

## 目录-第 5 章"远程控制-"基础"

<b>5</b>	<b>远程控制-基础.....</b>	<b>5.1</b>
	引论.....	5.1
	入门指南.....	5.2
	开始远程控制操作.....	5.3
	远程控制期间显示的内容.....	5.3
	使用IEC/IEEE总线远程控制.....	5.4
	设置设备地址.....	5.4
	返回手动操作.....	5.4
	使用RS - 232接口远程控制.....	5.5
	设置传输参数.....	5.5
	返回手动操作.....	5.5
	限制.....	5.5
	通过网络（RSIB接口）远程控制.....	5.6
	设置设备地址.....	5.6
	返回手动操作.....	5.6
	信息.....	5.7
	IEC/IEEE -总线接口信息.....	5.7
	设备信息（指令与设备响应）.....	5.8
	设备信息的结构和语法.....	5.9
	SCPI介绍.....	5.9
	指令结构.....	5.9
	指令行的结构.....	5.12
	查询的响应.....	5.12
	参数.....	5.13
	语法元素的概述.....	5.14
	设备模型与指令处理.....	5.15
	输入单元.....	5.15
	指令识别.....	5.16
	设备数据库和设备硬件.....	5.16
	输出单元.....	5.17
	指令次序和指令同步.....	5.17
	状态报告系统.....	5.18
	SCPI状态寄存器的结构.....	5.18
	状态寄存器概述.....	5.20
	状态寄存器说明.....	5.21
	状态字节（STB）和服务请求启动寄存器（SRE）.....	5.21
	IST标志和并行查询启动寄存器（PPE）.....	5.22
	事件-状态寄存器（ESR）和事件-状态-启动寄存器（ESE）.....	5.22
	STATus:OPERation寄存器.....	5.23
	STATus:QUEStionable寄存器.....	5.24
	STATus QUEStionable:ACPLimit寄存器.....	5.25
	STATus QUEStionable:FREQuency寄存器.....	5.26
	STATus QUEStionable:LIMit<1 2>寄存器.....	5.27
	STATus QUEStionable:LMARgin<1 2>寄存器.....	5.28
	STATus QUEStionable:POWEr寄存器.....	5.29

STATus-QUEStionable:SYNC寄存器.....	5.30
状态报告系统的应用 .....	5.31
服务请求、使用层次结构.....	5.31
串行查询.....	5.31
并行查询.....	5.32
使用指令查询.....	5.32
错误-队列查询.....	5.32
状态报告系统的恢复缺省值.....	5.33

## 5 远程控制-基础

在这一章中将介绍

- 如何使用远程控制操作FSP
- 概括地讲述了可编程设备的远程控制。解释了SCPI标准的指令结构和语法，指令的执行以及状态寄存器。
- FSP状态寄存器中使用的框图和表格。

第6章详细介绍了全部的远程控制功能。按照字母顺序——讲述根据SCPI建立的各个子系统。第6章最后的指令列表依据字母顺序列出了所有指令和他们的参数。

第7章是FSP编程范例。

第8章是远程控制接口和接口功能。

## 引论

根据IEC 625.1/IEEE 488.2和RS-232接口标准，设备配备了IEC总线接口。接口在设备背面，它可以连接一个远程控制器。此外，如果安装了选项B16，设备还可以通过本地局域网(LAN接口)进行远程控制。

设备支持SCPI:version 1997.0 (可编程设备标准指令集)。SCPI标准以IEEE488.2为基础，它的目的是规范化设备专用指令，差错处理和状态寄存器(见"SCPI入门"一节)。“自动测量控制-SCPI与IEEE 488.2教程”是根据John M.Pieper (R & S 编号0002.3536.00) 提供的有关SCPI的概念和定义的详细信息制定的。关于网络远程控制的叙述在“通过网络(RSIB接口)实现远程控制”一节中。

本节讲述了IEC/IEEE总线编程和控制器使用的基础知识。接口指令的说明可以参考有关的手册。

SCPI标准对指令语法、差错处理与状态寄存器设置的要求，会在以后几节里详细说明。我们使用表格对状态寄存器的比特分配做了简介。状态寄存器的详细说明是表格的补充。

IEC-总线编程的范例全部以VISUAL BASIC编写。

## 入门指南

以下给出了一个简短的操作程序，可以快速将设备打开并对其进行基本的功能设置。作为前提，不能改变IEC/IEEE总线地址的出厂设置，它的出厂设置为20。

1.使用IEC/IEEE-总线电缆将控制器连接到设备上。

2.在控制器上输入以下程序并开始运行它

```
CALL IBFIND ( "DEV1" , analyzer%)           '打开与设备连接的端口
CALL IBPAD   ( analyzer%, 20)                 '将设备地址告知控制器
CALL IBWRT   ( analyzer%, ' * RST; * CLS ')    '重置设备

CALL IBWRT   ( analyzer%, 'FREQ:CEN 100MHz ') '设置中心频率为100MHz
CALL IBWRT   ( analyzer%, 'FREQ:SPAN 10MHz ') '设置频跨为10MHz
CALL IBWRT   ( analyzer%, 'DISP:TRAC:Y:RLEV - 10dBm ') '设置参考电平为-10dBm
```

设备开始在95MHz到105MHz的频带内扫描

3.按下面板上的LOCAL键，返回手动控制

## 开始远程控制操作

接通电源后、设备就处于手动操作状态("LOCAL"状态)，可以使用面板进行操作。

它会切换到远程控制("REMOTE"状态)，

IEC/IEEE-总线                当它收到从控制器传来的编址指令。

如果它处于网络之中( RSIB接口)，当它收到控制器发出的指令。

RS-232                        当它收到控制器发出的"@\_REM"指令

在远程控制期间、面板的操作会被屏蔽掉。在使用面板或远程控制器将设备重置到手动状态之前，它会一直保持在远程控制状态。在手动状态和远程控制状态之间切换不会影响其它设置。

## 远程控制期间显示的内容

远程控制过程中，只显现LOCAL软键盘，可使用它返回手动操作状态。

此外、使用"SYSTem:DISPlay:UPDate OFF"指令(远程控制过程中的缺省值) 可以停止显示框图和结果，以获得设备的最佳性能。

在程序执行期间，建议使用"SYSTem:DISPlay:UPDate ON"激活结果显示，以配合设置的改变并且可以记录屏幕上测量曲线。

**注意** 如果设备只通过远程控制方式操作,推荐使用省电模式( POWER SAVE)。在这个模式下、在预设时间过去后，各种显示都会完全关闭。

## 使用IEC/IEEE总线远程控制

### 操作设置设备地址

要通过IEC-总线操作设备,就必须使用IEC/IEEE总线编址。IEC/IEEE总线的地址的出厂设置为20。它可以在菜单SETUP-GENERAL SETUP中手动修改,也可以使用IEC总线修改,从0到30都是可用地址。

#### 手动修改

- 调用菜单SETUP-GENERAL SETUP
- 在表GPIB-ADDRESS中输入需要的地址
- 使用ENTER键结束输入

#### 通过IEC/IEEE总线

CALL IBFIND ( " DEV1 ", analyzer%)	'打开与设备连接的端口
CALL IBPAD ( analyzer%, 20)	'将旧地址告知控制器
CALL IBWRT ( analyzer%, " SYST:COMM:GPIB:ADDR 18 ")	'设置设备新地址
CALL IBPAD ( analyzer%, 18)	'将新地址告知控制器

### 返回手动操作

可以使用面板或IEC/IEEE总线返回手动操作。

#### 手动地

- 按下LOCAL软键盘或PRESET键

**注意** -传送之前,指令处理必须全部完成。否则远程控制操作的传送会立即执行。

-使用通用指令LLO,可以屏蔽掉键盘,(见第8章、IEC/IEEE-总线接口-接口信息)以防止无意地传送。这样的话,只有通过IEC/IEEE总线才可以切换到手控方式。

