory:CLEar:TABLe[1]
举例: MEM:CLE:TABL 清空频响偏置表（共有 10 个）。

:MEMory:FREE[:ALL]?

功能: 查询用户配置空间未用的字节总数，以及已用的字节数。
返回串的形式为：<未用字节数>, <已用字节数>
查询: :MEMory:FREE[:ALL]?
设置: 不支持
举例: MEM:FREE?

:MEMory:FREE:STATe?

功能: 查询存储调用空间未用的字节总数，以及已用的字节数。
返回串的形式为：<未用字节数>, <已用字节数>
查询: :MEMory:FREE:STATe?
设置: 不支持
举例: MEM:FREE:STAT?

:MEMory:FREE:TABLe?

功能: 查询频响偏置表空间未用的字节总数，以及已用的字节数。
返回串的形式为：<未用字节数>, <已用字节数>
查询: :MEMory:FREE:TABLe?
设置: 不支持
举例: MEM:FREE:TABL?

:MEMory:NSTates?

功能: 查询存储调用状态个数，始终返回 10。
查询: :MEMory:NSTates?
设置: 不支持
举例: MEM:NST?

:MEMory:STATe:CATalog?

功能: 列举所有存储调用状态的名称。
查询: :MEMory:STATe:CATalog?
设置: 不支持
举例: MEM:STAT:CAT?

:MEMory:TABLe[1]:DEFine

功能: 查询或设置指定频响偏置表的名称。

查询: :MEMory:TABLe[1]:DEFine?
设置: :MEMory:TABLe[1]:DEFine <字符串>
举例: MEM:TABL:DEF? 查询频响偏置表的名称。
MEM:TABL:DEF "fdo0" 将频响偏置表命名为 fdo0。
错误信息: 如果指定的名称已经存在，则提示“-257,"文件名错误"”。

:MEMory:TABLe[1]:FREQuency

功能: 查询或设置指定频响偏置表的频率列表。
设置频率列表时，以前的频率列表将被清空，并且频率列表必须以升序排列。
相关命令为:MEMory:TABLe[1]:GAIN[:MAGNitude]
查询: :MEMory:TABLe[1]:FREQuency?
设置: :MEMory:TABLe[1]:FREQuency <数值数据 1>{, <数值数据 n>}
需要确保该频率列表覆盖探头的频率范围，如果被测信号的频率在该列表的范围之外，则将利用最低或最高频率处的值。
最大频率点数为 80。
频率的有效单位有：Hz，kHz，MHz，GHz
举例: MEM:TABL:FREQ? 查询频响偏置表的频率列表。
MEM:TABL:FREQ 50MHz, 40GHz 设置频响偏置表频率为 50MHz，40GHz
错误信息: 如果频率列表中的点数超过 80，则提示“-108,"参数不允许"”。
如果频率列表不是以递增的顺序排列，则提示“-220,"参数错误"”。
复位状态: 无

:MEMory:TABLe[1]:FREQuency:POINts?

功能: 查询频响偏置表的频率点数。
查询: :MEMory:TABLe[1]:FREQuency:POINts?
设置: 不支持
举例: MEM:TABL:FREQ:POIN? 查询频响偏置表的频率点数。

:MEMory:TABLe[1]:GAIN[:MAGNitude]

功能: 查询或设置指定频响偏置表的幅度增益列表。
设置增益列表时，以前的增益列表将被清空，
相关命令为:MEMory:TABLe[1]:FREQuency
查询: :MEMory:TABLe[1]:GAIN[:MAGNitude]?
设置: :MEMory:TABLe[1]:GAIN[:MAGNitude] <数值数据 1>{, <数值数据 n>}
最大增益点数为 80。
单位为 PCT，即 100 表示 100%
幅度增益的范围为 1.0e-009 到 1.0e+009，即-90dB 到 90dB。
举例: MEM:TABL:GAIN? 查询频响偏置表的增益列表。
MEM:TABL:GAIN 98, 102 设置频响偏置表的增益列表为 98%，102%。

:MEMory:TABLe[1]:GAIN[:MAGNitude]:POINts?

功能: 查询频响偏置表的幅度增益点数。

查询: :MEMory:TABLe[1]:GAIN[:MAGNitude]:POINts?
设置: 不支持
举例: MEM:TABL:GAIN:POIN? 查询频响偏置表的幅度增益点数。

:MEMory:TABLe:MOVE

功能: 重命名指定频响偏置表。该命令需要事先知道需要修改的频响偏置表的名称。
相关命令 MEMory:TABLe[1]:DEFine
推荐用 MEMory:TABLe[1]:DEFine。
查询: 不支持
设置: :MEMory:TABLe:MOVE <字符串 1>, <字符串 2>
<字符串 1>指出需要修改的频响偏置表的名称,
<字符串 2>指出修改后的频响偏置表的名称。
举例: MEM:TABL:MOVE "fdo0", "fdo1"
错误信息: 如果第一参数指定的频响偏置表不存在，则提示“-256,"文件名未找到"”。
如果第二参数指定的频响偏置表已经存在，则提示“-257,"文件名错误"”。
如果第二参数的长度超过 16 个字符，则提示“-257,"文件名错误"”。

:MEMory:TABLe:SELect

功能: 查询或设置当前频响偏置表。这与 AV2434 不同。AV2434 中支持探头校准表，可以用该命令选择探头校准表。而本案中，仅支持频响偏置表。
查询: :MEMory:TABLe:SELect?
设置: :MEMory:TABLe:SELect <字符串>
举例: MEM:TABL:SEL? 返回当前频响偏置表的名称。
MEM:TABL:SEL "fdo0" 设置当前频响偏置表为 fdo0。
错误信息: 如果由参数指定的频响偏置表不存在，则提示“-224,"无效参数值"”。

3.3.6 传感子系统命令（SENSe）

- [:SENSe[1]:AVERage[1]:COUNt....55
- [:SENSe[1]:AVERage:COUNt:AUTO....55
- [:SENSe[1]:AVERage:SDETect....55
- [:SENSe[1]:AVERage[1][:STATe]....55
- [:SENSe[1]:CORRection:CSET2[:SELect]....56
- [:SENSe[1]:CORRection:CSET:STATe....56
- [:SENSe[1]:CORRection:FDOFfset[:INPut][:MAGNitude]?....56
- [:SENSe[1]:CORRection:GAIN[1]|2|3|4:INPut][:MAGNitude]....56
- [:SENSe[1]:CORRection:GAIN[1]|2|3|4:INPut]:STATe....57
- [:SENSe[1]:FREQuency[:CW|FIXed]....57
- [:SENSe[1]:POWer:AC:RANGe....57
- [:SENSe[1]:POWer:AC:RANGe:AUTO....58

[:SENSe[1]:AVERage[1]:COUNt

功能: 查询或设置通道平均次数。
查询: [:SENSe[1]:AVERage[1]:COUNt? [MIN|MAX]
设置: [:SENSe[1]:AVERage[1]:COUNt <数值数据>
数值数据的有效值为：DEF，MIN，MAX，<NRf>，其中 DEF 和<NRf>仅用于设置。
<NRf>的范围为 1 到 1024，
DEF 为 8，
MIN 为 1，
MAX 为 1024。
举例: SENS1:AVER:COUN? 查询平均次数。
SENS1:AVER:COUN? MAX 查询可设置的平均次数的最大值。
SENS:AVER:COUN 28 设置平均次数为 28。
复位状态: 平均次数设置 8。

[:SENSe[1]:AVERage:COUNt:AUTO

功能: 查询或设置通道自动平均状态。该功能仅对连续波探头有效。
查询: [:SENSe[1]:AVERage:COUNt:AUTO?
设置: [:SENSe[1]:AVERage:COUNt:AUTO <布尔数据>
<布尔数据>的有效形式为：0，OFF，1，ON。
举例: SENS1:AVER:COUN:AUTO? 查询自动平均状态。
SENS:AVER:COUN:AUTO 1 打开自动平均。
限制: 仅对连续波探头有效。
复位状态: 对于连续波探头，自动平均设置为开。

[:SENSe[1]:AVERage:SDETect

功能: 查询或设置通道步进检测状态。该功能仅对连续波探头有效。
在自动平均模式，最近 4 次的平均测量值用于跟整个滤波器的值相比较，当二者的平均差超过 15%时，清空数字滤波器。然后滤波器开始存储新的测量值。该功能的用途在于，当功率变化比较明显时，能够缩短滤波时间。
查询: [:SENSe[1]:AVERage:SDETect?
设置: [:SENSe[1]:AVERage:SDETect <布尔数据>
<布尔数据>的有效形式为：0，OFF，1，ON。
举例: SENS1:AVER:SDET? 查询步进检测状态。
SENS:AVER:SDET 1 打开步进检测。
限制: 仅对连续波探头有效。
复位状态: 对于连续波探头，步进检测设置为开。

[:SENSe[1]:AVERage[1][:STATe]

功能: 查询或设置通道平均开关状态。
查询: [:SENSe[1]:AVERage[1][:STATe]?
设置: [:SENSe[1]:AVERage[1][:STATe] <布尔数据>

<布尔数据>的有效形式为：0，OFF，1，ON。

举例：SENS1:AVER? 查询平均开关状态。

SENS:AVER 1 打开平均开关。

复位状态：平均状态设置为开。

[:SENSe[1]:CORRection:CSET2[:SELect]

功能：本案中仅支持[:SENSe[1]:CORRection:CSET2[:SELect]。选择频响偏置表。

频响偏置表的数据两个通道共享，但开关状态是分开的。

查询：[:SENSe[1]:CORRection:CSET2[:SELect]?

设置：[:SENSe[1]:CORRection:CSET2[:SELect] <字符串>

<字符串>指出频响偏置表的名称。

举例：SENS:CORR:CSET2? 查询当前选择的频响偏置表。

SENS:CORR:CSET2 "fdo0" 选择名称为“fdo0”的频响偏置表。

错误信息：如果参数指定的频响偏置表不存在，则提示“-256,"文件名未找到"”。

复位状态：无影响。

[:SENSe[1]:CORRection:CSET:STATe

功能：本案中仅支持[:SENSe[1]:CORRection:CSET:STATe。查询或设置频响偏置表的启用状态。

频响偏置表的数据通道共享，但启用状态是分开的。

查询：[:SENSe[1]:CORRection:CSET:STATe?

设置：[:SENSe[1]:CORRection:CSET:STATe <布尔数据>

<布尔数据>的有效形式为：0，OFF，1，ON。

举例：SENS1:CORR:CSET:STAT? 查询频响偏置表启用状态。

SENS:CORR:CSET:STAT 0 禁止频响偏置表。

错误信息：如果当前选择的频响偏置表中的频率点数和幅度增益（偏置）点数不同，则提示“-226,"列表长度不相同"”。

如果当前选择的频响偏置表点数为 0，则提示“-221,"设置冲突"”。

复位状态：不影响频响偏置表的启用状态。

[:SENSe[1]:CORRection:FDOFfset[:INPut][:MAGNitude]?

功能：查询当前使用的频响偏置因子。单位为 PCT，即 100 表示 100%。

查询：[:SENSe[1]:CORRection:FDOFfset[:INPut][:MAGNitude]?

设置：不支持

举例：CORR:FDOF? 查询当前使用的频响偏置因子。

[:SENSe[1]:CORRection:GAIN[1]|2|3|4:INPut][:MAGNitude]

功能：GAIN[1]的等价形式为 CFACtor，表示探头的校准因子。前者符合 SCPI 规范，后者不符合。本案不支持 GAIN[1];

GAIN2 表示通道偏置。本案支持。既可查询，也可设置。

GAIN3 的等价形式为 DCYCle，表示占空比。本案不支持 GAIN3。

GAIN4 的等价形式为 FDOFfset，表示频响偏置因子。本案支持 GAIN4。频响偏置因子

仅提供查询。

查询: [:SENSe[1]:CORRection:GAIN2|4[:INPut][:MAGNitude]? [MIN|MAX]

设置: [:SENSe[1]:CORRection:GAIN2[:INPut][:MAGNitude] <数值数据>
数值数据的有效值为：DEF，MIN，MAX，<NRf>，其中 DEF 和<NRf>仅用于设置。
<NRf>的范围为-100 到 100dB，
DEF 为 0dB，
MIN 为-100dB，
MAX 为 100dB。

举例:

SENS:CORR:GAIN2?	查询通道偏置。
CORR:GAIN2? MAX	查询最大可设置的通道偏置值。
SENS1:CORR:GAIN4?	查询频响偏置值。
CORR:GAIN2 3.6	设置通道偏置为 3.6dB。

[:SENSe[1]:CORRection:GAIN[1]|2|3|4:INPut]:STATe

功能: GAIN[1]的等价形式为 CFACtor，表示探头的校准因子。前者符合 SCPI 规范，后者不符合。本案不支持 GAIN[1];
GAIN2 表示通道偏置。
GAIN3 的等价形式为 DCYCle，表示占空比。本案不支持 GAIN3。
GAIN4 的等价形式为 FDOFfset，表示频响偏置因子。本案不支持 GAIN4 状态的查询与设置。

查询: [:SENSe[1]:CORRection:GAIN2[:INPut]:STATe?

设置: [:SENSe[1]:CORRection:GAIN2[:INPut]:STATe <布尔数据>
<布尔数据>的有效形式为：0，OFF，1，ON。

举例:

SENS:CORR:GAIN2:STAT?	查询通道偏置启用状态。
CORR:GAIN2:STAT 1	启用通道偏置。

[:SENSe[1]:FREQuency[:CW|FIXed]

功能: 查询或设置频率。

查询: [:SENSe[1]:FREQuency[:CW|FIXed]? [MIN|MAX]

设置: [:SENSe[1]:FREQuency[:CW|FIXed] <数值数据>
数值数据的有效值为：DEF，MIN，MAX，<NRf>，其中 DEF 和<NRf>仅用于设置。
<NRf>的范围为 1e3 到 1e12，
DEF 为 50MHz，
MIN 为 1kHz，
MAX 为 1000GHz。

举例:

FREQ?	查询频率。
FREQ? MAX	查询最大可设置的频率。
SENS:FREQ 8GHz	设置频率为 8GHz

复位状态: 设置为 50MHz。

[:SENSe[1]:POWer:AC:RANGe

功能: 查询或设置探头的量程。该功能仅对连续波探头有效。0 表示低量程，1 表示高量程。

当用该命令设置量程后，自动量程状态变为手动，
即[SENSe[1]|SENSe:POWer:AC:RANGe:AUTO 设置为 OFF。

查询: [:SENSe[1]:POWer:AC:RANGe?

设置: [:SENSe[1]:POWer:AC:RANGe <0|1>

举例: SENS:POW:AC:RANG? 查询当前量程。
POW:AC:RANG 1 设置量程为高量程。

限制: 仅对连续波探头有效。

复位状态: 量程设置为高量程。

[:SENSe[1]:POWer:AC:RANGe:AUTO

功能: 查询或设置探头的自动量程开关状态。该功能仅对连续波探头有效。0 表示手动，1 表示自动。
当自动量程打开时，仪器根据功率信号自动调整量程，用最佳量程测量。相关命令:[:SENSe[1]:POWer:AC:RANGe。

查询: [:SENSe[1]:POWer:AC:RANGe:AUTO?

