Contains pages for

GPIB Operating Manual

for

RF and MICROWAVE TEST SETS

6200B SERIES

Includes GPIB operation for 6210 Reflection Analyzer

Part number 46882-264J
Issue 5

Creation date 30-Jul-96

Please open and fit to the supplied Ring Binder

RF and MICROWAVE TEST SETS

6200B SERIES

Includes GPIB operation for 6210 Reflection Analyzer

Printed in the UK

Manual Part No. 46882-264J
Issue 5

30-Jul-96

Contents

PREFACE

About this manual

This manual consists of three chapters and an appendix. Chapter 1 provides general information about the commands and conventions used when operating the instrument remotely using the General Purpose Interface Bus (GPIB). Chapter 2 provides an introduction to GPIB programming of the MTS, and includes a worked example of a typical measurement. Chapter 3 gives specific information about the GPIB command set, for both the MTS and the 6210 Reflection Analyzer. Appendix A describes the GPIB status reporting structure.

Although familiarity with local operation is assumed, this manual is essentially self-contained and gives all the information necessary to operate the MTS and Reflection Analyzer using the GPIB. Local operation is described in the MTS Operating Manual.

WARNINGS

Observe the warnings given in the 'Precautions' section of the Operating Manual for the 6200B Series RF and Microwave Test Sets.

Chapter 1

GPIB CONVENTIONS

Chapter 1
GPIB CONVENTIONS

CONTENTS

INTRODUCTION

The instruments comprising the 6200B series of Microwave Test Sets are equipped for remote operation via the GPIB interface. The interface, which conforms to IEEE 488.1 and IEEE 488.2, provides the following:

- Instrument control with full talk and listen capability. The instrument can be under the control of an external controller or a second MTS. Commands are also provided for controlling the 6210 Reflection Analyzer.
- Control of a GPIB plotter which uses the HPGL language.
- Control of a second MTS, for example when performing mixer measurements, where the instruments are set to sweep with a fixed frequency offset between them.

Before operating the instrument under GPIB control, the reader should already be familiar with making measurements with the MTS and with the general operation of the GPIB. Local operation is covered in the MTS Operating Manual; this chapter provides a general description of GPIB commands and conventions.

SCPI 1990.0 COMPATIBILITY

Commands are divided into a number of subsystems. Each subsystem contains groups of related commands.

The form of the subsystems owes much to the ideas contained in the Standard Commands for Programmable Instruments standard, SCPI 1990.0 and the STATus subsystem and its associated Status Reporting Structures conform to that standard.

Other subsystems are instrument-specific, although some of the features of SCPI, such as the conventions for naming and organising commands, have been adopted.

COMPOUND HEADERS

Compound headers allow a complex set of commands to be built up from a smaller set of basic elements in a "tree" structure. The elements of a compound header are separated by a colon ":". Each subsystem in this instrument is organised as a separate tree structure.

The use of compound headers brings a number of advantages. Commands are less cryptic compared with a traditional "flat" instrument command set, and compound header elements may appear more than once.

Example:

```
SOURce
    :VIOutput
        :CURRent
            :CONStant
            :STARt
            :STOP
        :VOLTage
            :CONStant
            :STARt
            :STOP
```

Here the compound header elements "CONStant", "STARt" and "STOP" appear for both the current and voltage functions.

Although it is possible to use the full compound header starting from the tree root every time (e.g. SOURCE:VIOUTPUT:VOLTAGE:START 2; SOURCE:VIOUTPUT:VOLTAGE:STOP 8), sequences of <COMMAND MESSAGE UNITS> and <QUERY MESSAGE UNITS> can often be shortened by taking advantage of the special rules which apply to compound headers.

Having "descended" the tree, (for example to create the <PROGRAM MESSAGE UNIT> SOURCE:VIOUTPUT:VOLTAGE:START 2), any other elements at that level may be included in the <PROGRAM MESSAGE> without repeating the entire path through the tree.

Example:

```
SOURCE:VIOUTPUT:VOLTAGE:START 2;STOP 8
```

is equivalent to the two <PROGRAM MESSAGES> SOURCE:VIOUTPUT:VOLTAGE:START 2 followed by SOURCE:VIOUTPUT:VOLTAGE:STOP 8.

Note the use of the <PROGRAM MESSAGE UNIT SEPARATOR> character ";" between <PROGRAM MESSAGE UNITS>.

Here is another example, this time using commands from the SYSTem subsystem (page 3-206).

```
SYSTEM:ADDRESS:PLOTTER 8;SOURCE 19
```

is equivalent to the two <PROGRAM MESSAGES>:

```
SYSTEM:ADDRESS:PLOTTER 8 and SYSTEM:ADDRESS:SOURCE 19
```

To return to the top of the tree so that another "branch" may be descended, a colon is used.

Example:

```
SYSTEM:ADDRESS:PLOTTER 8;SOURCE 19;:SYSTEM:DISPLAY:LCD
```

DEFAULT BRANCHES

Some elements within the compound header tree structure are enclosed within square brackets, "[" and "]". These elements may be omitted, if desired, to reduce the length of the compound header.

Example from the SYSTem subsystem (page 3-212):

```
SYSTEM:DISPLAY:STITLE ON
```

is equivalent to

```
SYSTEM:DISPLAY:STITLE:STATE ON
```

A potential error that can occur when taking advantage of default branches is to mistake the level reached within the tree, and hence the choice of commands available at that level.

Example:

```
SYSTEM:DISPLAY:STITLE ON; LCD HALF
```

will not work because LCD is not at the same level in the tree as SYSTEM:DISPLAY:STITLE: STATE, as may be verified by examining the following fragment of the SYSTem subsystem:

```
SYSTem
      :DISPlay
             :STITle
                    [:STATe]
                    STRing
             :LCD
```

ABBREVIATIONS

In general, compound header elements have a long and a short form. Following the convention adopted in SCPI, the short form of the element is printed in upper case characters, with any remaining characters in lower case. (This is merely a convenient way of showing both forms. The instrument does not distinguish between upper and lower case characters within a header).

Example:

FREQuency

The short form is "FREQ" and the long form "FREQUENCY". Other abbreviations such as "FRE" or "FREQUEN" are not allowed.

PROGRAM DATA

The following program data functional elements are accepted by the instrument:

<CPD> (also known as <CHARACTER PROGRAM DATA>)

<NRf> (also known as <DECIMAL NUMERIC PROGRAM DATA>)

<STRING PROGRAM DATA>

<ARBITRARY BLOCK PROGRAM DATA>

<BOOLEAN PROGRAM DATA>

All these functional elements, with the exception of <BOOLEAN PROGRAM DATA>, are defined in IEEE 488.2-1987.

The following informal definitions are provided as a guide:

<CPD>

Character program data is used to set a parameter to one of a number of states that are best described by short alphanumeric strings.

In this manual <CPD> strings are shown using the same conventions for abbreviation as program headers; the part of the string printed in upper case characters being the short form.

Example:

SOURce, COUNter and FREQuency are the possible sources of the frequency used to look up a power sensor calibration factor value (see page 3-47). They may be abbreviated to SOUR, COUN and FREQ respectively.

<NRf>

Flexible numeric representation (also known as <DECIMAL NUMERIC PROGRAM DATA>) covers integer and floating point representations.

Examples:

-466	Integer value.
4.91	Explicitly placed decimal point.
59.5E+2	Mantissa and Exponent representation

The format is known as "flexible" because any of the three representations may be used for any type of numeric parameter.

Example:

Suppose a parameter requires an integer value in the range 1 to 100, and the user wishes to set its value to 42, the following values will be accepted by the instrument.

42	Integer
42.0	Floating point.
4.2E1, 4200E-2	Floating point - Mantissa/exponent.
41.5	Rounded up to 42
42.4	Rounded down to 42

<STRING PROGRAM DATA>

String program data consists of a number of ASCII characters enclosed in quotes. Either a pair of single ('ASCII 39') or double ("ASCII 34") quotes may be used. If the quote character chosen to mark the beginning and end of the string also appears within it, it must be doubled.

Example:

'This string contains the word "Hello"'

will be interpreted as the string:

This string contains the word 'Hello'

When receiving string data, the MTS GPIB system interprets character codes as follows:

32 - 126	Standard ASCII characters
127	Copyright symbol ©
128	Mu symbol µ
129	Degree symbol °
130	Ohms symbol Ω

Any command received with string data containing any other code will result in an error message being displayed.

<ARBITRARY BLOCK PROGRAM DATA>

This format is used for the transmission of large quantities of 8-bit binary data.

Since it is not intended that the user should ever need to compile data of this type for transmission to the instrument, details of the format are not given here.

Note that data received from the instrument as <INDEFINITE LENGTH ARBITRARY BLOCK RESPONSE DATA> is already in a form suitable for transmission back to the instrument as <ARBITRARY BLOCK PROGRAM DATA>.

Also note that since only the indefinite length form is used, the data must be terminated by line feed with EOI asserted. This means that a command requiring <ARBITRARY BLOCK PROGRAM DATA> must be the last <PROGRAM MESSAGE UNIT> of the <PROGRAM MESSAGE>.

<BOOLEAN PROGRAM DATA>

This is not defined in IEEE 488.2-1987, but is a useful addition for programming parameters that have an "ON/OFF" function.

A parameter accepting <BOOLEAN PROGRAM DATA> will take the values OFF | 0 | ON | 1, where "0" has the same meaning as "OFF" and "1" has the same meaning as "ON".

RESPONSE DATA

The following response data functional elements are generated by the instrument:

<CRD> (also known as <CHARACTER RESPONSE DATA>)

<NR1>

<NR2>

<NR3>

<STRING RESPONSE DATA>

<INDEFINITE LENGTH ARBITRARY BLOCK RESPONSE DATA>

<ARBITRARY ASCII RESPONSE DATA>

<BOOLEAN RESPONSE DATA>

All these functional elements, with the exception of <BOOLEAN RESPONSE DATA>, are defined in IEEE 488.2-1987.

The following informal definitions are provided as a guide:

<CRD>

This type of response is returned when reading the value of a parameter which can take a number of discrete states. States are represented by short alphanumeric strings.

Example:

> SOUR, COUN and FREQ are the possible <CRD> responses if the parameter which determines the source of the cal factor look-up frequency is queried (see page 3-47).

Note that when setting the parameter, the long form (i.e. SOURCE, COUNTER and FREQUENCY) may be used, but when the parameter is queried, the short form is always returned.

<NR1>

This type of numeric response is used when returning the value of integer parameters, such as averaging number or number of measurement points.

Examples:

```
 15
 +3
 -57
```

<NR2>

This type of numeric response includes an explicitly placed decimal point, but no exponent.
Examples:

```
 17.91
-18.27
+18.83
```

<NR3>

This type of numeric response includes an explicitly placed decimal point and an exponent.

Examples:

```
1.756E+2
182.8E-3
```

<STRING RESPONSE DATA>

This takes a similar form to <STRING PROGRAM DATA> except that the delimiting character is always a double quote, ("ASCII 34").

<INDEFINITE LENGTH ARBITRARY BLOCK RESPONSE DATA>

This form of response is used when reading blocks of 8-bit binary data from the instrument. Examples include settings stores and trace memories.

The format comprises a '#' character followed by a '0' followed by the data, followed by a newline character (ASCII 10). EOI is asserted with the terminating newline character.

Because EOI is always used as a terminator, a <QUERY MESSAGE UNIT> which generates data in this form must be the last <QUERY MESSAGE UNIT> in the <PROGRAM MESSAGE>.

<ARBITRARY ASCII RESPONSE DATA>

This takes the form of an ASCII string terminated by newline (ASCII 10) with EOI asserted.

Notes on interpreting data returned in this format will be found in the descriptions for the few commands that use it.

Because EOI is always used as a terminator, a <QUERY MESSAGE UNIT> which generates data in this form must be the last <QUERY MESSAGE UNIT> in the <PROGRAM MESSAGE>.

<BOOLEAN RESPONSE DATA>

This is not defined in IEEE 488.2-1987, but is a useful addition for querying parameters that have an "ON/OFF" function.

The response is either "0" or "1", where "0" means "OFF" and "1" means "ON".

GPIB TERMINATORS

A **<PROGRAM MESSAGE TERMINATOR>** (as defined in IEEE 488.2-1987) can be a newline character (ASCII 10), a newline character with the ^END message asserted at the same time, or an ^END message asserted with the final character of the <PROGRAM MESSAGE>. The terminator may be preceded by any number of "white space" characters - i.e. any single ASII-encoded byte in the range 0 to 9 and 11 to 32 decimal.

A **<RESPONSE MESSAGE TERMINATOR>** (as defined in IEEE 488.2-1987) is a newline character with the ^END message asserted at the same time.

Many GPIB controllers terminate program messages with a newline character and, by default, accept newline as the response message terminator. When transferring binary data - which may contain embedded newline characters - it is necessary to ensure that the controller uses only ^END messages. Usually this requires the controller's GPIB interface to be set up to generate and detect ^END. Refer to the documentation supplied with the controller.

COMMAND LAYOUT

Each command is set out as follows:

1. Path from the subsystem root.

Example:

```
:SYSTem
        :ADDRess
                :SOURce
```

2. Parameters

The first line lists each parameter, stating its <PROGRAM DATA> functional element (as defined in IEEE 488.2-1987).

Subsequent lines explain the meaning of each parameter. For numeric parameters, such as those holding frequency or power values, the units are stated (e.g. W, Hz, dB, etc). For <CPD> (character program data) parameters, the available choices are listed, separated by the "OR" symbol, "|".

Optional parameters are shown enclosed in square brackets, [...].

Angle brackets <...> indicate that the enclosed parameter is described in more detail later in the text.

Example:

<CPD> [,<NRf>]

<measurement type> [,frequency (Hz)]

where <measurement type> is POWer | VOLTage | FREQuency

The first line states that the command takes one or two parameters, the second being optional. The first parameter is character program data and the second is a numeric value.

The semantic interpretation is given on the second line. The first parameter is <measurement type>, which can take the values POWer, VOLtage or FREQuency. The second parameter is a frequency, entered in units of Hz.

3. Description

Describes the purpose of the command. Where applicable, a cross-reference to the corresponding local (i.e. front panel) operation in Chapter 3 of the MTS Operating Manual is provided. This cross reference is given in the form of a menu title, preceded by the title of the hard key which is used to access that menu; the page number can be easily found by referring to the Chapter 3 contents list.

4. Example.

An example of the use of the command is provided. Examples always use the short form of the command.

5. Query Response

Query responses follow the same format as parameter definitions. The first line shows the response in terms of its IEEE 488.2 functional elements, and below it is given the semantics of the response.

Example:

<NR3>

frequency (Hz)

Chapter 2

GETTING STARTED

Chapter 2
GETTING STARTED

CONTENTS

LIST OF FIGURES

INTRODUCTION

This chapter provides an introduction to GPIB programming of the MTS, including a worked example of a typical measurement. GPIB programmers may find it a worthwhile exercise to work through the example measurements given in the Getting Started Manual in order to gain familiarity with the front panel operation of the instrument.

The GPIB Command Set

The first point to notice when controlling the MTS via the GPIB is that there is not a straightforward mapping between manual front panel operations and their GPIB equivalents. The man-machine interface (MMI) restricts the user's view of the instrument's functions by taking account of the active channel mode and active measurement, and making available appropriate function sub-sets as demanded by the context.

There is no corresponding mechanism for providing "context sensitive" GPIB commands. Instead, the commands are organised into "subsystems" of related functions, and the number of unique mnemonics required kept to a minimum by the use of compound headers. (See Chapter 1, 'GPIB Conventions' for an introduction to compound headers and other command set conventions.)

To help GPIB programmers know where to look for a particular command, there follows a brief overview of the command subsystems.

Common Commands

A selection of IEEE 488.2 common commands is provided. These all start with a "*" character. The most important of these is *RST, which places the instrument in a defined state. It is good practice to send *RST at the start of any GPIB program.

CHANnel & MEASurement

These contain commands relating to channels and measurements. The most important are the commands for selecting the number of displayed channels, selecting the number of measurements displayed within each channel and determining which channel and measurement are active.

INPut

The input subsystem contains commands associated with the signal inputs (with the exception of the counter). Detection mode, detector correction, detector zero and power sensor cal are examples of the available functions.

SOURce

Contains all commands concerned with the control of the synthesized source.

SCALar, ROUT, FLOCation and REFLect

Each channel mode has an associated subsystem providing commands for setting up and acquiring measurements. It is important to ensure that the active channel mode (CHANnel:MODE\?) is set appropriately before issuing commands from these subsystems.

MARKer

Provides marker functions for swept measurements.

HARDcopy

Controls all available hard copy functions.

STATus

Accesses the SCPI-compatible status reporting structure.

SYSTem

A miscellany of functions, most of which appear under the [UTILITY] key. Note that the GPIB trigger function is accessed via this subsystem.

Preparing the MTS for GPIB Operation

The MTS GPIB system can operate in two modes, controller or talker/listener. In controller mode, the MTS takes control of the bus in order to drive a plotter or an external source. Talker/listener mode must be selected before the instrument can accept commands from another controller.

[UTILITY]
[GPIB]

This menu allows the GPIB mode and addresses to be set. Ensure that Talker/Listener mode is selected.

[Instrument Address]
[Ext Source Address]
[Plotter Address]

GPIB addresses may be examined and, if necessary, changed using these soft keys.

[Controller Mode]
[Talk/Listen Mode]

These soft keys set the GPIB mode of the MTS.

Example: Low Pass Filter Characterisation

In this example, MTS GPIB commands are stated without making any assumptions about the controller and programming language to be used. These commands, of course, will need to be incorporated into the program language statements of the target controller. Here are some examples of how this would be done in practice, using the reset command, *RST. The instrument address is assumed to be 8.

***RST** — Command as printed in the example.

PRINT @8:"*RST" — Controller using TBASIC^{R} programming language (TransEra Corporation).

OUTPUT 708;"*RST" — Controller using HTBASIC^{TM} programming language (TransEra Corporation).

The example will measure the broadband insertion and return loss responses of a 13 GHz low pass filter and simultaneously display the results on channel 1. Channel 2 will be used to display the narrowband insertion loss of the filter in order to determine the passband ripple. The set-up for this example is shown in Fig. 2-1. As far as GPIB control is concerned, a similar sequence of logical steps can be followed as described in the Getting Started Manual for the MTS.

It may sometimes be necessary to send a DEVICE CLEAR command, if the GPIB system fails to respond to *RST or appears to lock up. Examples of this command are as follows:

DEVICE CLEAR	Command as printed in the example.
WRITE GPIB CMD_SDC(8)	Controller using TBASIC$^{R.}$
CLEAR 708	Controller using HTBASIC^{TM}

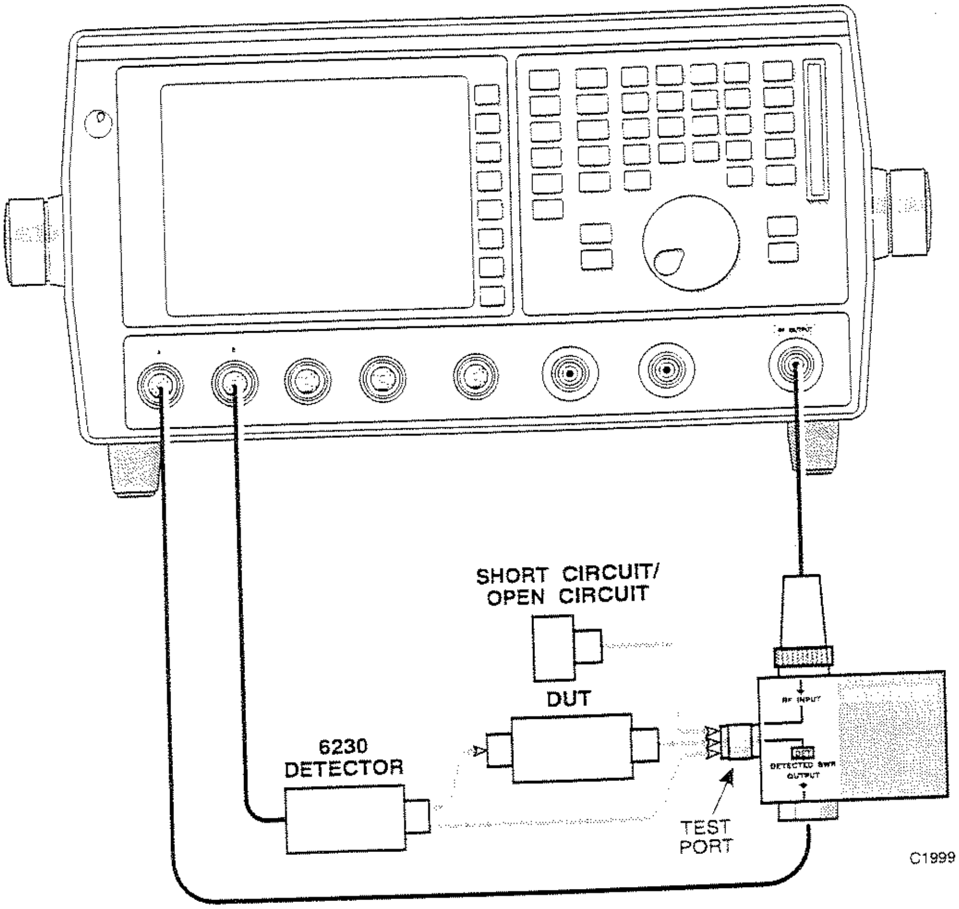


Fig. 2-1 Simultaneous Measurement of Insertion and Return Loss of a Low Pass Filter

Step 1 Preset the Instrument to a Known State

DEVICE CLEAR

*RST Preset the instrument:

Step 2 - Define the Display Configuration

Simultaneous broadband insertion and return loss measurements will be displayed on channel 1. Channel 2 will display the passband ripple (insertion loss) of the filter.

:CHANnel:NCHannels 2

Use the NCHannels (number of channels) function in the CHANnel sub-system to set up a dual channel display.

Remember that IEEE 488.2 requires a single space character between the command header and its parameter(s).

:CHANnel:ACTive 1

Make channel 1 the active channel.

Note that these two commands may be combined easily using the properties of compound headers described in Chapter 1. Thus:

```
:CHAN:NCH 2;ACT 1
```

From now on, only the abbreviated forms will be given, and contractions that take advantage of the compound header structure will be used whenever possible.

:MEAS:NME 2

Set the number of measurements on the active channel (channel 1) to 2.

Step 3 - Define the Source Conditions

:CHAN:COUP OFF; ACT 2

Disable channel coupling, (so that a different source set-up may be defined for each channel), and make channel 2 the active channel.

:SOUR:FREQ:STOP 13E9
:SOUR:RF ON

Set the source stop frequency to 13 GHz, and switch on the RF.

Step 4 - Define the Measurements

:MEAS:ACT 1

Make measurement 1 on channel 1 the active measurement.

:SCAL:MEAS:POW B

Define measurement 1 to display power from input B.

:CHAN:ACT 2;:MEAS:ACT 1

Make measurement 1 on channel 2 the active measurement.

:SCAL:MEAS:POW B

...and set up the same measurement from input B.

Step 5 - Calibrate the Measurement System

Calibrating the Insertion Loss Path

Although detector autozero is enabled automatically as part of the *RST conditions, it is necessary to perform a "full" zero when the detectors are changed. In a GPIB system it is good practice to perform a zero before a path calibration.

:INP:ZERO:DET; *OPC?

Initiate a detector zero and wait for the operation to finish. (*OPC? returns the value 1 when the zero is complete)

:CHAN:ACT 1;:MEAS:ACT 1

Make channel 1, measurement 1 active.

The insertion loss path is about to be calibrated. At this point in the program, the GPIB controller should issue a prompt to the user to make the through connection, then wait for confirmation from the user that the connection has been made.

```
:SCAL:PCAL:SAVE 1,THR
```

Perform the "THRough" calibration. saving the calibration data in path cal store 1.

Calibrating the Return Loss Path

```
:MEAS:ACT 2
```

Make measurement 2 on channel 1 the active measurement.

The program should prompt the user to connect a short circuit to the test port of the autotester, then wait for confirmation that the connection has been made.

```
:SCAL:PCAL:SAVE 2,SHOR
```

Save the short calibration data in path cal store 2.

Again the program should prompt the user - this time to replace the short circuit with an open circuit.

```
:SCAL:PCAL:SAVE 2,OPEN
```

The open cal data is averaged with the previously stored short data and saved in path cal store 2.

```
:CHAN:ACT 2;:MEAS:ACT 1
:SCAL:PCAL:SEL 1;STAT ON
```

Apply the through path cal to the narrow band insertion loss measurement displayed on channel 2.

Step 6 - Choose the Format

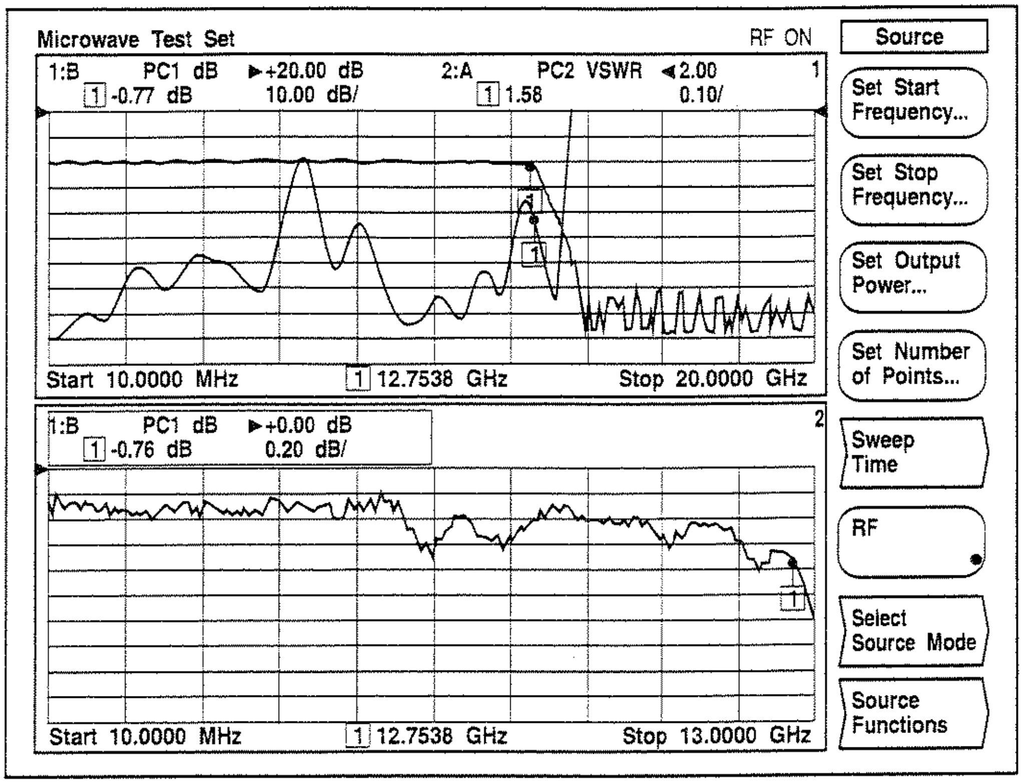
```
:CHAN:ACT 1;:MEAS:ACT 2
:SCAL:FORM VSWR
```

Display the return loss measurement in VSWR format.

Step 7 - Select Appropriate Scaling

```
:CHAN:ACT 2;:MEAS:ACT 1
:SCAL:SCAL:RLEV 0;DIV 0.2
```

Selects more appropriate values of reference level and scale division for the narrowband insertion loss measurement displayed on channel 2. (See Fig. 2-2)



Microwave Test Set RF ON

1:B PC1 dB ▶+20.00 dB 2:A PC2 VSWR ◀2.00 1
1 -0.77 dB 10.00 dB/ 1 1.58 0.10/

Start 10.0000 MHz 1 12.7538 GHz Stop 20.0000 GHz

1:B PC1 dB ▶+0.00 dB 2
1 -0.76 dB 0.20 dB/

Start 10.0000 MHz 1 12.7538 GHz Stop 13.0000 GHz

Fig. 2-2 Broadband Insertion and Return Loss and Narrowband Insertion Loss

Step 8 - Use Markers to Get Detailed Information About the Measurement

:CHAN:ACT 1;:MEAS:ACT 1	Ensure that the broadband insertion loss measurement is the active measurement.
:MARK:MAX;DELT ON	Place the active marker at the trace maximum, then switch on the delta marker.
:MARK:SEAR:DIR RIGH;RES?	This command returns a Boolean value indicating whether the search has succeeded or failed.
:MARK:ACT:POS?	Assuming the search succeeded, this command returns the frequency at which the filter response has fallen by 3 dB.

Step 9 - Create a Permanent Record of the Measurement Results

Newcomers to the MTS GPIB facilities will probably find it easier to create hard copy using a printer connected to the parallel interface, rather than a GPIB plotter. (The operations required to pass control to the MTS temporarily so that it can drive a plotter are described under the :HARDcopy:PLOT command.)

:HARD:CONF:MTAB OFF	Disable printing of the marker table (which is of limited value when only the active marker and delta marker are displayed).
:HARD:PRIN; *OPC?	Start the print, and wait for operation complete. *OPC? returns the value 1 when the last byte has been output to the printer.

Step 10 - Save the Instrument Settings for Future Use

Having just designed a program to set up the instrument from scratch, there seems little point in saving the set-up to a settings store. However, there are some aspects of the GPIB control of settings stores which are worth mentioning. It also provides an opportunity to explore the SYSTem subsystem, which has not yet been used.

:SYST:SETT:MAX?	The instrument's internal memory holds 10 settings stores. If a memory card is plugged in, there may be more stores available. This command returns the highest store number present.
:SYST:SETT:SAVE 1	Save settings to settings store 1.
:SYST:SETT:STOR? 1	Transfer the contents of settings store 1 to the controller as binary data.

Note that before attempting to read the response generated by this command, the controller must be set up to terminate GPIB input on receipt of an ^END message (not just a newline character without ^END).

If this is not done, and the binary data happens to contain an embedded newline character, data transfer will be terminated prematurely.

The final command in the example transfers the contents of a settings store to the GPIB controller as <INDEFINITE LENGTH ARBITRARY BLOCK RESPONSE DATA>. There is little to be gained by trying to modify the contents of the store, but the data may be saved for later transmission back to the instrument.

Chapter 3

GPIB COMMANDS

Chapter 3
GPIB COMMANDS

CONTENTS

COMMON COMMANDS

```
*CLS
*ESE\?
*ESR?
*IDN?
*OPT?
*OPC\?
*PCB
*RST
*SRE\?
*STB?
*TRG
*TST?
*WAI
```

*CLS

Description: Clear Status clears the Event Register, the Error Queue, the Operation Event Register and the Questionable Event Register.

*ESE

Parameters: <NRf>
mask (0 - 255)

Description: The Standard Event Status Enable command sets the Standard Event Enable register.

Example: `*ESE 2`
Set the Standard Event Status Enable register to 2 (00000010 in binary). This will allow RQC (Request Control) messages generated by the instrument to be reported in the Event Summary Bit. (RQC is issued by the instrument when it needs to take control of the bus to drive the plotter).

*ESE?

Response: <NR1>
mask (0 - 255)

Description: Read the Standard Event Status Enable register.

*ESR?

Response: <NR1>
register contents (0 - 255)

Description: Read the value of the Standard Event Status register.

*IDN?

Response: <ARBITRARY ASCII RESPONSE DATA>
<Manufacturer>, <Model>, <Serial No.>, <Software Part No. and Issue No.>
Where <Manufacturer> *is* Marconi Instruments
<Model> *is instrument model number in the form* 620xB *where*
x = 0, 1, 2, 3 or 4.
<Serial number> *is in the form* nnnnn/nnn *where* n *is an* ASCII
digit *in the range* 0 to 9.
<Software Part No. and Issue No.> *is in the form* nnnnn/nnn/nn.nn *where*
n *is an* ASCII digit *in the range* 0 to 9.