-键盘可以通过停用IEC/IEEE总线的REN线路重新使能。(见第8章、IEC/IEEE-总线接口-总线)。

#### 通过IEC总线:

```
.....
CALL IBLOC ( analyzer%)      '设置设备为手动操作
.....
```

## 通过RS-232-接口远程控制操作

### 设置传输参数

为了能够得到无差错的数据传输,设备和控制器的参数设置要完全相同。

参数可以手动地在COM PORT表格中的菜单SETUP-GENERAL SETUP中修改, 或者使用指令 SYSTem:COMMunicate:SERial :...远程控制操作。

COM接口的传输参数, 出厂设置如下:

波特率=9600、数据位数= 8、停止位=1、奇偶校验=NONE 和 所有者=INSTRUMENT。

远程控制操作时、接口应分配到操作系统 (所有者= OS) 所以控制特性包括@可以被辨别的接口。

#### 手动地

##### 设置COM接口

- 调用菜单SETUP-GENERAL SETUP
- 在表格COM PORT中选择期望的波特率、位、停止位、奇偶校验。
- 在表格COM PORT中设置所有者为OS。
- 使用ENTER键终止输入。

### 返回手动操作

通过面板或者RS-232标准接口都可以返回手动操作状态。

#### 手动地

- 按下LOCAL软键盘或PRESET键

##### 注意

- 在传送之前、必须完成指令处理否则远程控制操作的传送会立即执行
- 通过RS-232接口发送控制字符串" @LOC ", 键盘可以被再次使能 (参见第 8 章、RS-232-C接口-控制指令)。

#### 通过RS-232接口

.....

v24puts ( port、" @\_LOC " ) ; 把设备设置为手动。

.....

### 限制

如果设备通过RS-232-C标准接口进行远程控制, 以下限制有效

- 没有接口信息、只有控制字符串 (见第 8 章的接口描述、RS-232-C接口-控制指令)。
- 只有公共指令\* OPC?可以用来做同步指令, \* WAI和 \* OPC失效。
- 不能传输数据块。

## 通过网络（RSIB接口）远程控制

### 设置设备地址

要在网络中控制设备、必须使用预设的IP地址访问设备。设备的IP地址（设备地址）在网络配置中定义。

### 设置IP地址

- 调用菜单SETUP-GENERAL SETUP-CONFIGURE NETWORK。
- 选择"协议"列表。
- 在" Properties "中设置TCP/IP协议的IP地址（见选项FSP - b16一节）。

### 返回手动操作

可以手动使用面板或者RSIB接口远程控制返回手动操作。

### 手动地

- 按下LOCAL软键盘或者PRESET键。

### 注意

- 确信在传送之前指令全部执行完成，否则设备会立即转回远程控制操作。

### 使用RSIB接口

.....

调用RSDLLibloc ( analyzer%, ibsta%, iberr%, ibcntl &)'把设备设为手动控制

.....



## 信息

通过IEC总线的数据线传送的信息可以分成二类（参见第8章、IEC/IEEE-总线接口）

-接口信息和

-设备信息。

### IEC/IEEE -总线接口信息

接口信息通过IEC总线的数据线传输、" ATN"控制线处于激活状态。他们用来在控制器和设备之间通信，并且只能通过具有IEC/IEEE总线控制的控制器发送。接口指令可以再分为

-广播指令和

-定址指令。

广播指令作用于所有已连接到IEC/IEEE总线但还没有编址的设备，定址指令只作用于已编址为对应接收者的设备。与设备有关的接口信息都列在第8章IEC/IEEE总线接口-接口功能之中。

## 设备信息（指令与设备响应）

设备信息使用ASCII码，通过IEC总线的数据线传送、" ATN"控制线处于未激活状态。

按照在IEC/IEEE总线上发送的方向不同可以区分：

**-指令** 是控制器发送给设备的信息。他们管理设备的功能并请求查询信息。

指令按照二个标准细分::

按照他们对设备产生的作用

**设置指令** 决定了设备设置，比如重置设备或者设置中心频率。

**查询** 将资料输出到IEC/IEEE总线上、例如设备的识别或者标记查询。

依照他们在IEEE 488.2标准中的定义

**公共指令** 在IEEE 488.2标准中明确地定义了功能与表达式。功能包括管理标准状态寄存器、重置与自检。

**针对具体设备指令**

功能取决于设备的性能，如频率设定。SCPI委员会已将大多数的这种指令规范化了（对照." SCPI介绍"一节））。

**-设备响应** 是在查询之后设备发送给控制器的信息。他们可以包含测量结果、设备设置与有关设备状态的信息（对照." 查询的响应"一节））。

设备信息的结构与语法将在下面几节讲述。

## FSP设备信息的结构和语法

### SCPI介绍

SCPI ( 可编程设备标准指令集 ) 描述了一套跨越设备类型与生产商的可编程设备标准指令集。SCPI的目标是将大部分设备的具体指令标准化。为了这个目的、定义了一个针对同一设备内部或不同设备内部的相同功能都适用的模型。指令系统的产生就是为了完成这些功能。如此就可以使用相同的指令描述相同的功能。指令系统具有分层结构。

图 5 - 1使用一个树状结构图解说明了SENSe指令系统，它控制设备的具体设置,而与被测量信号的信号特征无关。SCPI基于标准IEEE 488.2，也就是说，它和这个标准所定义的公共指令有相同的语法元素。部分设备响应的语法比IEEE 488.2的限制更多(见"查询的响应"一节)。

### 指令结构

指令由所谓的指令头和 ,大多数场合下都有 ,一个或多个参数组成。指令头和参数通过一个"空隔"分隔( ASCII 码十进制的0到9、11到32、例如，空白 )。指令头可以由几个关键字组成。查询就是直接在指令头上添加一个问号而形成。

**注意**                    以下范例中使用的指令，并非在设备的每种情况中都能用到。

**公共指令**                公共指令有一个星号" \* "作为前缀、还包括指令头、以及，如果需要的话，一个或几个参数。

范例	* RST	RESET,重置设备
	* ESE 253	EVENT STATUS ENABLE,设置事件状态使能寄存器的比特
	*ESR?	EVENT STATUS QUERY、查询事件状态寄存器。