设置: [:SENSe[1]:POWer:AC:RANGe:AUTO <布尔数据>
<布尔数据>的有效形式为：0，OFF，1，ON。

举例: SENS:POW:AC:RANG:AUTO? 查询自动量程开关。
POW:AC:RANG:AUTO 0 禁止自动量程。

限制: 仅对连续波探头有效。

复位状态: 设置为自动量程。

3.3.7 状态子系统命令（STATus）

状态子系统命令通过监控设备状态寄存器、操作状态寄存器和疑问问题寄存器，检测仪器的状态。

- :STATus:DEVice:CONDition?.....73
- :STATus:DEVice:ENABle.....73
- :STATus:DEVice[:EVENt]?.....74
- :STATus:DEVice:NTRansition.....74
- :STATus:DEVice:PTRansition.....74
- :STATus:OPERation:CALibrating[:SUMMary]:CONDition?.....74
- :STATus:OPERation:CALibrating[:SUMMary]:ENABle.....75
- :STATus:OPERation:CALibrating[:SUMMary][:EVENt]?.....75
- :STATus:OPERation:CALibrating[:SUMMary]:NTRansition.....75
- :STATus:OPERation:CALibrating[:SUMMary]:PTRansition.....75
- :STATus:OPERation:CONDition?.....75
- :STATus:OPERation:ENABle.....76
- :STATus:OPERation[:EVENt]?.....76
- :STATus:OPERation:NTRansition.....76
- :STATus:OPERation:PTRansition.....76
- :STATus:OPERation:LLFail[:SUMMary]:CONDition?.....77
- :STATus:OPERation:LLFail[:SUMMary]:ENABle.....77

- :STATus:OPERation:LLFail[:SUMMary][:EVENt]?.....77
- :STATus:OPERation:LLFail[:SUMMary]:NTRansition.....77
- :STATus:OPERation:LLFail[:SUMMary]:PTRansition.....77
- :STATus:OPERation:SENSe[:SUMMary]:CONDition? 78
- :STATus:OPERation:SENSe[:SUMMary]:ENABle.....78
- :STATus:OPERation:SENSe[:SUMMary][:EVENt]?.....78
- :STATus:OPERation:SENSe[:SUMMary]:NTRansition.....78
- :STATus:OPERation:SENSe[:SUMMary]:PTRansition.....79
- :STATus:OPERation:ULFail[:SUMMary]:CONDition?.....79
- :STATus:OPERation:ULFail[:SUMMary]:ENABle.....79
- :STATus:OPERation:ULFail[:SUMMary][:EVENt]?.....79
- :STATus:OPERation:ULFail[:SUMMary]:NTRansition.....79
- :STATus:OPERation:ULFail[:SUMMary]:PTRansition.....80
- :STATus:PRESet.....80
- :STATus:QUEStionable:CALibration[:SUMMary]:CONDition?.....80
- :STATus:QUEStionable:CALibration[:SUMMary]:ENABle.....81
- :STATus:QUEStionable:CALibration[:SUMMary][:EVENt]?.....81
- :STATus:QUEStionable:CALibration[:SUMMary]:NTRansition.....81
- :STATus:QUEStionable:CALibration[:SUMMary]:PTRansition.....81
- :STATus:QUEStionable:CONDition?.....81
- :STATus:QUEStionable:ENABle.....82
- :STATus:QUEStionable[:EVENt]?.....82
- :STATus:QUEStionable:NTRansition.....82
- :STATus:QUEStionable:PTRansition.....82
- :STATus:QUEStionable:POWer[:SUMMary]:CONDition?.....83
- :STATus:QUEStionable:POWer[:SUMMary]:ENABle.....83
- :STATus:QUEStionable:POWer[:SUMMary][:EVENt]?.....83
- :STATus:QUEStionable:POWer[:SUMMary]:NTRansition.....83
- :STATus:QUEStionable:POWer[:SUMMary]:PTRansition.....84

表3.10　影响状态寄存器的命令或事件

状态寄存器	*RST	*CLS	开机	STATus:PRESet
SCPI 跳变过滤器(NTR 和 PTR)	无影响	无影响	预置	预置
SCPI 使能寄存器	无影响	无影响	预置	预置
SCPI 事件寄存器	无影响	清零	清零	无影响
SCPI 错误/事件队列使能	无影响	无影响	预置	预置
SCPI 错误/事件队列	无影响	清零	清零	无影响
IEEE488.2 寄存器 ESE SRE	无影响	无影响	清零	无影响
IEEE488.2 寄存器 SESR STB	无影响	清零	清零	无影响

预置状态说明：PTR 的预置值为 0x7fff (32767)；NTR 和使能寄存器均清零。

- 命令集.....60
- 跳变过滤器.....61
- 设备状态寄存器描述.....61
- 操作状态寄存器描述.....62
- 校准操作状态寄存器描述.....63
- 下限检测操作状态寄存器描述.....64
- 传感操作状态寄存器描述.....65
- 触发操作状态寄存器描述.....66
- 上限检测操作状态寄存器描述.....67
- 疑问状态寄存器描述.....69
- 校准疑问状态寄存器描述.....70
- 功率疑问状态寄存器描述.....71

3.3.7.1 命令集

通过如下命令集可以查询或设置状态寄存器的内容:

:CONDition?

查询状态寄存器的条件寄存器的值，返回格式为 < NR1 > 。范围为 0~32767。查询完毕后，条件寄存器的值保持不变。

:ENABle <NRf>|<非十进制数>

查询或设置状态寄存器的事件使能寄存器，最高位（Bit15）始终为 0。

[:EVENt?]

查询状态寄存器的事件寄存器，查询完毕后，清零该寄存器。

:NTRansition <NRf>|<非十进制数>

查询或设置状态寄存器的负跳变过滤器，最高位始终为 0。

:PTRansition <NRf>|<非十进制数>。

查询或设置状态寄存器的正跳变过滤器，最高位始终为 0。

本案支持的状态寄存器有:

STATus:DEVice
STATus:OPERation
STATus:OPERation:CALibrating[:SUMMary]
STATus:OPERation:LLFail[:SUMMary]
STATus:OPERation:SENSe[:SUMMary]
STATus:OPERation:ULFail[:SUMMary]
STATus:QUEStionable
STATus:QUEStionable
STATus:QUEStionable:CALibration[:SUMMary]
STATus:QUEStionable:POWer[:SUMMary]

例如:

可以用 :CONDition? 来查询校准操作寄存器 STATus:OPERation:CALibrating[:SUMMary]

STATus:OPERation:CALibrating[:SUMMary]:CONDition?

可以用 :ENABle 来查询或设置校准操作寄存器

STATus:OPERation:CALibrating[:SUMMary]
STATus:OPERation:CALibrating[:SUMMary]:ENABle

3.3.7.2 跳变过滤器

有关跳变过滤器的描述，请参考 SCPI-99 的第 9.2 节。下面给出简要介绍。

1） 正跳变（PTR）：当条件由 FALSE 事件设置为 TRUE。
2） 负跳变（NTR）：当条件由 TRUE 到 FALSE 时，事件设置为 TRUE。
3） 正跳变或负跳变：当条件由 FALSE 到 TRUE 或者 TRUE 到 FALSE 时，事件设置为 TRUE。
4） 清零正跳变和负跳变寄存器，将禁止事件报告。

3.3.7.3 设备状态寄存器描述（STATus:DEVice）

表3.11 设备状态寄存器描述

位	值	定义
0	1	未用
1	2	探头检测状态
2	4	保留
3	8	探头错误状态
4	16	保留
5	32	保留
6	64	保留
7-14	-	未用
15	-	始终为 0

1） 位 1 和表示探头检测状态。
   a） STATus:DEVice:CONDition?的返回值：1 表示检测到了探头，0 表示未接探头。
   b） STATus:DEVice[:EVENt]?的返回值：1 表明发生了探头连接或探头移走事件。0 表示未发生。查询完毕后，事件寄存器将被清零。
   c） 当 STATus:DEVice:NTRansition 设置为 1 时，如果检测到探头移去，则 STATus:DEVice[:EVENt]将被设置为 1。
   d） 当 STATus:DEVice:PTRansition 设置为 1 时，如果检测到探头接入，则 STATus:DEVice[:EVENt]将被设置为 1。
2） 位 3 表示探头错误。1 表示错误，0 表示未发现错误。

3.3 仪器子系统命令

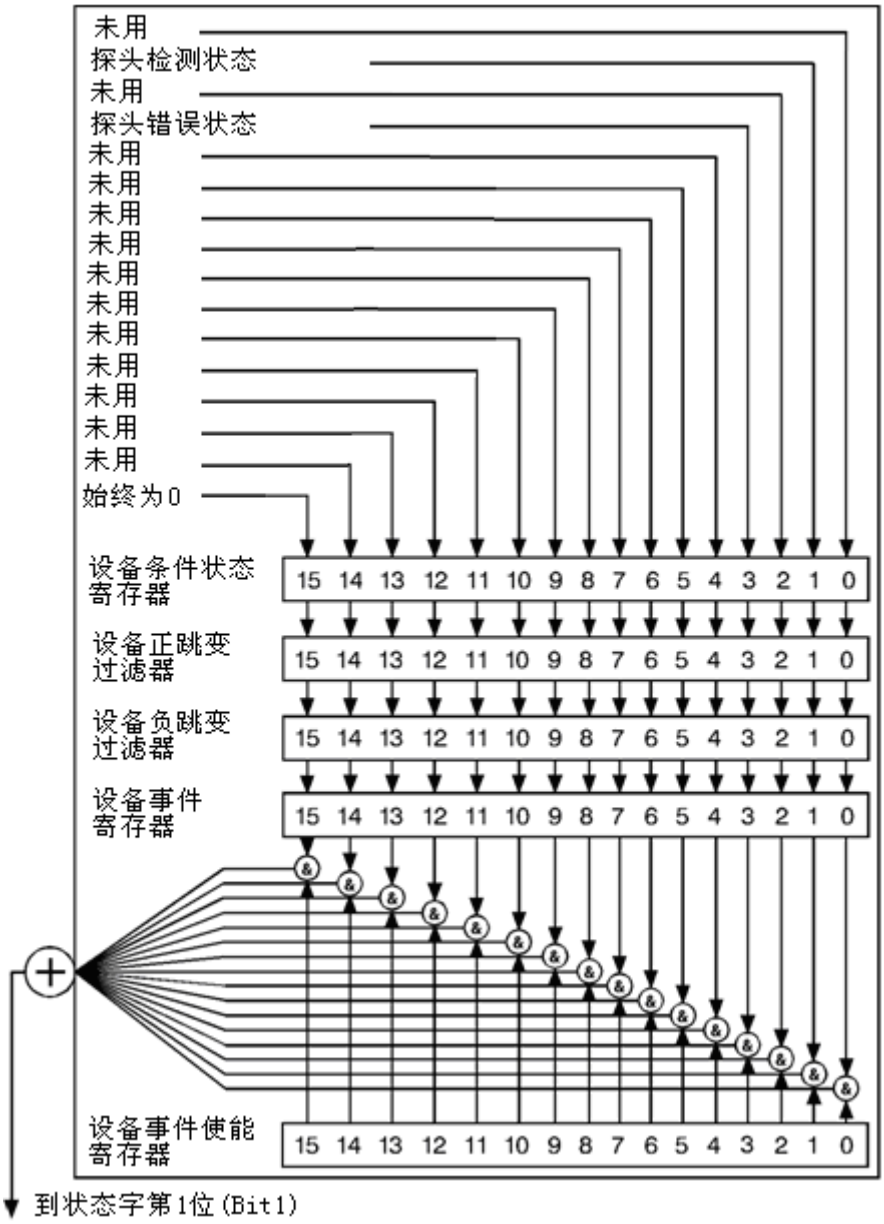


图3.1　　设备状态寄存器

3.3.7.4 操作状态寄存器描述（STATus:OPERation）

表3.12　操作状态寄存器描述

位	值	定义
0	1	校准概要
1-4	-	未用
5	32	等待触发概要
6-9	-	未用
10	1024	传感概要
11	2048	下限检测失败概要（LLF）
12	4096	上限检测失败概要（ULF）
13-14	-	未用
15	-	始终为 0

包括 6 组操作寄存器：

- STATus:OPERation
- STATus:OPERation:CALibrating[:SUMMary]
- STATus:OPERation:LLFail[:SUMMary]
- STATus:OPERation:SENSe[:SUMMary]
- STATus:OPERation:TRIGger[:SUMMary]
- STATus:OPERation:ULFail[:SUMMary]