The Identification Query command allows information about the instrument to be read.

*OPT?

Response: <NR1>
<options>
where <options> *is a* 16 bit value *with each set bit representing the presence of an option:*

Bit Number	Option Present If Bit Set
0	2 GHz Source
1	20 GHz Source
2	26 GHz Source
3	46 GHz Source
4	8 GHz Source
5	70 dB Step Attenuator
6	8.4 GHz Source
7	Reflection Analyzer
8	Rear Panel Output
9	90 dB Step Attenuator
10	Bias Tee (6210 option)
11	110 dB Step Attenuator
12	Floppy Disk
13	8.5 GHz Source
14	SPARE
15	Always zero

Description: Read hardware options present.

*OPC

Description: The Operation Complete command sets the Operation Complete bit in the Standard Event Status Register when execution of the preceding operation is complete.

*OPC should be the final <PROGRAM MESSAGE UNIT> of the <PROGRAM MESSAGE>.

Example: `:HARD:PRIN; *OPC`

Initiate a print of the currently displayed measurements. The Operation Complete bit will be set in the Standard Event Status Register when the instrument has finished printing.

*OPC?

Response: <NR1>

1

Description: The Operation Complete Query returns a '1' when the preceding operation has completed.

*OPC? should be the final <QUERY MESSAGE UNIT> of the <PROGRAM MESSAGE>.

Example: `:HARD:BSTAT; *OPC?`

Initiate a print of the instrument's build state. When printing has finished, the value '1' will be placed in the output queue.

*PCB

Parameters: <NRf>

controller address

Description: Some commands, for example :HARDcopy:PLOT, require the instrument to act temporarily as the GPIB controller. *PCB is used to tell the instrument the address of the System Controller so that control can be passed back to it on completion of the command.

Example: `*PCB 21`

Inform the instrument that the system controller's GPIB address is 21.

*RST

Description: Reset the instrument. This command has the same effect as :SYSTem:PRESet, and places the instrument in its default state, as defined in Appendix A.

If the GPIB system fails to respond to *RST, or appears to lock up, it may be cleared by sending a DEVICE CLEAR command. It is good practice to precede *RST with DEVICE CLEAR.

See [PRESET] key for more details.

*SRE

Parameters: <NRf>
mask (0 - 255)

Description: Set the Service Request Enable register.

Example:
```
*SRE 32
```
Set the Service Request Enable register to 32 (0010 0000 in binary) to enable service requests when the Standard Event Status Register Summary Bit is set.

*SRE?

Response: <NR1>
mask (0 - 255)

Description: Read the Service Request Enable register.

*STB?

Response: <NR1>
status byte (0 - 255)

Description: Read the Status Byte.
See pageApp. A-1 for details.

*TRG

Description: Trigger command. This command is equivalent to Group Execute Trigger.

Example:
```
:SYST:TRIG MEAS; *TRG
```
Place instrument into Trigger Measurement mode and trigger the measurement.

*TST?

Response: <NR1>
0

Description: Self Test Query. Returns a '0' when the GPIB interface and processor are operating.

*WAI

Description: The Wait to Continue command inhibits execution of an overlapped command until the execution of the preceding operation has been completed.

Example:

```
:HARD:PRINT; *WAI
```

Initiate a print of the current measurement and inhibit further commands until the print is finished.

DIGITAL SWEEP INTERFACE COMMANDS

[OP]DSEN
[OP]DSNP
[OP]DSSP
OPDSBP

These commands are provided for compatibility with the 6300 series of Microwave Sweep Generators.

The Digital Sweep Interface is a means by which the instrument, when operating as a slave source, can be swept over a range of discrete values, with each step change controlled by a digital signal. The GPIB Group Execute Trigger <GET> signal is used to initiate the step to the next discrete value.

A note on compatibility with existing programs:

The digital sweep scheme was originally developed before the adoption of IEEE488.2-1987. To conform to the requirements of the IEEE488.2-1987 parser used in the 6200 family of instruments, a <PROGRAM HEADER SEPARATOR> (space character) is required between the <PROGRAM HEADER> and the <PROGRAM DATA>.

An existing program containing the command "DSNP401" should therefore be modified so that the command reads "DSNP 401".

It should also be noted that the prefix 'OP', rather than the suffix '?', is used for the query form of the commands.

DSEN

Parameters: <BOOLEAN PROGRAM DATA>
state

Description: Enable/disable digital sweep.

Enabling digital sweep is equivalent to placing the instrument in Slave Source mode by sending the command :SYSTem:MODE SLAVe.

When digital sweep is disabled, the instrument is placed in Source-Only mode, equivalent to sending the command :SYSTem:MODE SOURce.

Example:
```
DSEN ON
```
Enable digital sweep mode.

OPDSEN

Response: <BOOLEAN RESPONSE DATA>
State

Description: Determine whether digital sweep mode is enabled.

DSNP

Parameters: <NRf>
number of points

Description: Set the number of points for the digital sweep.

It is recommended that for frequency sweeps, the maximum step between points should not exceed 60 MHz. This should be taken into account when setting the number of points.

Example:
```
DSNP 401
```
Set up a 401 point sweep

OPDSNP

Response: <NR1>
number of points

Description: Read the number of points.

DSSP

Parameters: <NRf>
point number

Description: Set the source output to a point number in the range 0 to number of points - 1.

Example: `DSSP 0`
Set the source output to the first point, point zero, ready for the next sweep.

OPDSSP

Response: <NR1>
point number

Description: Read the current point number.

OPDSBP

Response: <NR1>,<NR1>,...,<NR1>,<NR1>

point number, point number,...,point number, point number

The command returns 10 values separated by commas, corresponding to the point numbers for a maximum of 10 bandswitch points. Unused fields in the response contain the value -1.

Description: Read the point numbers corresponding to the source bandswitch positions. A longer settling time (20 ms) is required at the bandswitch points compared with other points in the sweep (360 µs).

A bandswitch point is defined as the first point in the new band.

Example: `OPDSBP`
Read the bandswitch positions. The example response shows that bandswitches occur at points 53 and 234. Longer dwell times will be required at these points.

Example Response: 53,234,-1,-1,-1,-1,-1,-1,-1,-1

USING THE DIGITAL SWEEP COMMANDS

1. Enable digital sweep

   Example:
   ```
   DSEN ON
   ```

2. Read bandswitch positions

   Example:
   ```
   OPDSBP
   ```

   Example Response:
   ```
   53,234,-1,-1,-1,-1,-1,-1,-1,-1
   ```

3. Start from point 0.

   Example:
   ```
   DSSP 0
   ```

4. Use <GET> signals to step the source to each point. The settling time at points 53 and 234 should be 20 ms to allow for bandswitching. At other points a settling time of 360 µs should be sufficient.

CHANnel SUBSYSTEM

```
CHANnel
        ACTive\?
        COUPling\?
        NCHannels\?
        MODE\?
```

:CHANnel
:ACTive

Parameters: <NRf>
active channel number

Description: Set the active channel.
Seee 'Display Group keys' for more details.

Example: `:CHAN:ACT 2`
Make channel 2 active.

:CHANnel
:ACTive?

Parameters: [<NRf>]
[active channel number]

Response:
<NR1>
active channel number

Example: `:CHAN:ACT?`
Determine which channel is active.

:CHANnel
:COUPling

Parameters: <BOOLEAN PROGRAM DATA>
state

Description: Set channel coupling on or off.
See 'Channel Coupling' for more details.

Example: `:CHAN:COUP OFF`
Disable channel coupling.

:CHANnel
:COUPling?

Parameters: [<BOOLEAN PROGRAM DATA>]
[state]

Response: <BOOLEAN RESPONSE DATA>
state

Example: `:CHAN:COUP?`
Determine whether channels are coupled.

:CHANnel
:NCHannels

Parameters: <NRf>
number of channels

Description: Set the number of channels displayed.
See 'Channels and Measurements' and 'Display Group Keys' for more details.

Example: `:CHAN:NCH 2`
Display two channels.

:CHANnel
:NCHannels?

Parameters: [<NRf>]
[number of channels]

Response: <NR1>
number of channels

Example: `:CHAN:NCH?`
Determine how many channels are displayed.

:CHANnel
:MODE

Parameters: <CPD>
<channel mode>
where <channel mode> *is* SCALar | READout | FLOCation | REFLect

Description: Set the mode of the currently active channel.
See 'Channels and Measurements' and Channel Mode Menu for more details.

Example: `:CHAN:MODE FLOC`
Make the active channel a fault location channel.

:CHANnel
:MODE?

Parameters: [<CPD>]
[<channel mode>]
where <channel mode> *is* SCALar | READout | FLOCation | REFLect

Response: <CRD>
<channel mode>

Example: `:CHAN:MODE?`
Determine the mode of the active channel.

FLOCation SUBSYSTEM

```
FLOCation
    ATTenuation\?
    AVERaging
        NUMBer\?
        RESTart
        [STATe]\?
    CALibration
        RESTore
        SAVE
        STORe\?
    DATA
        [ASCii]?
        BINary?
    ENHanced\?
    FORMat\?
    FREQuency
        CENTer\?
        [ENTRy]\?
        STARt\?
        STOP\?
    MCORrection\?
    MEDium\?
    POINts\?
    RANGe
        [ENTRy]\?
        VALue\?
    RVELocity\?
    SCALing
        AUTO\?
        DIVision\?
        POSition\?
        RLEVel\?
    UNITs\?
    WGCutoff\?
    WINDow\?
```

Note...

The FLOCation subsystem assumes that a Test Head will be used for the measurement, using a ratio measurement of B/C; consequently, there are no GPIB commands for setting up the measurement definition. If it is required to perform a single input fault location measurement, then the appropriate commands from the SCALar subsystem must be used, i.e. :SCALar:MEASure:POWer or :SCALar:MEASure:RATio.

:FLOCation
:ATTenuation

Parameters: <NRf>
attenuation per unit length (dB/m. or dB/ft.)

Description: Set the attenuation per unit length of the transmission line under test.
See [MEASURE] / F Loc Funcs for more details.

To determine the distance units in use (metres or feet) use :FLOC:UNIT?.

Example: `:FLOC:ATT 0.1`
Set the attenuation per unit length to 0.1 dB/m.

:FLOCation
:ATTenuation?

Parameters: [<NRf>]
[attenuation per unit length]

Response: <NR2>
attenuation per unit length

Example: `:FLOC:ATT?`
Read the transmission line attenuation per unit length.

:FLOCation
:AVERaging
:NUMBer

Parameters: <NRf>
averaging number

Description: Set the averaging number for a fault location channel. This is applied on the sweep after the current sweep.
See [MEASURE] / Averaging (Scalar Channel) for more details.

Example: `:FLOC:AVER:NUMB 16`
Set the averaging number to 16.

:FLOCation
:AVERaging
:NUMBer?

Parameters: [<NRf>]
[averaging number]

Response: <NR1>
averaging number

Example: `:FLOC:AVER:NUMB?`
Read the averaging number.

:FLOCation
:AVERaging
:RESTart

Description: Restart averaging. This is applied on the sweep after the current sweep.
See [MEASURE] / Measure for more details.

Example: `:FLOC:AVER:REST`
Restart averaging.

:FLOCation
:AVERaging
[:STATe]

Parameters: <BOOLEAN PROGRAM DATA>
state

Description: Set the averaging for a fault location channel on or off. This is applied on the sweep after the current sweep.
See [MEASURE] / Averaging (Scalar Channel) for more details.

Example: `:FLOC:AVER ON`
Enable averaging.

:FLOCation
:AVERaging
:STATe?

Parameters: [<BOOLEAN PROGRAM DATA>]
[state]

Response: <BOOLEAN RESPONSE DATA>
state

Example: `:FLOC:AVER:STAT?`
Determine whether averaging is enabled.

:FLOCation
:CALibration
:RESTore

Description: Restore the fault location measurement definition to that which was stored with the calibration data when the system was calibrated.
See [MEASURE] / Measure for more details.

Example: `:FLOC:CAL:REST`
Re-establish measurement conditions that were stored with the calibration.

:FLOCation

:CALibration

:SAVE

Description: Initiate a fault location calibration by saving the current measurement data and conditions to the fault location calibration store. Note that this command restarts averaging and requires the relevant number of sweeps to be completed, which is equal to the average number that has been set.
See [CAL] key for more details.

Example: `:FLOC:CAL:SAVE`
Perform a fault location calibration.

:FLOCation

:CALibration

:STORe

Parameters: <ARBITRARY BLOCK PROGRAM DATA>
store contents

Description: Write data to the fault location calibration store.

Example: `:FLOC:CAL:STOR` #...etc
Write to fault location calibration data store. (Only first byte of data is shown).

:FLOCation

:CALibration

:STORe?

Response: <INDEFINITE LENGTH ARBITRARY BLOCK RESPONSE DATA>
store contents

Example: `:FLOC:CAL:STOR?`
Read the contents of the fault location calibration store.

:FLOCation

:DATA

[:ASCii]?

Response: <NR2>,<NR2>,...,<NR2>
response at point 0,response at point 1...,response at point *n*-1

Description: Read fault location measurement data in ASCII format. Units of data returned depends on measurement format: dB or VSWR.

Example: `:FLOC?`
Read measurement data.

Example response: -10.12,-10.09,-10.03,...0.15,0.49

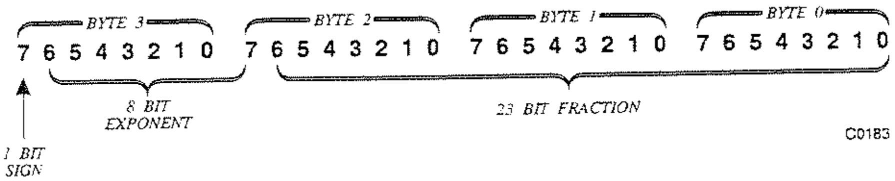
:FLOCation
:DATA
:BINary?

Response: <INDEFINITE LENGTH ARBITRARY BLOCK RESPONSE DATA>
<measurement data>
where <measurement data> *is* organised as follows:

The number of measurements in <measurement data> is equal to the number of measurement points (set using :FLOC:POINTS).

Each measurement consists of 4 bytes received in the order:
byte 0, byte 1, byte 2, byte 3.

These bytes hold a 32 bit (IEEE single precision) number conforming to the IEEE Standard of Binary Floating-Point Arithmetic, ANSI/IEEE Std 754-1985.



Description: Read fault location measurement data in Binary format. Binary format allows faster transfer of measurement data.

Example: `:FLOC:DATA:BIN?`
Read measurement data.

:FLOCation
:ENHanced

Parameters: <BOOLEAN PROGRAM DATA>
state

Description: Enable/disable fault location enhanced mode.
See [MEASURE] / Enhance Mode for more details.

Example: `:FLOC:ENH ON`
Enable enhanced mode.

:FLOCation
:ENHanced?

Parameters: [<BOOLEAN PROGRAM DATA>]
[state]

Response: <BOOLEAN RESPONSE DATA>
state

Example: `:FLOC:ENH?`
Determine whether fault location enhanced mode is enabled.

:FLOCation
:FORMat

Parameters: <CPD>
<settable fault location format>
where <settable fault location format> *is* VSWR | LOG

Description: Set the display format for a fault location measurement.
See [FORMAT] key for more details.

Example: `:FLOC:FORM VSWR`
Set VSWR format.

:FLOCation
:FORMat?

Parameters: [<CPD>]
[<fault location format>]
where <fault location format> *is* VSWR|LOG

Response: <CPD>
<fault location format>

Example: `:FLOC:FORM?`
Read the fault location format

:FLOCation
:FREQuency
:CENTer

Parameters: <NRf>
frequency (Hz)

Description: Set the sweep centre frequency for fault location measurements (when range entry mode is enabled).
See [MEASURE] / Set Up Meas for more details.

The instrument will also accept the English spelling, "CENTre".

Example: `:FLOC:FREQ:CENT 12.4E9`
Set centre frequency to 12.4 GHz.

:FLOCation
:FREQuency
:CENTer?

Parameters: [<NRf>]
[frequency (Hz)]

Response: <NR2>
frequency (Hz)

Example: `:FLOC:FREQ:CENT?`
Read the centre frequency.

:FLOCation
:FREQuency
[:ENTRy]

Parameters: <BOOLEAN PROGRAM DATA>
state

Description: Set frequency entry mode on or off.

This is coupled to the command which selects range entry mode (:FLOCation:RANGe:ENTRy) and is its inverse.
See [MEASURE] / Set Up Meas for more details.

Example:

```
:FLOC:FREQ ON
```

Enable frequency entry mode.

:FLOCation
:FREQuency
:ENTRy?

Parameters: [<BOOLEAN PROGRAM DATA>]
[state]

Response: <BOOLEAN RESPONSE DATA>
state

Example:

```
:FLOC:FREQ:ENTR?
```

Determine whether frequency entry mode is enabled.

:FLOCation
:FREQuency
:STARt

Parameters: <NRf>
frequency (Hz)

Description: Set the start frequency for fault location measurements (when frequency entry mode is enabled).
See [MEASURE] / Set Up Meas for more details.

Example:

```
:FLOC:FREQ:STAR 1E9
```

Set 1 GHz start frequency.

:FLOCation
:FREQuency
:STARt?

Parameters: [<NRf>]
[frequency (Hz)]

Response: <NR2>
frequency (Hz)

Example:

```
:FLOC:FREQ:STAR?
```

Read the fault location start frequency.

:FLOCation
:FREQuency
:STOP

Parameters: <NRf>
frequency (Hz)

Description: Set the stop frequency for fault location measurements (when frequency entry mode is enabled).
See [MEASURE] / Set Up Meas for more details.

Example: `:FLOC:FREQ:STOP 25E9`
Set 25 GHz stop frequency.

:FLOCation
:FREQuency
:STOP?

Parameters: [<NRf>]
[frequency (Hz)]

Response: <NR2>
frequency (Hz)

Example: `:FLOC:FREQ:STOP?`
Read the fault location stop frequency.

:FLOCation
:MCORrection

Parameters: <BOOLEAN PROGRAM DATA>
state

Description: Enable/disable masking correction.
See [MEASURE] / F Loc Funcs for more details.

Example: `:FLOC:MCOR ON`
Enable masking correction.

:FLOCation
:MCORrection?

Parameters: [<BOOLEAN PROGRAM DATA>]
[state]

Response: <BOOLEAN RESPONSE DATA>
state

Example: `:FLOC:MCOR?`
Determine whether masking correction is enabled.

:FLOCation
:MEDium

Parameters: <CPD>
<medium>
where <medium> *is* COAX | WAVeguide

Description: Set medium of transmission line under test.
See [MEASURE] / Config F Loc for more details.

Example: `:FLOC:MED COAX`
Inform instrument that measurements will be made on a coax transmission line.

:FLOCation
:MEDium?

Parameters: [<CPD>]
[<medium>]
where <medium> *is* COAX | WAVeguide

Response: <CRD>
<medium>

Example: `:FLOC:MED?`
Read the transmission line medium selected.

:FLOCation
:POINts

Parameters: <NRf>
number of points

Description: Set the number of measurement points for the fault location measurement.
See [MEASURE] / Set Up Meas for more details.

Example: `:FLOC:POIN 201`
Set number of points to 201.

:FLOCation
:POINts?

Parameters: [<NRf>]
[number of points]

Response: <NR1>
number of points

Example: `:FLOC:POIN?`
Read number of points.

:FLOCation
:RANGe
[:ENTRy]

Parameters: <BOOLEAN PROGRAM DATA>
state

Description: Set range entry mode on or off.

This is coupled to the command which selects frequency entry mode (:FLOCation:FREQuency:ENTRy) and is its inverse.
See [MEASURE] / Set Up Meas for more details.

Example: `:FLOC:RANG ON`
Enable range entry mode.

:FLOCation
:RANGe
:ENTRy?

Parameters: [<BOOLEAN PROGRAM DATA>]
[state]

Response: <BOOLEAN RESPONSE DATA>
state

Example: `:FLOC:RANG:ENTR?`
Determine whether range entry mode is enabled.

:FLOCation
:RANGe
:VALue

Parameters: <NRf>
range

Description: Sets the range of the fault location system.
The distance units will be interpreted as either metres or feet, as set up by the :FLOCation:UNITs command
See [MEASURE] / Set Up Meas for more details.

Example:

```
:FLOC:RANG:VAL 100
```

Set the fault location range to 100 m.

:FLOCation
:RANGe
:VALue?

Parameters: [<NRf>]
[range]

Response: <NR2>
range

Example:

```
:FLOC:RANG:VAL?
```

Determine the range that has been set.

:FLOCation
:RVELocity

Parameters: <NRf>
relative velocity

Description: Set the relative velocity of the currently selected medium (i.e. coax or waveguide).
See [MEASURE] / Set Up Meas for more details.

Example:

```
:FLOC:RVEL 0.8
```

Set the relative velocity to 0.8.

:FLOCation
:RVELocity?

Parameters: [<NRf>]
[relative velocity]

Response: <NR2>
relative velocity

Example:

```
:FLOC:RVEL?
```

Read the relative velocity.

:FLOCation
:SCALing
:AUTO

Parameters: <CPD>
<autoscale mode>
where <autoscale mode> *is* ONCE | CONTinuous | OFF

Description: Set the autoscale mode:

ONCE	Autoscaling for one cycle
CONTinuous	Enable continuous autoscaling
OFF	Disable continuous autoscaling

See [SCALING] key for more details.

Example:
```
:FLOC:SCAL:AUTO CONT
```
Enable continuous autoscaling.

:FLOCation
:SCALing
:AUTO?

Parameters: [<CPD>[
[<autoscale mode>]
where <autoscale mode> *is* ONCE | CONTinuous | OFF

Response: <CRD>
<continuous autoscale status>
where <continuous autoscale status> *is* CONT | OFF

Description: The scaling mode ONCE will never be returned by the query form of the command.

Example:
```
:FLOC:SCAL:AUTO?
```
Determine whether continuous scaling is enabled.

:FLOCation
:SCALing
:DIVision

Parameters: <NRf>
per division

Description: Set the graticule vertical scaling.
See [SCALING] key for more details.

Example: `:FLOC:SCAL:DIV 10`
Set the vertical scaling to 10 dB per division (assuming log format).

:FLOCation
:SCALing
DIVision?

Parameters: [<NRf>]
[per division]

Response: <NR2>
per division

Example: `:FLOC:SCAL:DIV?`
Read the vertical scaling.

:FLOCation
:SCALing
:POSition

Parameters: <NRf>
reference position

Description: Set the reference position.
See [SCALING] key for more details.

Example: `:FLOC:SCAL:POS 5`
Set the centre graticule line to be the reference line.

:FLOCation
:SCALing
:POSition?

Parameters: [<NRf>]
[reference position]

Response: <NR1>
reference position

Example: `:FLOC:SCAL:POS?`
Read the reference position.

:FLOCation
:SCALing
:RLEVel

Parameters: <NRf>
reference level

Description: Set the reference level.
Reference level units depend on the display format: dB or VSWR.

To determine the format in use (VSWR or LOG) use :FLOC:FORM?
See [SCALING] key for more details.

Example:

```
:FLOC:SCAL:RLEV -10
```

Set the reference level to -10 dB (assuming log format).

:FLOCation
:SCALing
:RLEVel?

Parameters: [<NRf>]
[reference level]

Response: <NR2>
reference level

Example:

```
:FLOC:SCAL:RLEV?
```

Read the reference level.

:FLOCation
:UNITs

Parameters: <CPD>
<distance units>
where <distance units> *is* FEET | METRes

Description: Set the distance unit to feet or metres.
See [MEASURE] / Config F Loc for more details.

Example:
```
:FLOC:UNIT METR
```
Set the distance units to metres.

:FLOCation
:UNITs?

Parameters: [<CPD>]
[<distance units>]
where <distance units> *is* FEET | METRes

Response: <CRD>
<distance units>

Example :
```
:FLOC:UNIT?
```
Determine the distance units currently in use.

:FLOCation
:WGCutoff

Parameters: <NRf>
waveguide cutoff frequency (Hz)

Description: Set the waveguide cutoff frequency.
See [MEASURE] / Set Up Meas for more details.

Example:
```
:FLOC:WGC 2.078E9
```
Set the waveguide cutoff frequency to 2.078 GHz.

:FLOCation
:WGCutoff?

Parameters: [<NRf>]
[waveguide cutoff frequency (Hz)]

Response: <NR2>
waveguide cutoff frequency (Hz)

Example:
```
:FLOC:WGC?
```
Read the waveguide cutoff frequency.

:FLOCation
:WINDow

Parameters: <CPD>
<window level>
where <window level> *is* LOW | MEDium | HIGH

Description: Set the window level.
See [MEASURE] / F Loc Funcs for more details.

Example: `:FLOC:WIND MED`
Set the window level to medium.

:FLOCation
:WINDow?

Parameters: [<CPD>]
[<window level>]
where <window level> *is* LOW | MEDium | HIGH

Response: <CRD>
<window level>

Example: `:FLOC:WIND?`
Read the window level

HARDcopy SUBSYSTEM

```
HARDcopy
    BUILd
    CONFigure
        DATE\?
        GANNotation\?
        GRATicule\?
        LIMits\?
        MANNotation\?
        MARKers\?
        MEASurement\?
        MTABle\?
        MTITle\?
        PRINter\?
        SCONditions\?
        STITle\?
        TYPE\?
    PLOT
        [ALL]
        TRACes
    PRINt
    SETTings
    TABLe
    TEST
```

:HARDcopy
:BUILd

Description: Start a print of the instrument build state.
See [UTILITY] / Status for more details.

Example: `:HARD:BUIL`

:HARDcopy
:CONFigure
:DATE

Parameters: <BOOLEAN PROGRAM DATA>
state

Description: Enable/disable plotting of date and time.
See [COPY] / Graph Attrib for more details.

Example: `:HARD:CONF:DATE ON`
Plot date and time.

:HARDcopy
:CONFigure
:DATE?

Parameters: [<BOOLEAN PROGRAM DATA>]
[state]

Response: <BOOLEAN RESPONSE DATA>
state

Example: `:HARD:CONF:DATE?`
Determine whether plotting of date and time is enabled.

:HARDcopy
:CONFigure
:GANNotation

Parameters: <BOOLEAN PROGRAM DATA>
state

Description: Enable/disable plotting of the graticule annotation.
See [COPY] / Graph Attrib for more details.

Example:
```
:HARD:CONF:GANN ON
```
Plot graticule annotation.

:HARDcopy
:CONFigure
:GANNotation?

Parameters: [<BOOLEAN PROGRAM DATA>]
[state]

Response: <BOOLEAN RESPONSE DATA>
state

Example:
```
:HARD:CONF:GANN?
```
Determine whether plotting of graticule annotation is enabled.

:HARDcopy
:CONFigure
:GRATicule

Parameters: <BOOLEAN PROGRAM DATA>
state

Description: Enable/disable plotting of graticules.
See [COPY] / Graph Attrib for more details.

Example:
```
:HARD:CONF:GRAT ON
```
Plot graticules.

:HARDcopy
:CONFigure
:GRATicule?

Parameters: [<BOOLEAN PROGRAM DATA>]
[state]

Response: <BOOLEAN RESPONSE DATA>
state

Example:
```
:HARD:CONF:GRAT?
```
Determine whether plotting of graticules is enabled.

:HARDcopy
:CONFigure
:LIMits

Parameters: <BOOLEAN PROGRAM DATA>
state

Description: Enable/disable plotting of limit lines.
See [COPY] / Graph Attrib for more details.

Example: `:HARD:CONF:LIM ON`
Plot limit lines.

:HARDcopy
:CONFigure
:LIMits?

Parameters: [<BOOLEAN PROGRAM DATA>]
[state]

Response: <BOOLEAN RESPONSE DATA>
state

Example: `:HARD:CONF:LIM?`
Determine whether plotting of limit lines is enabled.

:HARDcopy
:CONFigure
:MANNotation

Parameters: <BOOLEAN PROGRAM DATA>
state

Description: Enable/disable plotting of the measurement annotation.
See [COPY] / Graph Attrib for more details.

Example: `:HARD:CONF:MANN ON`
Plot measurement annotation.

:HARDcopy
:CONFigure
:MANNotation?

Parameters: [<BOOLEAN PROGRAM DATA>]
[state]

Response: <BOOLEAN RESPONSE DATA>
state

Example: `:HARD:CONF:MANN?`
Determine whether plotting of measurement annotation is enabled.

:HARDcopy
:CONFigure
:MARKers

Parameters: <BOOLEAN PROGRAM DATA>
state

Description: Enable/disable plotting of markers.
See [COPY] / Graph Attrib for more details.

Example: `:HARD:CONF:MARK ON`
Plot markers.

:HARDcopy
:CONFigure
:MARKers?

Parameters: [<BOOLEAN PROGRAM DATA>]
[state]

Response: <BOOLEAN RESPONSE DATA>
state

Example: `:HARD:CONF:MARK?`
Determine whether plotting of markers is enabled.

:HARDcopy
:CONFigure
:MEASurement

Parameters: <BOOLEAN PROGRAM DATA>
state

Description: Enable/disable plotting of measurements.
See [COPY] / Graph Attrib for more details.

Example: `:HARD:CONF:MEAS ON`
Plot measurements.

:HARDcopy
:CONFigure
:MEASurement?

Parameters: [<BOOLEAN PROGRAM DATA>]
[state]

Response: <BOOLEAN RESPONSE DATA>
state

Example: `:HARD:CONF:MEAS?`
Determine whether plotting of measurements is enabled.

:HARDcopy
:CONFigure
:MTABle

Parameters: <BOOLEAN PROGRAM DATA>
state

Description: Enable/disable plotting of marker table.
See [COPY] / Graph Attrib for more details.

Example: `:HARD:CONF:MTAB ON`
Plot marker table.

:HARDcopy
:CONFigure
:MTABle?

Parameters: [<BOOLEAN PROGRAM DATA>]
[state]

Response: <BOOLEAN RESPONSE DATA>
state

Example: `:HARD:CONF:MTAB?`
Determine whether plotting of the marker table is enabled.

:HARDcopy
:CONFigure
:MTITle

Parameters: <BOOLEAN PROGRAM DATA>
state

Description: Enable/disable plotting of measurement titles.
See [COPY] / Graph Attrib for more details.

Example: `:HARD:CONF:MTIT ON`
Plot measurement titles.

:HARDcopy
:CONFigure
:MTITle?

Parameters: [<BOOLEAN PROGRAM DATA>]
[state]

Response: <BOOLEAN RESPONSE DATA>
state

Example: `:HARD:CONF:MTIT?`
Determine whether plotting of measurement titles is enabled.

:HARDcopy
:CONFigure
:PRINter

Parameters: <CPD>
<printer>
where <printer> *is* EPSon | DESKjet | CANon | CDESkjet

Description: Select the printer to be either an Epson FX series printer, a Hewlett Packard DeskJet / LaserJet series printer, a Canon BJ series printer, or a Hewlett Packard Colour DeskJet series printer.

Example: `:HARD:CONF:PRIN DESK`
Select a Hewlett Packard DeskJet / LaserJet series printer.

:HARDcopy
:CONFigure
:PRINter?

Parameters: [<CPD>]
[<printer>]
where <printer> *is* EPSon | DESKjet | CANon | CDESkjet

Response: <CRD>
<printer>

Example: `:HARD:CONF:PRIN?`
Determine the currently selected printer type.

:HARDcopy
:CONFigure
:SCONditions

Parameters: <BOOLEAN PROGRAM DATA>
state

Description: Enable/disable plotting of the setup conditions.
See [COPY] / Graph Attrib for more details.

Example: `:HARD:CONF:SCON ON`
Plot setup conditions.

:HARDcopy
:CONFigure
:SCONditions?

Parameters: [<BOOLEAN PROGRAM DATA>]
[state]

Response: <BOOLEAN RESPONSE DATA>
state

Example: `:HARD:CONF:SCON?`

:HARDcopy
:CONFigure
:STITle

Parameters: <BOOLEAN PROGRAM DATA>
state

Description: Enable/disable plotting of screen title.
See [COPY] / Graph Attrib for more details.