## 针对具体设备指令

**层次** 针对具体设备指令具有分层结构（见图5-1）。不同的指令头组合代表了不同的层次。最高一层(根层)的指令头只包括一个关键字。这个关键字表示一整套指令系统。

**范例** SENSE 这个关键字表示SENSe指令系统。

下层指令,必须有完整的路径,由左边起,从最高一层开始,关键字之间使用冒号" "分隔。

**范例** SENSE:FREQuency: SPAN 10MHZ

这条指令位于SENSe系统的第三层。它用于设置频带宽度。

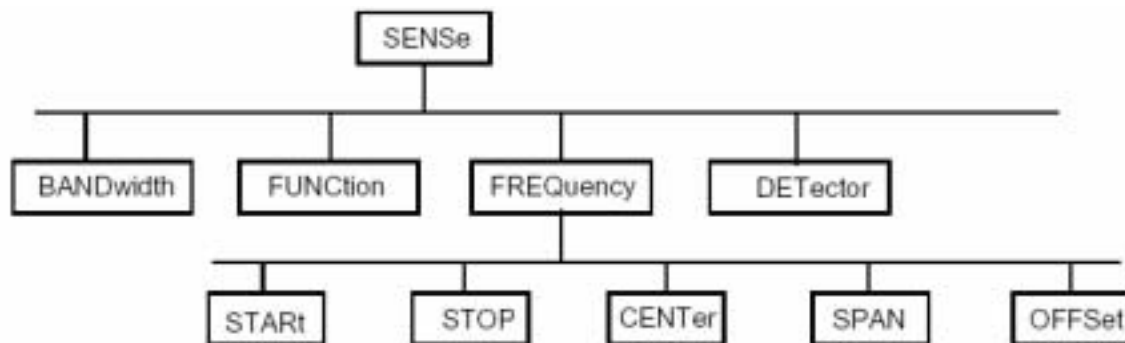


图5-1 SCPI指令系统的树状结构，SENSe系统的范例

一些关键字在同一指令系统的几个层中都存在。此时它们的作用取决于指令的结构、即、它们插入在指令头的哪个位置。

**范例** SOURce:FM:POLarity NORMal

这条指令包含第三层的关键字POLarity。它定义了调制器和调制信号之间的极性关系。

SOURce:FM:EXTernal:POLarity NORMal

这条指令包含第四层的关键字POLarity。它定义了指明外部信号源的情况下，调制压降和调制结果方向的极性。

- 可选关键字** 一些指令系统允许某个关键字可选地插入指令头中或者省略。这些关键字在说明中以方括号标明。这是出于对SCPI标准兼容性的考虑，设备必须可以辨认无缩写指令。一些指令由于可选关键字会显著地缩短。
- 范例 [SENSe]:BANDwidth[:RESolution]:AUTO
- 这条指令把设备的分辨率带宽和其他参数联系起来。下面的指令具有相同效果
- BANDwidth:AUTO
- 注意** 如果可选关键字的相关功能要求指定数字后缀，关键字就不可省略。
- 无缩写与缩写** 关键字有无缩写格式和缩写格式。可以随意键入缩写格式或无缩写格式、但其他任何形式的缩写都不可用。
- Beispiel :STATus:QUEStionable:ENABle 1 = STAT:QUES:ENAB 1
- 注意** 缩写格式只以大写字母组成、无缩写格式要求完整的单词拼写。表达式的大写和小写只为了简洁、对设备的作用与其无关。
- 参数** 指令头与参数之间必须有个"空格"。如果一个指令中有几个参数，则他们以逗号","分隔。少数查询可以输入MINimum, MAXimum和DEFault这样的参数。参数类型的说明在"参数"一节。
- 范例 SENSE:FREQUENCY:STOP?MAXimum                      响应 3.5E9
- 这是查询停止频率的最大值。
- 数字后缀** 如果设备具有几个功能或者几个同类的功能，例如输入,那么可以通过给指令添加后缀,选择所要求的功能。无后缀的输入被解释为后缀为1的输入。
- 范例: SYSTem:COMMunicate:SERial2:BAUD 9600
- 这条指令设置第二个串行接口的波特率。

## 指令行的结构

指令行可以由一条或者几条指令组成。 New Line ,带EOI的 New Line 或者最后跟一个EOI的数据字节都可以结束一个指令行。控制器的IEC/IEEE驱动通常会自动地在最后的数据字节后添加EOI。

同一指令行中的指令用分号";"分隔。如果下一个指令属于不同的指令系统、分号后面要再跟一冒号。

范例

```
CALL IBWRT ( analyzer%, " SENSE:FREQuency:CENTer 100MHz;:INPut:ATTenuation 10 ")
```

指令行中包含二条指令。第一个属于SENSe指令系统，用于设置设备的中心频率。第二个属于INPut指令系统，设置输入信号衰减。

如果连续的指令都属于相同的系统，有一层或者几层相同、指令行可以缩写。要缩写、分号后的第二条指令就要从低于相同层的那一层开始写起（参见图 5 -1）。这样的话，分号后的冒号必须省略。

范例

```
CALL IBWRT ( analyzer%, " SENSE:FREQuency:STARt 1E6;:SENSe:FREQuency:STOP 1E9 ")
```

指令行中没有缩写并包含二条用分号隔开的指令。两条指令都属于SENSe指令系统的FREQuency子系统，就是说，他们有二个相同层。当缩写指令行时、第二条指令从SENSe:FREQuency层下面开始写起。分号之后的冒号要省略。

缩写形式的指令行如下

```
CALL IBWRT ( analyzer%, " SENSE:FREQuency:STARt 1E6;STOP 1E9 ")
```

但是、一条新的指令行必须从完整的路径开始写起。

范例 CALL IBWRT ( analyzer, " SENSE:FREQuency:STARt 1E6 ")

```
CALL IBWRT ( analyzer%, " SENSE:FREQuency:STOP 1E9 ")
```

## 查询的响应

除非明确地指出，否则查询都是对设置指令而言的。它是在相关设置指令后增加一个问号构成的。按照SCPI规定、在一定程度上查询响应比IEEE 488.2标准更加严格。

1.被查询的参数传回时不带指令头。

范例 INPut:COUPling ? 响应 DC

极大值、极小值等特别数值、要使用专用的文本参数查询，然后它们会以数值形式传回。

范例 SENSe:FREQuency:STOP?MAX 响应 3.5E9

3.数值是不带有单位的输出。物理量的量纲会选用基本单位或Unit指令设置的单位。

范例 SENSe:FREQuency:CENTer ? 响应 1E6代表1兆赫

真值<逻辑值>的返回值为0（表示OFF）或1（表示ON）。

范例 SENSe:BANDwidth:AUTO ? 响应 1表示ON

文本（字符数据）以缩写格式返回（参见3.5.5一节）。

范例 SYSTem:COMMunicate:SERial:CONTrol:RTS? 响应 STAN（代表标准）

## 参数

大多数指令需要指定参数。参数与指令头之间必须用"空格"隔开。可以使用的参数包括数值、逻辑参数、文本、字符串和数据块。需要哪一种参数要看各自的指令和指令说明中指定的取值范围。

**数值** 数值可以有各种形式，带有符号的、有小数点和指数形式的。取值若超出设备分辨率则进位舍入。数值部分可以包含最多255个字符、指数范围在-32000到32000之间。指数用" E "或者" e "表示。仅输入指数值是不容许的。就物理量而言、可以输入单位。可用的前缀有G（千兆）、MA（兆）、（MOHM和MHZ也容许）、K（千）、M（毫）、U（百万分之一）和N（毫微）。如果没有添加单位、就使用基本单位。

范例

SENSe:FREQuency:STOP 1.5GHz = SENSe:FREQuency:STOP 1.5E9

**特别数值** 文本MINimum, MAXimum, DEFault, UP和DOWN被仪器作为特别数值解释。

查询时、会用到特别数值。

范例      设置指令      SENSe:FREQuency:STOP MAXimum

                 查询              SENSe:FREQuency:STOP?              响应    3.5E9

MIN/MAX      MINimum和MAXimum表示极小值和极大值。

DEF            DEFault表示已经保存在EPROM中的预置值。这个值会被\* RST指令调用 ,所以又称默认值。

UP/DOWN      UP/DOWN，按步长增加或者减少数值。步长可以用UP和DOWN在每个参数的步长设置指令中（见附录C、指令目录）进行修改。

INF/NINF      正无穷（INF）负无穷(NINF)分别表示数值9.9e37和-9.9e37。正负无穷大只作为设备发出的响应。

NAN            不定数(NAN)表示值9.91E37.NAN只作为设备发出的响应。这个值没有定义。它可以是零对零的除法、无限对无限的减法和漏测值的表现。

**逻辑参数** 逻辑参数有两态。ON状态(逻辑真)表示为ON或者一个不等于0的数值。OFF状态(逻辑假)表示为OFF或者数字值0。0或1作为查询的结果出现。

范例    设定指令      DISPlay:WINDow:STATe ON

                 查询              DISPlay:WINDow:STATe?              响应    1

文本	文本参数遵守关键字的语法规则，也就是说，可以使用缩写或者无缩写形式输入文本。与其它参数相似、他们与指令头之间必须用"空隔"隔开。查询返回的结果是缩写形式的文本。		
	范例	设定指令	INPut:COUPling GROund
		查询	INPut:COUPling ? 响应GRO
字符串	字符串的输入必须使用引号（'或者"）。 范例    SYSTem:LANGuage"SCPI "    或者 SYSTem:LANGuage' SCPI '		
数据块	数据块是适合于大量数据传输的传输格式。使用数据块参数的指令具有以下结构 范例    HEADer:HEADer #45168xxxxxxxx  数据块由ASCII码字符#引导。#后的数字表示随后有几个数字描述数据块的长度。范例中跟随的 4 个数字指出了长度是5168字节。数据字节紧随其后。在数据字节传输期间，所有的End或其他控制信号都会被忽视，直到全部字节传输完毕。		