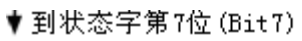


图3.2　操作状态寄存器

3.3.7.5 校准操作状态寄存器描述（STATus:OPERation:CALibrating[:SUMMary]）

表3.13　校准操作状态寄存器描述

位	值	定义
0	1	未用
1	2	校准状态
2	4	保留
3-14	-	未用
15	-	始终为 0

位 1 表示通道校准状态，1 表示正在校零或校准，0 表示校准完毕或者不是正在校零或校准。

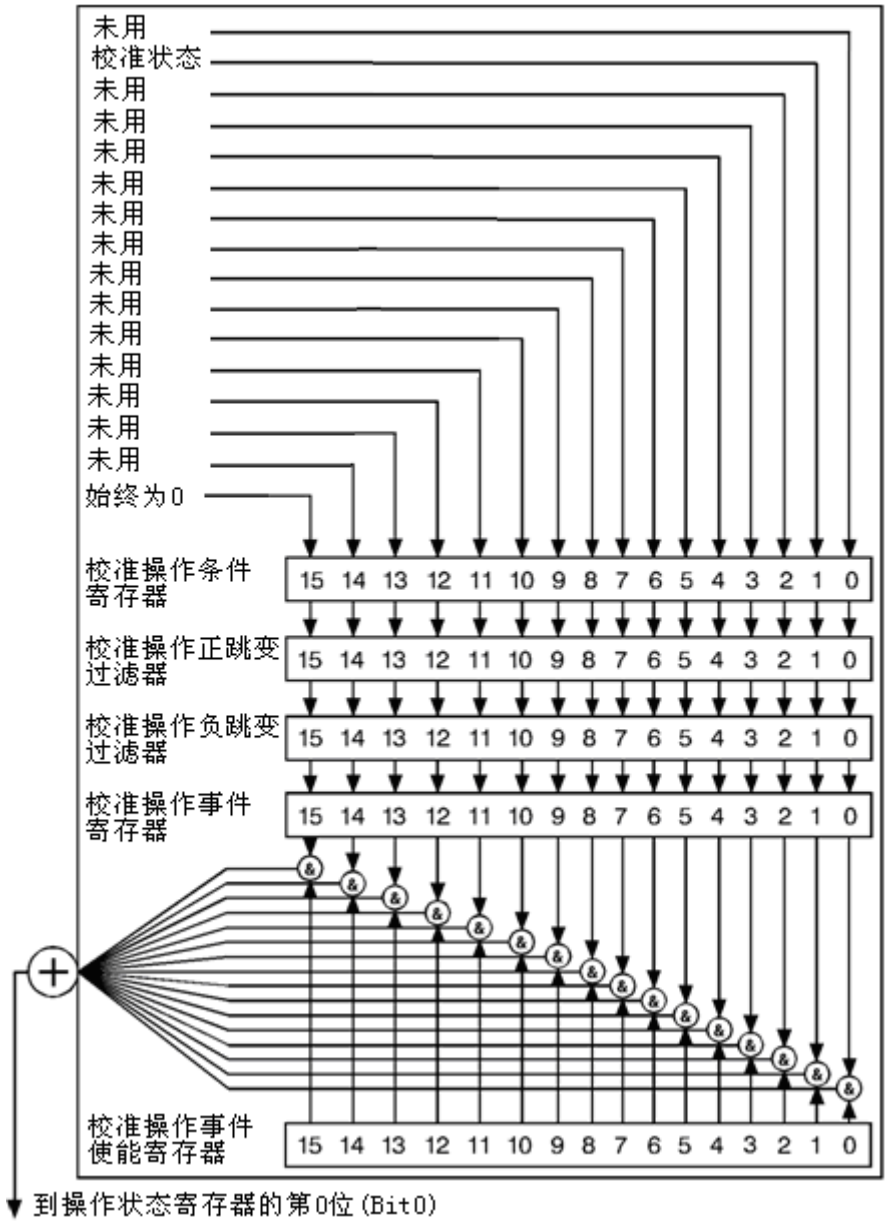


图3.3　　校准操作状态寄存器

3.3.7.6 下限检测操作状态寄存器描述（STATus:OPERation:LLFail[:SUMMary]）

表3.14　下限检测操作状态寄存器描述

位	值	定义
0-2	-	未用
3	8	下限检测状态
4	16	保留
5	32	保留
6	64	保留
7-14	-	未用
15	-	始终为 0

位 3 表示下限极限失败检测状态，1 表示超出下限

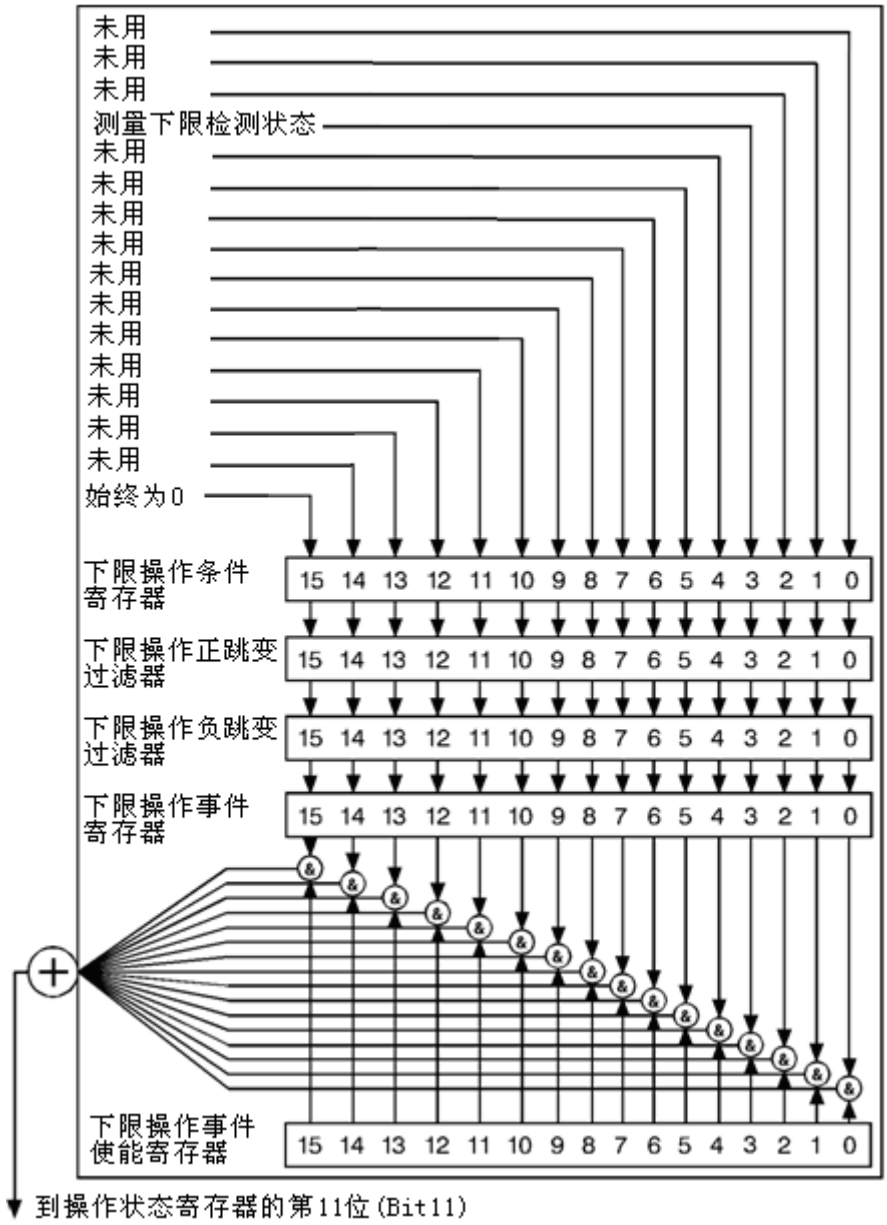


图3.4　　下限检测操作状态寄存器

3.3.7.7 传感操作状态寄存器描述（STATus:OPERation:SENSe[:SUMMary]）

表3.15　传感操作状态寄存器描述

位	值	定义
0	1	未用
1	2	读探头 EEPROM 状态
2	4	保留
3-14	-	未用
15	-	始终为 0

位 1 表示读探头 EEPROM 状态，1 表示正在读探头。

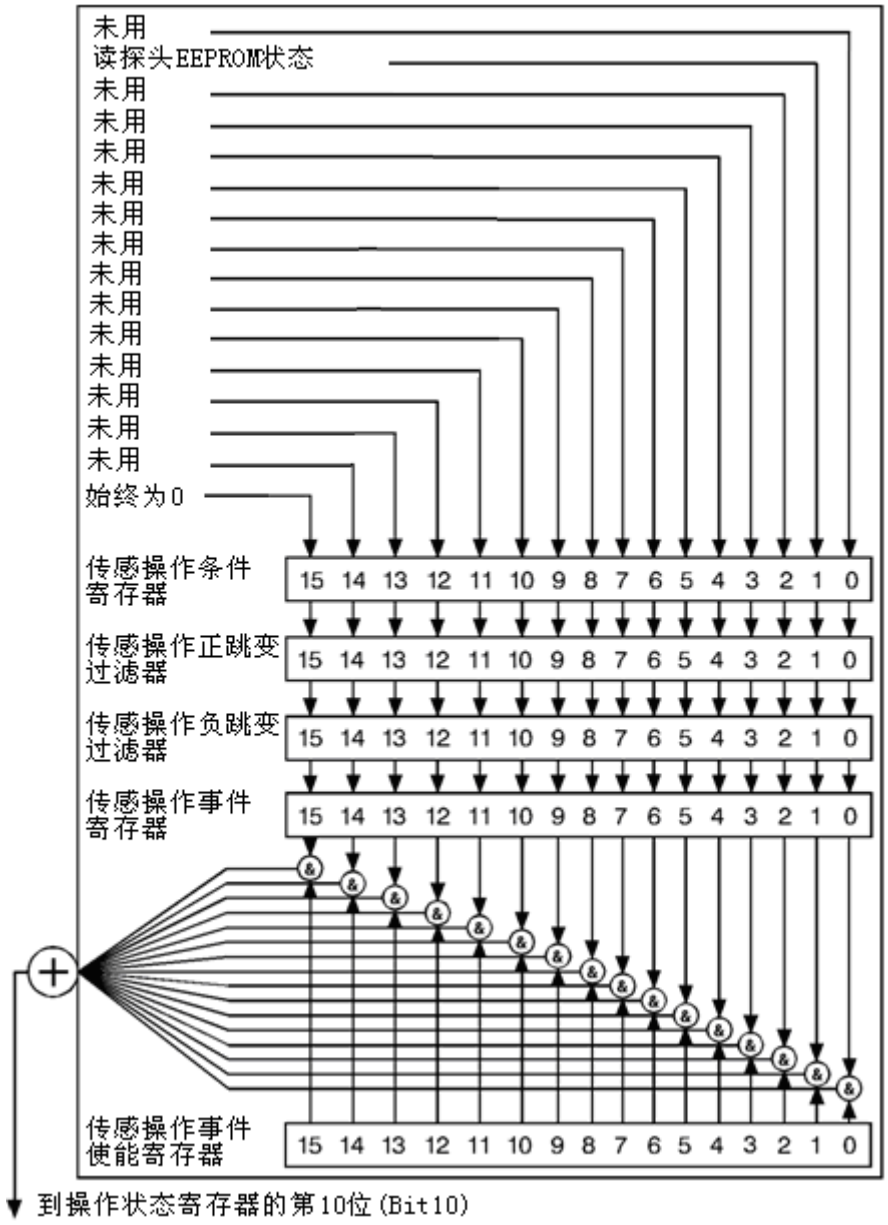


图3.5　传感操作状态寄存器

3.3.7.8 触发操作状态寄存器描述（STATus:OPERation:TRIGger[:SUMMary]）

表3.16　触发操作状态寄存器描述

位	值	定义
0	1	未用
1	2	触发状态
2	4	保留
3-14	-	未用
15	-	始终为 0

位 1 表示触发等待状态，1 表示正在等待触发。

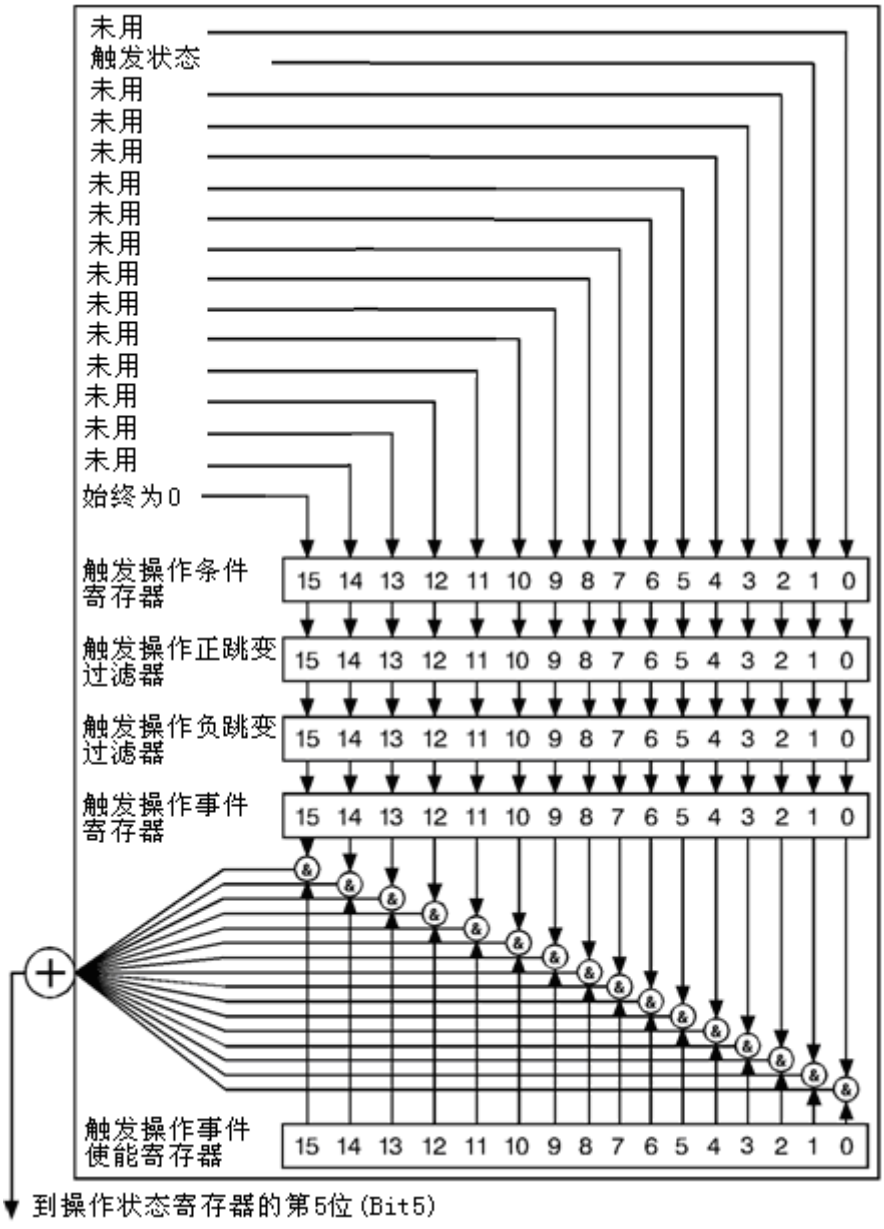


图3.6　触发操作状态寄存器

3.3.7.9 上限检测操作状态寄存器描述（STATus:OPERation:ULFail[:SUMMary]）

表3.17　上限检测操作状态寄存器描述

位	值	定义
0-2	-	未用
3	8	上限检测状态
4	16	保留
5	32	保留
6	64	保留
7-14	-	未用
15	-	始终为 0

位 3 表示相上限极限失败检测状态，1 表示超出上限。

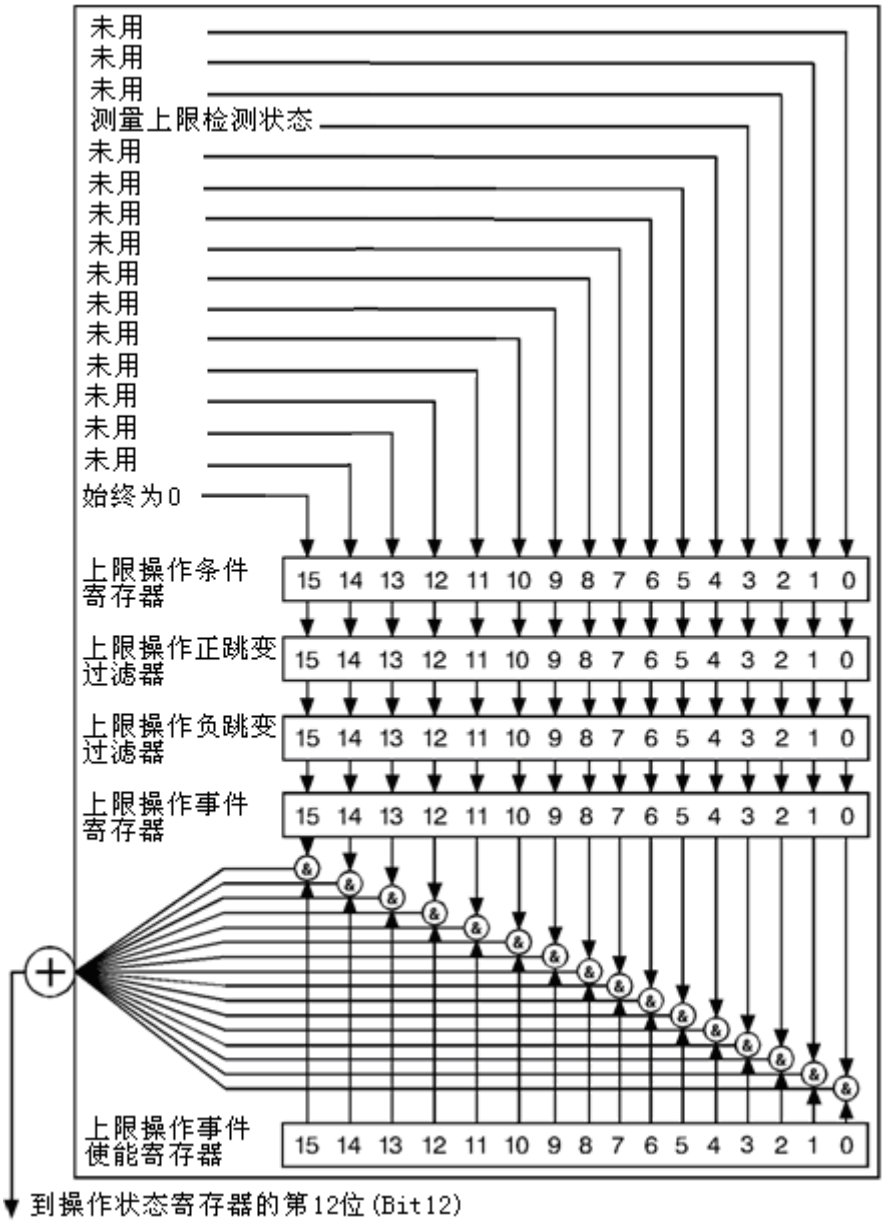


图3.7　上限检测操作状态寄存器

3.3.12.10 疑问状态寄存器描述（STATus:QUEStionable）

表3.18　疑问状态寄存器描述

位	值	定义
0-2	-	未用
3	8	功率概要
4-7	-	未用
8	256	校准概要
9	512	开机自测试
10-14	-	未用
15	-	始终为 0

3.3 仪器子系统命令

包括 3 组操作寄存器:

- STATus:QUEStionable
- STATus:QUEStionable:Power[:SUMMary]
- STATus:QUEStionable:CALibration[:SUMMary]