Example: `:HARD:CONF:STIT ON`
Plot screen title.

:HARDcopy
:CONFigure
:STITle?

Parameters: [<BOOLEAN PROGRAM DATA>]
[state]

Response: <BOOLEAN RESPONSE DATA>
state

Example: `:HARD:CONF:STIT?`
Determine whether plotting of screen titles is enabled.

:HARDcopy
:CONFigure
:TYPE

Parameters: <BOOLEAN PROGRAM DATA>
state

Description: Enable/disable plotting of instrument type number.
See [COPY] / Graph Attrib for more details.

Example: `:HARD:CONF:TYPE ON`
Plot instrument type number.

:HARDcopy
:CONFigure
:TYPE?

Parameters: [<BOOLEAN PROGRAM DATA>]
[state]

Response: <BOOLEAN RESPONSE DATA>
state

Example: `:HARD:CONF:TYPE?`
Determine whether plotting of the instrument type number is enabled.

:HARDcopy
:PLOT
[:ALL]

Description: Start a plot of the displayed measurements, including all extra information.

Plotting requires that the instrument must be given control of the GPIB. The recommended steps for plotting under GPIB control are as follows:

1. Use the common command *PCB to tell the instrument the address of the controller. The instrument needs to know this so that control can be passed back when the plot is complete.

Example:

```
*PCB 21
```

Inform the instrument that the controller is located at address 21.

2. Initiate the plot

Example:

```
:HARD:PLOT
```

3. Wait for the instrument to request control of the GPIB. The RQC bit (bit 1) of the Standard Event Register is set when the instrument is ready to accept control. This register may be read using *ESR?

Example:

```
*ESR?
```

4. Pass control to the instrument. Refer to the GPIB controller's operating manual for details.

5. When the plot has completed the instrument will pass control back to the controller.

:HARDcopy
:PLOT
:TRACes

Description: As for :HARDcopy:PLOT [:ALL], .but only the traces are plotted.

Example:

```
:HARD:PLOT:TRAC
```

Plot traces only.

:HARDcopy
:PRINt

Description: Start a print of the displayed measurements.

Example:

```
:HARD:PRIN
```

:HARDcopy
:SETTings

Parameters: <NRf>
store number

Description: Start a print of the selected settings store
See [SAVE/RECALL] / View Store for more details.

Example: `:HARD:SETT 2`
Start a printout of settings store 2.

:HARDcopy
:TABLe

Description: Start a print of the displayed measurements in tabular form.
See [COPY] / Copy for more details.

Example: `:HARD:TABL`

:HARDcopy
:TEST

Description: Start a print of the self test results.
See [UTILITY] / Status for more details.

Example: `:HARD:TEST`

INPut SUBSYSTEM

```
INPut
    ACDetection\?
    CALibrate
        DETector
        [SENSor]
    CFACtor
        FREQuency
            FROM\?
            [VALue]\?
        STORe
            [DATA]\?
            MAXimum?
            NUMBer\?
        [VALue]\?
    DCONfigure\?
    FCORrection
        FREQuency
            FROM\?
            [VALue]\?
        [STATe]\?
    LFACtor
        [VALue]\?
    OFFSet\?
    REFerence\?
    SCALar
        AUTO
        DETector\?
        TCORrection\?
    ZERO
        AUTO\?
        DETector
        SENSor
```

:INPut

:ACDetection

Parameters: <BOOLEAN PROGRAM DATA>
state

Description: Enable/disable AC detection for all scalar inputs.
See [MEASURE] / Gen Set-up for more details.

Example:
```
:INP:ACD ON
```
Enable AC detection.

:INPut

:ACDetection?

Parameters: <BOOLEAN PROGRAM DATA>
[state]

Response: <BOOLEAN RESPONSE DATA>
state

Example:
```
:INP:ACD?
```
Determine whether AC detection is enabled.

:INPut

:CALibrate

:DETector

Parameters: <CPD>
<input id>
where <input id> *is* A | B | C | D

Description: Calibrate the EEPROM detector connected to the specified input for linearity and sensitivity
See [MEASURE] / Detector Correction (Scalar Channel) for more details.

Example:
```
:INP:CAL:DET:C
```
Calibrate EEPROM detector on input C.

:INPut
:CALibrate
[:SENSor]

Description: Calibrate power sensor connected to input D. Power sensor should be connected to POWER REF output before command is sent.
See [CAL] / Sensor Cal (Scalar Channel) for more details.

Example:
```
:INP:CAL
```
Calibrate sensor

:INPut
:CFACtor
:FREQuency
:FROM

Parameters: <CPD>
<cal factor freq origin>
where <cal factor freq origin> *is* SOURce | COUNter | FREQuency

Description: Enable cal factor look-up from a sensor cal store and specify the origin of the frequency used to index the store.
See [MEASURE] / Sensor Corr (Scalar Channel) for more details.

Example:
```
:INP:CFAC:FREQ:FROM COUN
```
Allow frequency measurements from the counter to determine the cal factor frequency

:INPut
:CFACtor
:FREQuency
:FROM?

Parameters: [<CPD>]
[<cal factor freq origin>]
where <cal factor freq origin> *is* SOURce | COUNter | FREQuency

Response: <CRD>
<cal factor freq origin>

Description: If optional parameter is supplied, calibration look-up from a sensor cal store is enabled.
See [MEASURE] / Sensor Corr (Scalar Channel) for more details.

Example:
```
:INP:CFAC:FREQ:FROM?
```
Read the source of the cal factor look-up frequency

:INPut
:CFACtor
:FREQuency
[:VALue]

Parameters: <NRf>
frequency (Hz)

Description: Sets the frequency used to select a value of cal factor from the cal factor table held in a sensor cal store.

Before sending this command, :INPut:CFACtor:FREQuency:FROM FREQ should be sent to enable cal factor look-up from an explicitly set frequency. *See [MEASURE] / Sensor Corr (Scalar Channel) for more details.*

Example:

```
:INP:CFAC:FREQ 2.0E+9
```

Use cal factor from current sensor cal store corresponding to a frequency of 2.0 GHz.

:INPut
:CFACtor
:FREQuency
:VALue?

Parameters: <NRf>
[frequency (Hz)]

Response: <NR2>
frequency (Hz)

Example:

```
:INP:CFAC:FREQ:VAL?
```

Read frequency used to select cal factor

:INPut
:CFACtor
:STORe
[:DATA]

Parameters: <NRf>, <ARBITRARY BLOCK PROGRAM DATA>
sensor cal store number, store contents

Description: Send sensor cal data to the instrument's sensor cal store.

Example: `:INP:CFAC:STOR 1, #..etc.`
Send sensor cal data to sensor cal store 1 (only first byte of data is shown).

:INPut
:CFACtor
:STORe
:DATA?

Parameters: <NRf>[, <ARBITRARY BLOCK PROGRAM DATA>]
sensor cal store number[, store contents]

Response: <INDEFINITE LENGTH ARBITRARY BLOCK PROGRAM DATA>
store contents

Description: The contents of a sensor cal store may be read from for storage and later transferred back to the instrument.

Example: `:INP:CFAC:STOR:DATA? 2`
Read sensor cal data from sensor cal store 2.

:INPut
:CFACtor
:STORe
:MAXimum?

Response: <NR1>
Number of sensor cal stores available, including any located on a memory card.

Example: `:INP:CFAC:STOR:MAX?`
Determine how many sensor cal stores are present.

:INPut
:CFACtor
:STORe
:NUMBer

Parameters: <NRf>
sensor cal store number

Description: Select a sensor cal store. The store contains a 50 MHz cal factor, a linearity factor and a cal factor table.
See [MEASURE] / Sensor Corr for more details.

Example:

```
:INP:CFAC:STOR:NUMB 4
```

Use sensor cal store 4 for subsequent measurements.

:INPut
:CFACtor
:STORe
:NUMBer?

Parameters: [<NRf>]
[sensor cal store number]

Response: <NR1>
sensor cal store number

Example:

```
:INP:CFAC:STOR:NUMB?
```

Read sensor cal store currently in use.

:INPut
:CFACtor
[:VALue]

Parameters: <NRf>
cal factor

Description: Set explicit cal factor value to be used during measurements, and the 50 MHz cal factor used for sensor calibration, for a power sensor connected to input D.

If the cal factor was previously being obtained from a power sensor cal store, measurements following this command will be corrected using the explicitly set cal factor togther with the explicitly set linearity factor (set using :INPut:LFACtor)
See [MEASURE] / Sensor Corr for more details.

Example:
```
:INP:CFAC:VAL 90
```
Set cal factor to 90%

:INPut
:CFACtor
:VALue?

Parameters: [<NRf>]
[cal factor]

Response: <NR2>
cal factor

Description: If the optional parameter is supplied and the cal factor was previously being obtained from a sensor cal store, measurements following this command will be corrected using the explicitly set cal factor togther with the explicitly set linearity factor (which may be set using :INPut:LFACtor).
See [MEASURE] / Sensor Corr for more details.

Example:
```
:INP:CFAC:VAL?
```
Read the explicitly set cal factor.

:INPut

:DCONfigure

Parameters: <CPD>
<configuration>
where <configuration> *is* SCALar | PMETer

Description: Configure input D as a scalar detector or power meter sensor input.
See [MEASURE] / Input Config for more details.

Example:
```
:INP:DCON SCAL
```
Configure input D as a scalar detector input.

:INPut

:DCONfigure?

Parameters: [<CPD>]
[<configuration>]
where <configuration> *is* SCALar | PMETer

Response: <CRD>
<configuration>

Example:
```
INP:DCON?
```
Read the configuration for input D.

:INPut
:FCORrection
:FREQuency
:FROM

Parameters: <CPD>, <CPD>
<input id>, <flatness correction freq origin>
where <input id> *is* A | B | C | D
<flatness correction freq origin> *is* SOURce | COUNter | FREQuency

Description: Specify where the frequency used to index the EEPROM detector table for power meter readout measurements flatness correction comes from.
See [MEASURE] / Input A (B, C or D) for more details.

Example: `:INP:FCOR:FREQ:FROM A,COUN`
Allow frequency measurements from the counter to determine the flatness correction frequency on input A.

:INPut
:FCORrection
:FREQuency
:FROM?

Parameters: <CPD>[, <CPD>]
<input id>[, <flatness correction freq origin>]
where <input id> *is* A | B | C | D
<flatness correction freq origin> *is* SOURce | COUNter | FREQuency

Response: <CRD>
<flatness correction freq origin>

Example: `:INP:FCOR:FREQ:FROM? B`
Read the source of the flatness correction frequency for input B.

:INPut

:FCORrection

:FREQuency

[:VALue]

Parameters: <CPD>, <NRf>
<input id>, <frequency (Hz)>
where <input id> *is* A | B | C | D

Description: Select the flatness correction frequency to be user entered and set that frequency. Applies to power meter readouts only .
See [MEASURE] / Input A (B, C or D) for more details.

Example: `:INP:FCOR:FREQ A, 2.0E9`
Set the frequency used to index the EEPROM detector table to 2 GHz.

:INPut

:FCORrection

:FREQuency

:VALue?

Parameters: <CPD>[, <NRf>]
<input id>[, frequency (Hz)]
where <input id> *is* A | B | C | D

Response: <NR2>
frequency (Hz)

Example: `:INP:FCOR:FREQ:VAL? A`
Read the frequency used to index the EEPROM detector table for flatness correction on input A. The command is only valid if user selected freq is currently selected, using `:INP:FCOR:FREQ:FROM FREQ`

:INPut
:FCORrection
[:STATe]

Parameters: <CPD>,<BOOLEAN PROGRAM DATA>
<input id>, state
where <input id> *is* A | B | C | D

Description: Turn flatness correction on or off for the specified input.
See [MEASURE] / Input A (B, C or D) for more details.

Example: `:INP:FCOR B, ON`
Turn on flatness correction for input B

:INPut
:FCORrection
:STATe?

Parameters: <CPD>[,<BOOLEAN PROGRAM DATA>]
<input id>[, state]
where <input id> *is* A | B | C | D

Response: <BOOLEAN RESPONSE DATA>
state

Example: `:INP:FCOR:STAT? C`
Determine whether flatness correction is enabled on input C.

:INPut
:LFACtor
[:VALue]

Parameters: <NRf>
linearity factor

Description: Set sensor linearity factor for use when cal factor is not obtained from a sensor cal store.
See [MEASURE] / Sensor Corr for more details.

Example: `:INP:LFAC 8.5`
Set linearity factor to 8.5

:INPut
:LFACtor
:VALue?

Parameters: [<NRf>]
[linearity factor]

Response: <NR2>
linearity factor

Example: `:INP:LFAC:VAL?`
Read linearity factor value.

:INPut

:OFFSet

Parameters: <CPD>,<NRf>
<input id>, offset value (dB)
where <input id> *is* A | B | C | D

Description: Set offset to be applied to all measurements from specified input.
See [MEASURE] / Input Config for more details.

Example:

```
:INP:OFFS D,10
```

Offset all measurements from input D (whether scalar or power meter) by +10 dB.

:INPut

:OFFSet?

Parameters: <CPD>[,<NRf>]
<input id>[,offset value (dB)]
where <input id> *is* A | B | C | D

Response: <NR2>
offset value (dB)

Example:

```
:INP:OFFS? A
```

Read offset assigned to input A.

:INPut

:REFerence

Parameters: <BOOLEAN PROGRAM DATA>
state

Description: Switch power reference on or off.
See[CAL] / Cal for more details.

Example: `:INP:REF ON`
Set power reference on.

:INPut

:REFerence?

Parameters: [<BOOLEAN PROGRAM DATA>]
[state]

Response: <BOOLEAN RESPONSE DATA>
state

Example: `:INP:REF?`
Determine whether power reference is on or off.

:INPut

:SCALar

:AUTO

Parameters: <CPD>
<input id>
where <input id> *is* A | B | C | D

Description: Enable scalar detector autosensing for specified input.
See [MEASURE] / Input A (B, C or D) for more details.

Example: `:INP:SCAL:AUTO A`
Enable autosensing for input A

:INPut
:SCALar
:DETector

Parameters: <CPD>,<NRf>
<input id>,<detector type>
where <input id> *is* A | B | C | D *and*
<detector type> *is obtained from the table*:

Detector Type	Description	Detector Type	Description
1	6511	2	6512
3	6513	4	6514
5	6230	6	6233
7	6234	8	6240 Fault Location
9	6240 Return Loss	10	Reserved
11	Reserved	12	USER1
13	USER2	14	USER3
15	USER4	16	Detector correction off

Description: Set detector type for specified scalar input. Autosensing is disabled for that input. Selecting detector type 16 turns off detector correction.
See [MEASURE] / Input A (B, C or D) for more details.

Example:

```
:INP:SCAL:DET A,7
```

Set input A to accept a "6234" detector.

:INPut
:SCALar
:DETector?

Parameters: <CPD>[,<NRf>]
<input id>[,<detector type>]
where <input id> *is* A | B | C | D *and*
<detector type> *is obtained from the table above.*

Response: <NR1>
<detector type>

Description: If optional parameter <detector type> is not supplied, the detector autosensing state for the input remains unchanged. The query form may therefore be used to check that the instrument's autosensing function has correctly identified a detector.
See [MEASURE] / Input A (B, C or D) for more details.

Example:

```
:INP:SCAL:DET? C
```

Reads detector type set for input C.

:INPut
:SCALar
:TCORrection

Parameters: <CPD>,<BOOLEAN PROGRAM DATA>
<input id>,<state>
where <input id> *is* A | B | C | D

Description: Enable/disable temperature correction for the specified input.
See [MEASURE] / Input A (B, C or D) for more details.

Example: `:INP:SCAL:TCOR A, ON`
Enable temperature correction for input A.

:INPut
:SCALar
:TCORrection?

Parameters: <CPD>[,<BOOLEAN PROGRAM DATA>]
<input id>[,state]
where <input id> *is* A | B | C | D

Response: <BOOLEAN RESPONSE DATA>
state

Example: `:INP:SCAL:TCOR? A`
Determine whether temperature correction is enabled for input A.

:INPut
:ZERO
:AUTO

Parameters: <BOOLEAN PROGRAM DATA>
state

Description: Enable/disable automatic autozero for all scalar detectors.
See [CAL] / Det/Sen Zero for more details.

Example: `:INP:ZERO:AUTO ON`
Switch detector autozero on.

:INPut
:ZERO
:AUTO?

Parameters: [<BOOLEAN PROGRAM DATA>]
[state]

Response: <BOOLEAN RESPONSE DATA>
state

Example: `:INP:ZERO:AUTO?`
Determine whether detector autozero is enabled.

:INPut
:ZERO
:DETector

Description: Zero all scalar detectors
See [CAL] / Det/Sen Zero for more details.

Example:
```
:INP:ZERO:DET
Zero detectors.
```

:INPut
:ZERO
:SENSor

Description: Zero power meter sensor connected to input D.
See page [CAL] / Det/Sen Zero for more details.

Example:
```
:INP:ZERO:SENS
Zero sensor.
```

MARKer SUBSYSTEM

```
MARKer
    ACTive
        ASSign\?
        DATA?
        POSition\?
        [STATe]\?
    BANDwidth
        CFDF\?
        [DATA]?
        FROM\?
        TARGet\?
    COUPling\?
    DELTa
        POSition\?
        [STATe]\?
    LPEak
    MAXimum
    MINimum
    PKPK
        [DATA]?
        LIMit
            [STATe]\?
            VALue\?
    POSition\?
    RPEak
    SEARch
        DIRection\?
        RESult?
        TARGet\?
    SLOPe\?
    STATe\?
```

:MARKer
:ACTive
:ASSign

Parameters: <NRf>
marker number

Description: Designate a marker as the active marker.
See [MARKERS] / Set Up Mkrs for more details.

Example: `:MARK:ACT:ASS 4`
Make marker 4 the active marker.

:MARKer
:ACTive
:ASSign?

Parameters: [<NRf>]
[marker number]

Response: <NR1>
marker number

Example: `:MARK:ACT:ASS?`
Determine which is the active marker.

:MARKer
:ACTive
:DATA?

Response: <NR2>
active marker measurement

The active marker measurement returned should be interpreted according to the active measurement format:

Frequency	Hz
Log Power	dB or dBm
Linear Power	Watts
Volts	V
VSWR	Units

For a reflection analyzer channel, a complex (2 part) number will be returned.

Description: Read the measurement at the active marker position.

Example: `:MARK:ACT:DATA?`
Take a spot reading at the active marker position.

:MARKer
ACTive
:POSition

Parameters: <NRf>
domain value

The domain value will be interpreted according to the active channel domain:

Frequency	Hz
Power	dBm
Voltage	V
Current	A
Time	s
Distance	m or ft

Description: Set the active marker to a domain value.
See [MARKERS] / Markers for more details.

Example:
```
:MARK:ACT:POS 15E9
```
Set the active marker to 15 GHz.

:MARKer
:ACTive
:POSition?

Parameters: [<NRf>]
[domain value]

The domain value will be interpreted according to the active channel domain:

Frequency	Hz
Power	dBm
Voltage	V
Current	A
Time	s
Distance	m or ft

Response: <NR2>
domain value

Example:
```
:MARK:ACT:POS?
```
Read the active marker position.

:MARKer

:ACTive

[:STATe]

Parameters: <BOOLEAN PROGRAM DATA>
state

Description: Set the active marker on or off.
See [MARKERS] / Markers for more details.

Example:
```
:MARK:ACT ON
```
Switch on the active marker.

:MARKer

:ACTive

:STATe?

Parameters: [<BOOLEAN PROGRAM DATA>]
[state]

Response: <BOOLEAN RESPONSE DATA>
state

Example:
```
:MARK:ACT:STAT?
```
Determine whether the active marker is on.

:MARKer

:BANDwidth

:CFDF

Parameters: <BOOLEAN PROGRAM DATA>
state

Description: Set the marker centre frequency/delta frequency(CF/DF) calculation on or off. When it is turned on, the :MARKER:BANDwidth [:DATA]? command returns the CF/DF ratio in addition to the bandwidth and centre frequency values.
See [MARKERS] / Bandwidth for more details.

Example:
```
:MARK:BAND:CFDF ON
```
Enable the CF/DF calculation.

:MARKer

:BANDwidth

:CFDF?

Parameters: [<BOOLEAN PROGRAM DATA>]
[state]

Response: <BOOLEAN RESPONSE DATA>
state

Example:
```
:MARK:BAND:CFDF?
```
Determine whether the CF/DF calculation is enabled.

:MARKer
:BANDwidth
[:DATA]?

Response: <NR2>, <NR2>
bandwidth (Hz), centre frequency (Hz)

Description: Initiate a bandwidth search and return the bandwidth and centre frequency. If the search fails, the returned values will be 0. If the marker centre frequency/delta frequency calculations is enabled (:MARKer:BANDwidth:CFDF command), the value of this ratio is returned in addition to the bandwidth and centre frequency values.
See [MARKERS] / Bandwidth for more details.

Example:
```
:MARK:BAND?
```
Perform a bandwidth measurement.

:MARKer
:BANDwidth
:FROM

Parameters: <CPD>
<search start point>
where <search start point> *is* MAXMin | ACTive

Description: Set the starting point for the bandwidth search to either the active marker position or the maximum/minimum point on the trace.
See [MARKERS] / Bandwidth for more details.

Example:
```
:MARK:BAND:FROM MAXM
```
Set up the bandwidth search to start from max/min point on the trace.

:MARKer
:BANDwidth
:FROM?

Parameters: [<CPD>]
[<search start point>]
where <search start point> *is* MAXMin | ACTive

Response: <CRD>
<search start point>

Example:
```
:MARK:BAND:FROM?
```
Determine where the bandwidth search will start from.

:MARKer
:BANDwidth
:TARGet

Parameters: <NRf>
target (dB)

Description: Set the target for a marker bandwidth search.
See [MARKERS] / Bandwidth for more details.

Example: `:MARK:BAND:TARG -3`
Set up the search target for 3 dB bandwidth measurements.

:MARKer
:BANDwidth
:TARGet?

Parameters: [<NRf>]
[target (dB)]

Response: <NR2>
target (dB)

Example: `:MARK:BAND:TARG?`
Read the bandwidth search target.

:MARKer
:COUPling

Parameters: <BOOLEAN PROGRAM DATA>
state

Description: Enable/disable marker coupling between channels.

Note that marker coupling may only be disabled when channel coupling is also disabled.
See [MARKERS] / Set Up Mkrs for more details.

Example: `:MARK:COUP OFF`
Disable marker coupling.

:MARKer
:COUPling?

Parameters: [<BOOLEAN PROGRAM DATA>]
[state]

Response: <BOOLEAN RESPONSE DATA>
state

Example: `:MARK:COUP?`
Determine whether marker coupling is enabled.

:MARKer

:DELTa

:POSition

Parameters: <NRf>
domain value

The domain value will be interpreted according to the active channel domain:

Frequency	Hz
Power	dBm
Voltage	V
Current	A
Time	s
Distance	m or ft

Description: Set the delta marker to a domain value.
See [MARKERS] / Markers for more details.

Example:
```
:MARK:DELT:POS 12E9
```
Set the delta marker to 12 GHz.

:MARKer

:DELTa

:POSition?

Parameters: [<NRf>]
[domain value]

The domain value will be interpreted according to the active channel domain:

Frequency	Hz
Power	dBm
Voltage	V
Current	A
Time	s
Distance	m or ft

Response: <NR2>
domain value

Example:
```
:MARK:DELT:POS?
```
Read the delta marker position.

:MARKer
:DELTa
[:STATe]

Parameters: <BOOLEAN PROGRAM DATA>
state

Description: Set the delta marker on or off.
See [MARKERS] / Markers for more details.

Example: `:MARK:DELT ON`
Switch on the delta marker.

:MARKer
:DELTa
:STATe?

Parameters: [<BOOLEAN PROGRAM DATA>]
[state]

Response: <BOOLEAN RESPONSE DATA>
state

Example: `:MARK:DELT:STAT?`
Determine whether the delta marker is on.

:MARKer
:LPEak

Description: Place the active marker at the next peak to the left of its current position.
See [MARKERS] / Mkr Funcs (Fault Location Channel, Ref Analyzer Channel) for more details.

Example: `:MARK:LPE`

:MARKer
:MAXimum

Description: Place the active marker at the trace maximum.
See [MARKERS] / Mkr Funcs (Scalar Channel, Ref Analyzer Channel) for more details.

Example: `:MARK:MAX`

:MARKer
:MINimum

Description: Place the active marker at the trace minimum.
See [MARKERS] / Mkr Funcs (Scalar Channel, Ref Analyzer Channel) for more details.

Example: `:MARK:MIN`

:MARKer
:PKPK
[:DATA]?

Response: <BOOLEAN RESPONSE DATA>, <NR2>
pass/fail flag, measurement

Description: Initiate a peak-peak measurement, returning the result as a pass/fail flag. Note that the pass/fail flag will return a TRUE value when limit checking is disabled.
See [MARKERS] / Peak to Peak (Scalar Channel, Ref Analyzer Channel) for more details.

Example: `:MARK:PKPK?`
Perform a peak-peak measurement.

:MARKer
:PKPK
:LIMit
[:STATe]

Parameters: <BOOLEAN PROGRAM DATA>
state

Description: Set peak-peak measurement limit checking on or off.
See [MARKERS] / Peak to Peak (Scalar Channel, Ref Analyzer Channel) for more details.

Example: `:MARK:PKPK:LIM ON`
Enable peak-peak measurement limit checking.

:MARKer
:PKPK
:LIMit
:STATe?

Parameters: [<BOOLEAN PROGRAM DATA>]
[state]

Response: <BOOLEAN RESPONSE DATA>
state

Example: `:MARK:PKPK:LIM:STAT?`
Determine whether peak-peak measurement limit checking is enabled.

:MARKer
:PKPK
:LIMit
:VALue

Parameters: <NRf>
limit value (dB)

Description: Set the limit value for peak-peak limit checking.
See [MARKERS] / Peak to Peak (Scalar Channel, Ref Analyzer Channel) for more details.

Example: `:MARK:PKPK:LIM:VAL 0.8`
Set the peak-peak limit to 0.8 dB.

:MARKer
:PKPK
:LIMit
:VALue?

Parameters: [<NRf>]
[limit value (dB)]

Response: <NR2>
limit value (dB)

Example: `:MARK:PKPK:LIM:VAL?`
Read the limit value set for peak-peak measurements.

:MARKer

:POSition

Parameters: <NRf>, <NRf>
marker number, domain value

The domain value will be interpreted according to the active channel domain:

Frequency	Hz
Power	dBm
Voltage	V
Current	A
Time	s
Distance	m or ft

Description: Set a marker to a domain value.
See [MARKERS] / Set Up Mkrs for more details.

Example:

```
:MARK:POS 7,10E9
```

Place marker 7 at 10 GHz.

:MARKer

:POSition?

Parameters: [<NRf>, <NRf>]
[marker number, domain value]

Response: <NR 2>
domain value

Example:

```
:MARK:POS? 6
```

Read position of marker 6

:MARKer

:RPEak

Description: Place the active marker at the next peak to the right of its current position.
See [MARKERS] / Mkr Funcs (Fault Location Channel, Ref Analyzer Channel) for more details.

Example: `:MARK:RPE`

:MARKer

:SEARch

:DIRection

Parameters: <CPD>
<search direction>
where <search direction> *is* LEFT | RIGHt

Description: Set the direction for marker search operations.
See[MARKERS] / Search for more details.

Example: `:MARK:SEAR:DIR RIGH`
Set marker search right.

:MARKer

:SEARch

:DIRection?

Parameters: [<CPD>]
[<search direction>]
where <search direction> *is* LEFT | RIGHt

Response: <CRD>
<search direction>

Example: `:MARK:SEAR:DIR?`
Read the marker search direction.

:MARKer

:SEARch

:RESult?

Response: <BOOLEAN RESPONSE DATA>
target found flag

Description: Initiate a marker search and return the result as a success or failure.
See[MARKERS] / Search for more details.

Example: `:MARK:SEAR:RES?`
Perform a marker search.

:MARKer
:SEARch
:TARGet

Parameters: <NRf>
target

The target value will be interpreted according to the active measurement format:

Frequency	Hz
Log Power	dB or dBm
Linear Power	Watts
Voltage	V
VSWR	Units

Description: Set the search target value for marker search operations.
See[MARKERS] / Search for more details.

Example:

```
:MARK:SEAR:TARG -3.0
```

Set the search target to –3.0dB.

:MARKer
:SEARch
:TARGet?

Parameters: [<NRf>]
[target]

The target value will be interpreted according to the active measurement format:

Frequency	Hz
Log Power	dB or dBm
Linear Power	Watts
Voltage	V
VSWR	Units

Response: <NR2>
target

Example:

```
:MARK:SEAR:TARG?
```

Read the marker search target value.

:MARKer
:SLOPe

Parameters: <CPD>
<function>
where <function> *is* DBO | DBD | OFF

Description: Set the marker slope function:
DBO dB/octave
DBD dB/decade
OFF Disable slope function
See [MARKERS] / dB/O dB/D for more details.

Example:
```
:MARK:SLOP DBD
```
Set the marker slope function to dB/decade.

:MARKer
:SLOPe?

Parameters: [<CPD>]
[<function>]
where <function> *is* DBO | DBD | OFF

Response: <CRD>
<function>

Example:
```
:MARK:SLOP?
```
Determine the state of the marker slope function.

:MARKer
[:STATe]

Parameters: <NRf>, <BOOLEAN PROGRAM DATA>
marker number, state

Description: Set a marker on or off.
See [MARKERS] / Set Up Mkrs for more details.

Example:
```
:MARK 4,ON
```
Set marker 4 on.

:MARKer
:STATe?

Parameters: <NRf> [,<BOOLEAN PROGRAM DATA>]
marker number[,state]

Response: <BOOLEAN RESPONSE DATA>
state

Example:
```
:MARK:STAT? 5
```
Determine whether marker 5 is on.

MEASurement SUBSYSTEM

```
MEASurement
        ACTive\?
        DISPlay
                CENTre\?
                CSPan\?
                SPAN\?
                STARt\?
                STOP\?
        LIMit
                NSEGments?
                RESet
                SEGMent\?
                SELect\?
                [STATe]\?
        NMEas\?
        SMOothing
                APERture\?
                [STATe]\?
```

:MEASurement
:ACTive

Parameters: <NRf>
measurement number

Description: Set the active measurement within the active channel.
See 'Display Group Keys for more details.

Example:

```
:MEAS:ACT 2
```

Make measurement 2 active.

:MEASurement
:ACTive?

Parameters: [<NRf>]
[measurement number]

Response: <NR1>
measurement number

Example:

```
:MEAS:ACT?
```

Determine which measurement is active.

:MEASurement
:DISPlay
:CENTer

Parameters: <NRf>
domain centre value

Description: For a scalar channel, this sets the domain centre value (the :SOURce:MODE command is used to specify the domain). It has the same effect as :SOURce:FREQuency:CENTre in the frequency sweep mode.

For a fault location channel in the enhanced mode, this sets the centre value of the displayed distance (see [MEASURE]/Enhance Mode).

For a reflection analyzer channel, this command will produce an error message. The Reflection Analyzer sub-system contains its own commands for specifying the domain range that is displayed.

The instrument will also accept the English spelling, "CENTre"

Example:

```
:MEAS:DISP:CENT 15E9
```

Set the centre frequency to 15 GHz (assuming the domain is frequency)

:MEASurement
:DISPlay
:CENTer?

Parameters: [<NRf>]
[domain centre value]

Response: <NR2>
domain centret value

Example:

```
:MEAS:DISP;CENT?
```

Read the domain centre value.

:MEASurement
:DISPlay
:CSPan

Parameters: <BOOLEAN PROGRAM DATA>
state

Description: For a scalar channel with a frequency sweep, this selects centre / span display (ON) or start / stop display (OFF).