语法组件概观

下面简述语法组件的概观。



冒号用于分隔指令中的关键字。  
指令行中分号之后的冒号，表示跟随了最高一层的指令。



分号用于分隔一个指令行中的二个指令。它不改变路径。



逗点用于分隔一个指令中的几个参数。



问号表示查询。



星号表示公共指令。



引号用于引导一个字符串并终止它。



双剑号（#）用来引导数据块



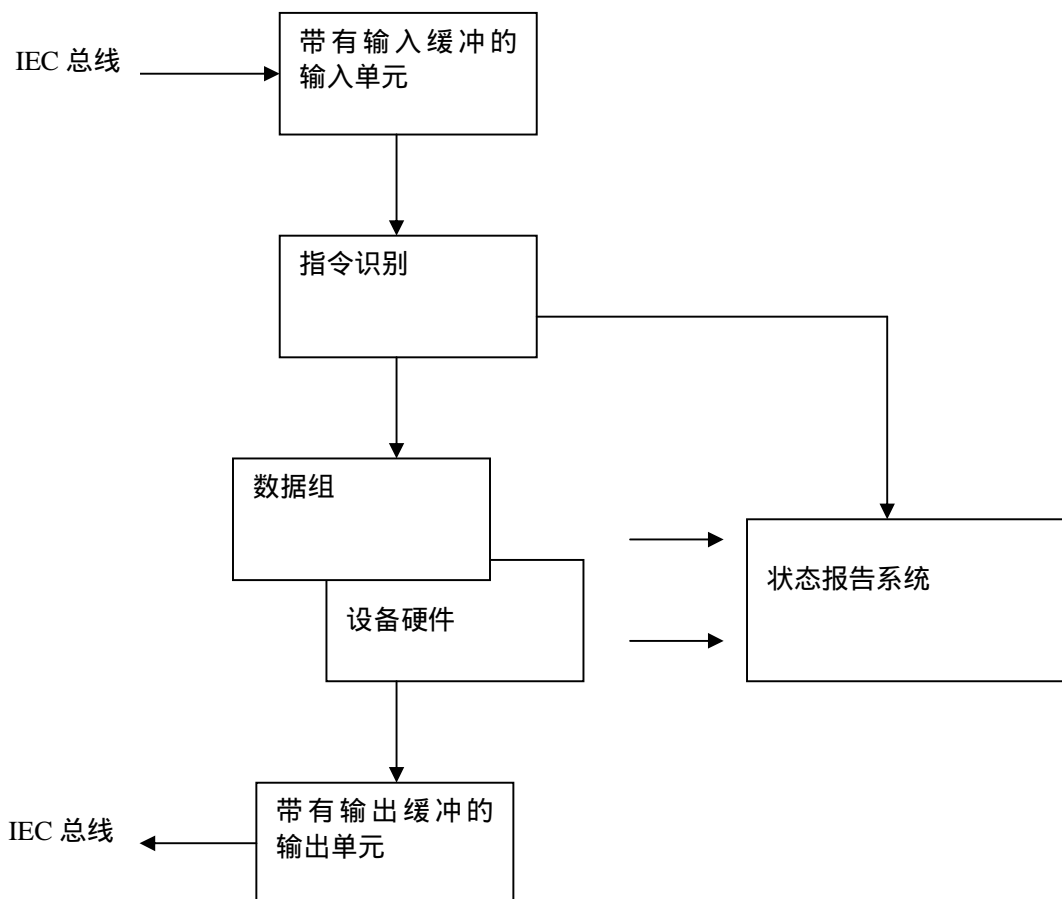
"空隔（ASCII码的0到9、11到32十进制、例如空白）用于分隔指令头和参数。



## 设备模型和指令处理

图5-2表示设备模型，从IEC总线指令执行的方面观察。每一个单独的组件彼此独立工作又同时运行。他们用所谓的“信息”通信。

图5-2 使用IEC总线远程控制操作的设备模型



## 输入单元

输入单元从IEC总线按字符接收指令，并将其聚集到输入缓冲器。一旦输入缓冲器充满或接收到一个定界符，如IEEE 488.2中定义的<程序信息终止>、或是接口信息DCL，输入单元就发送一个信号到指令识别器。

如果输入缓冲器充满、IEC-总线会停止传输并处理已收到的数据。然后再继续IEC-总线传输。然而，如果当接收到定界符时，缓存器仍未充满、那么输入单元可以在识别指令和执行指令的期间继续接收下一条指令。收到DCL后，输入缓冲器会立即清空并发送信息到指令识别单元。

## 指令识别

指令识别单元会分析从输入单元收到的数据，并按照数据接收的顺序进行处理。只有DCL的执行具有优先权、例如、GET（群执行触发器）、也只有指令接收之后才被执行。每个识别出的指令会立即被转到设备数据库中，但不会立刻执行。

指令中的句法误差由指令识别单元判明，再提供给状态报告系统。语法错误之外的指令行会被继续分析，如果可能执行的话还会执行。

如果指令识别系统发现了定界符（<程序信息分割号>或者<程序信息终止号>）或者DCL、它会立刻要求设备数据库将指令发送给设备硬件。接着它立即准备处理下一条指令。由于指令处理系统的工作方式，当硬件仍正在设置时，下一步的指令可以早已执行完毕（"并行执行"）。

## 设备数据库和设备硬件

这里讲述了"设备硬件"，它表示执行实际功能的设备部分-信号发生器、测量部分等，但控制器不包括在内。

设备数据库是使用软件实现的硬件设备细节再现。

IEC总线的设定指令会导致数据设置的变化，数据库管理单元会输入新的数值（例如频率）到数据库、然而只有当指令识别系统要求时他们才被送到硬件。

仅在被送到硬件设备之前，才检验数据之间以及数据与设备硬件之间的符合性，如果检测完成而无法执行、一个"执行错误"信号会送到状态报告系统。数据库的改动被取消、设备硬件也不重置。

IEC总线查询，引发数据库管理单元将被查询数据发送到输出单元。

## 状态报告系统

状态报告系统收集设备状态的信息，并按查询的要求进行输出。确切的结构与功能在3.8节中叙述。

输出单元

响应控制器的查询要求，输出单元从数据库管理单元收集相关信息。并按照SCPI规定处理信息，使其可以送到输出缓冲器。如果设备编址为信息源,输出缓冲器既不包含数据也没有等待数据库管理传送数据，输出单元则发送错误信息"查询无终结"到状态报告系统。如IEC总线上无数据转送、控制器会一直等，直到超出等待时限。这些特性在SCPI中定义。

指令次序和指令同步

上面已经解释了，所有的指令在一定状态下都可以并行执行。

为了防止并行执行指令、可以使用指令\* OPC, \* OPC?或者\* WAI。这三条指令，都要求在硬件已经设置完成并且稳定后方可执行。适当的编程、可以强制控制器等待，直到发生相应动作（对照。表格 5 - 1）。

表格5-1使用\* OPC, \* OPC?和 \* WAI的同步

指令	硬件稳定后的动作	控制器编程
* OPC	在ESR中设置操作-完成比特	-在ESE中设置比特0 -在SRE中设置比特5 -等待服务请求（SRQ）
* OPC?	写入" 1 "到输出缓冲器	将设备编址为信息源
* WAI	继续IEC总线握手	发送下一个指令

关于指令同步的范例在"程序范例"一章中可以找到。

对指令进行强制同步，直到指令执行完毕是为了得到期望的结果。被影响的指令或者需要为完成设备设置(例如自动设置范围)做超出一次的测定，或者需要较长的时段执行。如果在执行对应功能的期间，接收到新的指令，可能会导致测量异常中止或无效的测量数据。

下面列出了需要强制使用\* OPC?、\* OPC?或者\* WAI进行同步的指令

表格5-1 带有强制同步指令（并行指令）

指令	目的
INIT	开始测量
INIT:CONM	继续测量
CALC:MARK:FUNC:ZOOM	在标记1周围缩放频带
CALC:STAT:SCAL:AUTO ONCE	优化信号统计测量功能的电平设定
[ SENS ] POW:ACH:PRES:RLEV	优化邻道功率测量的电平设定