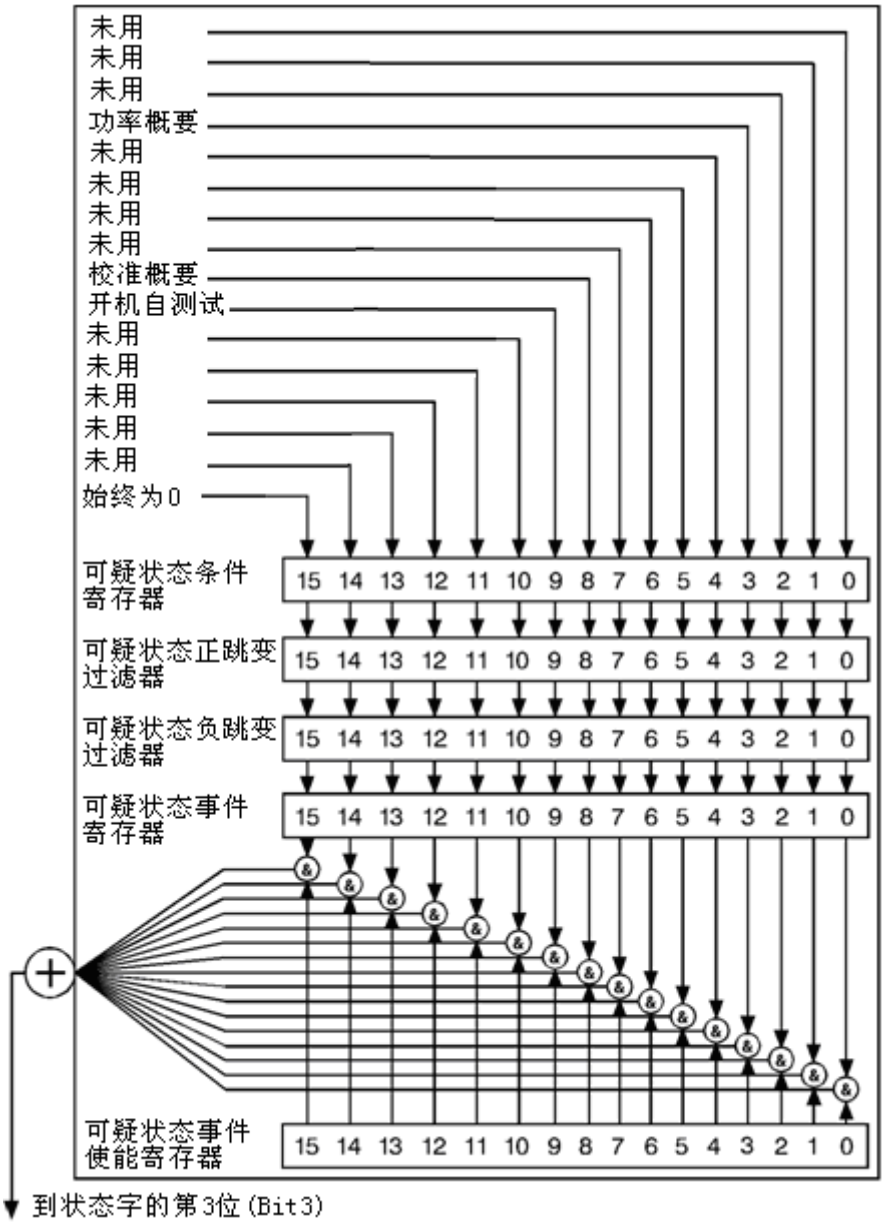


图3.8　　疑问状态寄存器

3.3.7.11 校准疑问状态寄存器描述（STATus:QUEStionable:CALibration[:SUMMary]）

表3.19　校准疑问状态寄存器描述

位	值	定义
0	1	未用
1	2	校零校准错误
2	4	保留
3-14	-	未用
15	-	始终为 0

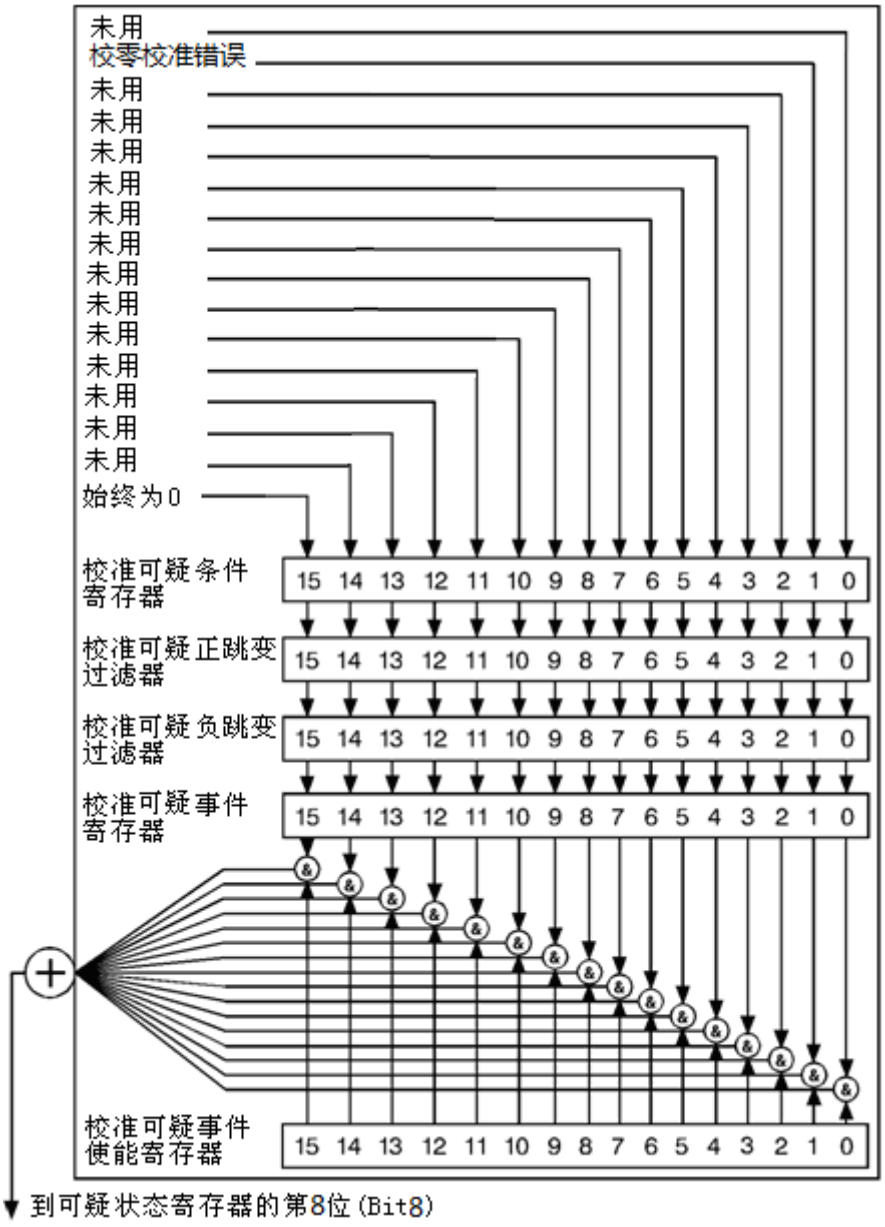


图3.9　　校准疑问状态寄存器

3.3.7.12 功率疑问状态寄存器描述（STATus:QUEStionable:POWer[:SUMMary]）

表3.20　功率疑问状态寄存器描述

位	值	定义
0	1	未用
1	2	输入过载
2	4	保留
3	8	需要校零
4	16	保留
5	32	数据无效，或者对数错误
6	64	保留
7	128	保留
8	256	保留
9-14	-	未用
15	-	始终为 0

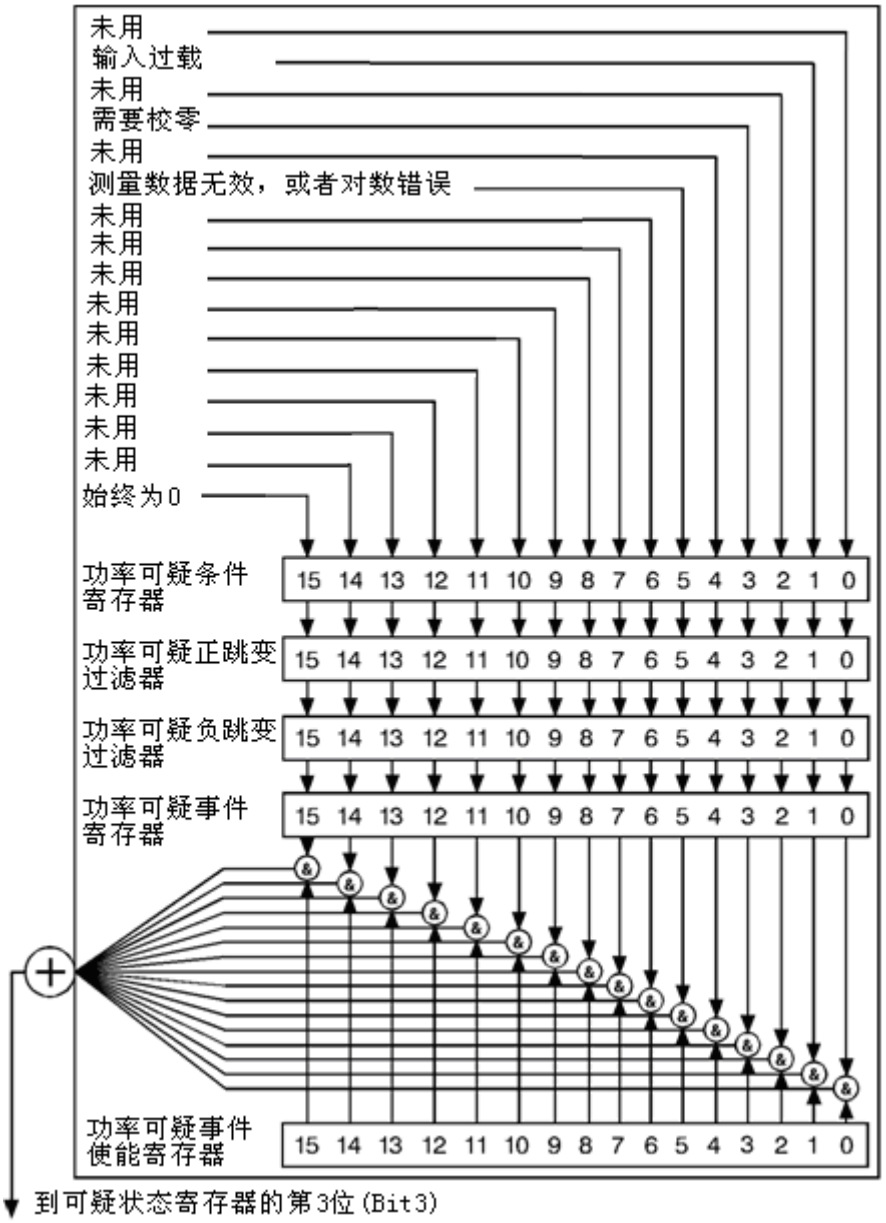


图3.10　功率疑问状态寄存器

:STATus:DEVice:CONDition?

功能:　查询设备状态条件寄存器中的值。
返回值中，如果位 1 非零，表示检测到探头；如果位 3 非零，表示探头错误。

查询:　:STATus:DEVice:CONDition?

设置:　不支持

举例:　STAT:DEV:COND?

:STATus:DEVice:ENABle

功能:　查询或设置设备状态事件使能寄存器。按位操作，0 表示禁止报告状态事件到上一级状态字的位 1（Bit1）中。1 表示使能。

查询:　:STATus:DEVice:ENABle?

设置: :STATus:DEVice:ENABle <NRf>|<非十进制数>
参数范围为 0 到 32767。

举例: STAT:DEV:ENAB? 查询设备状态事件使能寄存器
STAT:DEV:ENAB 6 使能位 1 和位 2，即允许报告探头检测事件到状态字中。
STAT:DEV:ENAB #B0110 意义同上，使用了二进制数。
STAT:DEV:ENAB #H06 意义同上，使用了十六进制数。

:STATus:DEVice[:EVENt]?

功能: 查询设备事件寄存器。查询完毕后，仪器自动清零该寄存器。
查询: :STATus:DEVice[:EVENt]?
设置: 不支持
举例: STAT:DEV?

:STATus:DEVice:NTRansition

功能: 查询或设置设备负跳变过滤器。
查询: :STATus:DEVice:NTRansition?
设置: :STATus:DEVice:NTRansition <NRf>|<非十进制数>
参数范围为 0 到 32767。
举例: STAT:DEV:NTR?
STAT:DEV:NTR #H06 负跳变过滤位 1 和位 2 的状态到事件寄存器中。
STAT:DEV:NTR #B0110 意义同上，使用了二进制数。
STAT:DEV:NTR 6 意义同上，使用了十进制数。

:STATus:DEVice:PTRansition

功能: 查询或设置设备正跳变过滤器。
查询: :STATus:DEVice:PTRansition?
设置: :STATus:DEVice:PTRansition <NRf>|<非十进制数>
参数范围为 0 到 32767。
举例: STAT:DEV:PTR?
STAT:DEV:PTR #H06 正跳变过滤位 1 和位 2 的状态到事件寄存器中。
STAT:DEV:PTR #B0110 意义同上，使用了二进制数。
STAT:DEV:PTR 6 意义同上，使用了十进制数。

:STATus:OPERation:CALibrating[:SUMMary]:CONDition?

功能: 查询校准操作状态条件寄存器中的值。
返回值中，如果位 1 非零，表示正在校零或校准。例如返回 2，表示正在校零校准。
查询: :STATus:OPERation:CALibrating[:SUMMary]:CONDition?
设置: 不支持
举例: STAT:OPER:CAL:COND?

:STATus:OPERation:CALibrating[:SUMMary]:ENABle

功能: 查询或设置校准操作事件使能寄存器。按位操作，0 表示禁止报告状态事件到操作状态的位 0（Bit0）中。1 表示使能。

查询: :STATus:OPERation:CALibrating[:SUMMary]:ENABle?

设置: :STATus:OPERation:CALibrating[:SUMMary]:ENABle <NRf>|<非十进制数>
参数范围为 0 到 32767。

举例: STAT:OPER:CAL:ENAB?

STAT:OPER:CAL:ENAB 6	使能位 1 和位 2，允许报告校准操作事件到操作状态。
STAT:OPER:CAL:ENAB #B0110	意义同上，使用了二进制数。
STAT:OPER:CAL:ENAB #H06	意义同上，使用了十六进制数。

:STATus:OPERation:CALibrating[:SUMMary][:EVENt]?

功能: 查询校准操作事件寄存器。查询完毕后，仪器自动清零该寄存器。

查询: :STATus:OPERation:CALibrating[:SUMMary][:EVENt]?

设置: 不支持

举例: STAT:OPER:CAL?

:STATus:OPERation:CALibrating[:SUMMary]:NTRansition

功能: 查询或设置校准操作负跳变过滤器。

查询: :STATus:OPERation:CALibrating[:SUMMary]:NTRansition?

设置: :STATus:OPERation:CALibrating[:SUMMary]:NTRansition <NRf>|<非十进制数>
参数范围为 0 到 32767。

举例: STAT:OPER:CAL:NTR?

STAT:OPER:CAL:NTR 6	负跳变过滤位 1 和位 2 的状态到事件寄存器中。
STAT:OPER:CAL:NTR #B0110	意义同上，使用了二进制数。
STAT:OPER:CAL:NTR #H06	意义同上，使用了十六进制数。

:STATus:OPERation:CALibrating[:SUMMary]:PTRansition

功能: 查询或设置校准操作正跳变过滤器。

查询: :STATus:OPERation:CALibrating[:SUMMary]:PTRansition?

设置: :STATus:OPERation:CALibrating[:SUMMary]:PTRansition <NRf>|<非十进制数>
参数范围为 0 到 32767。

举例: STAT:OPER:CAL:PTR?

STAT:OPER:CAL:PTR 6	正跳变过滤位 1 和位 2 的状态到事件寄存器中。
STAT:OPER:CAL:PTR #B0110	意义同上，使用了二进制数。
STAT:OPER:CAL:PTR #H06	意义同上，使用了十六进制数。

:STATus:OPERation:CONDition?

功能: 查询操作状态条件寄存器中的值。
返回值中，如果位 0 非零，表示检测到校准事件；如果位 5 非零，表示检测到等待触发

事件；如果位 10 非零，表示检测到探头接入或移去事件；如果位 11 非零，表示检测到下限检测事件；如果位 12 非零，表示检测到上限检测事件。

例如：如果通过该组寄存器检测校准事件，需要设置校准操作使能寄存器（STAT:OPER:CAL:ENAB）的相应位非 0。其它类同。

查询: :STATus:OPERation:CONDition?

设置: 不支持

举例: STAT:OPER:COND?

:STATus:OPERation:ENABle

功能: 查询或设置操作状态事件使能寄存器。按位操作，0 表示禁止报告状态事件到状态字的位 7（Bit7）中。1 表示使能。

查询: :STATus:OPERation:ENABle?

设置: :STATus:OPERation:ENABle <NRf>|<非十进制数>

参数范围为 0 到 32767。

举例: STAT:OPER:ENAB?

STAT:OPER:ENAB 1	使能位 1，即允许报告操作事件到状态字中。
STAT:OPER:ENAB #B0001	意义同上，使用了二进制数。
STAT:OPER:ENAB #H01	意义同上，使用了十六进制数。

:STATus:OPERation[:EVENt]?

功能: 查询操作状态事件寄存器。查询完毕后，仪器自动清零该寄存器。

查询: :STATus:OPERation[:EVENt]?

设置: 不支持

举例: STAT:OPER?

:STATus:OPERation:NTRansition

功能: 查询或设置操作状态负跳变过滤器。

查询: :STATus:OPERation:NTRansition?

设置: :STATus:OPERation:NTRansition <NRf>|<非十进制数>

参数范围为 0 到 32767。

举例: STAT:OPER:NTR?

STAT:OPER:NTR 1	负跳变过滤位 0 的状态到事件寄存器中。
STAT:OPER:NTR #B0001	意义同上，使用了二进制数。
STAT:OPER:NTR #H01	意义同上，使用了十六进制数。

:STATus:OPERation:PTRansition

功能: 查询或设置操作状态正跳变过滤器。

查询: :STATus:OPERation:NTRansition?

设置: :STATus:OPERation:NTRansition <NRf>|<非十进制数>

参数范围为 0 到 32767。

举例: STAT:OPER:NTR?

STAT:OPER:NTR	#H06	正跳变过滤位 1 和位 2 的状态到事件寄存器中。
STAT:OPER:NTR	#B0110	意义同上，使用了二进制数。
STAT:OPER:NTR	6	意义同上，使用了十进制数。

:STATus:OPERation:LLFail[:SUMMary]:CONDition?

功能: 查询下限检测操作状态条件寄存器中的值。
返回值中，如果位 3 非零，表示下限检测操作失败状态。
查询: :STATus:OPERation:LLFail[:SUMMary]:CONDition?
设置: 不支持
举例: STAT:OPER:LLF:COND?

:STATus:OPERation:LLFail[:SUMMary]:ENABle

功能: 查询或设置下限检测操作事件使能寄存器。按位操作，0 表示禁止报告状态事件到操作状态的位 11（Bit11）中。1 表示使能。
查询: :STATus:OPERation:LLFail[:SUMMary]:ENABle?
设置: :STATus:OPERation:LLFail[:SUMMary]:ENABle <NRf>|<非十进制数>
参数范围为 0 到 32767。
举例: STAT:OPER:LLF:ENAB?

STAT:OPER:LLF:ENAB	8	使能位 3，即允许报告下限检测操作事件到操作状态。
STAT:OPER:LLF:ENAB	#B1000	意义同上，使用了二进制数。
STAT:OPER:LLF:ENAB	#H08	意义同上，使用了十六进制数。

:STATus:OPERation:LLFail[:SUMMary][:EVENt]?

功能: 查询下限检测操作事件寄存器。查询完毕后，仪器自动清零该寄存器。
查询: :STATus:OPERation:LLFail[:SUMMary][:EVENt]?
设置: 不支持
举例: STAT:OPER:LLF?

:STATus:OPERation:LLFail[:SUMMary]:NTRansition

功能: 查询或设置下限检测操作负跳变过滤器。
查询: :STATus:OPERation:LLFail[:SUMMary]:NTRansition?
设置: :STATus:OPERation:LLFail[:SUMMary]:NTRansition <NRf>|<非十进制数>
参数范围为 0 到 32767。
举例: STAT:OPER:LLF:NTR?

STAT:OPER:LLF:NTR	8	负跳变过滤位 3 的状态到事件寄存器中。
STAT:OPER:LLF:NTR	#B1000	意义同上，使用了二进制数。
STAT:OPER:LLF:NTR	#H08	意义同上，使用了十六进制数。

:STATus:OPERation:LLFail[:SUMMary]:PTRansition

功能: 查询或设置下限检测操作正跳变过滤器。
查询: :STATus:OPERation:LLFail[:SUMMary]:PTRansition?

设置: :STATus:OPERation:LLFail[:SUMMary]:PTRansition <NRf>|<非十进制数>
参数范围为 0 到 32767。

举例: STAT:OPER:LLF:PTR?
STAT:OPER:LLF:PTR 8 正跳变过滤位 3 的状态到事件寄存器中。
STAT:OPER:LLF:PTR #B1000 意义同上，使用了二进制数。
STAT:OPER:LLF:PTR #H08 意义同上，使用了十进制数。

:STATus:OPERation:SENSe[:SUMMary]:CONDition?