For a fault location channel with enhanced mode turned on, this selects centre / span display (ON) or start / stop display (OFF).

For a reflection analyzer channel, this command will produce an error message. The Reflection Analyzer sub-system contains its own commands for selecting centre / span display.

Example:
```
:REFL:DISP:CSP ON
```
Set display to centre / span.

:MEASurement
:DISPlay
:CSPan?

Parameters: [<BOOLEAN PROGRAM DATA>]
[state]

Parameters: <BOOLEAN RESPONSE DATA>
state

Example:
```
:MEAS:DISP:CSP?
```
Determine if in centre / span or start / stop display mode.

:MEASurement
:DISPlay
:SPAN

Parameters: <NRf>
span value

Description: For a scalar channel, this sets the domain span value (the :SOURce:MODE command is used to specify the domain). It has the same effect as :SOURce:FREQuency:SPAN in the frequency sweep mode.

For a fault location channel in the enhanced mode, this sets the span value of the displayed distance (see [MEASURE]/Enhance Mode).

For a reflection analyzer channel, this command will produce an error message. The Reflection Analyzer sub-system contains its own commands for specifying the domain range that is displayed.

Example:
```
:MEAS:DISP:SPAN 10E9
```
Set the frequency span to 10 GHz (assuming the domain is frequency).

:MEASurement
:DISPlay
:SPAN?

Parameters: [<NRf>]
[span value]

Response: <NR2>
span value

Example:

```
:MEAS:DISP;SPAN?
```

Read the domain span value.

:MEASurement
:DISPlay
:STARt

Parameters: <NRf>
domain start value

Description: For a scalar channel, this sets the domain start value (the :SOURce:MODE command is used to specify the domain). It has the same effect as :SOURce:FREQuency:STARt in the frequency sweep mode.

For a fault location channel in the enhanced mode, this sets the start value of the displayed distance (see [MEASURE]/Enhance Mode).

For a reflection analyzer channel, this command will produce an error message. The Reflection Analyzer sub-system contains its own commands for specifying the domain range that is displayed.

Example:

```
:MEAS:DISP:STAR 10.0E9
```

Set the start frequency to 10 GHz (assuming the domain is frequency).

:MEASurement
:DISPlay
:STARt?

Parameters: [<NRf>]
[domain start value]

Response: <NR2>
domain start value

Example:

```
:MEAS:DISP:STAR?
```

Read the domain start value.

:MEASurement
:DISPlay
:STOP

Parameters: <NRf>
domain stop value

Description: For a scalar channel, this sets the domain stop value (the :SOURce:MODE command is used to specify the domain). It has the same effect as :SOURce:FREQUency:STOP in the frequency sweep mode.

For a fault location channel in the enhanced mode, this sets the stop value of the displayed distance (see [MEASURE]/Enhance Mode).

For a reflection analyzer channel, this command will produce an error message. The Reflection Analyzer sub-system contains its own commands for specifying the domain range that is displayed.

Example:
```
:MEAS:DISP:STOP 26.5E9
```
Set the stop frequency to 26.5 GHz (assuming the domain is frequency).

:MEASurement
:DISPlay
:STOP

Parameters: [<NRf>]
[domain stop value]

Response: <NR2>
domain stop value

Example:
```
:MEAS:DISP:STOP?
```
Read the domain stop value.

:MEASurement
:LIMit
:NSEGments?

Response: <NR1>
number of segments

Description: Read the number of segments in the limit specification assigned to the active measurement.
See [MEASURE] / Lim Checking for more details.

Example:
```
:MEAS:LIM:NSEG?
```
Read number of limit segments.

:MEASurement

:LIMit

:RESet

Description: Reset the limit segment counter in the limit specification to address the first limit segment. This command is used in conjunction with the :MEASurement:LIMit:SEGment command .
See [MEASURE] / Lim Checking for more details.

Example:
```
:MEAS:LIM:RES
```
Prepare specification for reading/writing limit segments.

:MEASurement
:LIMit
SEGMent

Parameters: <NRf>,<NRf>,<CPD>,<NRf>,<NRf>,<NRf>,<NRf>
start domain, stop domain, <limit type>, start upper, start lower, stop upper, stop lower

where <limit type> *is* FLAT | SLOPe | POINt

When <limit type> is FLAT or POINt, only start upper and start lower will be used. (The stop lower and stop upper parameters must be sent to conform with the command syntax, but their values will be ignored)

The domain and limit values are stored as real numbers which are interpreted according to the active channel domain and the active measurement format respectively:

DOMAIN	UNITS
Frequency	Hz
Power	dBm
Voltage	V
Current	A

FORMAT	UNITS
Frequency	Hz
Log Power	dB or dBm
Linear Power	W
Voltage	V
VSWR	Units
Log Mag	dB
Lin Mag	Units
Phase	Degrees
Real	Units
Imaginary	Units
Impetance	Ohms

Description: Write a segment to the selected limit specification.

A limit specification consists of one or more limit segments. To send a specification:

1. Send :MEAS:LIM:RES

This resets a counter that addresses the list of limit segments.

2. Send limit segments using :MEAS:LIM:SEGM

After each :MEAS:LIM:SEGM command is received, the counter increments automatically to address the next free segment.

Note that limit checking may only be applied to swept measurements.
See[MEASURE] / Lim Checking for more details.

Example:

```
:MEAS:LIM:RES; SEGM 2E9, 8E9, FLAT, +0.5, -0.5, 0, 0;
 SEGM 8E9, 14E9, SLOPE, +0.5, -0.5, +1.0, -1.0
```

A limit specification is defined in two segments. Between 2 GHz and 8 GHz flat limit lines are placed at ± 0.5 dB. Between 8 GHz and 14 GHz, a pair of sloped limit lines are introduced, starting at ± 0.5 dB at 8 GHz and expanding to ±1.0 dB at 14 GHz.

:MEASurement
:LIMit
:SEGMent?

Parameters: [<NRf>,<NRf>,<CPD>,<NRf>,<NRf>,<NRf>,<NRf>]
[start domain, stop domain, <limit type>, start upper, start lower, stop upper, stop lower]
where <limit type> *is* FLAT | SLOPe | POINt

When <limit type> is FLAT or POINt, only start upper and start lower will be used. (The stop lower and stop upper parameters must be sent to conform with the command syntax, but their values will be ignored)

The domain and limit values are stored as real numbers which are interpreted according to the active channel domain and the active measurement format respectively:

DOMAIN	UNITS	FORMAT	UNITS
Frequency	Hz	Frequency	Hz
Power	dBm	Log Power	dB or dBm
Voltage	V	Linear Power	W
Current	A	Voltage	V
		VSWR	Units
		Log Mag	dB
		Lin Mag	Units
		Phase	Degrees
		Real	Units
		Imaginary	Units
		Impetance	Ohms

Response: <NR2>,<NR2>,<CRD>,<NR2>,<NR2>,<NR2>,<NR2>
start domain, stop domain, <limit type>, start upper, start lower, stop upper, stop lower

When <limit type> is FLAT or POIN, only start upper and start lower will be valid. (The stop lower and stop upper parameters will be returned in accordance with the response syntax, but their values can be ignored)

Description: Read a limit segment from the currently selected limit store. A limit specification held in a store consists of one or more limit segments. To read a specification:

1. Send :MEAS:LIM:RES
This resets a counter that addresses the list of limit segments within the specification.

2. Send :MEAS:LIM:NSEG?
This returns the number of segments in the limit specification held in the current limit store.

3. Read limit segments using :MEAS:LIM:SEGM?
After each :MEAS:LIM:SEGM? command is received, the counter increments automatically to address the next segment. An error will be generated if an attempt is made to read beyond the last valid segment.
See [MEASURE] / Lim Checking for more details.

Example: `:MEAS:LIM:RES; NSEG?`
Reset the store pointer and read the number of segments.

Example: `:MEAS:LIM:SEGM?`
Read the first limit segment.

:MEASurement
:LIMit
:SELect

Parameters: <NRf>
specification number

Description: Assign a limit specification to the active measurement.
See [MEASURE] / Lim Checking for more details.

Example: `:MEAS:LIM:SEL 4`
Assign limit specification 4.

:MEASurement
:LIMit
:SELect?

Parameters: [<NRf>]
[specification number]

Response: <NR1>
specification number

Example: `:MEAS:LIM:SEL?`
Determine which limit specification is assigned.

:MEASurement
:LIMIT
[:STATe]

Parameters: <BOOLEAN PROGRAM DATA>
state

Description: Enable/disable limit checking.
See [MEASURE] / Lim Checking for more details.

Example: `:MEAS:LIM ON`
Switch limit checking on.

:MEASurement
:LIMit
:STATe?

Parameters: [<BOOLEAN PROGRAM DATA>]
[state]

Response: <BOOLEAN RESPONSE DATA>
state

Example: `:MEAS:LIM:STAT?`
Determine whether limit checking is enabled.

:MEASurement
:NMEas

Parameters: <NRf>
number of measurements

Description: Set the number of measurements displayed within the active channel.
See 'Channels and Measurements' and 'Display Group Keys' for more details.

Example: `:MEAS:NME 2`
Display two measurements.

:MEASurement
:NMEas?

Parameters: [<NRf>]
[number of measurements]

Response: <NR1>
number of measurements

Example: `:MEAS:NME?`
Determine how many measurements are displayed within the active channel.

:MEASurement
:SMOothing
:APERture

Parameters: <NRf>
aperture (%)

Description: Set the smoothing aperture.
See [MEASURE] / Smoothing for more details.

Example: `:MEAS:SMO:APER 10`
Set the smoothing aperture to 10%.

:MEASurement
:SMOothing
:APERture?

Parameters: [<NRf>]
[aperture (%)]

Response: <NR2>
aperture (%)

Example: `:MEAS:SMO:APER?`
Read the smoothing aperture.

:MEASurement
:SMOothing
[:STATe]

Parameters: <BOOLEAN PROGRAM DATA>
state

Description: Enable/disable smoothing.
See [MEASURE] / Smoothing for more details.

Example: `:MEAS:SMO ON`
Switch smoothing on.

:MEASurement
:SMOothing
:STATe?

Parameters: [<BOOLEAN PROGRAM DATA>]
[state]

Response: <BOOLEAN RESPONSE DATA>
state

Example: `:MEAS:SMO:STAT?`
Determine whether smoothing is enabled.

REFLect SUBSYSTEM

```
REFLect
    AVERaging
        NUMBer\?
        RESTart
        [STATe]\?
    CALibration
        ABORt
        CABLe\?
        DATA
            CABLe?
            CONNector?
            DATE?
            MAXimum?
            SEX?
            [STORe]\?
            TIME?
            TITLe?
            TYPE?
        KIT
            CONNector\?
            FIXed
                BFRequency\?
                OLENgth\?
                [SNUMber]\?
            IDENtity\?
            LOAD
                OLENgth\?
                [SNUMber]\?
            MAXimum?
            OFFSET1
                MAXFrequency\?
                MINFrequency\?
                OLENgth\?
                [SNUMBer]\?
            OFFSET2
                MAXFrequency\?
                MINFrequency\?
                OLENgth\?
                [SNUMBer]\?
            OPEN
                FCOeffs\?
                OLENgth\?
                [SNUMBer]\?
            [SELect]\?
            SEX\?
            SHORt
```

```
                        OLENgth\?
                        [SNUMBer]\?
                SLIDing
                        BFRequency\?
                        [SNUMBer]\?
                STORe\?
                WGCutoff\?
        MEASure
                FLOad
                OPEN
                OSHort
                SHORt
                SLOad
                TPORt
        PROCess
        RESTore
        SEX\?
        TYPE\?
        VALid\?
CIMPedance\?
DATA
        [ASCii]?
        BINary?
DCONstant\?
DELay
        LENGth\?
        [TIME]\?
DETector
        LCORrection\?
        TCONtrol\?
DISPlay
        CENTer\?
        CSPan\?
        FORMat\?
        SPAN\?
        STARt\?
        STOP\?
MEASure
        DEFinition?
        DIFFerence
        MEMory
                [ONLY]
                SETTings
        RATio
        SAVE
        S11
MEDium\?
POFFset\?
```

```
RVELocity\?
SCALing
        AUToscale
        DIVision\?
        POSition\?
        RLEVel\?
TDOMain
        DISPlay\?
        EMODe\?
        GATing
                CENTer\?
                COUPling\?
                SHAPe\?
                SPAN\?
                STARt\?
                [STATe]\?
                STOP\?
                TYPE\?
        RCOefficient\?
        RESPonse\?
        [STATe]\?
        TRANsform
        UNITs\?
        WINDow
                SHAPe\?
                USER\?
ZERO
```

:REFLect
:AVERaging
:NUMBer

Parameters: <NRf>
averaging number

Description: Set the averaging number for reflection analyzer measurements. This is applied on the sweep after the current sweep.
See [MEASURE] / Averaging (Scalar Channel) for more details.

Example: `:REFL:AVER:NUMB 16`
Set an averaging number of 16.

:REFLect
:AVERaging
:NUMBer?

Parameters: [<NRf>]
[averaging number]

Response: <NR1>
averaging number

Example: `:REFL:AVER:NUMB?`
Read the averaging number.

:REFLect
:AVERaging
:RESTart

Description: Restart reflection analyzer averaging. This is applied on the sweep after the current sweep.
See [MEASURE] / Measure (Scalar Channel) for more details.

Example: `:REFL:AVER:REST`
Restart averaging.

:REFLect
:AVERaging
[:STATe]

Parameters: <BOOLEAN PROGRAM DATA>
state

Description: Turn reflection analyzer averaging on and off.
See [MEASURE] / Averaging (Scalar Channel) for more details.

Example: `:REFL:AVER ON`
Enable averaging

:REFLect
:AVERaging
:STATe?

Parameters: [<BOOLEAN PROGRAM DATA>]
[state]

Response: <BOOLEAN RESPONSE DATA>
state

Example: `:REFL:AVER:STAT?`

REFLect
:CALibration
:ABORt

Description: Abort the current reflection analyzer calibration.
See [CAL] / Reflect Cal for more details.

Example: `:REFL:CAL:ABOR`
Abort calibration.

:REFLect
:CALibration
:CABLe

Parameters: <BOOLEAN PROGRAM DATA>
state

Description: Turn calibration at the end of a cable on and off.
See [CAL] / Set Up Cal for more details.

Example: `:REFL:CAL:CABL ON`
Enable calibration at the end of a cable.

:REFLect
:CALibration
:CABLe?

Parameters: [<BOOLEAN PROGRAM DATA>]
[state]

Response: <BOOLEAN RESPONSE DATA>
state

Example: `:REFL:CAL:CABL?`
Determine whether calibration at the end of a cable is enabled.

:REFLect
:CALibration
:DATA
:CABLe?

Parameters: <NRf>
store number

Description: Determines whether or not the cal associated with the calibration store was done at the end of the cable.

Response: <BOOLEAN RESPONSE DATA>
calibrated at end of cable

Example: `:REFL:CAL:DATA:CABL? 1`
Determine whether the cal associated with calibration store 1 was done at the end of a cable.

:REFLect
:CALibration
:DATA
:CONNector?

Parameters: <NRf>
store number

Description: Read the connector type associated with the calibration store:

C35 is a 3.5 mm connector
CN is an N-type connector
C7 is a 7 mm connector
USER is the user defined connector
WG is the waveguide section

Response: <CRD>
connector type
where connector type *is* C35 | CN | C7 | USER | WG

Example: `:REFL:CAL:DATA:CONN? 1`
Determine the connector type associated with calibration store 1.

:REFLect
:CALibration
:DATA
:DATe?

Parameters: <NRf>
store number

Description: Read the date associated with the calibration store.

Response: <NR1>, <NR1>, <NR1>
year, month, day

Example: `:REFL:CAL:DATA:DATE? 1`
Read the date associated with calibration store 1.

Example Response: 1993, 8, 20
The date is 20th August 1993

:REFLect
:CALibration
:DATA
:MAXimum?

Response: <NR1>
number of stores

Description: Read the number of reflection analyzer calibration stores available, including any located on a memory card.

Example: `:REFL:CAL:DATA:MAX?`
Determine how many calibration stores are available.

:REFLect
:CALibration
:DATA
:SEX?

Parameters: <NRf>
store number

Description: Read the connector sex associated with the calibration store.

Response: <CRD>
connector sex
where connector sex *is* MALE | FEM | NEUT
NEUT signifies an unsexed connector.

Example: `:REFL:CAL:DATA:SEX? 1`
Determine the connector sex associated with calibration store 1.

:REFLect
:CALibration
:DATA
[:STORe]

Parameters: <NRf>,<ARBITRARY BLOCK PROGRAM DATA>
store numbers, store contents

Description: Send data to the processed calibration data store.

Example:

```
:REFL:CAL:DATA 1, #...etc
```

Send processed calibration data to store 1. (Only first byte of data shown.)

:REFLect
:CALibration:
:DATA?
[:STORe]?

Parameters: <NRf>[,<ARBITRARY BLOCK PROGRAM DATA>]
store number [,store contents]

Response: <INDEFINITE LENGTH ARBITRARY BLOCK RESPONSE DATA>
store contents

Example:

```
:REFL:CAL:DATA? 2
```

Read the contents of processed calibration store 2.

:REFLect
:CALibration
:DATA
:TIME?

Parameters: <NRf>
store number

Description: Read the time associated with the calibration store.

Response: <NR1>, <NR1>, <NR1>
hour, minute, second

Example:

```
:REFL:CAL:DATA:TIME? 1
```

Read the time associated with calibration store 1.

Example Response: 14, 30, 0
The time is 2:30 pm.

ect
:CALibration
:DATA
:TITLe?

Parameters: <NRf>
store number

Description: Read the title associated with the calibration store.

Response: <STRING RESPONSE DATA>
title

Example: `:REFL:CAL:DATA:TITL? 1`
Read the title associated with calibration store 1.

:REFLect
:CALibration
:DATA
:TYPE?

Parameters: <NRf>
store number

Description: Read the connector type associated with the calibration store:

CSOF is coax short-open-fixed load calibration
CSOS is coax short-open-siding load calibration
WSOL is waveguide short-offset short- load calibration
WS12 is waveguide short-offset short 1-offset short 2 calibration

Response: <CRD>
calibration type
where calibration type *is* CSOF | CSOS | WSOL | WS12

Example: `:REFL:CAL:DATA:TYPE? 1`
Read the calibration type associated with calibration store 1.

:REFLect
:CALibration
:KIT
:CONNector

Parameters: <CPD>
<connector type>
where <connector type> is C35 | CN | C7 | USR | WG

Description: Specify the type of connector used in the cal kit.

C35 is a 3.5 mm connector
CN is a N-type connector
C7 is a 7 mm connector
USR is the user defined connector
WG is the waveguide section

See [CAL] / Connector for more details.

Example:

```
:REFL:CAL:KIT:CONN C7
```

Select 7 mm connector type.

:REFLect
:CALibration
:KIT
:CONNector?

Parameters: [<CPD>]
[<connector type>]

Response: <CRD>
<connector type>

Example:

```
:REFL:CAL:KIT:CONN?
```

Determine the connector type.

:REFLect
:CALibration
:KIT
:FIXed
:BFRequency

Parameters: <NRf>
break frequency (Hz)

Description: Set the cal kit store fixed load break frequency.
See [CAL] / Edit Cal Kit for more details.

Example:
```
:REFL:CAL:KIT:FIX:BFR 12.5E9
```
Set the break frequency to 12.5 GHz

:REFLect
:CALibration
:KIT
:FIXed
:BFRequency?

Parameters: [<NRf>]
[break frequency (Hz)]

Response: <NR2>
break frequency (Hz)

Example:
```
:REFL:CAL:KIT:FIX:BFR?
```
Read the break frequency.

:REFLect
:CALibration
:KIT
:FIXed
:OLENgth

Parameters: <NRf>
offset length (metres)

Description: Set the cal kit store fixed load offset length.
See [CAL] / Edit Cal Kit for more details.

Example:
```
:REFL:CAL:KIT:FIX:OLEN 1.5E-3
```
Set the offset length to 1.5 mm.

:REFLect
:CALibration
:KIT
:FIXed
:OLENgth?

Parameters: [<NRf>]
[offset length (metres)]

Response: <NR2>
offset length (metres)

Example:
```
:REFL:CAL:KIT:FIX:OLEN?
```
Read the value of offset length.

:REFLect
:CALibration
:KIT
:FIXed
[:SNUMber]

Parameters: <STRING PROGRAM DATA>
serial number

Description: Set the cal kit store fixed load serial number.
See [CAL] / Edit Cal Kit for more details.

Example:
```
:REFL:CAL:KIT:FIX:SNUM "123456"
```
Specify the fixed load serial number.

:REFLect
:CALibration
:KIT
:FIXed
[:SNUMber]?

Parameters: [<STRING PROGRAM DATA>]
[serial number]

Response: <STRING RESPONSE DATA>
serial number

Example:
```
:REFL:CAL:KIT:FIX:SNUM?
```
Read the fixed load serial number.

:REFLect
:CALibration
:KIT
:IDENtity

Parameters: <STRING PROGRAM DATA>
cal kit identity string

Description: Send the cal kit identity.
See [CAL] / Reflect Cal for more details.

Example:
```
:REFL:CAL:KIT:IDEN "Marconi"
```
Send the cal kit identity: Marconi

:REFLect
:CALibration
:KIT
:IDENtity?

Parameters: [<STRING PROGRAM DATA>]
[cal kit identity string]

Response: <STRING RESPONSE DATA>
cal kit identity string

Example:
```
:REFL:CAL:KIT:IDEN?
```
Read the cal kit identity.

:REFLect
:CALibration
:KIT
:LOAD
:OLENgth

Parameters: <NRf>
offset length (metres)

Description: Set the cal kit store load offset length.
See [CAL] / Edit Cal Kit for more details.

Example:
```
:REFL:CAL:KIT:LOAD:OLEN 1.5E-3
```
Set the offset length to 1.5 mm.

:REFLect
:CALibration
:KIT
:LOAD
:OLENgth?

Parameters: [<NRf>]
[offset length (metres)]

Response: <NR2>
offset length (metres)

Example:
```
:REFL:CAL:KIT:LOAD:OLEN?
```
Read the value of offset length.

:REFLect
:CALibration
:KIT
:LOAD
[:SNUMber]

Parameters: <STRING PROGRAM DATA>
serial number

Description: Set the cal kit store load serial number.
See [CAL] / Edit Cal Kit for more details.

Example:
```
:REFL:CAL:KIT:LOAD:SNUM "123456"
```
Specify the load serial number.

:REFLect
:CALibration
:KIT
:LOAD
[:SNUMber]?

Parameters: [<STRING PROGRAM DATA>]
[serial number]

Response: <STRING RESPONSE DATA>
serial number

Example:
```
:REFL:CAL:KIT:LOAD:SNUM?
```
Read the load serial number.

:REFLect
:CALibration
:KIT
:MAXimum?

Response: <NR1>
Maximum cal kit store number.

Description: Read the maximum cal kit store number.
See [CAL] / Edit Cal Kit for more details.

Example: `:REFL:CAL:KIT:MAX?`
Read the maximum cal kit store number.

:REFLect
:CALibration
:KIT
:OFFSET1
:MAXFrequency

Parameters: <NRf>
maximum frequency (Hz)

Description: Set the cal kit store offset load 1 maximum frequency. The instrument will also accept "OFF1" instead of "OFFSET1".
See [CAL] / Edit Cal Kit for more details.

Example: `:REFL:CAL:KIT:OFF1:MAXF 23.4E9`
Set the max frequency to 23.4 GHz

:REFLect
:CALibration
:KIT
:OFFSET1
:MAXFrequency?

Parameters: [<NRf>]
[maximum frequency (Hz)]

Response: <NR2>
maximum frequency (Hz)

Example: `:REFL:CAL:KIT:OFF1:MAXF?`
Read the maximum frequency.

:REFLect
:CALibration
:KIT
:OFFSET1
:MINFrequency

Parameters: <NRf>
minimum frequency (Hz)

Description: Set the cal kit store offset load 1 minimum frequency. The instrument will also accept "OFF1" instead of "OFFSET1".
See [CAL] / Edit Cal Kit for more details.

Example:
```
:REFL:CAL:KIT:OFF1:MINF 12.3E9
```
Set the min frequency to 12.3 GHz

:REFLect
:CALibration
:KIT
:OFFSET1
:MINFrequency?

Parameters: [<NRf>]
[minimum frequency (Hz)]

Response: <NR2>
minimum frequency (Hz)

Example:
```
:REFL:CAL:KIT:OFF1:MINF?
```
Read the min frequency.

:REFLect
:CALibration
:KIT
:OFFSET1
:OLENgth

Parameters: <NRf>
offset length (metres)

Description: Set the cal kit store load 1 offset length. The instrument will also accept "OFF1" instead of "OFFSET1".
See [CAL] / Edit Cal Kit for more details.

Example: `:REFL:CAL:KIT:OFF1:OLEN 1.5E-3`
Set the offset length to 1.5 mm.

:REFLect
:CALibration
:KIT
:OFFSET1
:OLENgth?

Parameters: [<NRf>]
[offset length (metres)]

Response: <NR2>
offset length (metres)

Example: `:REFL:CAL:KIT:OFF1:OLEN?`
Read the value of offset length.

:REFLect
:CALibration
:KIT
:OFFSET1
[:SNUMber]

Parameters: <STRING PROGRAM DATA>
serial number

Description: Set the cal kit store offset load 1 serial number.
See CAL] / Edit Cal Kit for more details.

Example: `:REFL:CAL:KIT:OFF1:SNUM "123456"`
Set the offset load 1 serial number.

:REFLect
:CALibration
:KIT
:OFFSET1
[:SNUMber]?

Parameters: [<STRING PROGRAM DATA>]
[serial number]

Response: <STRING RESPONSE DATA>
serial number

Example: `:REFL:CAL:KIT:OFF1:SNUM?`
Read the offset load 1 serial number.

:REFLect
:CALibration
:KIT
:OFFSET2
:MAXFrequency

Parameters: <NRf>
maximum frequency (Hz)

Description: Set the cal kit store offset load 2 maximum frequency. The instrument will also accept "OFF2" instead of "OFFSET2".
See CAL] / Edit Cal Kit for more details.

Example:
```
:REFL:CAL:KIT:OFF2:MAXF 23.4E9
```
Set the max frequency to 23.4 GHz

:REFLect
:CALibration
:KIT
:OFFSET2
:MAXFrequency?

Parameters: [<NRf>]
[maximum frequency (Hz)]

Response: <NR2>
maximum frequency (Hz)

Example:
```
:REFL:CAL:KIT:OFF2:MAXF?
```
Read the maximum frequency.

:REFLect
:CALibration
:KIT
:OFFSET2
:MINFrequency

Parameters: <NRf>
minimum frequency (Hz)

Description: Set the cal kit store offset load 2 minimum frequency. The instrument will also accept "OFF2" instead of "OFFSET2".
See [CAL] / Edit Cal Kit for more details.

Example:
```
:REFL:CAL:KIT:OFF2:MINF 12.34E9
```
Set the min frequency to 12.3 GHz

:REFLect
:CALibration
:KIT
:OFFSET2
:MINFrequency?

Parameters: [<NRf>]
[minimum frequency (Hz)]

Response: <NR2>
minimum frequency (Hz)

Example:
```
:REFL:CAL:KIT:OFF2:MINF?
```
Read the min frequency.

:REFLect :CALibration :KIT :OFFSET2 :OLENgth

Parameters: <NRf>
offset length (metres)

Description: Set the cal kit store load 2 offset length. The instrument will also accept "OFF2" instead of "OFFSET2".
See [CAL] / Edit Cal Kit for more details.

Example:
```
:REFL:CAL:KIT:OFF2:OLEN 1.5E-3
```
Set the offset length to 1.5 mm.

:REFLect :CALibration :KIT :OFFSET2 :OLENgth?

Parameters: [<NRf>]
[offset length (metres)]

Response: <NR2>
offset length (metres)

Example:
```
:REFL:CAL:KIT:OFF2:OLEN?
```
Read the value of offset length.

:REFLect :CALibration :KIT :OFFSET2 [:SNUMber]

Parameters: <STRING PROGRAM DATA>
serial number

Description: Set the cal kit store offset load 2 serial number.
See [CAL] / Edit Cal Kit for more details.

Example:
```
:REFL:CAL:KIT:OFF2:SNUM "123456"
```
Set the offset load 2 serial number.

:REFLect :CALibration :KIT :OFFSET2 [:SNUMber]?

Parameters: [<STRING PROGRAM DATA>]
[serial number]

Response: <STRING RESPONSE DATA>
serial number

Example:
```
:REFL:CAL:KIT:OFF2:SNUM?
```
Read the offset load 2 serial number.

:REFLect :CALibration :KIT :OPEN :FCOeffs

Parameters: <NRf>,<NRf>,<NRf>,<NRf>
coeff 1, coeff 2, coeff 3, coeff 4

Description: Set the cal kit store open circuit fringing coefficients.
See [CAL] / Edit Cal Kit for more details.

Example:

```
:REFL:CAL:KIT:OPEN:FCO 1.234E-14
8.123E-34, 5.432E-16, 4.321E-20
```

Set the fringing coefficients.

:REFLect :CALibration :KIT :OPEN :FCOeffs?

Parameters: [<NRf>,<NRf>,<NRf>,<NRf>]
[coeff 1, coeff 2, coeff 3, coeff 4]

Response: <NR2>,<NR2>,<NR2>,<NR2>
coeff 1, coeff 2, coeff 3, coeff 4

Example:

```
:REFL:CAL:KIT:OPEN:FCO?
```

Read the fringing coefficients.

:REFLect :CALibration :KIT :OPEN :OLENgth

Parameters: <NRf>
offset length (metres)

Description: Set the cal kit store open circuit offset length.
See [CAL] / Edit Cal Kit for more details.

Example:

```
:REFL:CAL:KIT:OPEN:OLEN 1.5E-3
```

Set the offset length to 1.5 mm.

:REFLect :CALibration :KIT :OPEN :OLENgth?

Parameters: [<NRf>]
[offset length (metres)]

Response: <NR2>
offset length (metres)

Example:

```
:REFL:CAL:KIT:OPEN:OLEN?
```

Read the value of offset length.

:REFLect :CALibration :KIT :OPEN [:SNUMber]

Parameters: <STRING PROGRAM DATA>
serial number

Description: Set the cal kit store open circuit serial number.
See [CAL] / Edit Cal Kit for more details.

Example: `:REFL:CAL:KIT:OPEN:SNUM "123456"`
Set the open circuit serial number.

:REFLect :CALibration :KIT :OPEN [:SNUMber]?

Parameters: [<STRING PROGRAM DATA>]
[serial number]

Response: <STRING RESPONSE DATA>
serial number

Example: `:REFL:CAL:KIT:OPEN:SNUM?`
Read the open circuit serial number.

:REFLect :CALibration :KIT [:SELect]

Parameters: <NRf>
cal kit number

Description: Select the cal kit to use for the measurement.
See [CAL] / Reflect Cal for more details.

Example: `:REFL:CAL:KIT 1`
Select cal kit 1.

:REFLect :CALibration :KIT :SELect?

Parameters: [<NRf>]
[cal kit number]

Response: <NR1>
cal kit number

Example: `:REFL:CAL:KIT:SEL?`
Determine the cal kit number currently in use.

:REFLect
:CALibration
:KIT
:SEX

Parameters: <CPD>
<connector sex>
where <connector sex> is MALE | FEMale

Description: Set the cal kit connector sex.
See [CAL] / Reflect Cal for more details.

Example: `:REFL:CAL:KIT:SEX MALE`
Select male type connector.

:REFLect
:CALibration
:KIT
:SEX?

Parameters: [<CPD>]
[<connector sex>]
where <connector sex> is MALE | FEMale

Response: <CRD>
<connector sex>
where <connector sex> is MALE | FEM | NEUT
NEUT signifies an unsexed connector

Example: `:REFL:CAL:KIT:SEX?`
Determine the connector sex of the calibration kit.