状态报告系统

状态报告系统（参看图 5 - 4 ）保存所有有关当前设备操作状态的信息,例如设备目前正在执行校准，已发生的错误。这些信息保存在状态寄存器和错误队列中。状态寄存器和错误队列可以使用IEC总线查询。

状态信息具有层次结构。在IEEE 488.2中定义的寄存器状态字节（STB）和它的关联屏蔽寄存器服务请求使能（SRE）构成了最高一层。STB从标准事件状态寄存器(ESR)和包含设备详细信息的STATus:OPERation和STATus:QUEStionable处采集信息，ESR与其关联屏蔽寄存器标准事件状态使能（ESE）是在IEEE 488.2中定义的，寄存器STATus:OPERation和STATus:QUEStionable是在SCPI中定义的。

IST标志("独立状态")和分配给它的并行查询使能寄存器(PPE)也是状态报告系统的一部分。IST标志,如同SRQ,把总体的设备状态整合在一个比特中。PPE对于IST标志就像SRE对于服务请求,他们执行同样的功能，输出缓冲器包含设备返回控制器的信息。它不属于状态报告系统，但是决定了STB中的MAV比特的值，如图5 - 4所示。

SCPI状态寄存器的结构

每个SCPI寄存器包括 5 个字段,每个字段都具有16比特并具有不同的功能（对照。图 5 - 3 ）。每一个比特都彼此独立，就是说，每个硬件状态都分配一个比特数，这个数对所有的五个字段都有效。例如：在所有五个字段的比特 3 的STATus:OPERation寄存器，都分配给了硬件状态"等待触发"。每一字段的比特15（最大比特）都设置为零。如此一来，寄存器中每一字段的内容都能够被控制器作为正整数处理。

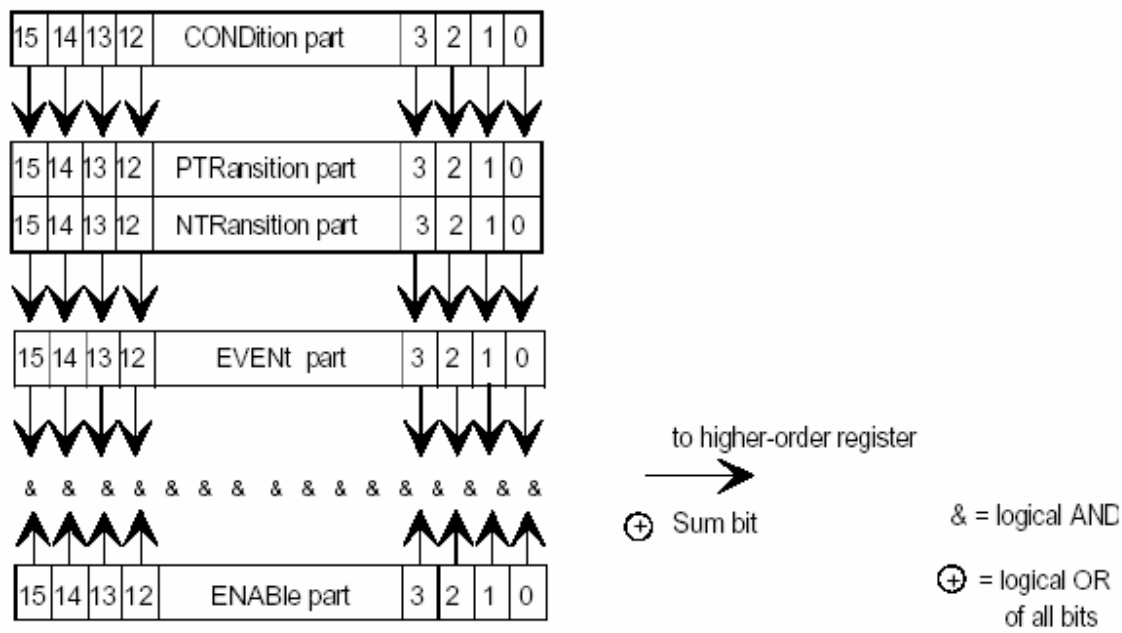


图5 - 3 状态-寄存器模型

**CONDition字段** CONDition字段是直接由硬件写入的或者是下一级寄存器的总比特。它反映当前的设备状态。这些记录只可读、不可写入也不可清除。读出不会影响它的内容。

**ptransition字段** 正方向-传送字段起到一个边缘检测器的作用。当CONDition字段中的一个比特从0变到1,相关的NTR比特会决定EVENT比特是否要置1。

PTR比特= 1 EVENT比特置1。

PTR比特= 0 EVENT比特不置1。

这个字段可以任意读写。读出不影响它的内容。

**ntransition字段** 反方向-传送字段也作为一个边缘检测器工作。当CONDition字段中的一个比特从0变到1,相关的NTR比特会决定EVENT比特是否要置1。

NTR-比特= 1 EVENT比特置1。

NTR-比特= 0 EVENT比特不置1。

这个字段可以任意读写。读出不影响它的内容。

使用这二个边缘寄存器字段,用户可以定义状态字段(无,0到1,1到0或者两者都是)的哪一个状态转移保存在EVENT字段中。

**EVENT字段** EVENT字段显示自从上次读以来是否又发生了某个事件、它是状态字段的"记忆"。它只显示边缘滤波器传递的事件。它由设备更新。

用户只能读出这一字段。在读出后、它的内容会被设置为零。在语言用法的方面,这个字段同总体寄存器往往符合。

**ENABLE字段** ENABLE字段决定相关的EVENT比特是否要改变概要比特(参看以下讲述)。每个比特的EVENT字段要和相关的ENABLE比特相与(符号'&')。这一字段的所有逻辑操作的结果会通过或操作(符号'+')传递到概要比特上。