功能: 查询传感操作状态条件寄存器中的值。
返回值中，如果位 1 非零，表示读探头 EEPROM 状态。

查询: :STATus:OPERation:SENSe[:SUMMary]:CONDition?

设置: 不支持

举例: STAT:OPER:SENS:COND?

:STATus:OPERation:SENSe[:SUMMary]:ENABle

功能: 查询或设置传感操作事件使能寄存器。按位操作，0 表示禁止传感操作状态事件到操作状态的位 10（Bit10）中。1 表示使能。

查询: :STATus:OPERation:SENSe[:SUMMary]:ENABle?

设置: :STATus:OPERation:SENSe[:SUMMary]:ENABle <NRf>|<非十进制数>
参数范围为 0 到 32767。

举例: STAT:OPER:SENS:ENAB?
STAT:OPER:SENS:ENAB 6 使能位 1 和位 2，允许报告传感操作事件到操作状态。
STAT:OPER:SENS:ENAB #B0110 意义同上，使用了二进制数。
STAT:OPER:SENS:ENAB #H06 意义同上，使用了十六进制数。

:STATus:OPERation:SENSe[:SUMMary][:EVENt]?

功能: 查询传感操作事件寄存器。查询完毕后，仪器自动清零该寄存器。

查询: :STATus:OPERation:SENSe[:SUMMary][:EVENt]?

设置: 不支持

举例: STAT:OPER:SENS?

:STATus:OPERation:SENSe[:SUMMary]:NTRansition

功能: 查询或设置传感操作负跳变过滤器。

查询: :STATus:OPERation:SENSe[:SUMMary]:NTRansition?

设置: :STATus:OPERation:SENSe[:SUMMary]:NTRansition <NRf>|<非十进制数>
参数范围为 0 到 32767。

举例: STAT:OPER:SENS:NTR?
STAT:OPER:SENS:NTR 6 负跳变过滤位 1 和位 2 的状态到事件寄存器中。
STAT:OPER:SENS:NTR #B0110 意义同上，使用了二进制数。
STAT:OPER:SENS:NTR #H06 意义同上，使用了十六进制数。

:STATus:OPERation:SENSe[:SUMMary]:PTRansition

功能: 查询或设置传感操作正跳变过滤器。
查询: :STATus:OPERation:SENSe[:SUMMary]:PTRansition?
设置: :STATus:OPERation:SENSe[:SUMMary]:PTRansition <NRf>|<非十进制数>
参数范围为 0 到 32767。
举例: STAT:OPER:SENS:PTR?
STAT:OPER:SENS:PTR 6 正跳变过滤位 1 和位 2 的状态到事件寄存器中。
STAT:OPER:SENS:PTR #B0110 意义同上，使用了二进制数。
STAT:OPER:SENS:PTR #H06 意义同上，使用了十六进制数。

:STATus:OPERation:ULFail[:SUMMary]:CONDition?

功能: 查询上限检测操作状态条件寄存器中的值。
返回值中，如果位 3 非零，表示上限检测操作失败状态。
查询: :STATus:OPERation:ULFail[:SUMMary]:CONDition?
设置: 不支持
举例: STAT:OPER:ULF:COND?

:STATus:OPERation:ULFail[:SUMMary]:ENABle

功能: 查询或设置上限检测操作事件使能寄存器。按位操作，0 表示禁止报告状态事件到操作状态的位 12（Bit12）中。1 表示使能。
查询: :STATus:OPERation:ULFail[:SUMMary]:ENABle?
设置: :STATus:OPERation:ULFail[:SUMMary]:ENABle <NRf>|<非十进制数>
参数范围为 0 到 32767。
举例: STAT:OPER:ULF:ENAB?
STAT:OPER:ULF:ENAB 8 使能位 3，允许报告上限检测操作事件到操作状态。
STAT:OPER:ULF:ENAB #B1000 意义同上，使用了二进制数。
STAT:OPER:ULF:ENAB #H08 意义同上，使用了十六进制数。

:STATus:OPERation:ULFail[:SUMMary][:EVENt]?

功能: 查询上限检测操作事件寄存器。查询完毕后，仪器自动清零该寄存器。
查询: :STATus:OPERation:ULFail[:SUMMary][:EVENt]?
设置: 不支持
举例: STAT:OPER:ULF?

:STATus:OPERation:ULFail[:SUMMary]:NTRansition

功能: 查询或设置上限检测操作负跳变过滤器。
查询: :STATus:OPERation:ULFail[:SUMMary]:NTRansition?
设置: :STATus:OPERation:ULFail[:SUMMary]:NTRansition <NRf>|<非十进制数>
参数范围为 0 到 32767。
举例: STAT:OPER:ULF:NTR?

STAT:OPER:ULF:NTR 8 负跳变过滤位 3 的状态到事件寄存器中。
STAT:OPER:ULF:NTR #B1000 意义同上，使用了二进制数。
STAT:OPER:ULF:NTR #H08 意义同上，使用了十六进制数。

:STATus:OPERation:ULFail[:SUMMary]:PTRansition

功能: 查询或设置上限检测操作正跳变过滤器。
查询: :STATus:OPERation:ULFail[:SUMMary]:PTRansition?
设置: :STATus:OPERation:ULFail[:SUMMary]:PTRansition <NRf>|<非十进制数>
参数范围为 0 到 32767。
举例: STAT:OPER:ULF:PTR?
STAT:OPER:ULF:PTR 8 正跳变过滤位 1 和位 2 的状态到事件寄存器中。
STAT:OPER:ULF:PTR #B1000 意义同上，使用了二进制数。
STAT:OPER:ULF:PTR #H08 意义同上，使用了十进制数。

:STATus:PRESet

功能: 预置一些状态寄存器如下，其它寄存器不变。
查询: 不支持
设置: :STATus:PRESet
举例: STAT:PRES

表3.21 状态寄存器复位状态描述

寄存器	子寄存器	预置状态
OPERation	ENABle	全 0
	PTR	全 1
	NTR	全 0
QUEStionable	ENABle	全 0
	PTR	全 1
	NTR	全 0
其它	ENABle	全 0
	PTR	全 1
	NTR	全 0

:STATus:QUEStionable:CALibration[:SUMMary]:CONDition?

功能: 查询校准疑问状态条件寄存器中的值。
返回值中，如果位 1 非零，表示校零校准错误。例如返回 2，表示校零校准错误。
查询: :STATus:OPERation:CALibrating[:SUMMary]:CONDition?
设置: 不支持
举例: STAT:OPER:CAL:COND?

:STATus:QUEStionable:CALibration[:SUMMary]:ENABle

功能: 查询或设置校准疑问事件使能寄存器。按位操作，0 表示禁止报告状态事件到疑问状态的位 8（Bit8）中。1 表示使能。

查询: :STATus:QUEStionable:CALibration[:SUMMary]:ENABle?

设置: :STATus:QUEStionable:CALibration[:SUMMary]:ENABle <NRf>|<非十进制数>
参数范围为 0 到 32767。

举例: STAT:QUES:CAL:ENAB?

STAT:QUES:CAL:ENAB 6	使能位 1 和 2，允许报告校准疑问事件到疑问状态。
STAT:QUES:CAL:ENAB #B0110	意义同上，使用了二进制数。
STAT:QUES:CAL:ENAB #H06	意义同上，使用了十六进制数。

:STATus:QUEStionable:CALibration[:SUMMary][:EVENt]?

功能: 查询校准疑问事件寄存器。查询完毕后，仪器自动清零该寄存器。

查询: :STATus:QUEStionable:CALibration[:SUMMary][:EVENt]?

设置: 不支持

举例: STAT:QUES:CAL?

:STATus:QUEStionable:CALibration[:SUMMary]:NTRansition

功能: 查询或设置校准疑问负跳变过滤器。

查询: :STATus:QUEStionable:CALibration[:SUMMary]:NTRansition?

设置: :STATus:QUEStionable:CALibration[:SUMMary]:NTRansition <NRf>|<非十进制数>
参数范围为 0 到 32767。

举例: STAT:QUES:CAL:NTR?

STAT:QUES:CAL:NTR 6	负跳变过滤位 1 和位 2 的状态到事件寄存器中。
STAT:QUES:CAL:NTR #B0110	意义同上，使用了二进制数。
STAT:QUES:CAL:NTR #H06	意义同上，使用了十六进制数。

:STATus:QUEStionable:CALibration[:SUMMary]:PTRansition

功能: 查询或设置校准疑问正跳变过滤器。

查询: :STATus:QUEStionable:CALibration[:SUMMary]:PTRansition?

设置: :STATus:QUEStionable:CALibration[:SUMMary]:PTRansition <NRf>|<非十进制数>
参数范围为 0 到 32767。

举例: STAT:QUES:CAL:PTR?

STAT:QUES:CAL:PTR 6	正跳变过滤位 1 和位 2 的状态到事件寄存器中。
STAT:QUES:CAL:PTR #B0110	意义同上，使用了二进制数。
STAT:QUES:CAL:PTR #H06	意义同上，使用了十六进制数。

:STATus:QUEStionable:CONDition?

功能: 查询疑问状态条件寄存器中的值。
返回值中，如果位 3 非零，表示检测到功率疑问事件；如果位 8 非零，表示检测到校准

疑问事件；如果位 9 非零，表示开机自测试失败。

例如：如果通过该组寄存器检测校准疑问事件，需要设置校准疑问使能寄存器（STAT:QUES:CAL:ENAB）的相应位非 0。其它类同。

查询: :STATus:QUEStionable:CONDition?

设置: 不支持

举例: STAT:QUES:COND?

:STATus:QUEStionable:ENABle

功能: 查询或设置疑问状态事件使能寄存器。按位操作，0 表示禁止报告状态事件到状态字的位 3（Bit3）中。1 表示使能。

查询: :STATus:QUEStionable:ENABle?

设置: :STATus:QUEStionable:ENABle <NRf>|<非十进制数>

参数范围为 0 到 32767。

举例: STAT:QUES:ENAB?

STAT:QUES:ENAB 8	使能位 1，即允许报告校准疑问事件到状态字中。
STAT:QUES:ENAB #B1000	意义同上，使用了二进制数。
STAT:QUES:ENAB #H08	意义同上，使用了十六进制数。

:STATus:QUEStionable[:EVENt]?

功能: 查询疑问状态事件寄存器。查询完毕后，仪器自动清零该寄存器。

查询: :STATus:QUEStionable[:EVENt]?

设置: 不支持

举例: STAT:QUES?

:STATus:QUEStionable:NTRansition

功能: 查询或设置疑问状态负跳变过滤器。

查询: :STATus:QUEStionable:NTRansition?

设置: :STATus:QUEStionable:NTRansition <NRf>|<非十进制数>

参数范围为 0 到 32767。

举例: STAT:QUES:NTR?

STAT:QUES:NTR 8	负跳变过滤位 3 的状态到事件寄存器中。
STAT:QUES:NTR #B1000	意义同上，使用了二进制数。
STAT:QUES:NTR #H08	意义同上，使用了十六进制数。

:STATus:QUEStionable:PTRansition

功能: 查询或设置疑问状态正跳变过滤器。

查询: :STATus:QUEStionable:PTRansition?

设置: :STATus:QUEStionable:PTRansition <NRf>|<非十进制数>

参数范围为 0 到 32767。

举例: STAT:QUES:PTR?

STAT:QUES:PTR 8	正跳变过滤位 3 的状态到事件寄存器中。
STAT:QUES:PTR #B0110	意义同上，使用了二进制数。

STAT:QUES:PTR　#H08　　意义同上，使用了十六进制数。

:STATus:QUEStionable:POWer[:SUMMary]:CONDition?

功能:　查询功率疑问状态条件寄存器中的值。
返回值中，
如果位 1 非零，表示输入过载；
如果位 3 非零，表示需要校零；
如果位 5 非零，表示数据无效，或者对数错误；
如果例如返回 8，表示需要校零。

查询:　:STATus:QUEStionable:POWer[:SUMMary]:CONDition?

设置:　不支持

举例:　STAT:QUES:POW:COND?

:STATus:QUEStionable:POWer[:SUMMary]:ENABle

功能:　查询或设置功率疑问事件使能寄存器。按位操作，0 表示禁止报告状态事件到疑问状态的位 3（Bit3）中。1 表示使能。

查询:　:STATus:QUEStionable:POWer[:SUMMary]:ENABle?

设置:　:STATus:QUEStionable:POWer[:SUMMary]:ENABle　<NRf>|<非十进制数>
参数范围为 0 到 32767。

举例:　STAT:QUES:POW:ENAB?
STAT:QUES:POW:ENAB　6　　使能位 1 和位 2，即允许报告功率疑问事件到疑问状态中。
STAT:QUES:POW:ENAB　#B0110　　意义同上，使用了二进制数。
STAT:QUES:POW:ENAB　#H06　　意义同上，使用了十六进制数。

:STATus:QUEStionable:POWer[:SUMMary][:EVENt]?

功能:　查询功率疑问事件寄存器。查询完毕后，仪器自动清零该寄存器。

查询:　:STATus:QUEStionable:POWer[:SUMMary][:EVENt]?

设置:　不支持

举例:　STAT:QUES:POW?

:STATus:QUEStionable:POWer[:SUMMary]:NTRansition

功能:　查询或设置功率疑问负跳变过滤器。

查询:　:STATus:QUEStionable:POWer[:SUMMary]:NTRansition?

设置:　:STATus:QUEStionable:POWer[:SUMMary]:NTRansition　<NRf>|<非十进制数>
参数范围为 0 到 32767。

举例:　STAT:QUES:POW:NTR?
STAT:QUES:POW:NTR　6　　负跳变过滤位 1 和位 2 的状态到事件寄存器中。
STAT:QUES:POW:NTR　#B0110　　意义同上，使用了二进制数。
STAT:QUES:POW:NTR　#H06　　意义同上，使用了十六进制数。

3.3 仪器子系统命令

:STATus:QUEStionable:POWer[:SUMMary]:PTRansition

功能: 查询或设置功率疑问正跳变过滤器。

查询: :STATus:QUEStionable:POWer[:SUMMary]:PTRansition?

设置: :STATus:QUEStionable:POWer[:SUMMary]:PTRansition <NRf>|<非十进制数>

参数范围为 0 到 32767。

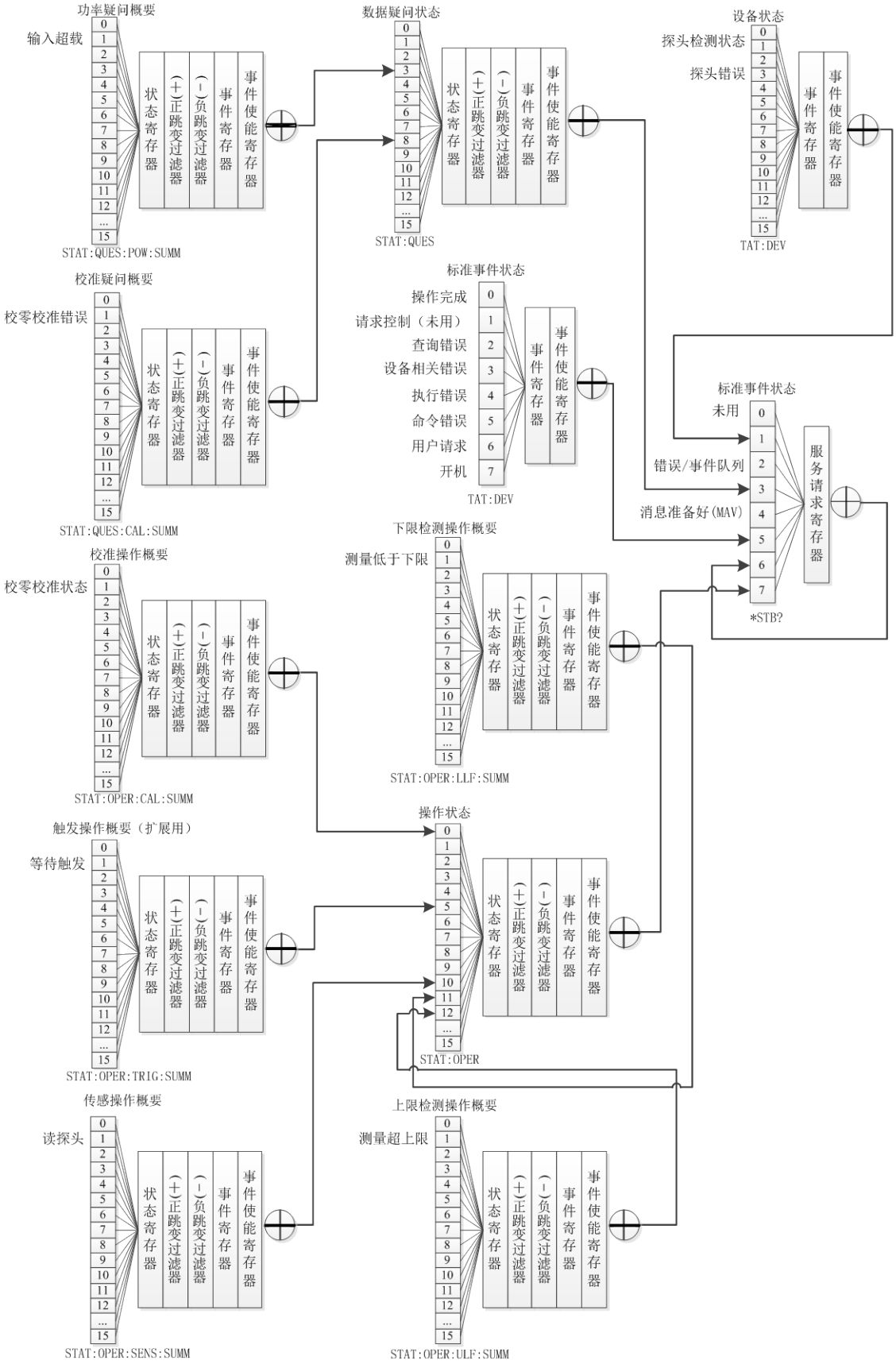
举例: STAT:QUES:POW:PTR?