:REFLect
:CALibration
:KIT
:SHORt
:OLENgth

Parameters: <NRf>
offset length (metres)

Description: Set the cal kit store short circuit offset length.
See [CAL] / Edit Cal Kit for more details.

Example: `:REFL:CAL:KIT:SHOR:OLEN 1.5E-3`
Set the short circuit offset length to 1.5 mm.

:REFLect
:CALibration
:KIT
:SHORt
:OLENgth?

Parameters: [<NRf>]
[offset length (metres)]

Response: <NR2>
offset length (metres)

Example: `:REFL:CAL:KIT:SHOR:OLEN?`
Read the value of offset length.

:REFLect
:CALibration
:KIT
:SHORt
[:SNUMber]

Parameters: <STRING PROGRAM DATA>
serial number

Description: Set the cal kit store short circuit serial number.
See [CAL] / Edit Cal Kit for more details.

Example: `:REFL:CAL:KIT:SHOR:SNUM "123456"`
Specify the short circuit serial number.

:REFLect
:CALibration
:KIT
:SHORt
[:SNUMber]?

Parameters: [<STRING PROGRAM DATA>]
[serial number]

Response: <STRING RESPONSE DATA>
serial number

Example: `:REFL:CAL:KIT:SHOR:SNUM?`
Read the short circuit serial number.

:REFLect
:CALibration
:KIT
:SLIDing
:BFRequency

Parameters: <NRf>
break frequency (Hz)

Description: Set the cal kit store sliding load break frequency.
See [CAL] / Edit Cal Kit for more details.

Example: `:REFL:CAL:KIT:SLID:BFR 23.4E9`
Set the break frequency to 23.4 GHz.

:REFLect
:CALibration
:KIT
:SLIDing
:BFRequency?

Parameters: [<NRf>]
[break frequency (Hz)]

Response: <NR2>
break frequency (Hz)

Example: `:REFL:CAL:KIT:SLID:BFR?`
Read the break frequency.

:REFLect :CALibration :KIT :SLIDing [:SNUMber]

Parameters: <STRING PROGRAM DATA>
serial number

Description: Set the cal kit store sliding load serial number.
See [CAL] / Edit Cal Kit for more details.

Example: `:REFL:CAL:KIT:SLID:SNUM "123456"`
Specify the sliding load serial number.

:REFLect :CALibration :KIT :SLIDing [:SNUMber]?

Parameters: [<STRING PROGRAM DATA>]
[serial number]

Response: <STRING RESPONSE DATA>
serial number

Example: `:REFL:CAL:KIT:SLID:SNUM?`
Read the sliding load serial number.

:REFLect :CALibration :KIT :STORe

Parameters: <NRf>,<ARBITRARY BLOCK PROGRAM DATA>
cal kit store number, store contents

Description: Send data to a cal kit store.

Example: `:REFL:CAL:KIT:STOR 2, #...etc`
Send data to cal kit store 2. (Only first byte of data shown.)

:REFLect :CALibration: :KIT :STORe?

Parameters: <NRf>[,<ARBITRARY BLOCK PROGRAM DATA>]
cal kit store number [,store contents]

Response: <INDEFINITE LENGTH ARBITRARY BLOCK RESPONSE DATA>
store contents

Example: `:REFL:CAL:KIT:STOR? 4`
Read the contents of cal kit store 4.

:REFLect
:CALibration
:KIT
:WGCutoff

Parameters: <NRf>
cutoff frequency (Hz)

Description: Set the cal kit waveguide cutoff frequency.
See [CAL] / Edit Cal Kit for more details.

Example:
```
:REFL:CAL:KIT:WGC 2.34E9
```
Set the cutoff frequency to 2.34 GHz.

:REFLect
:CALibration
:KIT
:WGCutoff?

Parameters: [<NRf>]
[cutoff frequency (Hz)]

Response: <NR2>
cutoff frequency (Hz)

Example:
```
:REFL:CAL:KIT:WGC?
```
Read the cutoff frequency.

:REFLect
:CALibration
:MEASure
:FLOad

Description: Measure a fixed load.
See [CAL] / Reflect Cal for more details.

Example:
```
:REFL:CAL:MEAS:FLO
```
Perform a fixed load measurement.

:REFLect
:CALibration
:MEASure
:OPEN

Description: Measure an open circuit.
See [CAL] / Reflect Cal for more details.

Example:

```
:REFL:CAL:MEAS:OPEN
```

Perform an open circuit measurement.

:REFLect
:CALibration
:MEASure
:OSHort

Parameters: <NRf>
offset short number

Description: Measure an offset short.
See [CAL] / Reflect Cal for more details.

Example:

```
:REFL:CAL:MEAS:OSH 1
```

Perform a measurement on offset short number 1.

:REFLect
:CALibration
:MEASure
:SHORt

Description: Measure a short circuit.
See [CAL] / Reflect Cal for more details.

Example:

```
:REFL:CAL:MEAS:SHOR
```

Perform a short circuit measurement.

:REFLect
:CALibration
:MEASure
:SLOad

Parameters: <NRf>
load position

Description: Measure a sliding load.
See [CAL] / Reflect Cal for more details.

Example:

```
:REFL:CAL:MEAS:SLO 2
```

Perform a sliding load measurement, with the load set to position 2.

:REFLect
:CALibration
:MEASure
:TPORt

Description: Measure a short circuit at the test port (for calibration at the end of a cable).
See [CAL] / Reflect Cal for more details.

Example:

```
:REFL:CAL:MEAS:TPOR
```

Measure a short circuit at the test port.

:REFLect
:CALibration
:PROCess

Parameters: <NRf>,<STRING DATA>
store number, calibration title

Description: Compute the calibration from the measured standards, store it in the specified cal store, and give it the title specified.
See [CAL] / Reflect Cal for more details.

Example:

```
:REFL:CAL:PROC 4, "Cal for low pass"
```

Perform the calibration and store it in cal store 4 with the title: Cal for low pass.

:REFLect
:CALibration
:RESTore

Parameters: <NRf>
store number

Description: Restore the calibration and calibration conditions from the selected calibration store.
See [CAL] / Cal for more details.

Example: `:REFL:CAL:REST 2`
Re-establish the calibration and cal conditions from calibration store 2.

:REFLect
:CALibration
:SEX

Parameters: <CPD>
<connector sex>
where <connector sex> *is* MALE | FEMale

Description: Set the sex of the pieces to be used for the calibration.
See [CAL] / Reflect Cal for more details.

Example: `:REFL:CAL:SEX MALE`
Select male type pieces.

:REFLect
:CALibration
:SEX?

Parameters: [<CPD>]
[<connector sex>]
where <connector sex> *is* MALE | FEMale

Response: <CRD>
<connector sex>
where <connector sex> *is* MALE | FEMale | NEUT
NEUT signifies an unsexed connector.

Example: `:REFL:CAL:SEX?`
Determine the sex of the pieces used for the calibration.

:REFLect
:CALibration
:TYPE

Parameters: <CPD>
<calibration type>
where <calibration type> is CSOF | CSOS | WSOL | WS12 | WSOS

Description: Select the calibration type.

CSOF is coax short-open-fixed load calibration
CSOS is coax short-open-sliding load calibration
WSOL is waveguide short-offset short load calibration
WS12 is waveguide short-offset short 1-offset short 2 calibration
WSOS is waveguide short - offset short - sliding load calibration

See [CAL] / Reflect Cal for more details.

Example:

```
:REFL:CAL:TYPE CSOF
```

Select CSOF calibration type.

:REFLect
:CALibration
:TYPE?

Parameters: [<CPD>]
[<calibration type>]
where <calibration type> is CSOF | CSOS | WSOL | WS12 | WSOS

Response: <CRD>
<calibration type>

Example:

```
:REFL:CAL:TYPE?
```

Determine the calibration type that has been selected.

:REFLect
:CALibration
:VALid?

Response: [<BOOLEAN RESPONSE DATA>]
<state>

Description: Returns the current calibration status for the active channel.

Example: `:REFL:CAL:VAL?`
Check for valid calibration for channel

:REFLect
:CIMPedance

Parameters: <NR2>
characteristic impedance (ohms)

Description: Set the characteristic impedance for the medium.
See [MEASURE] / Character Dev for more details.

Example: `:REFL:CIMP 75`
Set the characteristic impedance to 75 ohms.

:REFLect
:CIMPedance?

Parameters: [<NRf>]
[characteristic impedance (ohms)]

Response: <NR2>
characteristic impedance (ohms)

Example: `:REFL:CIMP?`
Read the characteristic impedance.

:REFLect
:DATA
[:ASCii]?

Response: <NR2>,<NR2> . . . <NR2>,<NR2>
response at point 0, response at point 1 . . . response at point n-1

Description: Read reflection analyzer data in ASCII format. Units of data returned depends on measurement format.

Each trace point is sent as a pair of numbers, real component first, in human readable form.

Example: `:REFL:DATA?`
Read reflection analyzer measurement data.

:REFLect
:DATA
:BINary?

Response: <INDEFINITE LENGTH ARBITRARY BLOCK RESPONSE DATA>
<measurement data>

Description: Read reflection analyzer measurement data in binary format . Binary format allows faster transfer of data.

Each trace point is sent as a pair of numbers, real component first, in machine readable form. Each number consist of a 4-bytes, and has the same format as the measurement data for a scalar channel (see :SCALar[:DATA]:BINary?)

Example: `:REFL:DATA:BIN?`
Read reflection analyzer measurement data.

:REFLect
:DCONstant

Parameters: <NRf>
dielectric constant

Description: Set the dielectric constant for the medium.
See [MEASURE] / Character Dev for more details.

Example: `:REFL:DCON 2.5`
Set dielectric constant to 2.5.

:REFLect
:DCONstant?

Parameters: [<NRf>]
[dielectric constant]

Response: <NR2>
dielectric constant

Example: `:REFL:DCON?`
Read the dielectric constant.

:REFLect
:DELay
:LENGth

Parameters: <NRf>
electrical delay length (metres)

Description: Set the electrical delay length for the medium.
See [MEASURE] / Elect Delay for more details.

Example: `:REFL:DEL:LENG 1.5`
Set an electrical delay length of 1.5 m.

:REFLect
:DELay
:LENGth?

Parameters: [<NRf>]
[electrical delay length (metres)]

Response: <NR2>
electrical delay length (metres)

Example: `:REFL:DEL:LENG?`
Read the electrical delay length.

:REFLect
:DELay
[:TIME]

Parameters: <NRf>
electrical delay time (seconds)

Description: Set the electrical delay time for the medium.
See [MEASURE] / Elect Delay for more details.

Example: `:REFL:DEL:10E-9`
Set an electrical delay time of 10 ns.

:REFLect
:DELay
:TIME?

Parameters: [<NRf>]
[electrical time delay (seconds)]

Response: <NR2>
electrical delay time (seconds)

Example: `:REFL:DEL:TIME?`
Read the electrical delay time.

:REFLect
:DETector
:LCORrection

Parameters: <BOOLEAN PROGRAM DATA>
state

Description: Set the detector linearity correction on or off.
This command can only be used when the instrument is in the data acquisition service mode.

Example:
```
:REFL:DET:LCOR ON
```
Enable detector linearity correction.

:REFLect
:DETector
:LCORrection?

Parameters: [<BOOLEAN PROGRAM DATA>]
[state]

Response: <BOOLEAN RESPONSE DATA>
state

Example:
```
:REFL:DET:LCOR?
```
Determine whether detector linearity correction is enabled.

:REFLect
:DETector
:TCONtrol

Parameters: <BOOLEAN PROGRAM DATA>
state

Description: Set the detector temperature control on or off.
This command can only be used when the instrument is in the data acquisition service mode.

Example:
```
:REFL:DET:TCON ON
```
Enable detector temperature control.

:REFLect
:DETector
:TCONtrol?

Parameters: [<BOOLEAN PROGRAM DATA>]
[state]

Response: <BOOLEAN RESPONSE DATA>
state

Example:
```
:REFL:DET:TCON?
```
Determine whether detector temperature control is enabled.

:REFLect
:DISPlay
:CENTer

Parameters: <NRf>
centre value (Hz, dBm, V, A, m, s)

Description: Set display centre value for the reflection analyzer. The instrument will also accept the English spelling, "CENTre".
See [MEASURE] / Display Zoom for more details.

Example: `:REFL:DISP:CENT 12.4E9`
Set centre frequency to 12.4 GHz (assuming domain is frequency).

:REFLect
:DISPlay
:CENTer?

Parameters: [<NRf>]
[centre value (Hz, dBm, V, A, m, s)]

Response: <NR2>
centre value (Hz, dBm, V, A, m, s)

Example: `:REFL:DISP:CENT?`
Read display centre value.

:REFLect
:DISPlay
:CSPan

Parameters: <BOOLEAN PROGRAM DATA>
state

Description: Select centre / span display (ON) or start / stop display (OFF).
See [MEASURE] / Display Zoom for more details.

Example: `:REFL:DISP:CSP ON`
Set display to centre / span.

:REFLect
:DISPlay
:CSPan?

Parameters: [<BOOLEAN PROGRAM DATA>]
[state]

Parameters: <BOOLEAN RESPONSE DATA>
state

Example: `:REFL:DISP:CSP?`
Determine if in centre / span or start / stop display mode.

:REFLect
:DISPlay
:FORMat

Parameters: <CPD>
<format>
where <format> is LIN | LOG | PHASe | VSWR | RE | IM | IMPedance |
PLIN | PLOG | REIM | SMITh |INVSmith

Description: Set the reflection analyzer display format:

LIN is lin magnitude
LOG is log magnitude
PHASe is phase angle
RE is the real component
IM is the imaginary component
IMPedance is impedance (step response, time domain only)
PLIN is polar linear
PLOG is polar log
REIM is polar real/imaginary
SMITh is smith
INVSmith is inverted smith

See [FORMAT] key for more details.

Example:
```
:REFL:DISP:FORM LOG
```
Select log magnitude format.

:REFLect
:DISPlay
:FORMat?

Parameters: [<CPD>]
[<format>]
where <format> is LIN | LOG | PHAS | VSWR | RE | IM | IMP |
PLIN | PLOG | REIM | SMIT | INVS

Response: <CRD>
<format>

Example:
```
:REFL:DISP:FORM?
```
Determine the display format currently in use.

:REFLect
:DISPlay
:SPAN

Parameters: <NRf>
span value (Hz, dBm, V, A, m, s)

Description: Set the display span about the current centre value for the reflection analyzer.
See [MEASURE] / Display Zoom for more details.

Example: `:REFL:DISP:SPAN 10E9`
Set frequency span to 10 GHz (assuming domain is frequency).

:REFLect
:DISPlay
:SPAN?

Parameters: [<NRf>]
[span value (Hz, dBm, V, A, m, s)]

Response: <NR2>
span value (Hz, dBm, V, A, m, s)

Example: `:REFL:DISP:SPAN?`
Read display span value.

:REFLect
:DISPlay
:STARt

Parameters: <NRf>
start value (Hz, dBm, V, A, m, s)

Description: Set display start value for the reflection analyzer.
See [MEASURE] / Display Zoom for more details.

Example: `:REFL:DISP:STAR 15E9`
Set display start frequency to 15 GHz (assuming domain is frequency).

:REFLect
:DISPlay
:STARt?

Parameters: [<NRf>]
[start value (Hz, dBm, V, A, m, s)]

Response: <NR2>
start value (Hz, dBm, V, A, m, s)

Example: `:REFL:DISP:STAR?`
Read display start value.

:REFLect
:DISPlay
:STOP

Parameters: <NRf>
stop value (Hz, dBm, V, A, m, s)

Description: Set display stop value for the reflection analyzer.
See [MEASURE] / Display Zoom for more details.

Example:
```
:REFL:DISP:STOP 25E9
```
Set display stop value to 25 GHz (assuming domain is frequency).

:REFLect
:DISPlay
:STOP?

Parameters: [<NRf>]
[stop value (Hz, dBm, V, A, m, s)]

Response: <NR2>
stop value (Hz, dBm, V, A, m, s)

Example:
```
:REFL:DISP:STOP?
```
Read display stop value.

:REFLect
:MEASure
:DEFinition?

Response: <CRD>
<measurement definition>
where <measurement definition> is S11 | DIFF | RAT | MEM

Description: Read the measurement definition for the reflection analyser measurement.

S11 is a measurement of S_{11}
DIFF is a live measurement - trace memory
RAT is a live measurement/trace memory
MEM is a measurement definition stored in memory

Example:
```
:REFL:MEAS:DEF?
```
Read the measurement definition.

:REFLect
:MEASure
:DIFFerence

Parameters: <NRf>
memory number

Description: Select live measurement - memory as the current measurement definition.
See [SAVE/RECALL] / Memory Op for more details.

Example:
```
:REFL:MEAS:DIFF 4
```
Subtract the measurement in memory number 4 from the live measurement.

:REFLect
:MEASure
:MEMory
[:ONLY]

Parameters: <NRf>
memory number

Description: Select memory as the current measurement definition and retain the current settings.
See [SAVE/RECALL] / Save/Recall for more details.

Example:
```
:REFL:MEAS:MEM 5
```
Display the measurement in memory 5 but keep the current instrument settings.

:REFLect
:MEASure
:MEMory
:SETTings

Parameters: <NRf>
memory number

Description: Select memory as the current measurement definition and restore the associated settings.
See [SAVE/RECALL] / Save/Recall for more details.

Example:
```
:REFL:MEAS:MEM:SETT 2
```
Display the measurement in memory 2 and use the associated instrument settings.

:REFLect
:MEASure
:RATio

Parameters: <NRf>
memory number

Description: Select live measurement / memory as the current measurement definition.
See [SAVE/RECALL] / Memory Op for more details.

Example:
```
:REFL:MEAS:RAT 2
```
Divide the live measurement by the measurement in memory 2.

:REFLect
:MEASure
:SAVE

Parameters: <NRf>
memory number

Description: Save the current measurement into memory.
See [SAVE/RECALL] / Save/Recall for more details.

Example:
```
:REFL:MEAS:SAV 3
```
Save measurement to memory 3.

:REFLect
:MEASure
:S11

Description: Select S_{11} as the current measurement definition.
See [MEASURE] / Measure for more details.

Example:
```
:REFL:MEAS:S11
```

:REFLect
:MEDium

Parameters: <CPU>
<medium>
where <medium> is COAX | WAV

Description: Specify the medium of the unit under test.
See [MEASURE] / Character Dev for more details.

Example:
```
:REFL:MED COAX
```
Inform instrument that measurements will be made on a coax transmission line.

:REFLect
:MEDium?

Parameters: [<CPD>]
[<medium>]
where <medium> is COAX | WAV

Response: <CRD>
<medium>

Example:
```
:REFL:MED?
```
Determine the medium type that has been selected.

:REFLect
:POFFset

Parameters: <NRf>
phase offset (degrees)

Description: Set the phase offset.
See [MEASURE] / Measure for more details.

Example: `:REFL:POFF 5`
Set phase offset to 5 degrees.

:REFLect
:POFFset?

Parameters: [<NRf>]
[phase offset (degrees)]

Response: <NR2>
phase offset (degrees)

Example: `:REFL:POFF?`
Read the phase offset value.

:REFLect
:RVELocity

Parameters: <NRf>
relative velocity

Description: Set the relative velocity for the medium.
See [MEASURE] / Character Dev for more details.

Example: `:REFL:RVEL 0.8`
Set relative velocity to 0.8.

:REFLect
:RVELocity?

Parameters: [<NRf>]
[relative velocity]

Response: <NR2>
relative velocity

Example: `:REFL:POFF?`
Read the relative velocity.

:REFLect
:SCALing
:AUToscale

Description: Force an autoscale operation on the current measurement.
See [SCALING] key for more details.

Example: `:REFL:SCAL:AUT`

:REFLect
:SCALing
:DIVision

Parameters: <NRf>
per division

Description: Set the graticule vertical scaling.
See [SCALING] key for more details.

Example: `:REFL:SCAL:DIV 2`
Set scaling to 2 dB/division (assuming log magnitude format).

:REFLect
:SCALing
:DIVision?

Parameters: [<NRf>]
[per division]

Response: <NR2>
per division

Example: `:REFL:SCAL:DIV?`
Read the vertical scaling.

:REFLect
:SCALing
:POSition

Parameters: <NRf>
reference position

Description: Set the reference position on cartesian formats.
See [SCALING] key for more details.

Example: `:REFL:SCAL:POS 5`
Select the centre graticule line to be the reference line.

:REFLect
:SCALing
:POSition?

Parameters: [<NRf>]
[reference position]

Response: <NR1>
reference position

Example: `:REFL:SCAL:POS?`
Read the reference position.

:REFLect
:SCALing
:RLEVel

Parameters: <NRf>
reference level

Description: Set the reference level. Reference level units depend on the display format. To determine the format in use, use :REFL:DISP:FORM?
See [SCALING] key for more details.

Example: `:REFL:SCAL:RLEV -10`
Set the reference level to -10 dB (assuming log magnitude format).

:REFLect
:SCALing
:RLEVel?

Parameters: [<NRf>]
[reference level]

Response: <NR2>
reference level

Example: `:REFL:SCAL:RLEV?`
Read the reference level.

:REFLect
:TDOMain
:DISPlay

Parameters: <CPD>
<domain>
where <domain> is TIME | DISTance

Description: Set the domain of the time domain display.
See [MEASURE] /Display Domain for more details.

Example:

```
:REFL:TDOM:DISP DIST
```

Select distance for the time domain display.

:REFLect
:TDOMain
:DISPlay?

Parameters: [<CPD>]
[<domain>]
where <domain> is TIME | DISTance

Response: <CRD>
<domain>

Example:

```
:REFL:TDOM:DISP?
```

Determine the domain that has been selected for the time domain display.

:REFLect
:TDOMain
:EMODe

Parameters: <CPD>
<extrapolation mode>
where <extrapolation mode> is AUTO | MANual | MATChed

Description: Set the extrapolation mode at DC for low pass time domain measurements.
See [MEASURE] / DC Extrap for more details.

Example:

```
:REFL:TDOM:EMOD AUTO
```

Set DC extrapolation mode to AUTO.

:REFLect
:TDOMain
:EMODe?

Parameters: [<CPD>]
[<extrapolation mode>]
where <extrapolation mode> is AUTO | MAN | MATC

Response: <CRD>
<extrapolation mode>

Example:

```
:REFL:TDOM:EMOD?
```

Read the extrapolation mode.

:REFLect
:TDOMain
:GATing
:CENTer

Parameters: <NRf>
centre value (m, s)

Description: Set the gating centre value for the reflection analyzer. The instrument will also accept the English spelling, "CENTre.
See [MEASURE] / Gate/Fence for more details.

Example: `:REFL:TDOM:GAT:CENT 1.5`
Set the centre value to 1.5 m.

:REFLect
:TDOMain
:GATing
:CENTer?

Parameters: [<NRf>]
[centre value (m, s)]

Response: <NR2>
centre value (m, s)

Example: `:REFL:TDOM:GAT:CENT?`
Read the gating centre value.

:REFLect
:TDOMain
:GATing
:COUPling

Parameters: <BOOLEAN PROGRAM DATA>
state

Description: Enable/disable gate/fence coupling.
See [MEASURE] / Gating Mode for more details.

Example: `:REFL:TDOM:GAT:COUP ON`
Enable gate/fence coupling.

:REFLect
:TDOMain
:GATing
:COUPling?

Parameters: [<BOOLEAN PROGRAM DATA>]
[state]

Response: <BOOLEAN RESPONSE DATA>
state

Example: `:REFL:TDOM:GAT:COUP?`
Determine whether gate/fence coupling is enabled.

:REFLect
:TDOMain
:GATing
:SHAPe

Parameters: <CPD>
<shape>
where <shape> is MINimum | NORMal | WIDE | MAXimum

Description: Set the gate/fence shape.
See [MEASURE] / G/F Shape for more details.

Example: `:REFL:TDOM:GAT:SHAP WIDE`
Set gate/fence shape to wide.

:REFLect
:TDOMain
:GATing
:SHAPe?

Parameters: [<CPD>]
[<shape>]
where <shape> is MINimum | NORM | WIDE | MAX

Response: <CRD>
<shape>

Example: `:REFL:TDOM:GAT:SHAP?`
Read the gate/fence shape.

:REFLect
:TDOMain
:GATing
:SPAN

Parameters: <NRf>
span (m, s)

Description: Set the gating span for the reflection analyzer.
See [MEASURE] / Gate/Fence for more details.

Example: `:REFL:TDOM:GAT:SPAN 0.5`
Set the gating span to 0.5 m.

:REFLect
:TDOMain
:GATing
:SPAN?

Parameters: [<NRf>]
[span (m, s)]

Response: <NR2>
span (m, s)

Example: `:REFL:TDOM:GAT:SPAN?`
Read the gating span value.

:REFLect
:TDOMain
:GATing
:STARt

Parameters: <NRf>
start value (m, s)

Description: Set the gating start value for the reflection analyzer.
See [MEASURE] / Gate/Fence for more details.

Example:
```
:REFL:TDOM:GAT:STAR 1.25
```
Set gating start value to 1.25 m.

:REFLect
:TDOMain
:GATing
:STARt?

Parameters: [<NRf>]
[start value (m, s)]

Response: <NR2>
start value (m, s)

Example:
```
:REFL:TDOM:GAT:STAR?
```
Read the gating start value.

:REFLect
:TDOMain
:GATing
[:STATe]

Parameters: <BOOLEAN PROGRAM DATA>
state

Description: Enable/disable gating.
See [MEASURE] / Gate/Fence for more details.

Example:
```
:REFL:TDOM:GAT OFF
```
Disable gating.

:REFLect
:TDOMain
:GATing
:STATe?

Parameters: [<BOOLEAN PROGRAM DATA>]
[state]

Response: <BOOLEAN RESPONSE DATA>
state

Example:
```
:REFL:TDOM:GAT:STAT?
```
Determine whether gating is enabled.

:REFLect
:TDOMain
:GATing
:STOP

Parameters: <NRf>
stop value (m, s)

Description: Set the gating stop value for the reflection analyzer.
See [MEASURE] / Gate/Fence for more details.

Example:

```
:REFL:TDOM:GAT:STOP 1.75
```

Set gating stop value to 1.75 m.

:REFLect
:TDOMain
:GATing
:STOP?

Parameters: [<NRf>]
[stop value (m, s)]

Response: <NR2>
stop value (m, s)

Example:

```
:REFL:TDOM:GAT:STOP?
```

Read the gating stop value.

:REFLect
:TDOMain
:GATing
:TYPE

Parameters: <CPD>
<type>
where <type> is GATE | FENCe

Description: Set the gate/fence type.
See [MEASURE] / Gating Mode for more details.

Example:

```
:REFL:TDOM:GAT:TYPE FENC
```

Select fencing.

:REFLect
:TDOMain
:GATing
:TYPE?

Parameters: [<CPD>]
[<type>]
where <type> is GATE | FENCe

Response: <CRD>
<type>

Example:

```
:REFL:TDOM:GAT:TYPE?
```

Determine whether gating or fencing has been selected.

:REFLect
:TDOMain
:RCOefficient

Parameters: <NRf>
reflection coefficient

Description: Set the reflection coefficient at DC for low pass time domain measurements.
See [MEASURE] / DC Extrap for more details.

Example: `:REFL:TDOM:RCO 0.2`
Set the DC reflection coefficient to 0.2.

:REFLect
:TDOMain
:RCOefficient?

Parameters: [<NRf>]
[reflection coefficient]

Response: <NR2>
reflection coefficient

Example: `:REFL:TDOM:RCO?`
Read the DC reflection coefficient.

:REFLect
:TDOMain
:RESPonse

Parameters: <CPD>
<response>
where <response> is IMPulse | STEP

Description: Set the response type.
See [MEASURE] / Time Dom Funcs for more details.

Example: `:REFL:TDOM:RESP IMP`
Set response type to impulse.

:REFLect
:TDOMain
:RESPonse?

Parameters: [<CPD>]
[<response>]
where <response> is IMP | STEP

Response: <CRD>
<response>

Example: `:REFL:TDOM:RESP?`
Read the response type.

:REFLect
:TDOMain
[:STATe]

Parameters: <BOOLEAN PROGRAM DATA>
state

Description: Set the time domain state.
See [MEASURE] / Measure for more details.

Example: `:REFL:TDOM:STAT ON`
Select time domain.

:REFLect
:TDOMain
:STATe?

Parameters: [<BOOLEAN PROGRAM DATA>]
[state]

Response: <BOOLEAN RESPONSE DATA>
state

Example: `:REFL:TDOM:STAT?`
Determine whether time domain has been selected.

:REFLect
:TDOMain
:TRANsform

Parameters: <CPD>
<transform>
where <transform> is LOW | BAND

Description: Set the tranform type.
LOW is Low Pass Transform.
BAND is Band Pass Transform.
See [MEASURE] / Time Dom Funcs for more details.

Example: `:REFL:TDOM:TRAN LOW`
Select low pass transform.

:REFLect
:TDOMain
:TRANsform?

Parameters: [<CPD>]
[<transform>]
where <transform> is LOW | BAND

Response: <CRD>
<transform>

Example: `:REFL:TDOM:TRAN?`
Read the transform type.

:REFLect
:TDOMain
:UNITs

Parameters: <CPD>
<units>
where <units> is FEET | METRes

Description: Set the time domain distance units.
See [MEASURE] / Display Domain for more details.

Example: `:REFL:TDOM:UNIT METR`
Set distance units to metres.

:REFLect
:TDOMain
:UNITs?

Parameters: [<CPD>]
[<units>]
where <units> is FEET | METR

Response: <CRD>
<units>

Example: `:REFL:TDOM:UNIT?`
Read the distance units.

:REFLect
:TDOMain
:WINDow
:SHAPe

Parameters: <CPD>
<shape>
where <shape> is MINimum | NORMal | WIDE | USER

Description: Set the windowing shape.
See [MEASURE] / Windowing for more details.

Example: `:REFL:TDOM:WIND:SHAP NORM`
Set windowing shape to normal.

:REFLect
:TDOMain
:WINDow
:SHAPe?

Parameters: [<CPD>]
[<shape>]
where <shape> is MIN | NORM | WIDE | USER

Response: <CRD>
<shape>

Example: `:REFL:TDOM:WIND:SHAP?`
Read the windowing shape.

:REFLect
:TDOMain
:WINDow
:USER

Parameters: <NRf>
order

Description: Set the windowing order for user type window.
See [MEASURE] / Windowing for more details.

Example:
```
:REFL:TDOM:WIND:USER 6
```
Set the windowing order to 6.

:REFLect
:TDOMain
:WINDow
:USER?

Parameters: [<NRf>]
[order]

Response: <NR1>
order

Example:
```
:REFL:TDOM:WIND:USER?
```
Read the windowing order.

:REFLect
:WGCutoff

Parameters: <NRf>
waveguide cutoff frequency (Hz)

Description: Set the waveguide cutoff frequency.
See [MEASURE] /Character Dev for more details.

Example: `:REFL:WGC 3.122E9`
Set the waveguide cutoff frequency to 3.122 GHz.

:REFLect
:WGCutoff?

Parameters: [<NRf>]
[waveguide cutoff frequency (Hz)]

Response: <NR2>
waveguide cutoff frequency (Hz)

Example: `:REFL:WGC?`
Read the waveguide cutoff frequency.

REFLect
:ZERO

Description: Zero the detectors in the reflection analyzer.
See [CAL] / Cal for more details.

Example: `:REFL:ZERO`
Perform a reflection analyzer zero.