ENABLE-比特= 0 相关的EVENT比特不影响概要比特

ENABLE-比特= 1 如果关联EVENT比特是"1"、概要比特也置"1"。

这字段用户可以任意读写。读出不影响它的内容。

**Sum比特** 如上所述,概要比特是由各个寄存器的EVENT和ENABLE字段产生的。然后结果进入上一层寄存器的CONDition字段的一个比特。

设备自动为每个寄存器产生概要比特。某些事件、例如一个尚未锁定的锁相回路、可能导致一个贯穿所有层次的服务请求。

**注意** 如果STB是依照SCPI建立的,那么在IEEE 488.2中定义的服务请求使能寄存器SRE能够作为STB的ENABLE字段。以此类推、ESE可以作为ESR的ENABLE字段。

## 状态寄存器概观

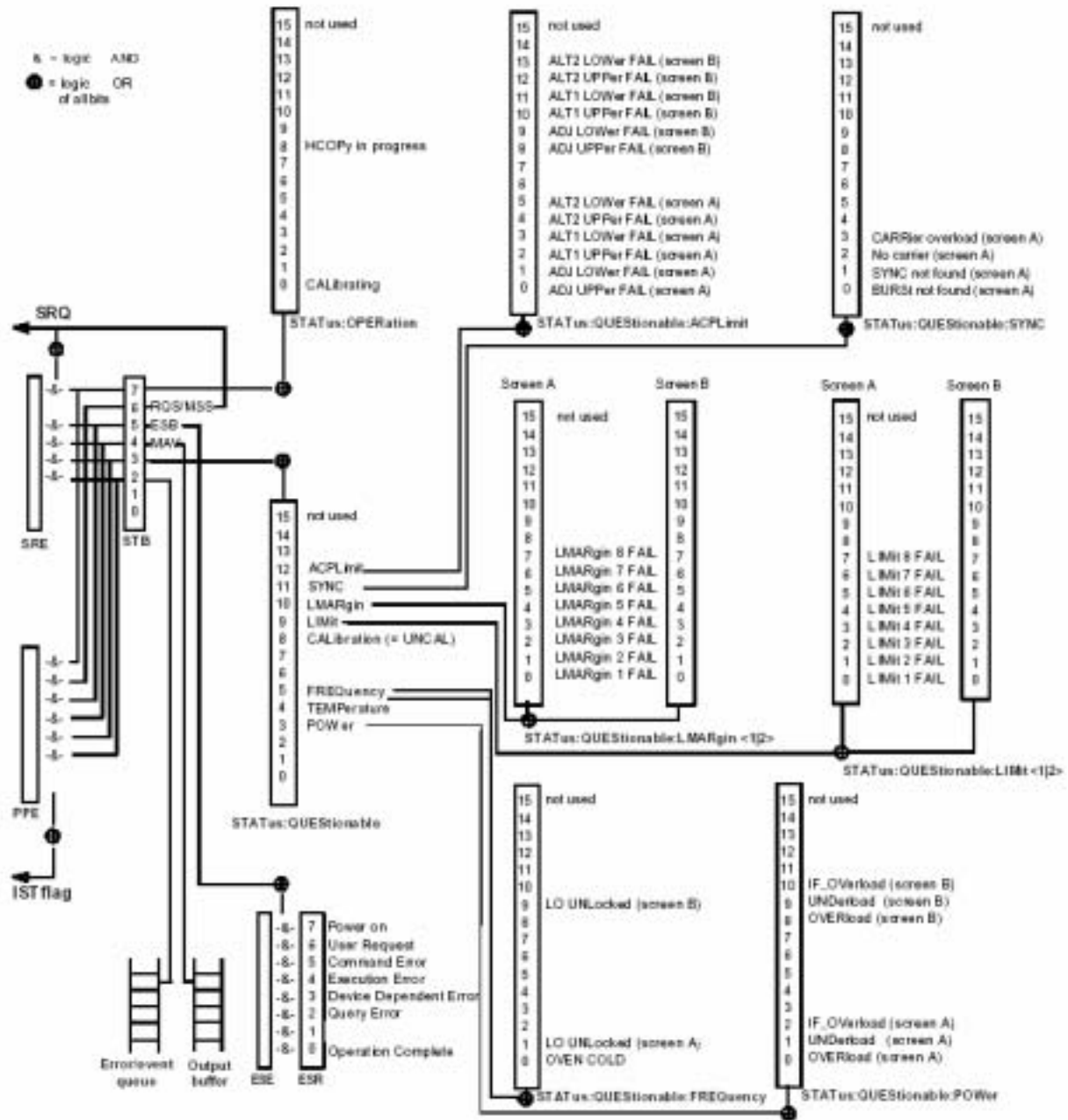


图5-4 状态寄存器概观

状态寄存器说明

状态字节（STB）和服务请求使能寄存器（SRE）

STB已在IEEE 488.2中定义了.它收集下级的寄存器信息，提供一个粗略的设备状态概观。它可以和SCPI寄存器的CONDition字段相比较，并在SCPI层次中担当最上层。比特 6 的作用是状态字节剩余比特的sum比特。使用指令" \* STB ? "或者串行查询可以读出状态字节。

STB包含SRE。它的功能使它相当于SCPI寄存器的ENABLE字段。STB的每个比特都在SRE中对应一个比特。而SRE的比特6则被忽略。如果SRE中的一个比特和相关的STB比特从0变化到1,在IEC总线上就会产生一个服务请求( SRQ)、当配置合适并且它可以被进一步处理时，那么服务请求会触发控制器中断。

SRE可以使用指令" \* SRE "设置，使用指令" \* SRE? "查询。

表格5 - 2种状态字节中各比特的含意

比特编号	意义
2	<b>错误队列非空</b>  错误队列有了输入的时候这个比特置一。  如果SRE设定这个比特使能,每个错误队列的输入都会引起一个服务请求。通过查询错误队列可以判明那个错误并确定更多的细节。查询能给出一个更详细的错误信息。推荐使用这个程序，因为它大大减少了涉及IEC总线控制的问题。
3	<b>QUESTionable状态概要比特</b>  如果QUESTionable中的EVENT比特置" 1 "这个比特也置" 1 " 状态寄存器和相关的使能比特置1。  一个比特置" 1 "表示一种不可靠的设备状态,通过查询QUESTionable状态寄存器可以确定更多的细节。
4	<b>MAV比特（信息可用）</b>  如果输出缓冲器有一条有效信息可以读出，那么这个比特置" 1 "。  这个比特使控制器能够自动从设备读取数据(参看第7章,程序范例)。
5	<b>ESB比特</b>  事件状态寄存器的概要比特。如果事件状态使能寄存器已经使能，并且事件状态寄存器的一个比特置" 1 "，它会随之置" 1 "。  这个比特置" 1 "意味发生了一个错误或者一个事件，通过查询事件状态寄存器可以确定更多的细节。
6	<b>MSS比特（主要状态概要比特）</b>  如果设备触发了一次服务请求，这个比特置" 1 "。这种情况是，如果这个寄存器的另一个比特同它在服务请求使能寄存器SRE中的屏蔽位置" 1 "。
7	<b>操作状态寄存器概要比特</b>  如果操作-状态寄存器中的一个EVENT比特置" 1 "而且相关的ENABLE比特置" 1 ",那么这个比特置" 1 "。  比特置" 1 "表示设备正在执行操作。可以通过查询操作-状态寄存器改变操作类型。

## IST标志和并行查询使能寄存器（PPE）

与SRQ类似, IST标志在单一比特中结合了总体状态信息。它可以使用并行查询(参看3.8.4.3节)或者使用"\* IST?"指令进行查询。

并行查询使能寄存器（PPE）确定STB的哪个比特会影响IST标志。STB比特与对应PPE比特相"与",同比特6在SRE中的使用相反。IST标志由所有的结果进行"或"运算产生。PPE可以使用"\* PRE"设定,使用"\* PRE?"指令读取。

## 事件-状态寄存器（ESR）和事件-状态-使能寄存器（ESE）

ESR已经在IEEE 488.2.中定义,它可与SCPI寄存器的EVENTt字段相比。

事件状态寄存器可以使用"\* ESR?"指令读取。

ESE是它的相关的使能字段。它可以使用"\* ESE"指令设定,使用"\* ESE?"指令读取。

表格5 - 3事件状态寄存器中的比特意义

比特编号	意义
0	<b>操作完成</b> 在收到* OPC指令之时, 前面所有的指令都已经执行完毕, 这个比特置"1"。
1	这个比特不使用
2	<b>查询错误</b> 如果控制器不发送查询而要从设备读出数据,或者没有取出请求数据反而发送了新的指令到设备, 这个比特置"1"。原因往往是查询不规格并由此无法执行。
3	<b>设备-相关错误</b> 如果发生设备-相关错误, 这个比特置"1"。一个错误信息跟随着一个-300到-399之间的编号, 或者是一个正的错误编号、那表示有更多的误差的细节、输入了错误队列(参看第9章、错误信息)。
4	<b>执行错误</b> 如果接收到一个句法上正确,但是,因为其他理由无法执行的指令, 这个比特置"1"。一个错误信息跟随一个-200到-300之间的编号、那表示有更多的误差的细节、输入了错误队列(参看第9章、错误信息)。
5	<b>指令错误</b> 如果接收的指令不明确或句法不正确, 这个比特置"1"。一个错误信息跟随一个-100到-200之间的编号, 那表示有更多的误差的细节, 输入了错误队列(参看第9章"错误信息")。
6	<b>用户请求</b> 按下LOCAL键后, 这个比特置"1"。
7	<b>接通电源(电源电压开启)</b> 当设备打开电源, 这个比特置"1"。



**STATus:OPERation寄存器**

在状态字段中、这个寄存器包含设备正在执行的操作信息,在事件段中,这个寄存器包含设备自从上次读取以来执行的操作的信息。它可以使用"STATus:OPERation:CONDition?"或者"STATus:OPERation[:EVENT?]"指令读取。