STAT:QUES:POW:PTR 6 正跳变过滤位 1 和位 2 的状态到事件寄存器中。

STAT:QUES:POW:PTR #B0110 意义同上，使用了二进制数。

STAT:QUES:POW:PTR #H06 意义同上，使用了十六进制数。

附：状态框图


功率疑问概要
输入超载
状态寄存器
(+)正跳变过滤器
(-)负跳变过滤器
事件寄存器
事件使能寄存器
STAT:QUES:POW:SUMM
数据疑问状态
STAT:QUES
设备状态
探头检测状态
探头错误
TAT:DEV
校准疑问概要
校零校准错误
STAT:QUES:CAL:SUMM
标准事件状态
操作完成
请求控制（未用）
查询错误
设备相关错误
执行错误
命令错误
用户请求
开机
TAT:DEV
未用
错误/事件队列
消息准备好(MAV)
服务请求寄存器
*STB?
校准操作概要
校零校准状态
STAT:OPER:CAL:SUMM
下限检测操作概要
测量低于下限
STAT:OPER:LLF:SUMM
触发操作概要（扩展用）
等待触发
STAT:OPER:TRIG:SUMM
操作状态
STAT:OPER
传感操作概要
读探头
STAT:OPER:SENS:SUMM
上限检测操作概要
测量超上限
STAT:OPER:ULF:SUMM

3.3.8 系统子系统命令（SYSTem）

- :SYSTem:ERRor[:NEXT]?.....86
- :SYSTem:HELP:HEADers?.....86
- :SYSTem:PRESet.....86
- :SYSTem:VERSion?.....86

:SYSTem:ERRor[:NEXT]?

功能: 从仪器的错误队列中返回仪器的错误代码和错误信息。当产生错误时，错误代码和错误信息存入错误队列中。该命令每执行一次，错误队列中的该信息将被移去。错误信息出队列的顺序是先进先出（FIFO），即最老的信息最先出队列。可以用*CLS 命令清空错误队列。当错误队列为空时，执行该命令，将返回“0，"No Error"”。错误队列的中最多可以容纳 1023 个错误信息。

查询: :SYSTem:ERRor[:NEXT]?

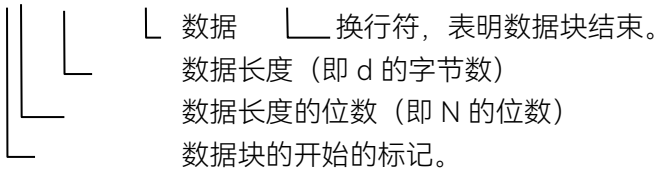
设置: 不支持

举例: SYST:ERR?

复位状态: 无影响。

:SYSTem:HELP:HEADers?

功能: 查询仪器支持的命令列表。数据格式为<任意数据块> 见 IEEE 488.2 第 7.7.6 节。#nNNN...Nddd……ddd<LF>



数据块的形式为#510331…..<LF>中，n = 5，N = 10331。

查询: :SYSTem:HELP:HEADers?

设置: 不支持

举例: SYST:HELP:HEAD?

:SYSTem:PRESet

功能: 复位仪器

查询: 不支持

设置: :SYSTem:PRESet [字符数据]

[字符数据]的形式如下（DEFault 是缺省参数，即不带参数时的参数。）:

举例: SYST:PRES DEF 复位

复位状态:

:SYSTem:VERSion?

功能: 查询仪器使用的 SCPI 版本号。返回形式为 YYYY.X，其中，YYYY 表示年份，X 表示版

本号。本案返回 1999.0。

查询: :SYSTem:VERSion?

设置: 不支持

举例: SYST:VERS?

3.3.9 触发子系统命令（INITiate/TRIGger）

- :ABORt[1]....87
- :INITiate[1]:CONTinuous....87
- :INITiate:CONTinuous:ALL....87
- :INITiate:CONTinuous:SEQuence[1]....88
- :INITiate[1][:IMMediate]....88
- :INITiate[:IMMediate]:ALL....88
- :INITiate[:IMMediate]:SEQuence[1]....88
- :TRIGger[1][:IMMediate]....89

:ABORt[1]

功能: 停止测量。发送 INITiate:CONTinuous ON 后，重启测量。

查询: 不支持

设置: ABORt[1]

举例: ABOR 停止测量。

说明: 本仪器没有设置触发空闲状态，即触发一直处于等待状态。

:INITiate[1]:CONTinuous

功能: 查询或设置仪器触发状态：单步和连续
当设置为单步（OFF）时，等待触发直到设置为连续（ON），或者接收到 INITiate:IMMediate。
该命令与 INITiate:CONTinuous:SEQuence[1]等价。

查询: :INITiate[1]:CONTinuous?

设置: :INITiate[1]:CONTinuous <布尔数据>
OFF|0 单步
ON|1 连续

举例: INIT:CONT? 查询触发状态，0 单步；1 连续。
INIT:CONT ON 设置连续触发状态。

说明: 对于连续波探头，不能设置为单步触发和连续触发；对于参数 OFF，相当于停止测量停，对于参数 ON，相当于启动测量。

复位状态: 设置为 ON

:INITiate:CONTinuous:ALL

功能: 查询或设置仪器的所有通道的触发状态：单步和连续

当设置为单步（OFF）时，等待触发直到设置为连续（ON），或者接收到INITiate:IMMediate。

查询: :INITiate:CONTinuous:ALL?

设置: :INITiate:CONTinuous:ALL <布尔数据>
OFF|0 单步
ON|1 连续

举例: INIT:CONT:ALL? 查询所有通道的触发状态，所有通道均为连续触发时，为 1，否则为 0。
INIT:CONT:ALL ON 设置所有通道为连续触发状态。

复位状态: 设置为 ON

:INITiate:CONTinuous:SEQuence[1]

功能: 查询或设置仪器的触发状态：单步和连续
当设置为单步（OFF）时，等待触发直到设置为连续（ON），或者接收到INITiate:IMMediate。
该命令与 INITiate[1]:CONTinuous 等价

查询: :INITiate:CONTinuous:SEQuence[1]?

设置: :INITiate:CONTinuous:SEQuence[1] <布尔数据>
OFF|0 单步
ON|1 连续

举例: INIT:CONT:SEQ? 查询触发状态，0 单步；1 连续。
INIT:CONT:SEQ1 ON 设置连续触发状态。

:INITiate[1][:IMMediate]

功能: 设置仪器处于等待触发状态。当收到触发事件时，测量开始。
等价命令 INITiate:[IMMediate]:SEQuence[1]

查询: 不支持

设置: :INITiate[1][:IMMediate]

举例: INIT 设置为等待触发状态。

:INITiate[:IMMediate]:ALL

功能: 设置仪器的所有通道处于等待触发状态。当收到触发事件时，测量开始。

查询: 不支持

设置: :INITiate[:IMMediate][:ALL]

举例: INIT:ALL 将仪器的所有通道设置为等待触发状态。

:INITiate[:IMMediate]:SEQuence[1]

功能: 设置仪器处于等待触发状态。当收到触发事件时，测量开始。
等价命令:INITiate[1][:IMMediate]

查询: 不支持

设置: :INITiate[:IMMediate]:SEQuence[1]
举例: INIT:SEQ 设置为等待触发状态。

:TRIGger[1][:IMMediate]

功能: 该命令使得仪器立即处于触发等待状态。
等价命令是:
INITiate[1]:[IMMediate]
TRIGger[:SEQuence[1]][:IMMediate]
查询: 不支持
设置: :TRIGger[1][:IMMediate]
举例: TRIG 使得仪器立即处于触发等待状态。

3.3.10 单位子系统命令（UNIT）

- :UNIT[1]:POWer.....89
- :UNIT[1]:POWer:RATio.....89

:UNIT[1]:POWer

功能: 查询或设置功率单位。菜单操作中是线性和对数。
绝对功率测量的单位为 W 和 dBm，分别对应于线性和对数。
比值测量和相对功率测量的单位为%和 dB，分别对应于线性和对数。
查询: :UNIT[1]:POWer?
设置: :UNIT[1]:POWer <字符数据>
有效的字符数据有:
DBM 或 0：对数显示。
W 或 1：线性显示
举例: UNIT:POW? 查询功率单位。
UNIT1:POW W 设置功率单位为 W。

:UNIT[1]:POWer:RATio

功能: 查询或设置比值测量功率单位。菜单操作中是线性和对数。
查询: :UNIT[1]:POWer:RATio?
设置: :UNIT[1]:POWer:RATio <字符数据>
有效的字符数据有:
DB 或 0：对数显示。
PCT 或 1：线性显示（PCT 表示%）
举例: UNIT:POW:RAT? 查询功率单位。
UNIT1:POW:RAT PCT 设置功率单位为 PCT。

3.3.11 服务子系统命令（SERVice）

- :SERVice:LANGuage.....90
- :SERVice:SENSor[1]:CDATe?.....90
- :SERVice:SENSor[1]:CPLace?.....90
- :SERVice:SENSor[1]:FREQuency:MAXimum?.....90
- :SERVice:SENSor[1]:FREQuency:MINimum?.....91
- :SERVice:SENSor[1]:SNUMber?.....91
- :SERVice:SENSor[1]:TYPE?.....91
- :SERVice:SNUMber.....91

:SERVice:LANGuage

功能: 查询或设置语言选择。
查询: :SERVice:LANGuage?
设置: :SERVice:LANGuage <CHINese|ENGLish|0|1>
CHINese 或 0 选择中文
ENGLish 或 1 选择英文
举例: SERV:LANG? 查询语言选择。
SERV:LANG CHIN 设置语言为中文。
复位状态: 不影响。

:SERVice:SENSor[1]:CDATe?

功能: 查询探头的校准日期。校准日期存储在探头的 EEPROM 中。
查询: :SERVice:SENSor[1]:CDATe?
设置: 不支持
举例: SERV:SENS:CDAT? 查询探头校准日期。

:SERVice:SENSor[1]:CPLace?

功能: 查询探头的校准地址。校准地址存储在探头的 EEPROM 中。
查询: :SERVice:SENSor[1]:CPLace?
设置: 不支持
举例: SERV:SENS:CPL? 查询探头校准地址。

:SERVice:SENSor[1]:FREQuency:MAXimum?

功能: 查询最大频率。最大频率存储在探头的 EEPROM 中。
查询: :SERVice:SENSor[1]:FREQuency:MAXimum?
设置: 不支持
举例: SERV:SENS:FREQ:MAX? 查询探头最大频率。

:SERVice:SENSor[1]:FREQuency:MINimum?

功能: 查询最小频率。最小频率存储在探头的 EEPROM 中。
查询: :SERVice:SENSor[1]:FREQuency:MINimum?
设置: 不支持
举例: SERV:SENS:FREQ:MIN? 查询探头最小频率。

:SERVice:SENSor[1]:SNUMber?

功能: 查询串号。探头的串号存储在探头的 EEPROM 中。
查询: :SERVice:SENSor[1]:SNUMber?
设置: 不支持
举例: SERV:SENS:SNUM? 查询探头串号。

:SERVice:SENSor[1]:TYPE?

功能: 查询类型。探头的类型存储在探头的 EEPROM 中。
查询: :SERVice:SENSor[1]:TYPE?
设置: 不支持
举例: SERV:SENS:TYPE? 查询探头类型。

:SERVice:SNUMber

功能: 查询或设置仪器的串号。
查询: :SERVice:SNUMber?
设置: :SERVice:SNUMber <字符数据>
暂时保留。
举例: SERV:SNUM?

4 编程示例

- 基本操作示例.....93
- 高级操作示例.....98

4.1 基本操作示例

下面举例说明如何使用 VISA 库实现仪器程控编程的基本方法。以 C++语言为例。

- VISA库.....93
- 示例运行环境.....94
- 初始化和设置默认状态.....95
- 发送设置命令.....96
- 读取测量仪器状态.....96
- 命令同步.....96

4.1.1 VISA 库

VISA 是标准的 I/O 函数库及其相关规范的总称。其中，VISA 库函数是一套可方便调用的函数，其核心函数能够控制各种类型器件，无需考虑器件的接口类型和不同 I/O 接口软件的使用方法。这些库函数用于编写仪器的驱动程序，完成计算机与仪器间的命令和数据传输，以实现对仪器的程控。通过初始化寻址字符串（“VISA 资源字符串”），可建立具备程控端口（LAN、USB、GPIB 及 RS-232 等）的仪器的连接。

为实现远程控制首先需要安装 VISA 库。其中，VISA 库封装了底层的 VXI、GPIB、LAN 及 USB 接口的底层传输函数，方便用户直接调用。该仪器支持的编程接口为：USB。这些接口与 VISA 库和编程语言结合使用可以远程控制仪器。目前常使用 Keysight 公司为用户提供的 Keysight I/O Library 作为底层 I/O 库。

图 4.1 以 USB 接口为例显示了程控接口、VISA 库、编程语言和仪器之间的关系。

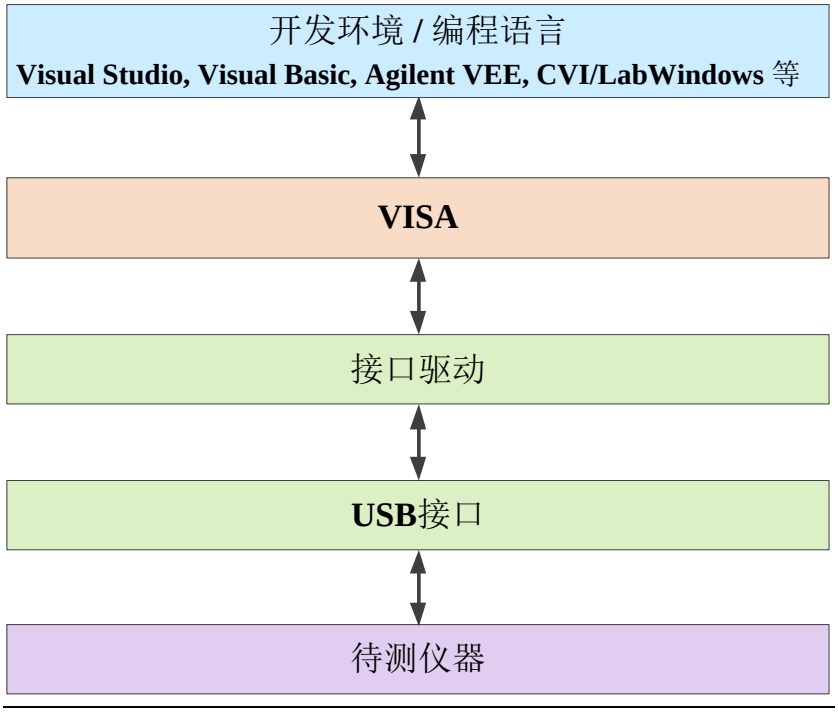


图4.1　　程控软硬件层

4.1.2 示例运行环境

4.1.2.1 配置要求

本章所描述的编程示例，已在如下配置的计算机上运行成功。

- IBM 兼容、奔腾及以上的 PC 机;
- Windows 2000、Windows XP、Windows7 或、Windows8、Windows10 操作系统;
- Visual Studio 2010/2012 集成开发环境;
- NI 的 VISA 库或者 Keysight 的 VISA 库。

4.1.2.2 包含的文件

运行 C/C++编写的程序示例，您必须在 VC 的项目中包含必须的文件
如果您使用 VISA 库必须进行以下步骤:

- 添加 visa32.lib 到源文件;
- 添加 visa.h 到头文件。

关于 VISA 库的更详细资料请分别在 NI 公司和 Keysight 公司的网站查阅。

4.1.3 初始化和设置默认状态

程序开始时首先需要初始化 VISA 资源管理器，打开并建立 VISA 库与仪器的通信连接。具体步骤如下：

4.1.3.1 生成全局变量

首先生成其它程序模块需要调用的全局变量，例如：仪器句柄变量。以下示例程序需要包含下面的全局变量：

```
ViSession iDevHandle;
ViSession iDefaultRM;
const char rgcDevResource[VI_FIND_BUFLEN] = "USB0::1204::4100::1803001::0::INSTR";
const analyzerTimeout = 5000;
```

其中，常量rgcDevResource代表仪器描述符。

4.1.3.2 初始化控者

```
/*****************************************************************************
下面的示例说明了如何打开并建立VISA库与仪器（仪器描述符指定）的通信连接。
初始化控者：打开默认资源管理器并且返回仪器句柄iDevHandle。
*****************************************************************************/
void InitController()
{
    ViStatus iStatus;
    iStatus = viOpenDefaultRM(&iDefaultRM);
    iStatus = viOpen(iDefaultRM, rgcDevResource, VI_NULL, VI_NULL, &iDevHandle);
}
```