ROUT SUBSYSTEM

```
ROUT
        AVERaging
                AUTO\?
                NUMBer\?
                RESTart
                [STATe]\?
        [DATA?]
        DCYCle
                [STATe]\?
                VALue\?
        FORMat\?
        LIMit
                MAXimum\?
                MINimum\?
                [STATe]\?
        MEASure
                DEFinition?
                DIFFerence\?
                FREQuency
                POWer\?
                RATio\?
        PEAK
                MAXimum?
                MINimum?
                RESet
                [STATe]\?
        PIReset
        RELative
                SAVE
                [STATe]\?
        RESolution
                PMETer\?
                COUNter\?
```

:ROUT
:AVERaging
:AUTO

Parameters: <CPD> , <BOOLEAN PROGRAM DATA>
<input id>, state
where <input id> *is* A | B | C | D

Description: Enable automatic averaging number selection for the device connected to the specified input.
See [MEASURE] / Input A (B, C or D) for more details.

Example:
```
:ROUT:AVER:AUTO D,ON
```
Enable automatic averaging number selection for input D.

:ROUT
:AVERaging
:AUTO?

Parameters: <CPD>[, <BOOLEAN PROGRAM DATA>]
<input id>[, state]
where <input id> *is* A | B | C | D

Response: <BOOLEAN RESPONSE DATA>
state

Example:
```
:ROUT:AVER:AUTO? D
```
Determine whether automatic averaging number selection is enabled for input D.

:ROUT
:AVERaging
:NUMBer

Parameters: <CPD> , <NRf>
<input id>, average number
where <input id> *is* A | B | C | D

Description: Set the average number for the specified input. This will only take effect if auto averaging is not enabled (See :ROUT:AVERaging:AUTO)
See [MEASURE] / Input A (B, C or D) for more details.

Note that for a readout channel, averaging is set up per input and not per measurement as is the case for a scalar channel.

Example:
```
:ROUT:AVER:NUMB D,16
```
Set an average number of 16 for input D.

:ROUT
:AVERaging
:NUMBer?

Parameters: <CPD>[, NRf]
<input id>[, average number]
where <input id> *is* A | B | C | D

Response: <NR1>
average number

Example:
```
:ROUT:AVER:NUMB? D
```
Read average number for input D.

:ROUT
:AVERaging
:RESTart

Parameters: <CPD>
<input id>
where <input id> *is* A | B | C | D

Description: Restart averaging for the readout power measurement.
See [MEASURE] / Measure for more details.

Example: `:ROUT:AVER:REST D`
Restart averaging on input D.

:ROUT
:AVERaging
[:STATe]

Parameters: <CPD> , <BOOLEAN PROGRAM DATA>
<input id>, state
where <input id> *is* A | B | C | D

Description: Enable/disable averaging for the specified input.
Note that for a readout channel, averaging is set up per input and not per measurement as is the case for a scalar channel.
See [MEASURE] / Input A (B, C or D) for more details.

Example: `:ROUT:AVER D,ON`
Enable averaging for input D.

:ROUT
:AVERaging
:STATe?

Parameters: <CPD>[, <BOOLEAN PROGRAM DATA>]
<input id>[, state]
where <input id> *is* A | B | C | D

Response: <BOOLEAN RESPONSE DATA>
state

Example: `:ROUT:AVER:STAT? A`
Determine whether averaging is enabled for input A.

:ROUT
[:DATA]?

Response: <NR2>
measurement

Description: Read the current value of the active measurement. This may either be a frequency or a power value depending on the measurement definition. The returned units depend on the active measurement format:

Frequency	Hz
Log Power	dB or dBm
Linear Power	W
Percent	Units (%)

Example: `:ROUT?`
Read the active measurement.

:ROUT
:DCYCle
[:STATe]

Parameters: <BOOLEAN PROGRAM DATA>
state

Description: Set power measurement duty cycle correction on or off.
See [MEASURE] / Duty Cycle for more details.

Example: `:ROUT:DCYC ON`
Enable duty cycle correction.

:ROUT
:DCYCle
:STATe?

Parameters: [<BOOLEAN PROGRAM DATA>]
[state]

Response: <BOOLEAN RESPONSE DATA>
state

Example: `:ROUT:DCYC:STAT?`
Determine whether duty cycle correction is enabled.

:ROUT
:DCYCle
:VALue

Parameters: <NRf>
duty cycle (%)

Description: Set the duty cycle value for power measurement duty cycle correction.
See [MEASURE] / Duty Cycle for more details.

Example:
```
:ROUT:DCYC:VAL 50
```
Set the instrument to measure a pulsed power source with a duty cycle of 50%.

:ROUT
:DCYCle
:VALue?

Parameters: [<NRf>]
[duty cycle (%)]

Response: <NR2>
duty cycle (%)

Example:
```
:ROUT:DCYC?
```
Read the duty cycle value.

:ROUT
:FORMat

Parameters: <CPD>
<settable readout format>
where <settable readout format> *is* WATTs | PERCent | LOG | VOLTs

Description: Set the display format for a power measurement. The choice of valid formats depends on the measurement definition.
See [FORMAT] key for more details.

Example:
```
:ROUT:FORM WATT
```
Set Watts format.

:ROUT
:FORMat?

Parameters: [<CPD>]
[<settable readout format>]
where <settable readout format> *is* WATTs | PERCent | LOG | VOLTs

Response: <CRD>
<readable readout format>
where <readable readout format> *is* WATT | PERC | LOG | FREQ | VOLT

Description: Note that the query form will indicate that frequency format is in use when a frequency measurement is defined.

Example:
```
:ROUT:FORM?
```
Read the format.

:ROUT
:LIMit
:MAXimum

Parameters: <NRf>
limit value

Description: Set a measurement maximum limit value for use when limit checking is enabled. The limit value units are interpreted according to the active measurement display format:

Frequency	Hz
Log Power	dB or dBm
Linear Power	W
Percent	Units (%)

See [MEASURE] / Lim Checking for more details.

Example: `:ROUT:LIM:MAX -10`
Set a limit value of -10 dBm (assuming log format previously selected).

:ROUT
:LIMit
:MAXimum?

Parameters: [<NRf>]
[limit value]

Response: <NR2>
limit value

Example: `:ROUT:LIM:MAX?`
Read the maximum limit value.

:ROUT
:LIMit
:MINimum

Parameters: <NRf>
limit value

Description: Set a measurement minimum limit value for use when limit checking is enabled. The limit value units are interpreted according to the active measurement display format:

Frequency	Hz
Log Power	dB or dBm
Linear Power	W
Percent	Units (%)

See [MEASURE] / Lim Checking for more details.

Example:
```
:ROUT:LIM:MIN -10
```
Set a limit value of -10 dBm (assuming log format previously selected).

:ROUT
:LIMit
:MINimum?

Parameters: [<NRf>]
[limit value]

Response: <NR2>
limit value

Example:
```
:ROUT:LIM:MIN?
```
Read the minimum limit value.

:ROUT
:LIMit
[:STATe]

Parameters: <BOOLEAN PROGRAM DATA>
state

Description: Set limit checking on or off. Out of limits reports are issued via the status reporting structures (see page *App. A*-4).
See [MEASURE] / Lim Checking for more details.

Example:
```
:ROUT:LIM ON
```
Enable limit checking.

:ROUT
:LIMit
:STATe?

Parameters: [<BOOLEAN PROGRAM DATA>]
[state]

Response: <BOOLEAN RESPONSE DATA>
state

Example:
```
:ROUT:LIM:STAT?
```
Determine whether limit checking is enabled.

:ROUT
:MEASure
:DEFinition?

Response: <CRD>
<measurement definition>
where <measurement definition> *is* FREQ | POW | DIFF | RAT

Description: Determine the readout measurement definition for the active measurement on the active channel:

FREQ	Frequency measurement
POW	Absolute power measurement
DIFF	Power difference measurement
RAT	Power ratio measurement

Example:
```
:ROUT:MEAS:DEF?
```
Read the measurement definition.

:MEASure

:DIFFerence

Parameters: <CPD> , <CPD>
<minuend>, <subtrahend>
where <minuend> *is* A | B | C | D *and*
<subtrahend> *is* A | B | C | D
The minuend and subtrahend must be different inputs.

Description: Set up a difference measurement between the specified inputs.
See [MEASURE] / Input Diff for more details.

Example:

```
:ROUT:MEAS:DIFF A,B
```

Measure the difference A-B.

ROUT:

:MEASure

:DIFFerence?

Parameters: [<CPD> , <CPD>]
[<minuend>, <subtrahend>]
where <minuend> *is* A | B | C | D *and*
<subtrahend> *is* A | B | C | D
The minuend and subtrahend must be different inputs.

Response: <CRD> , <CRD>
<minuend>, <subtrahend>

Description: If the optional parameters are supplied, a difference measurement will be set up using the inputs specified.

If the optional parameters are not supplied, the instrument will respond with the currently selected minuend and subtrahend. Before this command is sent, :ROUT:MEAS:DEF? should be used to verify that the active measurement is a difference measurement.
See [MEASURE] / Input Diff for more details.

Example:

```
:ROUT:MEAS:DIFF?
```

Read the difference measurement minuend and subtrahend.

:ROUT

:MEASure

:FREQuency

Description: Enable frequency measurements.
See [MEASURE] / Measure for more details.

Example:

```
:ROUT:MEAS:FREQ
```

Set up a frequency measurement.

:ROUT
:MEASure
:POWer

Parameters: <CPD>
<input id>
where <input id> *is* A | B | C | D

Description: Define an absolute power measurement from the specified input.
See [MEASURE] / Single Input for more details.

Example:
```
:ROUT:MEAS:POW D
```
Measure absolute power from input D.

:ROUT
:MEASure
:POWer?

Parameters: [<CPD>]
[<input id>]
where <input id> *is* A | B | C | D.

Response: <CRD>
<input id>

Description: If the optional parameter is supplied an absolute power measurement will be set up from the specified input.

If the optional parameter is not supplied, the instrument will respond with the currently selected input for absolute power measurements. Before this command is sent, :ROUT:MEAS:DEF? should be used to verify that the active measurement is an absolute power measurement.
See [MEASURE] / Single Input for more details.

Example:
```
:ROUT:MEAS:POW?
```
Determine which input is selected for the absolute power measurement.

:ROUT
:MEASure
:RATio

Parameters: <CPD> , <CPD>
<numerator>, <denominator>
where <numerator> *is* A | B | C | D *and*
<denominator> *is* A | B | C | D
The numerator and denominator must be different inputs.

Description: Set up a ratio measurement from the specified inputs.
See [MEASURE] / Input Ratio for more details.

Example:
```
:ROUT:MEAS:RAT A,C
```
Measure the ratio A/C.

:ROUT
:MEASure
:RATio?

Parameters: [<CPD> , <CPD>]
[<numerator>, <denominator>]
where <numerator> *is* A | B | C | D *and*
<denominator> *is* A | B | C | D
The numerator and denominator must be different inputs.

Response: <CRD> , <CRD>
<numerator>, <denominator>

Description: If the optional parameters are supplied, a ratio measurement will be set up from the specified inputs.

If the optional parameters are not supplied, the instrument will respond with the currently selected numerator and denominator. Before this command is sent, :ROUT:MEAS:DEF? should be used to verify that the active measurement is a ratio measurement.
See [MEASURE] / Input Ratio for more details.

Example:
```
:ROUT:MEAS:RAT?
```
Read the ratio measurement numerator and denominator.

:ROUT

:PEAK

:MAXimum?

Response: <NR2>
maximum value

Description: Read the maximum value of the measurement since peak hold was last reset. The value returned should be interpreted according to the active measurement format:

Frequency	Hz
Log Power	dB or dBm
Linear Power	W
Percent	Units (%)

See [MEASURE] / Max Min Hold for more details.

Example:
```
:ROUT:PEAK:MAX?
```
Read the maximum measurement since peak hold was last reset.

:ROUT

:PEAK

:MINimum?

Response: <NR2>
minimum value

Description: Read the minimum value of the measurement since peak hold was last reset. The units of the value returned should be interpreted according to the active measurement format:

Frequency	Hz
Log Power	dB or dBm
Linear Power	W
Percent	Units (%)

See [MEASURE] / Max Min Hold for more details.

Example:
```
:ROUT:PEAK:MIN?
```
Read the minimum measurement since peak hold was last reset.

:ROUT

:PEAK

:RESet

Description: Reset the peak hold function. The current measurement is written to both the maximum and minimum stores.
See [MEASURE] / Max Min Hold for more details.

Example:
```
:ROUT:PEAK:RES
```
Reset peak hold.

:ROUT
:PEAK
[:STATe]

Parameters: <BOOLEAN PROGRAM DATA>
state

Description: Set peak hold on or off.
See [MEASURE] / Max Min Hold for more details.

Example: `:ROUT:PEAK ON`
Enable peak hold.

:ROUT
:PEAK
:STATe?

Parameters: [<BOOLEAN PROGRAM DATA>]
[state]

Response: <BOOLEAN RESPONSE DATA>
state

Example: `:ROUT:PEAK:STAT?`
Determine whether peak hold is enabled.

:ROUT
:PIReset

Description: Reset the readout display peaking indicator.
See [MEASURE] / Measure for more details.

Example: `:ROUT:PIR`
Reset the peaking indicator.

:ROUT
:RELative
:SAVE

Description: Save the active measurement into the relative measurement store to establish a reference level for relative measurements.
See [MEASURE] / dB Rel and [MEASURE] / Freq Rel for more details.

Example: `:ROUT:REL:SAVE`
Make future measurements relative to the current measurement.

:ROUT

:RELative

[:STATe]

Parameters: <BOOLEAN PROGRAM DATA>
state

Description: Enable/disable a measurement relative to a reference measurement held in the relative measurement store. See also :ROUT:RELative:SAVE
See [MEASURE] / dB Rel and [MEASURE] / Freq Rel for more details.

Example: `:ROUT:REL ON`
Enable relative measurements.

:ROUT

:RELative:

:STATe?

Parameters: [<BOOLEAN PROGRAM DATA>]
[state]

Response: <BOOLEAN RESPONSE DATA>
state

Example: `:ROUT:REL:STAT?`
Determine whether relative measurement mode is enabled.

:ROUT

:RESolution

:COUNter

Parameters: <NRf>
resolution (Hz)

Description: Set the counter resolution.
See [MEASURE] / Cntr Res for more details.

Example: `:ROUT:RES:COUN 100`
Set the counter resolution to 100 Hz.

:ROUT

:RESolution

:COUNter?

Parameters: [<NRf>]

[resolution (Hz)]

Response: <NR2>

resolution (Hz)

Example: `:ROUT:RES:COUN?`
Determine the counter resolution that has been set.

:ROUT

:RESolution

:PMETer

Parameters: <CPD>, <CPD>
<input id>, <resolution>
where <input id> *is* A|B|C|D
<resolution> *is*:

AUTO	(the same as auto averaging on).
HIGH	(0.01 dB)
MEDium	(0.1 dB)
LOW	(1 dB)
OFF	(the same as averagng off)

Description: Set the power meter resolution.
See [MEASURE] / Input A (B, C or D) for more details.

Example:

```
:ROUT:RES:D, HIGH
```

Set the power meter resolution to HIGH on input D.

:ROUT

:RESolution

:PMETer?

Parameters: <CPD> [,<CPD>]
<input id> [,<resolution>]
where <resolution> *is* AUTO | HIGH | MED | LOW | OFF

Response: <CRD>
<resolution>

Example:

```
:ROUT:RES:PMET? D
```

Determine the power meter resolution that has been selected for input D.

SCALar SUBSYSTEM

```
SCALar
    AVERaging
        NUMBer\?
        RESTart
        [STATe]\?
    DATA
        [ASCii]?
        BINary?
    DETector
        CALibration
            [STATe]?
    FORMat\?
    MEASure
        DEFinition?
        FREQuency
        MEMory
            [ONLY]\?
            SETTings\?
        POWer\?
        RATio\?
    MEMory
        [DATA]\?
        MAXimum?
    MOPeration
        SELect\?
        [STATe]\?
    PCAL
        OPEN
            [ONLY]
                [RLOSs]
                SEILoss
            MERGe
                [RLOSs]
                SEILoss
        *SAVE
        SEILoss?
        SELect\?
        SHORt
            [ONLY]
                [RLOSs]
                SEILoss
            MERGe
                [RLOSs]
                SEILoss
        [STATe]\?
        STORe\?
        THRough
SAVE
SCALing
        AUTO\?
        DIVision\?
        POSition\?
        RLEVel\?
```

Note...

The command marked '*' is an older version of the path cal commands. It is recommended, however, that the new commands (OPEN, SHORt and THRough) are used for new GPIB programs.

:SCALar
:AVERaging
:NUMBer

Parameters: <NRf>
averaging number

Description: Set the averaging number for scalar measurements. This is applied on the sweep after the current sweep.
See [MEASURE] / Averaging for more details.

Example: `:SCAL:AVER:NUMB 16`
Set an averaging number of 16.

:SCALar
:AVERaging
:NUMBer?

Parameters: [<NRf>]
[averaging number]

Response: <NR1>
averaging number

Example: `:SCAL:AVER:NUMB?`
Read the scalar averaging number.

:SCALar
:AVERaging
:RESTart

Description: Restart averaging. This is applied on the sweep after the current sweep.
See [MEASURE] / Measure for more details.

Example: `:SCAL:AVER:REST`
Restart averaging.

:SCALar
:AVERaging
[:STATe]

Parameters: <BOOLEAN PROGRAM DATA>
state

Description: Set scalar averaging on or off. This is applied on the sweep after the current sweep.
See [MEASURE] / Averaging for more details.

Example:
```
:SCAL:AVER ON
```
Set scalar averaging on.

:SCALar
:AVERaging
:STATE?

Parameters: [<BOOLEAN PROGRAM DATA>]
[state]

Response: <BOOLEAN RESPONSE DATA>
state

Example:
```
:SCAL:AVER:STAT?
```
Determine whether averaging is enabled.

:SCALar
[:DATA]
[:ASCii]?

Response: <NR2>,<NR2>,...,<NR2>
response at point 0,response at point 1...,response at point *n*-1

Description: Read scalar measurement data in ASCII format. Units of data returned depends on measurement format: dB, dBm, Watts, VSWR, Hz, etc.

Example:
```
:SCAL?
```
Read scalar measurement data.

Example response: -10.12,-10.09,-10.03,...0.15,0.49

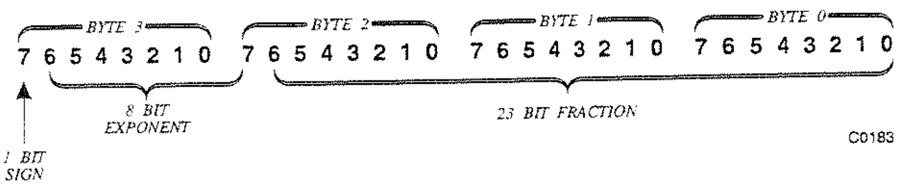
:SCALar
:DATA
:BINary?

Response: <INDEFINITE LENGTH ARBITRARY BLOCK RESPONSE DATA>
<measurement data>
where <measurement data> *is* organised as follows:

The number of measurements in <measurement data> is equal to the number of measurement points (set using :SOURce:SWEep:POINTS).

Each measurement consists of 4 bytes received in the order:
byte 0, byte 1, byte 2, byte 3.

These bytes hold a 32 bit (IEEE single precision) number conforming to the IEEE Standard of Binary Floating-Point Arithmetic, ANSI/IEEE Std 754-1985.



Description: Read scalar measurement data in Binary format. Binary format allows faster transfer of measurement data.

Example: `:SCAL:DATA:BIN?`
Read measurement data.

:SCALar
:DETector
:CALibration
[:STATe]?

Response: <BOOLEAN RESPONSE DATA>
state

Example : `:SCAL:DET:CAL?`
Determine if last EEPROM detector cal passed.

:SCALar
:FORMat

Parameters: <CPD>
<settable scalar format>
where <settable scalar format> *is* VSWR | WATT | VOLT | LOG

Description: Set the scalar measurement display format.
See [FORMAT] key for more details.

Example: `:SCAL:FORM LOG`
Display the measurement as dB or dBm.

:SCALar
:FORMat?

Parameters: [<CPD>]
[<settable scalar format>]
where <settable scalar format> *is* VSWR | WATT | VOLT | LOG

Response: <CRD>
<readable scalar format>
where <readable scalar format> *is* VSWR | WATT | VOLT | LOG | FREQ

Description: The frequency format is set automatically when a swept measurement of frequency is defined. It may be read (but not set) using the FORMat command.

Example: `:SCAL:FORM?`
Read scalar format.

:SCALar
:MEASure
:DEFinition?

Response: <CRD>
<measurement definition>
where <measurement definition> *is* POW | RAT | MEM | FREQ

Description: Determine the scalar measurement definition for the active measurement on the active channel:

POW	Absolute power measurement
RAT	Ratio measurement
MEM	Trace memory display
FREQ	Swept measurement of frequency

Example: `:SCAL:MEAS:DEF?`
Read the measurement definition.

:SCALar
:MEASure
:FREQuency

Description: Set up a swept measurement of frequency.
Note that this measurement requires that the swept domain is either voltage or current. This may be set using :SOURce:MODE.
See [MEASURE] / Measure for more details.

Example:
```
:SCAL:MEAS:FREQ
```

:SCALar
:MEASure
:MEMory
[:ONLY]

Parameters: <NRf>
memory number

Description: Display a trace memory.
See [SAVE/RECALL] / Save/Recall for more details.

Example:
```
:SCAL:MEAS:MEM 2
```
Display trace memory 2.

:SCALar
:MEASure
MEMory?
:ONLY?

Parameters: [<NRf>]
[memory number]

Response: <NR1>
memory number

Description: If the optional parameter is supplied, a trace memory will be displayed.
If the optional parameter is not supplied, the instrument will respond with the number of the displayed trace memory. In the latter case, :SCAL:MEAS:DEF? should be used to verify that a trace memory is being displayed.
See [SAVE/RECALL] / Save/Recall for more details.

Example:
```
:SCAL:MEAS:MEM:ONLY?
```
Read the number of the displayed trace memory.

:SCALar
:MEASure
:MEMory
:SETTings

Parameters: <NRf>
memory number

Description: Display a trace memory, and restore the settings associated with it.
See [SAVE/RECALL] / Save/Recall for more details.

Example: `:SCAL:MEAS:MEM:SETT 2`
Display trace memory 2.

:SCALar
:MEASure
MEMory
:SETTings?

Parameters: [<NRf>]
[memory number]

Response: <NR1>
memory number

Description: If the optional parameter is supplied, a trace memory will be displayed, and the settings associated with it restored.
If the optional parameter is not supplied, the instrument will respond with the number of the displayed trace memory. In the latter case, :SCAL:MEAS:DEF? should be used to verify that a trace memory is being displayed.
See [SAVE/RECALL] / Save/Recall for more details.

Example: `:SCAL:MEAS:MEM:SETT?`
Read the number of the displayed trace memory.

:SCALar

:MEASure

:POWer

Parameters: <CPD>
<input id>
where <input id> *is* A | B | C | D

Description: Set up an absolute power measurement from the specified input. This is applied on the sweep after the current sweep.
See [MEASURE] / Single Input for more details.

Example: `:SCAL:MEAS:POW A`
Measure absolute power from input A.

:SCALar

:MEASure

:POWer?

Parameters: [<CPD>]
[<input id>]
where <input id> *is* A | B | C | D

Response: <CRD>
<input id>

Description: If the optional parameter is supplied, an absolute power measurement will be set up from the specified input.
If the optional parameter is not supplied, the instrument will respond with the currently selected input for absolute power measurements. In the latter case, :SCAL:MEAS:DEF? should be used to verify that the current measurement is an absolute power measurement.
See [MEASURE] / Single Input for more details.

Example: `:SCAL:MEAS:POW?`
Determine which input is selected for the absolute power measurement.

:SCALar
:MEASure
:RATio

Parameters: <CPD>,<CPD>
<numerator>,<denominator>
where <numerator> *is* A | B | C | D *and* <denominator> *is* A | B | C | D
The numerator and denominator must be different inputs.

Description: Set up a ratio measurement from the specified inputs. This is applied on the sweep after the current sweep.
See [MEASURE] / Input Ratio for more details.

Example: `:SCAL:MEAS:RAT A,C`
Measure the ratio A/C.

:SCALar
:MEASure
:RATio?

Parameters: [<CPD>,<CPD>]
[numerator,denominator]

Response: <CRD>,<CRD>
numerator,denominator

Description: If the optional parameters are supplied, a ratio measurement will be set up from the specified inputs.
If the optional parameters are not supplied, the instrument will respond with the currently selected numerator and denominator. In the latter case, :SCAL:MEAS:DEF? should be used to verify that the current measurement is a ratio measurement.
See [MEASURE] / Input Ratio for more details.

Example: `:SCAL:MEAS:RAT?`
Read the ratio measurement numerator and denominator.

:SCALar

:MEMory

[:DATA]

Parameters: <NRf>,<ARBITRARY BLOCK PROGRAM DATA>
memory number, memory contents

Description: Write trace memory data to a trace memory store.

Example: `:SCAL:MEM 3,#...etc`
Write memory data to store 3. (Only the first character of the memory data is shown).

:SCALar

:MEMory

:DATA?

Parameters: <NRf>[,<ARBITRARY BLOCK PROGRAM DATA>]
memory number[,memory contents]

Response: <INDEFINITE LENGTH ARBITRARY BLOCK RESPONSE DATA>
memory contents

Example : `:SCAL:MEM:DATA? 1`
Read the data stored in trace memory 1.

:SCALar

:MEMory

:MAXimum?

Response: <NR1>
number of stores

Description: Read the number of trace memory stores available, including any located on a memory card.

Example: `:SCAL:MEM:MAX?`
Determine how many trace memory stores are available.

:SCALar
:MOPeration
:SELect

Parameters: <NRf>
memory number

Description: Select a trace memory for use in a memory operation with a live measurement.
See [SAVE/RECALL] / Memory Op for more details.

Example: `:SCAL:MOP:SEL 1`
Select trace memory 1 for use in a memory operation.

:SCALar
:MOPeration
:SELect?

Parameters: [<NRf>]
[memory number]

Response: <NR1>
memory number

Example: `:SCAL:MOP:SEL?`
Read the store number of the trace memory selected for a memory operation.

:SCALar
:MOPeration
[:STATe]

Parameters: <BOOLEAN PROGRAM DATA>
state

Description: Enable/disable a mathematical operation between a trace memory and a live measurement. This is applied on the sweep after the current sweep.
See [SAVE/RECALL] / Memory Op for more details.

Example: `:SCAL:MOP ON`
Enable the memory operation

:SCALar
:MOPeration
:STATe?

Parameters: [<BOOLEAN PROGRAM DATA>]
[state]

Response: <BOOLEAN RESPONSE DATA>
state

Example: `:SCAL:MOP:STAT?`
Determine whether a memory operation is enabled.

:SCALar
:PCAL
:OPEN
[:ONLY]
[:RLOSs]

Parameters: <NRf>
store number

Description: Initiate an 'open' path calibration, storing the data in the specified path calibration store. Note that this command restarts averaging and requires the relevant number of sweeps to be completed, which is equal to the average number that has been set.
See [CAL] / Cal for more details.

Example: `:SCAL:PCAL:OPEN 3`
Perform an 'open' path calibration. The data will be stored in path cal store 3.

:SCALar
:PCAL
:OPEN
[:ONLY]
:SEILoss

Parameters: <NRf>
store number

Description: Initiate an 'open' path calibration for a single-ended insertion loss measurement, storing the data in the specified path calibration store. Note that this command restarts averaging and requires the relevant number of sweeps to be completed, which is equal to the average number that has been set.
See [CAL] / Cal for more details.

Example: `:SCAL:PCAL:OPEN:SEIL 3`
Perform an 'open' path calibration. The data will be stored in path cal store 3.

:SCALar
:PCAL
:OPEN
:MERGe
[:RLOSs]

Parameters: <NRf>
store number

Description: Initiate an 'open' path calibration, merging the data with the already stored in the specified path calibration store. This command is used after a :SCAL:PCAL:SHOR to perform the second half of a Short & Open path calibration. Note that this command restarts averaging and requires the relevant number of sweeps to be completed, which is equal to the average number that has been set.
See [CAL] / Cal for more details.

Example:

```
:SCAL:PCAL:OPEN:MERG 2
```

Perform an 'open' path calibration and merge with the data already present in path cal store 2 (which would have been created previously for a 'short' path cal).

:SCALar
:PCAL
:OPEN
:MERGe
:SEILoss

Parameters: <NRf>
store number

Description: Initiate an 'open' path calibration for a single-ended insertion loss measurement, merging the data with the already stored in the specified path calibration store. This command is used after a :SCAL:PCAL:SHOR:SEIL to perform the second half of a Short & Open path calibration. Note that this command restarts averaging and requires the relevant number of sweeps to be completed, which is equal to the average number that has been set.
See [CAL] / Cal for more details.

Example:

```
:SCAL:PCAL:OPEN:MERG:SEIL 2
```

Perform an 'open' path calibration and merge with the data already present in path cal store 2 (which would have been created previously for a 'short' path cal).

:SCALar
:PCAL
:SAVE

Parameters: <NRf>,<CPD>
store number.<cal type>
where <cal type> *is* SHORt | OPEN | THRough

Description: Initiate a path calibration of type specified by <cal type>, storing the data in the specified path calibration store. Note that this command restarts averaging and requires the relevant number of sweeps to be completed, which is equal to the average number that has been set.

This is an older version of the path cal command; the new commands OPEN, SHORt and THRough should be used for new GPIB programs.
See [CAL] / Cal for more details.

Example:
```
:SCAL:PCAL:SAVE 1,THR
```
Perform a "Through" path calibration. The data will be stored in path cal store 1.

:SCALar
:PCAL
:SEILoss?

Response: <BOOLEAN RESPONSE DATA>
state

Example:
```
:SCAL:PCAL:SEIL?
```
Determine if the path cal for the active trace is a single-ended insertion loss path calibration.

:SCALar
:PCAL
:SELect

Parameters: <NRf>
store number

Description: Assign a path calibration store to the active measurement. This is applied on the sweep after the current sweep.
See [CAL] / Cal for more details.

Example:
```
:SCAL:PCAL:SEL 1
```
Assign path cal store 1 to the measurement.

:SCALar
:PCAL
:SELect?

Parameters: [<NRf>]
[store number]

Response: <NR1>
store number

Example:
```
:SCAL:PCAL:SEL?
```
Determine which path cal store is assigned to the measurement.

:SCALar
:PCAL
:SHORt
[:ONLY]
[:RLOSs]

Parameters: <NRf>
store number

Description: Initiate a 'short' path calibration, storing the data in the specified path calibration store. Note that this command restarts averaging and requires the relevant number of sweeps to be completed, which is equal to the average number that has been set.
See [CAL] / Cal for more details.

Example:
```
:SCAL:PCAL:SHOR 2
```
Perform a 'short' path calibration. The data will be stored in path cal store 2.

:SCALar
:PCAL
:SHORt
[:ONLY]
:SEILoss

Parameters: <NRf>
store number

Description: Initiate a 'short' path calibration for a single-ended insertion loss measurement , storing the data in the specified path calibration store. Note that this command restarts averaging and requires the relevant number of sweeps to be completed, which is equal to the average number that has been set.
See [CAL] / Cal for more details.

Example: `:SCAL:PCAL:SHOR:SEIL 2`
Perform a 'short' path calibration. The data will be stored in path cal store 2.

:SCALar
:PCAL
:SHORt
:MERGe
[:RLOSs]

Parameters: <NRf>
store number

Description: Initiate a 'short' path calibration, merging the data with the already stored in the specified path calibration store. This command is used after a :SCAL:PCAL:OPEN to perform the second half of a Short & Open path calibration. Note that this command restarts averaging and requires the relevant number of sweeps to be completed, which is equal to the average number that has been set.
See [CAL] / Cal for more details.

Example: `:SCAL:PCAL:SHOR:MERG 3`
Perform a 'short' path calibration and merge with the data already present in path cal store 3 (which would have been created previously for an 'open' path cal).