表格5 - 4 在STATus.OPERation寄存器中的比特的意义

比特编号	意义
0	<b>校准</b> 只要设备在执行校准,这个比特就置"1"。
1到7	这些比特不使用
8	<b>硬拷贝进行中</b> 当设备打印硬拷贝时,这个比特置"1"。
9到14	这些比特不使用
15	这个比特常为0

**STATus:QUEStionable寄存器**

这个寄存器包括了当设备没有按照技术要求操作时可能出现的不稳定状态信息。它可以使用 STATus:QUEStionable:CONDition?和STATus:QUEStionable [:EVENT]?指令查询

.表格5-5种STATus:QUEStionable寄存器中的比特含义

比特编号	意义
0到2	这些比特不使用
3	<b>POWER</b> 如果发生电源不可靠的情况(也请参看"STATus QUEStionable:POWer寄存器"一节), 这个比特置"1"。
4	<b>TEMPerature</b> 如果不可靠的温度出现, 这个比特置"1"。
5	<b>FREQuency</b> 如果频率不可靠(参看"状态:可疑:频率寄存器"一节),这个比特置"1"。
6到7	这些比特不使用
8	<b>CALibration</b> 如果测量执行时仪器未校准(= ^标签"UNCAL"), 这个比特置"1"。
9	LIMit (针对具体设备) 如果突破了极限值(参见状态:可疑:极限寄存器一节), 这个比特置"1"。
10	LMARgin (针对具体设备) 如果突破了显示边际(参见STATus QUEStionable:LMARgin<1 2>寄存器一节), 这个比特置"1"。
11	SYNC (设备-相关) 如果在GSM MS模式下测量或预测量的时候、中频同步失败或者没有发现脉冲, 那么这个比特置"1"。 如果在GSM MS模式下预测量、结果与预期值严重不符合(参见"STATus-QUEStionable:SYNC寄存器"), 这个比特也置"1"。
12	ACPLimit (针对具体设备) 如果邻道功率测量的极限被突破(参见节"状态:可疑:邻道功率极限寄存器"), 这个比特置"1"。
13到14	这些比特不使用
15	这个比特常为0。

**STATus:QUEStionable:ACPLimit寄存器**

这个寄存器包含了邻道功率测量极限的信息。它可以通过指令

STATus:QUEStionable:ACPLimit:CONDition?'和' STATus:QUEStionable:ACPLimit [:EVENT]?'查询

表格5 - 6状态：可疑：邻道功率极限寄存器的比特含义

比特编号	意义
0	<b>ADJ UPPer FAIL (屏幕A)</b> 如果屏幕A中上邻道极限被超过，这个比特置"1"。
1	<b>ADJ LOWer FAIL (屏幕A)</b> 如果屏幕A下邻道极限被超过，这个比特置"1"。
2	<b>ALT1 UPPer FAIL (屏幕A)</b> 如果屏幕A第一上更替信道极限被超出，这个比特置"1"。
3	<b>ALT1 LOWer FAIL (屏幕A)</b> 如果屏幕A第一下更替信道极限被超出，这个比特置"1"。
4	<b>ALT2 UPPer FAIL (屏幕A)</b> 如果屏幕A第二上更替信道极限被超出，这个比特置"1"。
5	<b>ALT2 LOWer FAIL (屏幕A)</b> 如果屏幕A第二下更替信道极限被超出，这个比特置"1"。
6 到 7	未使用
8	<b>ADJ UPPer FAIL (屏幕B)</b> 如果屏幕B中上邻道极限被超过，这个比特置"1"。
9	<b>ADJ LOWer FAIL (屏幕B)</b> 如果屏幕B中下邻道极限被超出，这个比特置"1"。
10	<b>ALT1 UPPer FAIL (屏幕B)</b> 如果屏幕B第一上更替信道极限被超出，这个比特置"1"。
11	<b>ALT1 LOWer FAIL (屏幕B)</b> 如果屏幕B第一下更替信道极限被超出，这个比特置"1"。
12	<b>ALT2 UPPer FAIL (屏幕B)</b> 如果屏幕B第二上更替信道极限被超出，这个比特置"1"。
13	<b>ALT2 LOWer FAIL (屏幕B)</b> 如果屏幕B第二下更替信道极限被超出，这个比特置"1"。
14	未使用
15	这个比特是总是设置为0。

STATus QUEStionable:FREQuency寄存器

这个寄存器包含有关参考源和本机振荡器的信息。  
它可以使用指令是STATus:QUEStionable:FREQuency:CONDition?和"  
STATus :QUEStionable:FREQuency [ :EVENT]?查询。

表格5 - 7状态：可疑：频率寄存器的比特含义

比特编号	意义
0	<b>OVEN COLD</b> 如果参考振荡器还没有达到它的工作温度，这个比特置"1"。将显示'OCXO'。
1	LO UNLocked（屏幕A） 如果本机振荡器不再锁定，这个比特置"1"。于是将显示'LOUNL'。
2到8	未使用
9	LO UNLocked（屏幕B） 如果本机振荡器不再锁定，这个比特置"1"。然后将显示 'LOUNL '
10到14	未使用
15	这个比特常为0。

**STATus QUEStionable:LIMit<1|2>寄存器**

这个寄存器包含对应测量窗中要注意的临界线的信息（极限1与屏幕A对应、极限2与屏幕B对应）。它可以使用STATus:QUEStionable:LIMit<1|2>:CONDition?和STATus:QUEStionable:LIMit<1|2> [:EVENT]?指令查询。

表格5 - 8STATus QUEStionable:LIMit<1|2>寄存器的比特含义

比特编号	意义
0	<b>LIMit 1 FAIL</b> 如果临界线1被突破，这个比特置"1"。
1	<b>LIMit 2 FAIL</b> 如果临界线2被突破，这个比特置"1"。
2	<b>LIMit 3 FAIL</b> 如果临界线3被突破，这个比特置"1"。
3	<b>LIMit 4 FAIL</b> 如果临界线4被突破，这个比特置"1"。
4	<b>LIMit 5 FAIL</b> 如果临界线5被突破，这个比特置"1"。
5	<b>LIMit 6 FAIL</b> 如果临界线6被突破，这个比特置"1"。
6	<b>LIMit 7 FAIL</b> 如果临界线7被突破，这个比特置"1"。
7	<b>LIMit 8 FAIL</b> 如果临界线8被突破，这个比特置"1"。
8到14	未使用
15	这个比特常为0。

STATus QUEStionable:LMARgin<1|2>寄存器

这个寄存器包含在对应测量窗中要注意的屏幕边际极限（LMARgin1对应屏幕A、LMARgin2对应屏幕B）。它可以使用指令STATus:QUEStionable:LMARgin<1|2>:CONDition?和"STATus :QUEStionable:LMARgin<1|2> [ :EVENT]?查询。

表格5 - 9STATus QUEStionable:LMARgin<1|2>寄存器的比特含义

比特编号	意义
0	<b>LMARgin 1 FAIL</b> 如果显示边际极限1被突破，这个比特置"1"。
1	<b>LMARgin 2 FAIL</b> 如果显示边际极限2被突破，这个比特置"1"。
2	<b>LMARgin 3 FAIL</b> 如果显示边际极限3被突破，这个比特置"1"。
3	<b>LMARgin 4 FAIL</b> 如果显示边际极限4被突破，这个比特置"1"。
4	<b>LMARgin1 5 FAIL</b> 如果显示边际极限5被突破，这个比特置"1"。
5	<b>LMARgin1 6 FAIL</b> 如果显示边际极限6被突破，这个比特置"1"。
6	<b>LMARgin1 7 FAIL</b> 如果显示边际极限7被突破，这个比特置"1"。
7	<b>LMARgin1 8 FAIL</b> 如果显示边际极限8被突破，这个比特置"1"。
8到14	未使用
15	这个比特常为0。

**STATus QUEStionable:POWer寄存器**

这个寄存器包含所有的设备可能发生的过载的信息。

它可以使用STATus:QUEStionable:POWer:CONDition?和" STATus:QUEStionable:POWer [ :EVENT]?指令查询。