4.1.3.3 初始化仪器

```
/*****************************************************************************/
下面的示例初始化仪器默认状态，并且清空状态寄存器。
/*****************************************************************************/
void InitDevice()
{
    ViStatus iStatus;
    ViUInt32 uiRetCnt;
    iStatus = viWrite(iDevHandle, "*CLS\n", strlen("*CLS\n"), &uiRetCnt); //复位状态
```

```
        寄存器
        iStatus = viWrite(iDevHandle, "*RST\n", strlen("*RST\n"), &uiRetCnt); //复位仪器
}
```

4.1.4 发送设置命令

```
/*****************************************************************************/
下面的示例说明如何设置87230的频率。
/*****************************************************************************/
void SimpleSettings()
{
    ViStatus iStatus;
    ViUInt32 uiRetCnt;
    //设置频率128MHz
    iStatus = viWrite(iDevHandle, "FREQ 1.28e8\n", strlen("FREQ 1.28e8\n"),
    &uiRetCnt);
}
```

4.1.5 读取测量仪器状态

```
/*****************************************************************************/
下面的示例说明了如何读取仪器的设置状态。
/*****************************************************************************/
void ReadSettings()
{
    ViStatus iStatus;
    ViUInt32 uiRetCnt;
    char rgcBuf[256];

    //查询频率
    iStatus = viWrite(iDevHandle, "FREQ?\n", strlen("FREQ?\n"), &uiRetCnt);
    Sleep(10);
    iStatus = viRead(iDevHandle, rgcBuf, sizeof(rgcBuf), &uiRetCnt);
    //打印调试信息
    printf("频率为%s", rgcBuf);
}
```

4.1.6 命令同步

```
/*****************************************************************************/
下面以扫描过程为例说明了命令同步的方法。
```

```
/*****************************************************************************/
void SweepSync()
{
    ViStatus iStatus;
    ViUInt32 uiRetCnt;
    ViEventType eType;
    ViEvent eEvent;
    int iStat;
    char rgcOpcOk[2];

    /*********************************************************************/
    /* 命令INITiate[:IMMediate]启动单次扫描（连续扫描关闭时INIT:CONT OFF）*/
    /* 单次扫描结束时，才能执行命令缓冲区中的下一条命令。                  */
    /*********************************************************************/
    iStatus = viWrite(iDevHandle, "INIT:CONT OFF", 13, &uiRetCnt);
    //等待扫描结束的方法1：使用 *WAI
    iStatus = viWrite(iDevHandle, "ABOR;INIT:IMM;*WAI", 18, &uiRetCnt);
    //等待扫描结束的方法2：使用 *OPC?
    iStatus = viWrite(iDevHandle, "ABOR;INIT:IMM; *OPC?", 20, &uiRetCnt);
    iStatus = viRead(iDevHandle, rgcOpcOk, 2, &uiRetCnt); //等待*OPC返回“1”
    //等待扫描结束的方法3：使用 *OPC
    //为了使用GPIB服务请求，设置"Disable Auto Serial Poll"为"yes"
    iStatus = viWrite(iDevHandle, "*SRE 32", 7, &uiRetCnt);
    iStatus = viWrite(iDevHandle, "*ESE 1", 6, &uiRetCnt); //使能服务请求ESR
    //设置事件使能位，操作完成
    iStatus = viEnableEvent(iDevHandle, VI_EVENT_SERVICE_REQ, VI_QUEUE,
    VI_NULL);
    //使能SRQ事件
    iStatus = viWrite(iDevHandle, "ABOR;INIT:IMM;*OPC", 18, &uiRetCnt);
    //与OPC同步启动扫描
    iStatus = viWaitOnEvent(iDevHandle, VI_EVENT_SERVICE_REQ, 10000, &eType,
    &eEvent)
    //等待服务请求
    iStatus = viReadSTB(iDevHandle, &iStat);
    iStatus = viClose(eEvent); //关闭事件句柄
    //禁止SRQ事件
    iStatus = viDisableEvent(iDevHandle, VI_EVENT_SERVICE_REQ, VI_QUEUE);
    //主程序继续……
}
```

4.2 高级操作示例

- USBTMC程控示例..... 98

4.2.1 USBTMC 程控示例

- 使用示例之前..... 98
- 使用VISA库和C语言来实现设置和查询功能..... 98

4.2.1.1 使用示例之前

正确安装 VISA 库和 USBTMC 设备。调用 VISA 库的过程大致如下:

1) 用 viOpenDefaultRM 打开缺省的资源管理器，得到资源管理句柄 defaultRM。
2) 打开设备，有两种情况。

a) 已知被控仪器（设备）的资源地址。

用 viOpen 打开指定资源地址的仪器，得到仪器的会话句柄 vi，示例见 UsbTest1。

b) 不清楚被控仪器（设备）的资源地址。

用 viFindRsrc 和 viFindNext 查找可用的设备资源，用 viOpen 打开指定资源地址的仪器，得到仪器的会话句柄 vi，示例见 UsbTest2。

3) 用 viPrintf、viWrite 等函数进行设备写操作，用 viScanf、viRead 等函数进行设备读操作，用 viQueryf 等函数进行设备读写操作。
4) 用 viClose 函数关闭仪器句柄和资源管理句柄。

4.2.1.2 使用 VISA 库和 C 语言来实现设置和查询功能

```
/*************************************************************
本例用于查询和设置探头的频率，并最终恢复设置前的频率。
启动VC，添加必须的文件。把下述代码输入您的.cpp文件
*************************************************************/
#include "stdafx.h"
#include <visa.h>
#include <stdio.h>
#include <stdlib.h>

// USB地址，其中，“201202006”为87230的串号，每个87230的串号各不相同。
void MyDeviceIo(ViSession uiDev);
#define POWER_METER_USB_ADDR     "1204::4100::201803001"
#define ShowMsg(cMsg)   printf(cMsg)
//采用已知探头USB地址的方式进行设备打开。
void UsbTest1(void)
```

```
{
    ViSession uiRm; //缺省的资源管理器句柄
    ViSession uiDev; //设备句柄
    ViStatus iStatus = VI_SUCCESS; //状态
    ViChar rgcDesc[VI_FIND_BUFLEN]; //设备描述符

    sprintf(rgcDesc, "USB0::%s::INSTR", POWER_METER_USB_ADDR);
    iStatus = viOpenDefaultRM(&uiRm); //打开缺省的资源管理器
    if (iStatus)
    {
        ShowMsg("无法打开资源管理器，请重新检查设备并连接\n");
    }
    else
    {
        iStatus = viOpen(uiRm, rgcDesc, VI_NULL, 2000, &uiDev);
        if (iStatus)
        {
            ShowMsg("设备无法打开，请重新检查设备并连接\n");
        }
        else
        {
            MyDeviceIo(uiDev);          //成功打开一个USB设备，开始进行程控操作.
            viClose(uiDev);          //关闭设备
        }
        viClose(uiRm); //关闭默认管理句柄
    }
}

//搜索USBTMC设备，用于不知道USB设备的VID、PID、序列号的情况。
#define MAX_USBTMC_NUM      32                  //最多可能有多少个USBTMC设备
#define USBTMC_RSRC     "USB[0-9]*::?*INSTR"   //USBTMC设备资源匹配正则表达式
void UsbTest2(void)
{
    ViSession uiRm; //缺省的资源管理器句柄
    ViSession uiDev[MAX_USBTMC_NUM]; //设备句柄
    ViUInt32 uiCnt; //共找到多个设备
    ViFindList findList; //设备列表
    ViStatus iStatus = VI_SUCCESS; //状态
    ViChar rgcDesc[MAX_USBTMC_NUM][VI_FIND_BUFLEN]; //资源描述符
    ViUInt32 i; //循环变量
    iStatus = viOpenUiRm(&uiRm); //打开缺省的资源管理器
    if (iStatus)
```

```
    {
        ShowMsg("无法打开资源管理器，请重新检查设备并连接\n");
    }
    else
    {
        iStatus = viFindRsrc(uiRm, USBTMC_RSRC, &findList, &uiCnt, rgcDesc[0]);
        for (i=0; i<uiCnt; i++)
        {
            if ((0 == i) || (VI_SUCCESS == viFindNext(findList, rgcDesc[i])))
            {
                if (VI_SUCCESS == viOpen(uiRm, rgcDesc[i], VI_NULL, 2000, &uiDev[i]))
                {
                    MyDeviceIo(uiDev[i]);  //成功打开一个USB设备，开始进行程控操作.
                    viClose(uiDev[i]);  //关闭设备
                }
            }
        }
        viClose(uiRm); //关闭默认管理句柄
    }
}
void MyDeviceIo(ViSession uiDev)
{
    ViChar rgcBuf[VI_FIND_BUFLEN]; //缓冲区
    ViByte rgbRead[VI_FIND_BUFLEN]; //读取的缓冲区
    ViReal64 rgdFreq[2]; //频率
    ViUInt32 uiRetCnt; //读取的字节数
    //1）先查询探头的频率，并存入nFreq[0]
    viPrintf(uiDev, "SENS:FREQ?\n");
    Sleep(10);
    viRead(uiDev, rgbRead, sizeof(rgbRead), &uiRetCnt);
    rgbRead[uiRetCnt] = 0;
    ShowMsg((PCHAR)rgbRead);
    sscanf((PCHAR)&rgbRead[0], "%lf", &rgdFreq[0]);
    sprintf(rgcBuf, "探头的频率为：%lg\n", rgdFreq[0]);
    ShowMsg(rgcBuf);
    //2）设置探头的频率为16.78GHz
    viPrintf(uiDev, "SENS:FREQ  %lfGHz\n", 16.78);
    //3）查询探头的频率，并存入nFreq[1]
    viPrintf(uiDev, "SENS:FREQ?\n");
    viScanf(uiDev, "%t", rgcBuf);          //把查询结果放入数组
    ShowMsg(rgcBuf);
    sscanf(rgcBuf, "%lf", &rgdFreq[1]);
```

```
    sprintf(rgcBuf, "探头的频率为：%lg\n", rgdFreq[1]);
    ShowMsg(rgcBuf);
    //4）还原探头的频率
    viPrintf(uiDev, "SENS:FREQ  %lg\n", rgdFreq[0]);
}
```

5 错误说明

本章将告诉您如何发现问题并接受售后服务。并说明仪器出错信息。

- 错误信息.....103
- 返修方法.....108

5.1 错误信息

仪器采用两种途径记录测量过程中出现的错误：前面板显示错误信息队列和 SCPI（远程控制模式）错误信息队列，两种错误信息队列分别存储管理。

- 本地错误信息.....103
- 程控错误信息.....104

5.1.1 本地错误信息

- 错误信息查看.....103
- 错误信息说明.....103

1) 错误信息查看

通过界面操作方法:

如果使用过程中在仪器的右下脚显示有错误提示信息，则说明仪器软件运行或硬件出现问题。您根据错误代码可以大致判断问题类型，并采取相应措施排除故障。

在一个时刻，仪器错误显示区只能显示一条错误提示信息。由于仪器可能同时存在若干问题，执行下面的操作就可以看到所有错误提示信息：

步骤 1. 按【系统】，然后按[错误列表]，将会弹出错误列表窗口。

步骤 2. 提示信息会显示在窗口中。

步骤 3. 用鼠标可以浏览错误信息，关闭对话窗口。

步骤 4. 选择清除错误列表按钮可以清除历史错误信息。

2) 错误信息说明

仪器测量过程中若检测到错误，状态指示区右侧会显示告警或错误信息（错误缩写 + 详细错误说明），并以状态条的不同颜色标注错误级别：

表5.1　字体颜色区分状态指示区错误级别

颜色	错误级别	错误说明
红色	严重错误	测量过程中的严重错误，出现此类错误时，仪器无法正常工作。
红色	错误	测量过程中出现的错误，例如因数据丢失或错误的设置导致测量无法正常完成。
黑色	正常	无错误信息显示。

5.1.2 程控错误信息

1)　错误信息格式及说明

远程控制模式下，错误信息记录在状态报告系统中的错误/事件队列中，可由命令“SYSTem:ERRor?”查询错误信息，格式如下：

“<错误代码>, "<错误队列中错误信息>; <详细错误信息描述>”

举例:

“-110,"数据超界；输入参数超出下界。”

程控错误信息包括两种类型:

- SCPI标准定义的负值错误代码，该类错误信息在此不做具体说明。
- 仪器特性正值错误代码，具体说明如下表:

表5.2　仪器特性错误信息说明列表

错误代码	错误说明
-101	Invalid character 无效字符：命令串（命令或参数）中有无效字符。 例如：AVER:COUN !6
-102	Syntax error 语法错误：命令串语法无效。 例如: DISPlay:ACT,　CH1
-108	Parameter not allowed 参数不允许：命令的参数太多，或者无参数命令跟着参数。 例如:TRAC:AUT　ON
-109	Missing parameter 缺少参数：命令的参数太少。 例如：AVER
-112	Program mnemonic too long 命令串过长：命令中的单个段的字符超过12个字符。 例如：OUTPutROSCillatorSTATe ON
-113	Undefined header 未定义头：仪器收到无法识别的命令。可能原因：命令的拼写错误，或者缩写错误等。 例如:CALI:AUTO

错误代码	错误说明
-121	Invalid character in number 数值中存在无效字符：数值型参数中有无效字符。 例如: SENS:CORR:GAIN2 #12
-123	Exponent too large 指数太大：数值型参数的指数超过32000。 例如: SENS:CORR:GAIN2 1E32001
-124	Too many digits 位数太多：数值型参数的尾数超过255位，不含前面的0。
-128	Numeric data not allowed 数值型参数不允许：不能接收数值型参数的命令收到了一个数值。
-131	Invalid suffix 后缀无效：数值型参数的后缀不正确。 例如：FREQ 10GZ
-134	Suffix too long 后缀太长：后缀超过12个字符。 例如：FREQ 10GHHHHHHHHHHHHHHHHHHZ
-138	Suffix not allowed 后缀不允许：数值型参数不得跟后缀。 例如: SENS:CORR:GAIN2 12HZ
-148	Character data not allowed 字符数据不允许：检查是否需要增加引号。 例如：MEM:CLE State_1 正确：MEM:CLE "State_1"
-151	Invalid string data 无效字符串数据：检查字符串的单引号或双引号是否匹配。 例如：MEM:CLE“State1
-158	String data not allowed 字符型数据不允许：检查参数类型是否有效。 例如：OUTP:ROSC "ON"
-161	Invalid block data 无效块数据：根据IEEE 488.2第7.7.6节检查。
-168	Block data not allowed 块数据不允许：检测到了有效的数据块，但命令不支持数据块。 例如：OUTP:ROSC #15FETC?
-178	Expression data not allowed 表达式数据不允许：检测到了有效的表达式，但该仪器不允许表达式。 例如:SENS:CORR:GAIN2 (1+3)
-211	Trigger ignored 触发被忽略：当仪器不处于等待触发状态时，收到了TRIG:IMM，*TRG等命令。
-213	Init ignored

5.1 错误信息

错误代码	错误说明
	初始化被忽略：当仪器已经初始化了时，收到了测量初始化命令。 例如：INIT:CONT ON INIT
-214	Trigger deadlock 触发死锁。
-220	Parameter error；Frequency list must be in ascending order. 参数错误，频率列表必须按升序排列。
-221	Settings conflict 设置冲突：发生冲突的原因有多种，如在统计测量时，设置触发延时等。
-222	Data out of range 数据超范围：数值型数据不在有效的范围之内。 例如：AVER:COUN 100000
-224	Illegal parameter value 非法参数值：收到离散的参数，但对于该命令无效。
-226	Lists not same length 列表长度不相同。
-230	Data corrupt or stale；Please calibrate 数据无效或者损坏；请校准。
-231	Data questionable；CAL ERROR 数据疑问；校准错误：仪器校准失败。最可能的原因校准时是没有连接探头到校准源的输出端。 Data questionable；CAL ERROR Ch1 数据疑问；校准错误：仪器校准失败。最可能的原因校准时是没有连接探头到校准源的输出端。 Data questionable；Input Overload 数据疑问；输入过载：功率输入超过探头的功率上限。 Data questionable；Input Overload Ch1 数据疑问；输入过载：功率输入超过探头的功率上限。 Data questionable；Lower window log error 数据疑问；下窗口取对数错误：进行差值测量时，测量结果为0，并显示单位为对数。 Data questionable；Upper window log error 数据疑问；上窗口取对数错误：进行差值测量时，测量结果为0，并显示单位为对数。 Data questionable；ZERO ERROR 数据疑问；校零错误：仪器校零失败。最可能的原因校零时是有功率信号输入。 Data questionable；ZERO ERROR Ch1 数据疑问；校零错误：仪器校零失败。最可能的原因校零时是有功率信号输入。
-241	Hardware missing 缺少硬件：仪器无法执行该命令，原因是未接探头，或者探头类型不匹配。
-310	System error；Sensor EEPROM Read Failed – critical data not found or unreadable 系统错误；探头EEPROM读失败 – 关键数据未发现，或者不可读 System error；Sensor EEPROM Read Failed – unknown EEPROM table format 系统错误；探头EEPROM读失败 – 未知EEPROM格式。