:SCALar
:PCAL
:SHORt
:MERGe
:SEILoss

Parameters: <NRf>
store number

Description: Initiate a 'short' path calibration, merging the data with the already stored in the specified path calibration store. This command is used after a :SCAL:PCAL:OPEN:SEIL to perform the second half of a Short & Open path calibration. Note that this command restarts averaging and requires the relevant number of sweeps to be completed, which is equal to the average number that has been set.
See [CAL] / Cal for more details.

Example: `:SCAL:PCAL:SHOR:MERG:SEIL 3`
Perform a 'short' path calibration and merge with the data already present in path cal store 3 (which would have been created previously for an 'open' path cal).

:SCALar
:PCAL
[:STATe]

Parameters: <BOOLEAN PROGRAM DATA>
state

Description: Enable/disable path calibration. This is applied on the sweep after the current sweep.
See [CAL] / Cal for more details.

Example: `:SCAL:PCAL OFF`
Switch path calibration off. This is the normal use for the command, since saving a measurement to a path cal memory enables path calibration automatically.

:SCALar
:PCAL
:STATe?

Parameters: [<BOOLEAN PROGRAM DATA>]
[state]

Response: <BOOLEAN RESPONSE DATA>
state

Example: `:SCAL:PCAL:STAT?`
Determine whether path calibration is enabled.

:SCALar
:PCAL
:STORe

Parameters: <NRf>,<ARBITRARY BLOCK PROGRAM DATA>
store number,store contents

Description: Write data to a path calibration store.
See [CAL] / Cal for more details.

Example: `:SCAL:PCAL:STOR 1,#...etc`
Write path cal data to store 1. (Only first character of data is shown).

:SCALar
:PCAL
:STORe?

Parameters: <NRf>[,<ARBITRARY BLOCK PROGRAM DATA>]
store number[,store contents]

Response: <INDEFINITE LENGTH ARBITRARY BLOCK RESPONSE DATA>
store contents

Description: Read (and optionally write) data from a path calibration store.

Example : `:SCAL:PCAL:STOR? 2`
Read the contents of path calibration store 2.

:SCALar
:PCAL
THRough

Parameters: <NRf>
store number

Description: Initiate an 'through' path calibration, storing the data in the specified path calibration store. Note that this command restarts averaging and requires the relevant number of sweeps to be completed, which is equal to the average number that has been set.
See [CAL] / Cal for more details.

Example: `:SCAL:PCAL:THR 4`
Perform a 'throgh' path calibration. The data will be stored in path cal store 4.

:SCALar
:SAVe

Parameters: <NRf>
memory number

Description: Save the active measurement to a trace memory.
See [SAVE/RECALL] / Save/Recall for more details.

Example: `:SCAL:SAV 3`
Save measurement to trace memory 3.

:SCALar
:SCALing
:AUTO

Parameters: <CPD>
<autoscale mode>
where <autoscale mode> *is* ONCE | CONTinuous | OFF

Description: Set the autoscale mode:

ONCE	Autoscale once
CONTinuous	Enable continuous autoscaling
OFF	Disable continuous autoscaling

See [SCALING] key for more details.

Example: `:SCAL:SCAL:AUTO CONT`
Enable continuous autoscaling.

:SCALar
:SCALing
:AUTO?

Parameters: [<CPD>]
[<autoscale mode>]
where <autoscale mode> *is* ONCE | CONTinuous | OFF

Response: <CRD>
<continuous autoscale status>
where <continuous autoscale status> *is* CONT | OFF

Description: The scaling mode ONCE will never be returned by the query form of the command.

Example : `:SCAL:SCAL:AUTO?`
Determine whether continuous autoscaling is enabled.

:SCALar
:SCALing
:DIVision

Parameters: <NRf>
per division

Description: Set the graticule vertical scaling.
See [SCALING] key for more details.

Example: `:SCAL:SCAL:DIV 2`
Set a scale of 2 dB per division (assuming log format).

:SCALar
:SCALing
:DIVision?

Parameters: [<NRf>]
[per division]

Response: <NR2>
per division

Example: `:SCAL:SCAL:DIV?`
Read the vertical scaling.

:SCALar
:SCALing
:POSition

Parameters: <NRf>
reference position

Description: Set the reference position.
See [SCALING] key for more details.

Example: `:SCAL:SCAL:POS 5`
Select the centre graticule line to be the reference line.

:SCALar
:SCALing
:POSition?

Parameters: [<NRf>]
[reference position]

Response: <NR1>
reference position

Example: `:SCAL:SCAL:POS?`
Read the reference position.

:SCALar
:SCALing
:RLEVel

Parameters: <NRf>
reference level

Description: Set the reference level.
Reference level units depend on the display format: dB or dBm, W, VSWR, V, or Hz.
To determine the format in use, use :SCAL:FORM?.
See [SCALING] key for more details.

Example: `:SCAL:SCAL:RLEV -10`
Set the reference level to –10dB (assuming log format).

:SCALar
:SCALing:
RLEVel?

Parameters: [<NRf>]
[reference level]

Response: <NR2>
reference level

Example: `:SCAL:SCAL:RLEV?`
Read the reference level.

SOURce SUBSYSTEM

```
SOURce
        BLANking\?
        CALibration
                FSTore?
                MAXimum?
                PPSTore?
                SELect\?
                UPSTore\?
        CWFilter\?
        DOMain?
        FREQuency
                CENTer\?
                [CW]\?
                SPAN\?
                STANdard\?
                STARt\?
                STOP\?
        INTerleave?
        LEVelling\?
        LLISt
                NSEGments?
                RESet
                SEGMent\?
        MODE\?
        POWer
                LEVel\?
                STARt\?
                STOP\?
        RF\?
        SBANdwidth\?
        SELect\?
        SETup
        SWEep
                AUTO\?
                POINts\?
                TIME\?
        VIOutput
                CURRent
                        CONStant\?
                        STARt\?
                        STOP\?
                MODE\?
                VOLTage
                        CONStant\?
                        STARt\?
                        STOP\?
                VPGScale\?
```

:SOURce

:BLANking

Parameters: <BOOLEAN PROGRAM DATA>
state

Description: Set source RF blanking on or off.
See [SOURCE] / Source Funcs for more details.

Example: `:SOUR:BLAN ON`
Set source RF blanking on.

:SOURce

:BLANking?

Parameters: [<BOOLEAN PROGRAM DATA>]
[state]

Response: <BOOLEAN RESPONSE DATA>
state

Example: `:SOUR:BLAN?`
Determine whether source RF blanking is on or off.

:SOURce

:CALibration

:FSTore?

Response: <INDEFINITE LENGTH ARBITRARY BLOCK RESPONSE DATA>
store contents

Description: Read the source frequency calibration store data. (Note that this command is provided for diagnostic purposes. It is not possible to transfer frequency calibration data to the instrument).

Example: `:SOUR:CAL:FST?`
Read the source frequency calibration store data.

:SOURce

:CALibration

:MAXimum?

Response: <NR1>
number of stores

Description: Read number of user power calibration stores available, including those located on a memory card.

Example: `:SOUR:CAL:MAX?`
Determine number of stores available.

:SOURce

:CALibration

:PPSTore?

Response: <INDEFINITE LENGTH ARBITRARY BLOCK RESPONSE DATA>

Description: Read the source primary power calibration store data. (Note that this command is provided for diagnostic purposes. It is not possible to transfer primary power calibration data to the instrument).

Example: `:SOUR:CAL:PPST?`
Read the source primary power calibration store data.

:SOURce

:CALibration

:SELect

Parameters: <CPD>
<source power calibration store>
where <source power calibration store> *is* PRIMary | USR1 | USR2

Description: Select a source power calibration: default primary calibration, or one of the two user calibrations.
See [UTILITY] / Select Pwr Cal for more details.

Example: `:SOUR:CAL:SEL USR1`
Select user source power calibration 1.

:SOURce

:CALibration

:SELect?

Parameters: [<CPD>]
[<source power calibration store>]
where <source power calibration store> *is* PRIMary | USR1 | USR2

Response: <CRD>
<source power calibration store>

Example: `:SOUR:CAL:SEL?`
Determine the source power calibration in use.

:SOURce
:CALibration
:UPSTore

Parameters: <NRf>,<ARBITRARY BLOCK PROGRAM DATA>
User power calibration store number, store contents

Description: Send data to user source power calibration store.

Example: `:SOUR:CAL:UPST 1, #...etc`
Send user source power calibration data to user source power calibration store 1.
(Only first byte of data shown).

:SOURce
:CALibration
:UPSTore?

Parameters: <NRf>[,<ARBITRARY BLOCK PROGRAM DATA>]
user power calibration store number [,store contents]

Response: <INDEFINITE LENGTH ARBITRARY BLOCK RESPONSE DATA>
store contents

Example: `:SOUR:CAL:UPST? 2`
Read power calibration data from user source power calibration store 2.

:SOURce
:CWFilter

Parameters: <CPD>
<CW filter mode>
where <CW filter mode> *is* OFF | ON | AUTO

Description: Set the source CW filter mode.
See [SOURCE] / CW & Loop BW for more details.

Example: `:SOUR:CWF AUTO`
Set the CW filter to its default, Auto mode.

:SOURce
:CWFilter?

Parameters: [<CPD>]
[<CW filter mode>]
where <CW filter mode> *is* OFF | ON | AUTO

Response: <CRD>
<CW filter mode>

Example: `:SOUR:CWF?`
Read the source CW filter mode.

:SOURce

:DOMain?

Response: <NR2>, <NR2>, ...
domain at point 0, domain at point 1, ...

Description: Read the domain values for every point in the sweep.

Example:

```
:SOUR:DOM?
```

Read source domain values.

:SOURce

:FREQuency

:CENTer

Parameters: <NRf>
frequency (Hz)

Description: Set the sweep centre frequency. (:SOUR:MODE FREQ should be used to enable frequency sweep mode).
The instrument will also accept the English spelling, "CENTre".
See [SOURCE] / Source (Centre/Span Frequency Sweep Mode) for more details.

Example:

```
:SOUR:FREQ:CENT 12.4E9
```

Set centre frequency to 12.4 GHz.

:SOURce

:FREQuency

:CENTer?

Parameters: [<NRf>]
[frequency (Hz)]

Response: <NR2>
frequency (Hz)

Example:

```
:SOUR:FREQ:CENT?
```

Read centre frequency.

:SOURce

:FREQuency

[:CW]

Parameters: <NRf>
frequency (Hz)

Description: Set the source frequency for CW mode or power sweep mode.
See [SOURCE] / Source (CW Mode)

Example: `:SOUR:FREQ:CW 26.5E9`
Set the CW frequency to 26.5 GHz.

:SOURce

:FREQuency

:CW?

Parameters: [<NRf>]
[frequency (Hz)]

Response: <NR2>
frequency (Hz)

Example: `:SOUR:FREQ:CW?`
Read the CW frequency.

:SOURce

:FREQuency

:SPAN

Parameters: <NRf>
frequency (Hz)

Description: Set the sweep frequency span about the current centre frequency. (:SOUR:MODE FREQ should be used to enable frequency sweep mode).
See [SOURCE] / Source (Centre/Span Frequency Sweep Mode) for more details.

Example: `:SOUR:FREQ:CENT 5E9;SPAN 100E6`
Set a frequency span of 100 MHz about a centre frequency of 5 GHz.

:SOURce

:FREQuency

:SPAN?

Parameters: [<NRf>]
[frequency (Hz)]

Response: <NR2>
frequency (Hz)

Example: `:SOUR:FREQ:SPAN?`
Read swept frequency span.

:SOURce
:FREQuency
:STANdard

Parameters: <CPD>
<frequency standard>
where <frequency standard> *is* INT | EX1 | EX10

Description: Select a source frequency standard: internal, external 1 MHz or external 10 MHz.
See [SOURCE] / Lev & F Std for more details.

Example: `:SOUR:FREQ:STAN EX1`
Select the rear panel FREQ STD INPUT/OUTPUT to accept a 1 MHz external frequency standard.

:SOURce
:FREQuency
:STANdard?

Parameters: [<CPD>]
[<frequency standard>]
where <frequency standard> *is* INT | EX1 | EX10

Response: <CRD>
<frequency standard>

Example: `:SOUR:FREQ:STAN?`
Determine the frequency standard in use.

:SOURce
:FREQuency
:STARt

Parameters: <NRf>
frequency (Hz)

Description: Set the sweep start frequency. (:SOUR:MODE FREQ should be used to enable frequency sweep mode).
See [SOURCE] / Source (Start/Stop Frequency Sweep Mode) for more details.

Example: `:SOUR:FREQ:STAR 25E6`
Set start frequency to 25 MHz.

:SOURce
:FREQuency
:STARt?

Parameters: [<NRf>]
[frequency (Hz)]

Response: <NR2>
frequency (Hz)

Example: `:SOUR:FREQ:STAR?`
Read start frequency.

:SOURce

:FREQuency

:STOP

Parameters: <NRf>
frequency (Hz)

Description: Set the sweep stop frequency. (:SOUR:MODE FREQ should be used to enable frequency sweep mode).
See [SOURCE] / Source (Start/Stop Frequency Sweep Mode) for more details.

Example: `:SOUR:FREQ:STOP 15E9`
Set stop frequency to 15 GHz.

:SOURce

:FREQuency

:STOP?

Parameters: [<NRf>]
[frequency (Hz)]

Response: <NR2>
frequency (Hz)

Example: `:SOUR:FREQ:STOP?`
Read stop frequency.

:SOURce
:INTerleave?

Parameters: <NR1>
number of interleave points

Description: Read the number of interleave points of the source.

Example: `:SOUR:INT?`
Determine the number of interleave points.

:SOURce
:LEVelling

Parameters: <CPD>
<levelling mode>
where <levelling mode> *is* INTernal | POSitive | NEGative | PMETer

Description: Set the source levelling mode:

INTernal	Internal levelling
POSitive	External levelling - for +ve detectors
NEGative	External levelling - for -ve detectors
PMETer	External levelling - power meter

See [SOURCE] / Lev & F Std for more details.

Example: `:SOUR:LEV PMET`
Enable power meter levelling.

:SOURce
:LEVelling?

Parameters: [<CPD>]
[<levelling mode>]
where <levelling mode> *is* INTernal | POSitive | NEGative | PMETer

Response: <CRD>
<levelling mode>

Example: `:SOUR:LEV?`
Read the source levelling mode.

:SOURce

:LLISt

:NSEGments?

Response: <NR1>
number of segments

Description: Read the number of segments that comprise a linear list sweep (for reflection analyzer channel).
See [SOURCE] / Source (Linear List Frequency Sweep) for more details.

Example: `:SOUR:LLIS:NSEG?`
Read number of segments.

:SOURce

:LLISt

:RESet

Description: Resets the internal linear list segment counter so that the next segment to be read or written is segment 0.
Used in conjunction with the :SOURce:LLISt:SEGMent command.

Example: `:SOUR:LLIS:RES`
Prepare segment list for reading/writing segments.

:SOURce
:LLISt
:SEGMent

Parameters:: <NRf>, <NRf>, <NRf>
start frequency (Hz), stop frequency (Hz), number of points

Description: Set up a linear list segment (for reflection analyzer channel).
To send a segment list:

1. Send :SOUR:LLIS:RES
   This resets the counter that addresses the list of segments.

2. Send segment data using: :SOUR:LLIS:SEGM
   After each :SOUR:LLIS:SEGM command is received, the counter increments automatically to address the next free segment.

See [SOURCE] / Source (Linear List Frequency Sweep) for more details.

Example:

```
:SOUR:LLIS:RES;SEGM 1E9, 2E9, 200; SEGM 5E9, 8E9, 100;
SEGM 10E9, 12E9, 100; SEGM 15E9, 17E9, 200
```

A linear list sweep is defined in 4 segments: 1 to 2 GHz/200 points, 5 to 8 GHz/100 points, 10 to 12 GHz/100 points, 15 to 17 GHz/200 points.

:SOURce
:LLISt
:SEGMent?

Parameters: [<NRf>, <NRf>, <NRf>]
[start frequency (Hz), stop frequency (Hz), number of points]

Response: <NR2>, <NR2>, <NR2>
start frequency (Hz), stop frequency (Hz), number of points

Description: Read a segment from the linear list segment store. To read all the segments in the list:

1. Send :SOUR:LLIS:RES
   This resets the counter that addresses the list of segments.

2. Send: :SOUR:LLIS:NSEG?
   This returns the number of segments in the list.

3. Read the segments using :SOUR:LLIS:SEGM?
   After each :SOUR:LLIS:SEGM? command is received, the counter increments automatically to address the next segment. An error will be generated if an attempt is made to read beyond the last valid segment.

Example:

```
:SOUR:LLIS:RES;NSEG?
```

Reset the store pointer and read the number of segments.

Example:

```
:SOUR:LLIS:SEGM?
```

Read the first segment.

:SOURce

:MODE

Parameters: <CPD>

<source mode>

where <source mode> *is* VOLTage | CURRent | POWer | FREQuency | CW | LIST | WAVeguide | LOWPass

Description: Set the source mode:

VOLTage	Voltage sweep (with CW)
CURRent	Current sweep (with CW)
POWer	Power sweep
FREQuency	Frequency sweep
CW	CW output
LIST	Linear list sweep
WAVeguide	Waveguide (warped) sweep
LOWPass	Low-pass sweep (for low-pass time domain)

The last three only apply to reflection analyzer measurements.

See [SOURCE] / Source Mode for more details.

Example:

```
:SOUR:MODE POW
```

Set power sweep mode.

:SOURce

:MODE?

Parameters: [<CPD>]

[<source mode>]

where <source mode> *is* VOLTage | CURRent | POWer | FREQuency | CW | LIST | WAVeguide | LOWPass

Response: <CRD>

<source mode>

Example:

```
:SOUR:MODE?
```

Read the source mode.

:SOURce

:POWer

:LEVel

Parameters: <NRf>
power level (dBm)

Description: Set the source output power level for CW and swept frequency modes.

Example: `:SOUR:POW:LEV -10`
Set the power level to −10 dBm.

:SOURce

:POWer

:LEVel?

Parameters: [<NRf>]
[power level (dBm)]

Response: <NR2>
power level (dBm)

Example: `:SOUR:POW:LEV? -99`
Set the source output power to minimum and read the actual level set.

:SOURce

:POWer

:STARt

Parameters: <NRf>
power level (dBm)

Description: Set the power sweep start. (Power sweep mode should be selected using :SOUR:MODE POW).
See [SOURCE] / Source (Power Sweep) for more details.

Example: `:SOUR:POW:STAR -10`
Set the start power to −10 dBm.

:SOURce

:POWer

:STARt?

Parameters: [<NRf>]
[power level (dBm)]

Response: <NR2>
power level (dBm)

Example: `:SOUR:POW:STAR?`
Read the start power.

:SOURce
:POWer
:STOP

Parameters: <NRf>
power level (dBm)

Description: Set the power sweep stop. (Power sweep mode should be selected using :SOUR:MODE POW).
See [SOURCE] / Source (Power Sweep) for more details.

Example: `:SOUR:POW:STOP 0`
Set the start power to 0 dBm.

:SOURce
:POWer
:STOP?

Parameters: [<NRf>]
[power level (dBm)]

Response: <NR2>
power level (dBm)

Example: `:SOUR:POW:STOP?`
Read the stop power.

:SOURce
:RF

Parameters: <BOOLEAN PROGRAM DATA>
state

Description: Set RF output on or off.

Example: `:SOUR:RF ON`
Set RF on.

:SOURce
:RF?

Parameters: [<BOOLEAN PROGRAM DATA>]
[state]

Response: <BOOLEAN RESPONSE DATA>
state

Example: `:SOUR:RF?`
Determine whether RF is on or off.

:SOURce
:SBANdwidth

Parameters: <CPD>
<loop bandwidth mode>
where <loop bandwidth mode> *is* LOW | HIGH | AUTO

Description: Set the synthesizer loop bandwith mode.
See [SOURCE] / CW & Loop BW for more details.

Example: `:SOUR:SBAN AUTO`
Set the synthesizer loop bandwidth mode to automatic.

:SOURce
:SBANdwidth?

Parameters: [<CPD>]
[<loop bandwidth mode>]
where <loop bandwidth mode> *is* LOW | HIGH | AUTO

Response: <CRD>
<loop bandwidth mode>

Example: `:SOUR:SBAN?`
Read the synthesizer loop bandwidth mode.

:SOURce
:SELect

Parameters: <CPD>
<source id>
where <source id> *is* INTernal | EXTernal

Description: Select either the internal or external source for set-up. Subsequent commands will apply to the selected source.
See [SOURCE] / Int/Ext Source for more details.

Example: `:SOUR:SEL EXT`
Ensure that all following source commands will apply to the external source.

:SOURce
:SELect?

Parameters: [<CPD>]
[<source id>]
where <source id> *is* INTernal | EXTernal

Response: <CRD>
<source id>

Example: `:SOUR:SEL?`
Determine which source set-up will be modified by future commands.

:SOURce
:SETup

Parameters: <ARBITRARY BLOCK PROGRAM DATA>
setup data

Description: Send data for source setup.

Example: `:SOUR:SET #...etc.`
Send source setup data. (Only first byte of data shown).

:SOURce
:SWEep
:AUTO

Parameters: <BOOLEAN PROGRAM DATA>
state

Description: Enable/disable automatic sweep time.
See[SOURCE] / Sweep Time for more details.

Example: `:SOUR:SWE:AUTO ON`
Enable automatic sweep time.

:SOURce
:SWEep
:AUTO?

Parameters: [<BOOLEAN PROGRAM DATA>]
[state]

Response: <BOOLEAN RESPONSE DATA>
state

Example: `:SOUR:SWE:AUTO?`
Determine whether automatic sweep time is on or off.

:SOURce
:SWEep
:POINts

Parameters: <NRf>
number of points

Description: Set number of points for swept modes. Does not affect the number of points in a fault location sweep.
See [SOURCE] / Source (Start/Stop Frequency Sweep Mode) for more details.

Example:
```
:SOUR:SWE:POIN 201
```
Set a 201 point sweep.

:SOURce
:SWEep
:POINts?

Parameters: [<NRf>]
[number of points]

Response: <NR1>
number of points

Example:
```
:SOUR:SWE:POIN?
```
Read number of points.

:SOURce
:SWEep
:TIME

Parameters: <NRf>
sweep time (seconds)

Description: Set the sweep time. (User-set sweep time should first be enabled using :SOUR:SWE:AUTO OFF).
See[SOURCE] / Sweep Time for more details.

Example:
```
:SOUR:SWE:TIME 0.5
```
Set a sweep time of 500 ms.

:SOURce
:SWEep
:TIME?

Parameters: [<NRf>]
[sweep time (seconds)]

Response: <NR2>
sweep time (seconds)

Example:
```
:SOUR:SWE:TIME?
```
Read the user-set sweep time.

:SOURce
:VIOutput
:CURRent
:CONStant

Parameters: <NRf>
current (A)

Description: Set the value of constant current output from VOLTAGE/CURRENT OUTPUT on rear panel. (Use :SOUR:VIO:MODE CCUR to select constant current mode).
See [SOURCE] / V/I Output for more details.

Example:
```
:SOUR:VIO:CURR:CONS 0.1
```
Set constant current output to 100 mA.

:SOURce
:VIOutput
:CURRent
:CONStant?

Parameters: [<NRf>]
current (A)

Response: <NR2>
current (A)

Example:
```
:SOUR:VIO:CURR:CONS?
```
Read constant current value.

:SOURce
:VIOutput
:CURRent
:STARt

Parameters: <NRf>
current (A)

Description: Set current sweep start. (Current sweep mode is enabled using SOURce:MODE command).
See [SOURCE] / Source (Current Sweep) for more details.

Example:
```
:SOUR:VIO:CURR:STAR 0.1
```
Set current sweep start to 100 mA.

:SOURce
:VIOutput
:CURRent
:STARt?

Parameters: [<NRf>]
[current (A)]

Response: <NR2>
current (A)

Example:
```
:SOUR:VIO:CURR:STAR?
```
Read the current sweep start.

:SOURce
:VIOutput
:CURRent
:STOP

Parameters: <NRf>
current (A)

Description: Set current sweep stop. (Current sweep mode is enabled using SOURce:MODE command).
See[SOURCE] / Source (Current Sweep) for more details.

Example:
```
:SOUR:VIO:CURR:STOP 0.15
```
Set current sweep stop to 150 mA.

:SOURce
:VIOutput
:CURRent
:STOP?

Parameters: [<NRf>]
[current (A)]

Response: <NR2>
current (A)

Example:
```
:SOUR:VIO:CURR:STOP?
```
Read the current sweep stop.

:SOURce
:VIOutput
:MODE

Parameters: <CPD>
<settable voltage/current output mode>
where <settable voltage/current output mode> *is*
VGHZ | CVOLtage | CCUR rent | CRECorder |VRAMp

Description: Set the mode for the VOLTAGE/CURRENT OUTPUT located on the rear panel. The available modes are volts per GHz, constant voltage, constant current, chart recorder and voltage ramp (0-10 V).
See [SOURCE] / V/I Output for more details.

Example:

```
:SOUR:VIO:MODE CREC
```

Set the VOLTAGE/CURRENT OUTPUT to chart recorder mode.

:SOURce
:VIOutput
:MODE?

Parameters: [<CPD>]
[<settable voltage/current output mode>]
where <settable voltage/current output mode> *is*
VGHZ | CVOLtage | CCUR rent | CRECorder |VRAMp

Response: <CRD>
<readable voltage/current output mode>
where <readable voltage/current output mode> *is*
VGHZ | CVOL | CCUR | CREC | VSW | CSW |VRAM

Description: Read (and optionally set) the VOLTAGE/CURRENT OUTPUT mode. Note that voltage sweep (VSW) and current sweep (CSW) modes may only be set using the :SOUR:MODE command.
See [SOURCE] / V/I Output for more details.

Example:

```
:SOUR:VIO:MODE?
```

Read the VOLTAGE/CURRENT OUTPUT mode.

:SOURce :VIOutput :VOLTage :CONStant

Parameters: <NRf>
voltage (V)

Description: Set the value of constant voltage output from VOLTAGE/CURRENT OUTPUT on rear panel. (Use :SOUR:VIO:MODE CVOL to select constant voltage mode).
See [SOURCE] / V/I Output for more details.

Example:
```
:SOUR:VIO:VOLT:CONS 0.2
```
Set constant voltage output to 200 mV.

:SOURce :VIOutput :VOLTage :CONStant?

Parameters: [<NRf>]
[voltage (V)]

Response: <NR2>
voltage (V)

Example:
```
:SOUR:VIO:VOLT:CONST?
```
Read constant voltage value.

:SOURce :VIOutput :VOLTage :STARt

Parameters: <NRf>
voltage (V)

Description: Set voltage sweep start. (Voltage sweep mode is enabled using SOURce:MODE command).
See [SOURCE] / Source (Voltage Sweep) for more details.

Example:
```
:SOUR:VIO:VOLT:STAR 2.5
```
Set voltage sweep start to 2.5 V.

:SOURce :VIOutput :VOLTage :STARt?

Parameters: [<NRf>]
[voltage (V)]

Response: <NR2>
voltage (V)

Example:
```
:SOUR:VIO:VOLT:STAR?
```
Read the voltage sweep start.

:SOURce
:VIOutput
:VOLTage
:STOP

Parameters: <NRf>
voltage (V)

Description: Set voltage sweep stop. (Voltage sweep mode is enabled using SOURce:MODE command).
See [SOURCE] / Source (Voltage Sweep) for more details.

Example: `:SOUR:VIO:VOLT:STOP 8.5`
Set voltage sweep stop to 8.5 V.

:SOURce
:VIOutput
:VOLTage
:STOP?

Parameters: [<NRf>]
[voltage (V)]

Response: <NR2>
voltage (V)

Example: `:SOUR:VIO:VOLT:STOP?`
Read the voltage sweep stop.

:SOURce
:VIOutput
:VPGScale

Parameters: <NRf>
scale factor

Description: Set the volts per GHz scaling to either 1.0 V/GHz or 0.5 V/GHz.
See [SOURCE] / V/I Output for more details.

Example: `:SOUR:VIO:VPGS 0.5`
Set 0.5 V/GHz.

:SOURce
:VIOutput
:VPGScale?

Parameters: [<NRf>]
[scale factor]

Response: <NR2>
scale factor

Example: `:SOUR:VIO:VPGS?`
Determine the volts per GHz scaling.

STATus COMMANDS

```
STATus
        OPERation
                CONDition?
                ENABle\?
                [EVENt]?
        PRESet
        QUEStionable
                CONDition?
                ENABle\?
                [EVENt]?
```

:STATus
:OPERation
:CONDition?

Response: <NR1>
operation status register contents

The meaning of each bit in the Operation Status Register is given on page App. A-3.

Description: Read the contents of the Operation Status Register.

Example: `:STAT:OPER:COND?`

:STATus
:OPERation
:ENABle

Parameters: <NRf>
mask

Description: Sets the enable mask which allows true conditions in the Operation Status Event Register to be reported in the summary bit. If a bit is 1 in the enable register and its associated event bit makes a transition to true, a positive transition will occur in the associated summary bit.
See page App. A-3 for more details

Example: `:STAT:OPER:ENAB 32`
Program the status register with the mask value 32 (0000 0000 0010 0000 in binary) to enable a positive transition in the summary bit when the instrument is waiting for trigger.

:STATus
:OPERation
:ENABle?

Parameters: [<NRf>]
[mask]

Response: <NR1>
mask

Example: `:STAT:OPER:ENAB?`
Read the enable mask for the Operation Status Register.

:STATus
:OPERation
[:EVENt]?

Response: <NR1>
event register contents

Description: Read the contents of the event register associated with the Operation Status Register.
See page App. A-4 for more details

Example: `STAT:OPER?`
Determine what caused the event.

:STATus
:PRESet

Description: Preset the OPERation:ENABle Register and the QUEStionable:ENABle Register to zero.

Example: `:STAT:PRES`

:STATus
:QUEStionable
:CONDition?

Response: <NR1>
questionable status register contents

The meaning of each bit in the Questionable Status Register is given on page *App. A-4*.

Description: Read the contents of the Questionable Status Register.

Example: `:STAT:QUES:COND?`

:STATus
:QUEStionable
:ENABle

Parameters: <NRf>
mask

Description: Sets the enable mask which allows true conditions in the Questionable Status Event Register to be reported in the summary bit. If a bit is 1 in the enable register and its associated event bit makes a transition to true, a positive transition will occur in the associated summary bit.
See page App. A-4 for more details

Example:
```
:STAT:QUES:ENAB 8
```
Program the status register with the mask value 8 (0000 0000 0000 1000 in binary) to enable a positive transition in the summary bit when the source becomes unlevelled.

:STATus
:QUEStionable
:ENABle?

Parameters: [<NRf>]
[mask]

Response: <NR1>
mask

Example:
```
:STAT:QUES:ENAB?
```
Read the enable mask for the Questionable Status Register.

:STATus
:QUEStionable
[:EVENt]?

Response: <NR1>
event register contents

Description: Read the contents of the event register associated with the Questionable Status Register.
See page App. A-4 for more details

Example:
```
STAT:QUES?
```
Determine what caused the event.

SYSTem SUBSYSTEM

```
SYSTem
    ADDRess
        PLOTter\?
        SOURce\?
    APPLication
        LOAD
        PARameter\?
        RUN
    DATE\?
    DISPlay
        CPALette\?
        FANNotation\?
        FMODe\?
        FOFFset\?
        FSCaling\?
        LCD\?
        MTITle
            [STATe]\?
            STRing\?
        POFFset\?
        STITle
            [STATe]\?
            STRing\?
    ERRor?
    HOLD
    KEYBoard
        COUNtry\?
    MODE\?
    PRESet
    SERVice
        DACQ
            CORRection\?
            MUX\?
            RANGe\?
            [STATe]\?
        SOURce
            ADC?
            FLAGs?
            MUX\?
            [STATe]\?
    SETTings
        CURRent?
        MAXimum?
        RECall
        SAVE
        STORe\?
    TIME\?
    TRIGger\?
```

:SYSTem
:ADDRess
:PLOTter

Parameters: <NRf>
address

Description: Set the GPIB address that the instrument will use for controlling an HPGL plotter.
See [UTILITY] / GPIB for more details.