表格5 - 10STATus QUEStionable:POWer寄存器的比特含义

比特编号	意义
<b>0</b>	OVERload (屏幕A) 如果射频输入过载，这个比特置"1"。然后将显示' OVLD'。
<b>1</b>	UNDerload ( 屏幕A) 如果射频输入轻载，这个比特置"1"。然后将显示' UNLD'。
<b>2</b>	IF_OVerload ( 屏幕A) 如果IF路径过载，这个比特置"1"。然后将显示' IFOVL '。
3到7	未使用
<b>8</b>	OVERload ( 屏幕B) 如果射频输入过载，这个比特置"1"。然后将显示' OVLD'。
<b>9</b>	UNDerload ( 屏幕B) 如果射频输入轻载，这个比特置"1"。然后将显示' UNLD'。
<b>10</b>	IF_OVerload ( 屏幕B) 如果IF路径过载，这个比特置"1"。然后将显示' IFOVL '。
11到14	未使用
15	这个比特常为0。

**STATus-QUEStionable:SYNC寄存器**

只有GSM MS模式中使用这个寄存器。它包含有关同步、未发现的脉冲、预测量结果超出或没有达到期望值的信息。这个比特可以使用" STATus:QUEStionable:SYNC:CONDition? " " STATus:QUEStionable:SYNC [ :EVENT]? "指令查询。

表格5 - 11STATus-QUEStionable:SYNC寄存器的比特含义

比特编号	意义
0	<b>BURSt not found ( 屏幕A )</b> 在GSM MS模式下,由于因为相位/频率的错误( PFE )或载波功率时间比( PVT ),在测量/预测量中没有发现脉冲, , 这个比特置" 1 "。如果在测量/预测量中发现了脉冲,这个比特会复位。
1	<b>SYNC not found ( 屏幕A )</b> 在GSM MS模式下,由于因为相位/频率的错误( PFE )或载波功率时间比( PVT ),没有在测量/预测量中发现中速同步序列(训练序列), 这个比特置" 1 "。 如果在测量/预测量中发现了中速同步序列(训练序列)、 这个比特复位。
2	<b>No carrier ( 屏幕A )</b> 在GSM MS模式中,如果由于调制造成的预测量设置的载波功率时间比( PVT)和频谱电平值过低, 这个比特置" 1 "。这个比特在预测量开始时复位(参见第2章,指定的预测量说明)。
3	<b>Carrier overload ( 屏幕A )</b> 在GSM MS模式中,如果由于调制造成的预测量设置的载波功率时间比( PVT)和频谱电平值过高, 这个比特置" 1 "。这个比特在预测量开始时复位(参见第2章,指定的预测量说明)。
4- 14	未使用。
15	这个比特常为0。



## 状态报告系统的应用

为了更有效地使用状态报告系统、它所包含的信息一定要送到控制器并且在那里做进一步的处理。在下文中讲述了几个方法。详细的程序范例在第 7 章、程序范例中。

### 服务请求、层次结构的使用

在某种情况下、设备可以发送服务请求 (SRQ) 到控制器。通常服务请求会引起控制器中断、而控制程序可以采取相应动作。如图5-4所示, 在SRE使能的情况下, 如果状态字节的第2, 3, 4, 5或者第7比特中的一个或者几个位置"1", 那么SRQ会被初始化。这些比特的每一个都整合了更进一步的寄存器信息, 如错误队列或者输出缓冲器。状态寄存器的ENABLE段的对应设置可以实现对任意一个状态寄存器中的任意比特初始化SRQ。为了使服务请求可以正常工作, 使能寄存器SRE和ESE中的所有比特应该设置为"1"。

范例 (也请参看图5 - 4和第 7 章、程序范例) 使用"\*OPC"指令在扫描的最后产生一个SRQ。

- CALL IBWRT ( analyzer%, " \* ESE 1 ")将ESE的比特0置"1" (操作完成)
- CALL IBWRT ( analyzer%, " \* SRE 32 ")将SRE (ESB)?中的比特5置"1"

设置完成后、设备会产生一个SRQ。

只有SRQ可能使仪器自动运行。各个控制器程序都应该设置成使仪器在发生故障的情况下可以初始化服务请求。程序应该对服务请求做出恰当的反应。在第 7 章、程序范例中有一个关于服务请求例行程序的详例。

### 串行查询

在串行查询中、使用"\*STB"指令、会查询仪器的一个状态字节。无论如何, 查询是通过接口信息实现的并显而易见地迅速。这种串行查询方法已经在IEEE 488.1中定义了, 并且它曾经是可以对不同仪器查询状态字节的唯一标准。这种方法对不遵守SCPI或者IEEE 488.2的仪器也适用。VISUAL BASIC执行串行查询的指令是"IBRSP ()"。串行查询主要用于快速察看连接到IEC总线的几个仪器的状态。

## 并行查询

在并行查询中、控制器使用一条指令，在各个数据线上分别传达1比特的信息，最多可以同时查询八套仪器，也就是、设置分配给各个仪器的数据线为逻辑"0"或者"1"。以此类推，SRE寄存器决定了在哪个状态下会产生SRQ、相对的一个并行查询使能寄存器（PPE）与STB按位相"与"就得到比特6。结果相"或"、然后作为响应发送到（或转化为响应）控制器的并行查询。结果还可以通过"\*IST"指令查询而不必并行查询。仪器最初必须使用quick - BASIC指令"IBPPC（）"，设定成并行查询。这些指令分配一条数据线到仪器并且决定是否转化响应。并行查询本身的实现使用"IBRPP（）"。

如果有多台仪器连接到IEC总线，并行-查询方法主要用于在一个SRQ之后迅速找出是哪个仪器发送了服务请求。为此，SRE和PPE必须设置为同样的值。并行查询的详例在第7章、程序范例中。

## 使用指令查询

每个状态寄存器的各个字段都能够通过查询读取。每一个指令都在第3.8.3.节的寄存器详细说明中做了解释，他们的返回值会是一个表示寄存器查询的位模式的编号。这些编号的评估要受到控制器程序的影响。

查询通常在SRQ之后使用，以便获得关于SRQ的原因的更多详细信息。

## 错误-队列查询

仪器的每个错误状态都会引起一个输入到错误队列。错误队列的输入，是详尽的纯文本形式的错误信息，它可以通过手动操作在ERROR菜单中检查，或者通过IEC总线使用"SYSTem:ERRor?"指令查询。每次使用"SYSTem:ERRor?"都会从错误队列调出一个输入。如果再没有错误信息了,仪器以0响应, "没有错误"。

应该在控制器程序中的每个SRQ之后查询错误队列，以得到比状态寄存器更确切的出错原因。特别在控制器程序的测试阶段，应该定期查询错误队列，因为从控制器发送到仪器的造成故障的指令在那里也有记录。

## 状态报告系统的恢复缺省值

表格5 - 12包括了引起状态报告系统复位的各种指令和事件。除\* RST和SYSTem:PRESet之外，没有任何指令，能影响到正在起作用的仪器设置。尤其是、DCL不会改变仪器设置。

表格5 - 12复位后的仪器功能

事件  效果	打开电源		DCL,SDC ( 设备清零 ,被选 设备清零 )	*RST 或 SYSTem:PRESet	STATus:PRESet	*CLS
	Power-On-Status- 清零					
	0	1				
STB,ESR清零		是				是
SRE,ESE 清零		是				
PPE 清零		是				
寄存器的EVENTt 字段清零		是				是
所 有 OPERation 和 QUESTionable 寄 存 器 的 Enable 字段清零 , 其他寄 存器全部填满 1		是			是	
PTRansition字段填满1 NTRansition 清零		是			是	
错误队列清零	是	是				是
输出缓存清零	是	是	是	1 )	1 )	1 )
命令处理和输入缓存清零	是	是	是			

1) 每个指令在指令行中都占首位，那就是说,他们直接跟随一个<PROGRAM MESSAGE TERMINATOR>清零输出缓冲器。