错误代码	错误说明
-321	Out of memory 超出内存。
-330	Self-test Failed; 自测试失败。
-350	Queue overflow 队列溢出：错误队列满，随后的错误将不再记录。
-410	Query INTERRUPTED 查询被中断：某个命令需要往输出缓冲区发送数据，但发送缓冲区已经包含了以前命令发送的数据（不覆盖以前的数据）。输出缓冲区在关机或者收到*RST命令时清空。详细情况请参考IEEE488.2第6.3.2.3节。
-420	Query UNTERMINATED 查询未终止：仪器被设置为讲（也就是往接口总线上发送数据），但未收到将数据发送到输出缓冲区的命令。详细情况请参考IEEE 488.2第6.3.2.2节。 例如：在执行CONFigure命令（该命令不产生数据）之后，试图从远程接口读取数据。
-430	Query DEADLOCKED 查询死锁：输出缓冲区无法容纳命令产生太多的数据，并且输出缓冲区已满。命令继续执行但数据丢失。详细情况请参考IEEE 488.2第6.3.1.7节。
-440	Query UNTERMINATED after indefinite response 在不定响应后，查询未终止：某些组合查询可能产生非法响应信息。如果一个产生不定响应（不定长度的任意块响应或任意ASCII响应数据）的查询命令不是最后一个查询命令，仪器将报告查询错误，并且在该查询命令之后将不再发送响应。详细情况请参考IEEE 488.2第6.5.7.5节。

2) 错误信息类型

错误事件只对应一种错误信息，下面分类说明错误信息类型：

- **查询错误（-499 to -400）**：表明仪器的输出队列控制检测到 IEEE 488.2, Chapter 6 中描述的消息交换协议错误。此时，事件状态寄存器的查询错误位（bit2）置位（具体请参阅 IEEE 488.2, 6.5）。此时不能成功从输出队列中读取数据。
- **仪器特性错误（-399 to -300, 201 to 703, and 800 to 810）**：表明仪器操作未成功，原因可能是不正常的硬件或固件状态，该类错误代码常用于仪器自检测操作。此时，事件状态寄存器的仪器特性错误位（bit3）置位。
- **执行错误（-299 to -200）**：表明仪器执行测量过程中检测到错误。此时，事件状态寄存器的执行错误位（bit4）置位。
- **命令错误（-199 to -100）**：表明仪器命令解析过程中检测到的语法错误，一般是错误的命令格式导致。此时，事件状态寄存器的命令错误位（bit5）置位。

5.2 返修方法

- 联系我们.....108
- 包装与邮寄.....108

5.2.1 联系我们

若87230出现问题，首先观察错误信息并保存，分析可能的原因并参考用户手册中章节“7.2 故障诊断与排除”中提供的方法，予以先期排查解决问题。若未解决，请根据下面的联系方式与我公司服务咨询中心联系并提供收集的错误信息，我们将以最快的速度协助您解决问题。

联系方式:

服务咨询: 0532-86889847 400-1684191
技术支持: 0532-86880796
质量监督: 0532-86886614
传　　真: 0532-86889056
网　　址: www.ceyear.com
电子邮箱: techbb@ceyear.com
地　　址: 山东省青岛市黄岛区香江路98号
邮　　编: 266555

5.2.2 包装与邮寄

当您的仪器出现难以解决的问题时，可通过电话或传真与我们联系。如果经联系确认是仪器需要返修时，请您用原包装材料和包装箱包装仪器，并按下面的步骤进行包装:

1) 写一份有关仪器故障现象的详细说明，与仪器一同放入包装箱。
2) 用原包装材料将仪器包装好，以减少可能的损坏。
3) 在外包装纸箱四角摆放好衬垫，将仪器放入外包装箱。
4) 用胶带密封好包装箱口，并用尼龙带加固包装箱。
5) 在箱体上标明“易碎！ 勿碰！ 小心轻放！”字样。
6) 请按精密仪器进行托运。
7) 保留所有运输单据的副本。

注 意

包装仪器需注意

使用其它材料包装仪器，可能会损坏仪器。禁止使用聚苯乙烯小球作为包装材料，它们一方面不能充分保护仪器，另一方面会被产生的静电吸入仪器风扇中，对仪器造成损坏。

提 示

仪器的包装和运输

运输或者搬运本仪器（例如，因发货期间的损坏）时，请严格遵守用户手册章节“3.1.1.1 开箱”中描述的注意事项。

附 录

- 附录A SCPI命令按子系统分类速查表 111

附录 A SCPI 命令按子系统分类速查表

附表5.1 SCPI 命令按子系统分类速查表

序号	命令	功能简要说明
38	*CLS	清空仪器状态数据结构
38	*ESE	查询或设置标准事件状态使能寄存器
38	*ESR?	查询标准事件状态寄存器的值
38	*IDN?	查询仪器的标识串
39	*OPC	当所有等待执行的操作完成时，设置标准事件状态寄存器中的操作结束位
39	*RCL	调用指定的存储调用寄存器中的状态
39	*RST	复位仪器
39	*SAV	存储仪器状态到指定的寄存器中
39	*SRE	查询或设置服务请求寄存器
40	*STB?	查询状态字
40	*TRG	触发正在等待触发的所能通道
41	*TST?	执行自测试
41	*WAI	使仪器处于等待状态
87	:ABORt[1]	停止仪器测量
42	:CALCulate[1]:LIMit:CLEar:AUTO	控制何时清零极限 FCO（失败计数）
42	:CALCulate[1]:LIMit:CLEar[:IMMediate]	清零的 FCO（失败计数）
42	:CALCulate[1]:LIMit:FAIL?	查询是否有超过极限
42	:CALCulate[1]:LIMit:FCOunt?	查询的极限检测失败计数(FCO)
43	:CALCulate[1]:LIMit:LOWer[:DATA]	查询或设置测量极限的下限
44	:CALCulate[1]:LIMit:STATe	查询或设置测量极限检测开关
44	:CALCulate[1]:LIMit:UPPer[:DATA]	查询或设置测量极限的上限
45	:CALCulate[1]:MATH[:EXPRession]	查询或设置指定测量表达式

附录 A SCPI 命令按子系统分类速查表

45	:CALCulate[1]:MATH[:EXPRession]:CATalogue?	列举所有测量表达式
45	:CALCulate[1]:RELative[:MAGNitude]:AUTO	设置用于相对测量的参考值
45	:CALCulate[1]:RELative[:MAGNitude]:VALue?	查询用于相对测量的参考值
46	:CALCulate[1]:RELative:STATe	查询或设置相对测量开关状态
46	:CALibration[1]:ZERO:AUTO	校零仪器
46	:CALibration[1]:ZERO:TYPE	查询或设置探头的校零类型
47	:CONFigure[1][:SCALar][:POWer][:AC]	查询或设置的功率测量模式
47	:CONFigure[1][:SCALar][:POWer][:AC]:RELative	设置的绝对功率测量模式，打开相对测量
47	:FETCh[1][:SCALar][:POWer][:AC]?	设置为绝对功率测量，并关闭相对测量，返回测量值
48	:FETCh[1][:SCALar][:POWer][:AC]:RELative?	设置为绝对功率测量，并打开相对测量，返回测量值
49	:FORMat[:READings]:BORDer	查询或设置二进制数据的传输顺序
49	:FORMat[:READings][:DATA]	查询或设置数据传输格式
87	:INITiate[1]:CONTinuous	查询或设置发状态
87	:INITiate:CONTinuous:ALL	查询或设置所有通道的触发状态
88	:INITiate:CONTinuous:SEQuence[1]	查询或设置触发状态
88	:INITiate[1][:IMMediate]	设置仪器处于等待触发状态
88	:INITiate[:IMMediate]:ALL	设置所有通道处于等待触发状态
88	:INITiate[:IMMediate]:SEQuence[1]	设置仪器处于等待触发状态
48	:MEASure[1][:SCALar][:POWer][:AC]?	设置为绝对功率测量，并关闭相对测量，返回测量值
48	:MEASure[1][:SCALar][:POWer][:AC]:RELative?	设置为绝对功率测量，并打开相对测量，返回测量值
50	:MEMory:CATalog[:ALL]?	列举仪器中的用户配置，包括存储调用配置、频响偏置表（FDO）等
50	:MEMory:CATalog:STATe?	列举仪器中存储调用配置
51	:MEMory:CATalog:TABLe?	列举仪器中频响偏置表
51	:MEMory:CLEar[:NAME]	用于清零指定的频响偏置表或者存储调用表
52	:MEMory:CLEar:TABLe[1]	清空指定的频响偏置表
52	:MEMory:FREE[:ALL]?	查询用户配置空间未用的字节总数，以及已用的字节数
52	:MEMory:FREE:STATe?	查询存储调用空间未用的字节总数，以及已用的字节数

52	:MEMory:FREE:TABLe?	查询频响偏置表空间未用的字节总数，以及已用的字节数
52	:MEMory:NSTates?	查询存储调用状态个数，始终返回 10
52	:MEMory:STATe:CATalog?	列举所有存储调用状态的名称
52	:MEMory:TABLe[1]:DEFine	查询或设置指定频响偏置表的名称
53	:MEMory:TABLe[1]:FREQuency	查询或设置指定频响偏置表的频率列表
53	:MEMory:TABLe[1]:FREQuency:POINts?	查询指定频响偏置表的频率点数
53	:MEMory:TABLe[1]:GAIN[:MAGNitude]	查询或设置指定频响偏置表的幅度增益列表
53	:MEMory:TABLe[1]:GAIN[:MAGNitude]:POINts?	查询指定频响偏置表的幅度增益点数
54	:MEMory:TABLe:MOVE	重命名指定频响偏置表
54	:MEMory:TABLe:SELect	查询或设置当前频响偏置表
49	:READ[1][:SCALar][:POWer][:AC]?	设置为绝对功率测量，并关闭相对测量，返回测量值
49	:READ[1][:SCALar][:POWer][:AC]:RELative?	设置为绝对功率测量，并打开相对测量，返回测量值
55	[:SENSe[1]:AVERage[1]:COUNt	查询或设置通道平均次数
55	[:SENSe[1]:AVERage:COUNt:AUTO	查询或设置通道自动平均状态
55	[:SENSe[1]:AVERage:SDETect	查询或设置通道步进检测状态
55	[:SENSe[1]:AVERage[1][:STATe]	查询或设置通道平均开关状态
56	[:SENSe[1]:CORRection:CSET2[:SELect]	选择频响偏置表
56	[:SENSe[1]:CORRection:CSET:STATe	查询或设置频响偏置表的启用状态
56	[:SENSe[1]:CORRection:FDOFfset|GAIN4[:INPut][:MAGNitude]?	查询频响偏置因子
56	[:SENSe[1]:CORRection:GAIN2:[INPut][:MAGNitude]	查询或设置通道偏置
57	[:SENSe[1]:CORRection:GAIN2:[INPut]:STATe	查询或设置道偏置启用状态
57	[:SENSe[1]:FREQuency[:CW|FIXed]	查询或设置通道频率
57	[:SENSe[1]:POWer:AC:RANGe	查询或设置探头的量程
58	[:SENSe[1]:POWer:AC:RANGe:AUTO	查询或设置探头的自动量程开关状态
90	:SERVice:LANGuage	查询或设置语言选择
90	:SERVice:SENSor[1]:CDATe?	查询探头的校准日期
90	:SERVice:SENSor[1]:CPLace?	查询探头的校准地点
90	:SERVice:SENSor[1]:FREQuency:MAXimum?	查询最大频率

附录 A SCPI 命令按子系统分类速查表

91	:SERVice:SENSor[1]:FREQuency:MINimum?	查询最小频率
91	:SERVice:SENSor[1]:SNUMber?	查询串号
91	:SERVice:SENSor[1]:TYPE?	查询类型
91	:SERVice:SNUMber	查询或设置仪器的串号
73	:STATus:DEVice:CONDition?	查询设备状态条件寄存器中的值
73	:STATus:DEVice:ENABle	查询或设置设备状态事件使能寄存器
74	:STATus:DEVice[:EVENt]?	查询设备事件寄存器
74	:STATus:DEVice:NTRansition	查询或设置设备负跳变过滤器
74	:STATus:DEVice:PTRansition	查询或设置设备正跳变过滤器
74	:STATus:OPERation:CALibrating[:SUMMary]:CONDition?	查询校准操作状态条件寄存器中的值
75	:STATus:OPERation:CALibrating[:SUMMary]:ENABle	查询或设置校准操作事件使能寄存器
75	:STATus:OPERation:CALibrating[:SUMMary][:EVENt]?	查询校准操作事件寄存器
75	:STATus:OPERation:CALibrating[:SUMMary]:NTRansition	查询或设置校准操作负跳变过滤器
75	:STATus:OPERation:CALibrating[:SUMMary]:PTRansition	查询或设置校准操作正跳变过滤器
75	:STATus:OPERation:CONDition?	查询操作状态条件寄存器中的值
76	:STATus:OPERation:ENABle	查询或设置操作状态事件使能寄存器
76	:STATus:OPERation[:EVENt]?	查询操作状态事件寄存器
76	:STATus:OPERation:LLFail[:SUMMary]:CONDition?	查询下限检测操作状态条件寄存器中的值
76	:STATus:OPERation:LLFail[:SUMMary]:ENABle	查询或设置下限检测操作事件使能寄存器
77	:STATus:OPERation:LLFail[:SUMMary][:EVENt]?	查询下限检测操作事件寄存器
77	:STATus:OPERation:LLFail[:SUMMary]:NTRansition	查询或设置下限检测操作负跳变过滤器
77	:STATus:OPERation:LLFail[:SUMMary]:PTRansition	查询或设置下限检测操作正跳变过滤器
77	:STATus:OPERation:NTRansition	查询或设置操作状态负跳变过滤器
77	:STATus:OPERation:PTRansition	查询或设置操作状态正跳变过滤器
78	:STATus:OPERation:SENSe[:SUMMary]:CONDition?	查询传感操作状态条件寄存器中的值

78	:STATus:OPERation:SENSe[:SUMMary]:ENABle	查询或设置传感操作事件使能寄存器
78	:STATus:OPERation:SENSe[:SUMMary][:EVENt]?	查询传感操作事件寄存器
78	:STATus:OPERation:SENSe[:SUMMary]:NTRansition	查询或设置传感操作负跳变过滤器
79	:STATus:OPERation:SENSe[:SUMMary]:PTRansition	查询或设置传感操作正跳变过滤器
79	:STATus:OPERation:ULFail[:SUMMary]:CONDition?	查询上限检测操作状态条件寄存器中的值
79	:STATus:OPERation:ULFail[:SUMMary]:ENABle	查询或设置上限检测操作事件使能寄存器
79	:STATus:OPERation:ULFail[:SUMMary][:EVENt]?	查询上限检测操作事件寄存器
79	:STATus:OPERation:ULFail[:SUMMary]:NTRansition	查询或设置上限检测操作负跳变过滤器
80	:STATus:OPERation:ULFail[:SUMMary]:PTRansition	查询或设置上限检测操作正跳变过滤器
80	:STATus:PRESet	预置一些状态寄存器
80	:STATus:QUEStionable:CALibration[:SUMMary]:CONDition?	查询校准可疑状态条件寄存器中的值
81	:STATus:QUEStionable:CALibration[:SUMMary]:ENABle	查询或设置校准可疑事件使能寄存器
81	:STATus:QUEStionable:CALibration[:SUMMary][:EVENt]?	查询校准操作事件寄存器
81	:STATus:QUEStionable:CALibration[:SUMMary]:NTRansition	查询或设置校准可疑负跳变过滤器
81	:STATus:QUEStionable:CALibration[:SUMMary]:PTRansition	查询或设置校准可疑正跳变过滤器
81	:STATus:QUEStionable:CONDition?	查询可疑状态条件寄存器中的值
82	:STATus:QUEStionable:ENABle	查询或设置可疑状态事件使能寄存器
82	:STATus:QUEStionable[:EVENt]?	查询可疑状态事件寄存器
82	:STATus:QUEStionable:NTRansition	查询或设置可疑状态负跳变过滤器
82	:STATus:QUEStionable:POWer[:SUMMary]:CONDition?	查询功率可疑状态条件寄存器中的值
83	:STATus:QUEStionable:POWer[:SUMMary]:ENABle	查询或设置功率可疑事件使能寄存器
83	:STATus:QUEStionable:POWer[:SUMMary][:EVENt]?	查询功率可疑事件寄存器
83	:STATus:QUEStionable:POWer[:SUMMary]:NTRansition	查询或设置功率可疑负跳变过滤器

附录 A SCPI 命令按子系统分类速查表

83	:STATus:QUEStionable:POWer[:SUMMary]:PTRansition	查询或设置功率可疑正跳变过滤器
84	:STATus:QUEStionable:PTRansition	查询或设置可疑状态正跳变过滤器
86	:SYSTem:ERRor[:NEXT]?	从仪器的错误队列中返回错误代码和错误信息
86	:SYSTem:HELP:HEADers?	查询仪器支持的命令列表
86	:SYSTem:PRESet	复位仪器到参数指定的状态
86	:SYSTem:VERSion?	查询仪器使用的 SCPI 版本号
89	:TRIGger[1][:IMMediate]	设置仪器处于等待触发状态
89	:UNIT[1]:POWer	查询或设置功率单位
89	:UNIT[1]:POWer:RATio	查询或设置比值测量功率单位