Example:
```
:SYST:ADDR:PLOT 5
```
Set plotter address to 5.

:SYSTem
:ADDRess
:PLOTter?

Parameters: [<NRf>]
[address]

Response: <NR1>
address

Example:
```
:SYST:ADDR:PLOT?
```
Read GPIB address used for controlling the plotter.

:SYSTem
:ADDRess
:SOURce

Parameters: <NRf>
address

Description: Set the GPIB address that the instrument will use for controlling an external source (or second instrument).
See [UTILITY] / GPIB for more details.

Example:
```
:SYST:ADDR:SOUR 18
```
Set source address to 18.

:SYSTem
:ADDRess
:SOURce?

Parameters: [<NRf>]
[address]

Response: <NR1>
address

Example:
```
:SYST:ADDR:SOUR?
```
Read GPIB address used for controlling the external source.

:SYSTem

:APPLication

:LOAD

Parameters: <STRING PROGRAM DATA>
name of application to be loaded

Description: Loads the named application.

Example: `:SYST:APPL:LOAD "Gain Compression"`
Load the Gain Compression application.

:SYSTem

:APPLication

:PARameter

Parameters: <NRf>,<NRf>
<parameter identity>,<parameter value>,
where <parameter identity> *and* <parameter value> *are application dependent*
See the application User Manual for details.

Description: Set the parameters for the current application.

:SYSTem

:APPLication

: PARameter?

Parameters: <NRf>
<parameter identity>

Response: <NR2>
parameter value

Description: Reads the value of the specified parameter .

:SYSTem

:APPLication

:RUN

Parameters: <NRf>
<function identity>
where <function identity> is *application dependent*
See the application User Manual for details.

Description: Runs the specified function in the current application.

Example: `:SYST:APPL:RUN 2`
Perform function number 2 in the current application.

:SYSTem
:DATE

Parameters: <NRf>, <NRf>, <NRf>
year, month, day

Description: Sets the date of the real-time clock.
See UTILITY / Date & Time for more details.

Example: `:SYST:DATE, 1993, 8, 20`
Set the date to 20th August 1993.

:SYSTem
:DATE?

Parameters: [<NRf>, <NRf>, <NRf>]
[year, month, day]

Response: <NR1>, <NR1>, <NR1>
year, month, day

Example: `:SYST:DATE?`
Read the date of the real-time clock.

:SYSTem
:DISPlay
:CPALette

Parameters: <CPD>
<palette>
where <palette> *is* COLour|WHITe|BLACk|GREen

Description: Sets the colour palette that the LCD uses to either the standard colour palette (COL), white on black (WHIT), black on white (BLAC) or shades of green (GRE).
See [UTILITY] / Colour Palette for more details.

Example: `:SYST:DISP:CPAL WHIT`
Set the colour palette to white on black.

:SYSTem
:DISPLay
:CPALette?

Parameters: [<CPD>]
[<palette>]
where <palette> *is* COLour|WHITe|BLACk|GREen

Response: <CRD>
<palette>

Example: `:SYST:DISP:CPAL?`
Determine which colour palette is being used.

:SYSTem
:DISPlay
:FANNotation

Parameters: <BOOLEAN PROGRAM DATA>
state

Description: Set frequency annotation on display (and hard copy) on or off.
See [UTILITY] / Display Set-up for more details.

Example: `:SYST:DISP:FANN OFF`
Remove frequency annotation from display.

:SYSTem
:DISPlay
:FANNotation?

Parameters: [<BOOLEAN PROGRAM DATA>]
[state]

Response: <BOOLEAN RESPONSE DATA>
state

Example: `:SYST:DISP:FANN?`
Determine whether frequency annotation is displayed.

:SYSTem
:DISPlay
:FMODe

Parameters: <CPD>
<scaling mode>
where <scaling mode> *is* USER | 6255 | 6256

Description: Set frequency scaling mode as follows:

USER Frequency offset and scaling is specified using the :SYST:DISP:FOFF and :SYST:DISP:FSC commands.

6255 Frequency scaling factor = 4.

6256 Frequency scaling factor = 6.

When the mode is set to '6255' or '6256' the :SYST:DISP:FOFF and :SYST:DISP :FSC commands will result in an error.

See [MEASURE] / Scale & Offset for more details.

Example:

```
:SYST:DISP:FMOD 6255
```

Set the frequency scaling factor to 4.

:SYSTem
:DISPlay
:FMODe?

Parameters: [<CPD>]
[<scaling mode>]
where <scaling mode> *is* USER | 6255 | 6256

Response: <CRD>
<scaling mode>

Example:

```
:SYST:DISP:FMOD?
```

Determine the frequency scaling mode that is being used.

:SYSTem
:DISPlay
:FOFFset

Parameters: <NRf>
frequency offset (Hz)

Description: Set displayed frequency offset for a scalar channel.
See [MEASURE] / Scale & Offset for more details.

Example:
```
:SYST:DISP:FOFF 5.0E+9
```
Offset displayed frequency information by 5 GHz.

:SYSTem
:DISPlay
:FOFFset?

Parameters: [<NRf>]
[frequency offset (Hz)]

Response: <NR2>
frequency offset (Hz)

Example:
```
:SYST:DISP:FOFF?
```
Read frequency offset.

:SYSTem
:DISPlay:
:FSCaling

Parameters: <NRf>
frequency scale factor

Description: Set displayed frequency scale factor for a scalar channel.
See [MEASURE] / Scale & Offset for more details.

Example:
```
:SYST:DISP:FSC 1.5
```
Apply a displayed frequency scale factor of 1.5.

:SYSTem
:DISPlay
:FSCaling?

Parameters: [<NRf>]
[frequency scale factor]

Response: <NR2>
frequency scale factor

Example:
```
:SYST:DISP:FSC?
```
Read the frequency scale factor.

:SYSTem
:DISPlay
:LCD

Parameters: <CPD>
<staus>
where <status> *is* OFF | HALF | FULL

Description: Set the backlight to be either Off, Half Brightness or Full Brightness.

Example:
```
:SYST:DISP:LCD FULL
```
Set the display toFull Brightness.

:SYSTem
:DISPlay
:LCD?

Parameters: [<CPD>]
[<status>]
where <status> *is* OFF | HALF | FULL

Response: <CRD>
<status>

Example:
```
:SYST:DISP:LCD?
```
Return the current state of the backlight.

:SYSTem
:DISPlay
:MTITle
[:STATe]

Parameters: <BOOLEAN PROGRAM DATA>
state

Description: Set measurement title on or off for the active measurement.
See [UTILITY] / Channel 1 Meas Titles for more details.

Example:

```
:SYST:DISP:MTIT ON
```

Enable measurement title display.

:SYSTem
:DISPlay
:MTITle
:STATe?

Parameters: [<BOOLEAN PROGRAM DATA>]
[state]

Response: <BOOLEAN RESPONSE DATA>
state

Example:

```
:SYST:DISP:MTIT:STAT?
```

Determine whether measurement title is displayed.

:SYSTem
:DISPlay
:MTITle
:STRing

Parameters: <STRING PROGRAM DATA>
measurement title string

Description: Send a measurement title for display. Use :SYST:DISP:MTIT ON to display the measurement title.

Example:

```
:SYST:DISP:MTIT:STR "13.6 GHz LPF"
```

Send the measurement title: 13.6 GHz LPF.

:SYSTem
:DISPlay
:MTITle
:STRing?

Parameters: [<STRING PROGRAM DATA>]
[measurement title string]

Response: <STRING RESPONSE DATA>
measurement title string

Example:

```
:SYST:DISP:MTIT:STR?
```

Read the measurement title.

:SYSTem
:DISPlay
:POFFset

Parameters: <NRf>
power offset (dB)

Description: Set displayed power offset.
See [MEASURE] / Scale & Offset for more details.

Example: `:SYST:DISP:POFF 1.5`
Apply a displayed power offset of 1.5 dB.

:SYSTem
:DISPlay
:POFFset?

Parameters: [<NRf>]
[power offset (dB)]

Response: <NR2>
power offset (dB)

Example: `:SYST:DISP:POFF?`
Read the displayed power offset.

:SYSTem
:DISPlay
:STITle
[:STATe]

Parameters: <BOOLEAN PROGRAM DATA>
state

Description: Set screen title on or off.
See [UTILITY] / Titles for more details.

Example: `:SYST:DISP:STIT ON`
Enable screen title display.

:SYSTem
:DISPlay
:STITle
:STATe?

Parameters: [<BOOLEAN PROGRAM DATA>]
[state]

Response: <BOOLEAN RESPONSE DATA>
state

Example: `:SYST:DISP:STIT:STAT?`
Determine whether screen title is displayed.

:SYSTem
:DISPlay
:STITle
:STRing

Parameters: <STRING PROGRAM DATA>
screen title string

Description: Send a screen title for display. Use :SYST:DISP:STIT ON to display the screen title.

Example:

```
:SYST:DISP:STIT:STR "This is a SCREEN TITLE"
```

Send the screen title: This is a SCREEN TITLE.

:SYSTem
:DISPlay
:STITle
:STRing?

Parameters: [<STRING PROGRAM DATA>]
[screen title string]

Response: <STRING RESPONSE DATA>
screen title string

Example:

```
:SYST:DISP:STIT:STR?
```

Read the screen title.

:SYSTem
:ERRor?

Response: <NR1>,<STRING RESPONSE DATA>
error number, error message string

Description: Read the SCPI 1990.0 error number and error message from the head of the error queue.

Example:

```
:SYST:ERR?
```

Example Response:

```
-112,"Program mnemonic too long"
```

:SYSTem
:HOLD

Parameters: <BOOLEAN PROGRAM DATA>
state

Description: Hold active measurement at the end of the current update.
See [HOLD] key description for more details.

Example:

```
:SYST:HOLD OFF
```

Release held measurement.

:SYSTem

:KEYBoard

:COUNtry

Parameters: <CPD>
<country>
where <country> *is* FRANce | GERMany | SPAin | UK | USA

Description: Specifies the country for the external IBM keyboard that is to be used with the MTS, ie France, Germany, Spain, United Kingdom or the USA.
See [UTILITY]/International for more details.

Example:

```
:SYST:KEYB:COUN SPA
```

Set up the instrument to use a Spanish keyboard.

:SYSTem

:KEYBoard

:COUNtry?

Parameters: [<CPD>]
[<country>]
where <country> *is* FRANce | GERMany | SPAin | UK | USA

Response: <CRD>
<country>

Example:

```
:SYST:KEYB:COUN?
```

Determine the keyboard type the instrument is set up to use.

:SYSTem

:MODE

Parameters: <CPD>
<mode>
where <mode> *is* MTS | SOURce | SLAVe

Description: Set the instrument to operate in measurement mode (MTS), in source-only mode (SOURce) or as a source with digital sweep control, (SLAVe).

Example:

```
:SYST:MODE MTS
```

Set the instrument in MTS mode.

:SYSTem

:MODE?

Parameters: [<CPD>]
[<mode>]
where <mode> *is* MTS | SOURce | SLAVe

Response: <CRD>
<mode>

Example:

```
:SYST:MODE?
```

Read the instrument mode.

:SYSTem

:PRESet

Description: Place the instrument in its default state, as defined in Appendix A.
See also *RST Common Commands

Example:
```
:SYST:PRES
```
Preset the instrument.

:SYSTem

:SERVice

:DACQ

:CORRection

Parameters: <BOOLEAN PROGRAM DATA>
state

Description: Set data acquisition correction on or off.
This command is only available when data acquisition service mode is enabled.
Use :SYST:SERV:DACQ ON to enable the service mode.
See [UTILITY] / DAcq Diag for more details.

Example:
```
:SYST:SERV:DACQ:CORR OFF
```
Disable data acquisition correction.

:SYSTem

:SERVice

:DACQ

:CORRection?

Parameters: [<BOOLEAN PROGRAM DATA>]
[state]

Response: <BOOLEAN RESPONSE DATA>
state

Example:
```
:SYST:SERV:DACQ:CORR?
```
Determine whether data acquisition correction is enabled.

:SYSTem

:SERVice

:DACQ

:MUX

Parameters: <NRf>
multiplexer setting

Description: Set data acquisition multiplexer setting.

This command is only available when data acquisition service mode is enabled.
Use: SYST:SERV:DACQ ON to enable the service mode.
See [UTILITY] / Set Up Input for more details.

Example:
```
:SYST:SERV:DACQ:MUX 2
```
Set data acquisition amplifier multiplexer setting.

:SYSTem

:SERVice

:DACQ

:MUX?

Parameters: [<NRf>]
[multiplexer setting]

Response: <NR1>
multiplexer setting

Example:
```
:SYST:SERV:DACQ:MUX?
```
Read the data acquisition multiplexer setting.

:SYSTem
:SERVice
:DACQ
:RANGe

Parameters: <NRf>
amplifier range

Description: Set data acquisition amplifier range.
This command is only available when data acquisition service mode is enabled.
Use :SYST:SERV:DACQ ON to enable the service mode.
See [UTILITY] / DAcq Diag for more details.

Example:
```
:SYST:SERV:DACQ:RANG 2
```
Set data acquisition amplifier range to 2.

:SYSTem
:SERVice
:DACQ
:RANGe?

Parameters: [<NRf>]
[amplifier range]

Response: <NR1>
amplifier range

Example:
```
:SYST:SERV:DACQ:RANG?
```
Read the data acquisition amplifier range.

:SYSTem
:SERVice
:DACQ
[:STATe]

Parameters: <BOOLEAN PROGRAM DATA>
state

Description: Enable/disable the data acquisition service mode.
See [UTILITY] / DAcq Diag for more details.

Example: `:SYST:SERV:DACQ ON`
Enable data acquisition service mode

:SYSTem
:SERVice
:DACQ
:STATe?

Parameters: [<BOOLEAN PROGRAM DATA>]
[state]

Response: <BOOLEAN RESPONSE DATA>
state

Example: `:SYST:SERV:DACQ:STAT?`
Determine whether instrument is in data acquisition service mode.

:SYSTem
:SERVice
:SOURce
:ADC?

Response: <NR1>
source ADC value

Description: Read the source ADC value.

Example: `:SYST:SERV:SOUR:ADC?`
Determine the source ADC value.

:SYSTem
:SERVice
:SOURce
:FLAGs?

Response: <NR1>
source status word

Description: Read the source status word.

Example: `:SYST:SERV:SOUR:FLAG?`
Determine the source status word.

:SYSTem
:SERVice
:SOURce
:MUX

Parameters: <NRf>
multiplexer setting

Description: Set the source multiplexer setting.
This command is only available when the source service mode is enabled.
See [UTILITY] / Set Up Input for more details.

Example: `:SYST:SERV:SOUR:MUX 2`
Set the source multiplexer setting.

:SYSTem
:SERVice
:SOURce
:MUX?

Parameters: [<NRf>]
[multiplexer setting]

Response: <NR1>
multiplexer setting

Example: `:SYST:SERV:SOUR:MUX?`
Read the source multiplexer setting.

:SYSTem
:SERVice
:SOURce
:[STATe]

Parameters: <BOOLEAN PROGRAM DATA>
state

Description: Enable/disable the source service mode.

Example: `:SYST:SERV:SOUR ON`
Enable source service mode.

:SYSTem
:SERVice
:SOURce
:STATe?

Parameters: [<BOOLEAN PROGRAM DATA>]
[state]

Response: <BOOLEAN RESPONSE DATA>
state

Example: `:SYST:SERV:SOUR:STAT?`
Determine whether instrument is in source service mode.

:SYSTem
:SETTings
:CURRent?

Response: <INDEFINITE LENGTH ARBITRARY BLOCK RESPONSE DATA>

Description Read settings data that instrument is currently using.

Example: `:SYST:SETT:CURR?`
Read current settings data

:SYSTem
:SETTings
:MAXimum?

Response: <NR1>
number of instrument settings stores

Description: Read the total number of instrument settings stores available, including those located on a memory card.

Example: `:SYST:SETT:MAX?`
Determine how many settings stores are available.

:SYSTem
:SETTings
:RECall

Parameters: <NRf>
settings store number

Description: Recall instrument settings from a non-volatile store.
See [SAVE/RECALL] / Save/Recall for more details.

Example: `:SYST:SETT:REC 3`
Recall settings from store 3.

:SYSTem
:SETTings
:SAVE

Parameters: <NRf>
settings store number

Description: Save current instrument settings to a non-volatile store.
See [SAVE/RECALL] / Save/Recall for more details.

Example: `:SYST:SETT:SAVE 3`
Save instrument settings in store number 3.

:SYSTem
:SETTings
:STORe

Parameters: <NRf> , <ARBITRARY BLOCK PROGRAM DATA>
settings store number, store contents

Description: Send instrument settings data to a non-volatile store.

Example: `:SYST:SETT:STOR 4,#...etc`
Send settings data (of which only the first byte is shown) to instrument settings store 4.

:SYSTem
:SETTings
:STORe?

Parameters: <NRf>[,<ARBITRARY BLOCK PROGRAM DATA>]
settings store number, [store contents]

Response: <INDEFINITE LENGTH ARBITRARY BLOCK RESPONSE DATA>
store contents

Example: `:SYST:SETT:STOR? 5`
Read settings data from instrument settings store 5.

:SYSTem
:TIME

Parameters: <NRf>, <NRf>, <NRf>
hours, minutes, seconds

Description: Sets the time of the real-time clock.
See the UTILITY / Date & Time menu for more details.

Example: `:SYST:TIME, 14, 30, 0`
Set the time to 2:30 pm.

:SYSTem
:TIME?

Parameters: [<NRf>, <NRf>, <NRf>]
[hours, minutes, seconds]

Response: <NR1>, <NR1>, <NR1>
hours, minutes, seconds

Example: `:SYST:TIME?`
Read the time of the real-time clock.

SYSTem
:TRIGger

Parameters: <CPD>
<trigger mode>
where <trigger mode> *is* AUTO | MEASurement

Description: Set the GPIB trigger mode.
AUTO is the normal "free run" mode. When set to MEASurement mode, the instrument updates all displayed channels on receipt of a group execute trigger or *TRG common command.

Example: `:SYST:TRIG AUTO`
Enable auto-trigger mode.

:SYSTem
:TRIGger?

Parameters: [<CPD>]
[<trigger mode>]
where <trigger mode> *is* AUTO | MEASurement.

Response: <CRD>
<trigger mode>

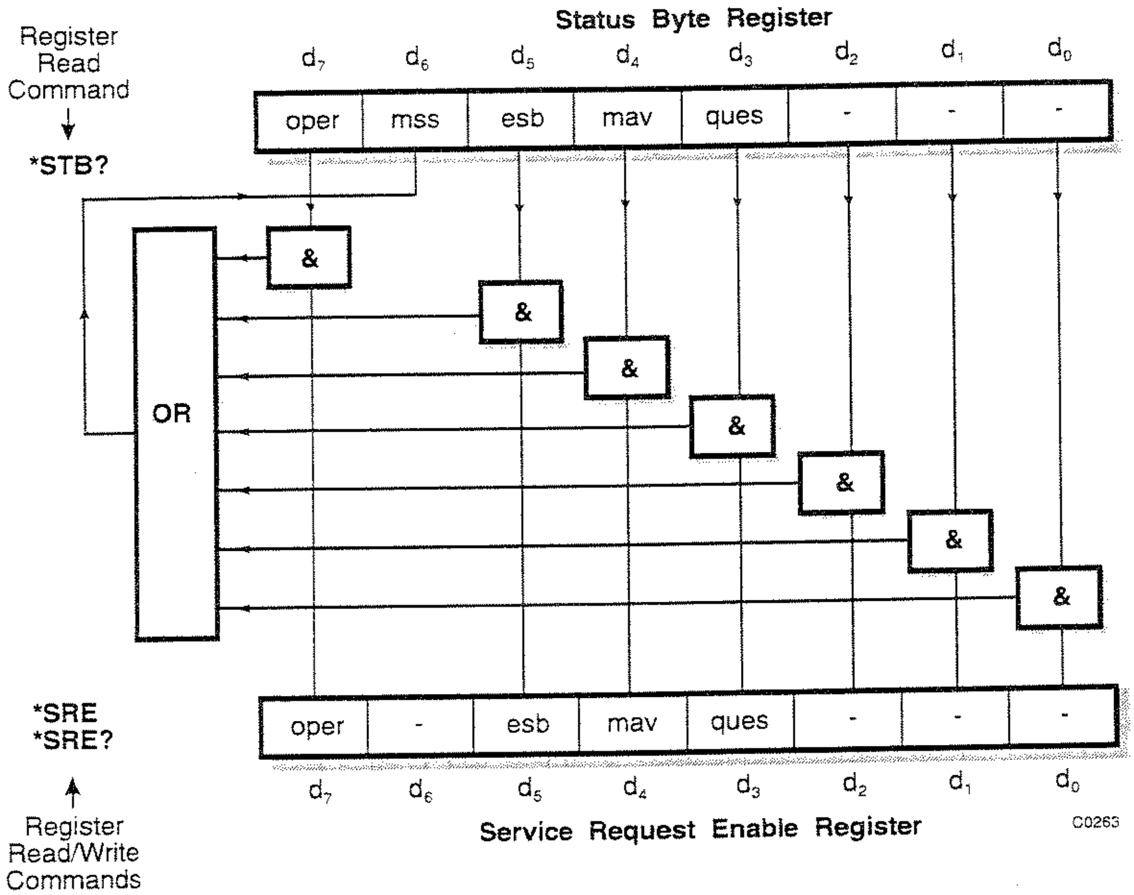
Example: `:SYST:TRIG?`
Determine trigger mode.

Chapter 4

GPIB STATUS REPORTING STRUCTURE

Appendix A
GPIB STATUS REPORTING STRUCTURE

STATUS BYTE WHEN READ BY *STB



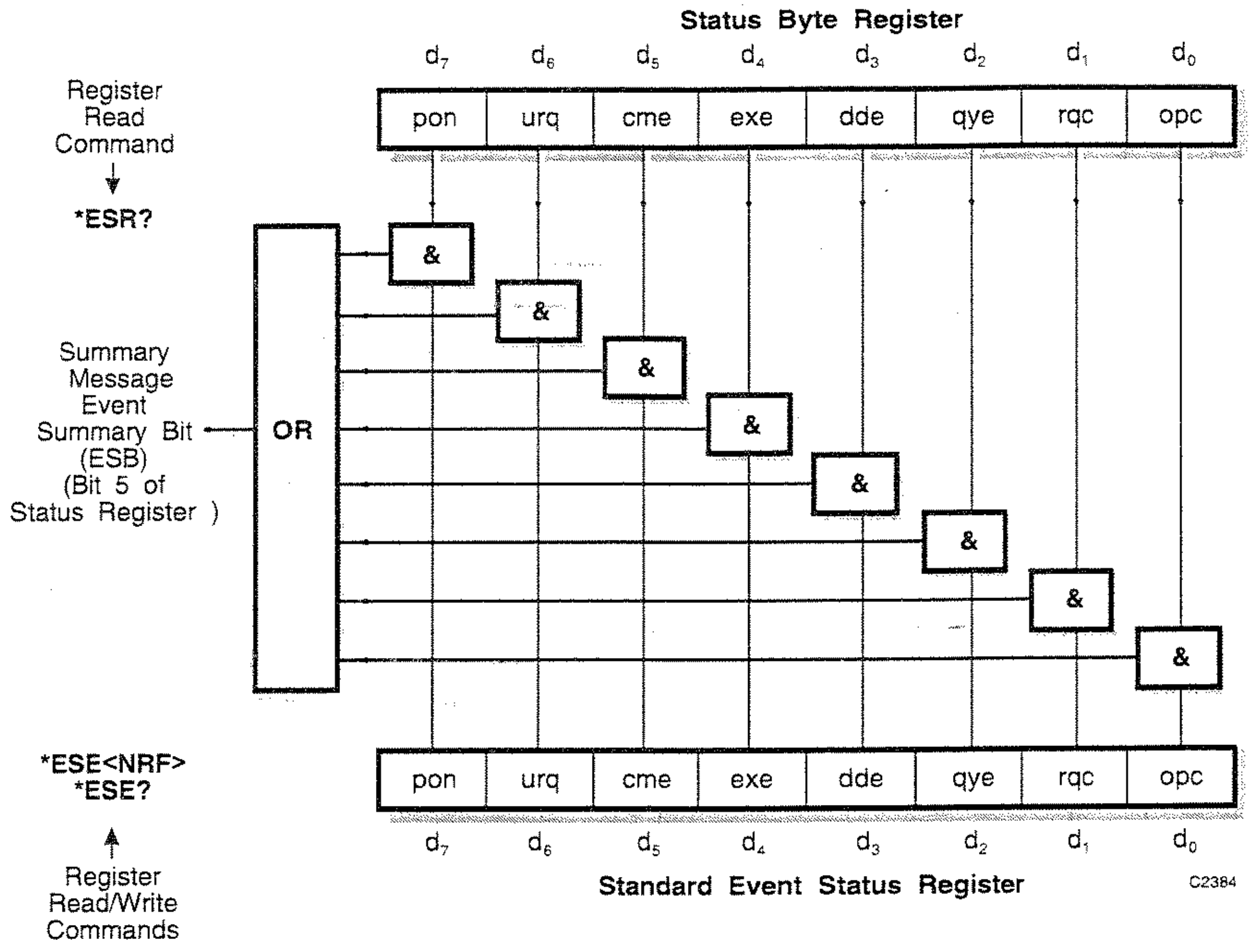
d_0	Not used	
d_1	Not used	
d_2	Not used	
d_3	QUES	Questionable Status Register Summary Bit
d_4	MAV	Message available in output queue
d_5	ESB	Event Status Register Summary Bit
d_6	MSS	True when (Status Byte > 0) AND (Enable Reg > 0)
d_7	OPER	Operation Status Register Summary Bit

Notes...

When read by Serial Poll (rather than *STB?), d_6 contains RQS (Request Service) as defined in IEEE 488.2.

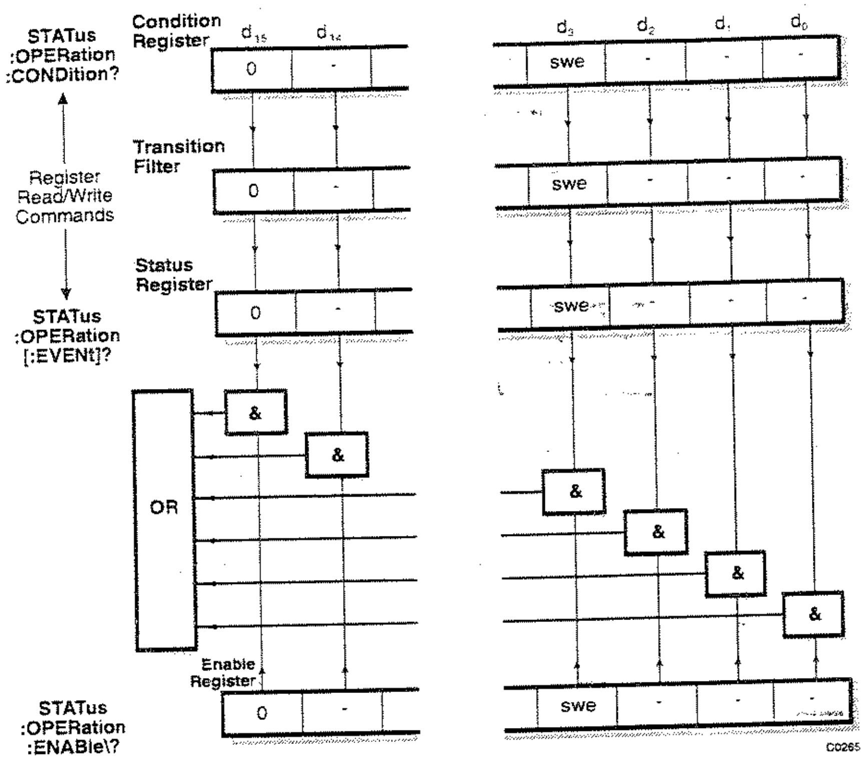
*SRE? always returns 0 for bit d_6.

STANDARD EVENT REGISTER (AS DEFINED IN IEEE 488.2 1987)


Status Byte Register
d7 d6 d5 d4 d3 d2 d1 d0
pon urq cme exe dde qye rqc opc
Register Read Command
*ESR?
Summary Message Event Summary Bit (ESB) (Bit 5 of Status Register)
OR
&
*ESE<NRF>
*ESE?
Register Read/Write Commands
Standard Event Status Register
C2384

d_0	OPC	Operation Complete.
d_1	RQC	Request Control. The instrument is requesting permission to become the IEEE 488.1 controller in charge.
d_2	QYE	Query Error
d_3	DDE	Device-Specific Error
d_4	EXE	Execution Error
d_5	CME	Command Error
d_6	URQ	User Request - Not implemented in this instrument.
d_7	PON	Power on.

OPERATION STATUS REGISTER



BIT	MNEM	TRANSITION	DESCRIPTION
d_0	MCAL	NEG	Measuring for calibration
d_1			Not Used
d_2			Not Used
d_3	SWE	NEG	Sweeping
d_4			Not Used
d_5	TRIG	POS	Waiting for Trigger
d_6			Not Used
d_7			Not Used
d_8	AV11	NEG	Averaging: Chan 1 Meas 1
d_9	AV12	NEG	Averaging: Chan 1 Meas 2
d_{10}	AV21	NEG	Averaging: Chan 2 Meas 1
d_{11}	AV22	NEG	Averaging: Chan 2 Meas 2
d_{12}			Not Used
d_{13}			Not Used
d_{14}	ABSY	not applicable	Application busy
d_{15}			Always zero

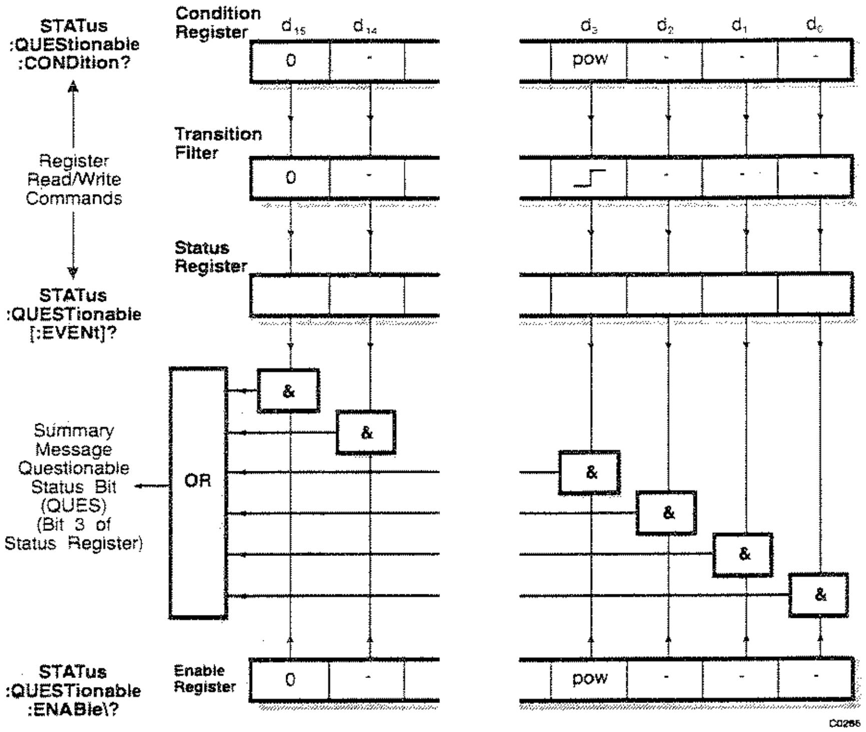
Notes...

When displaying a power ratio or difference measurement on a readout channel, each input may be assigned a different averaging number. A measurement A/B could be defined, for example, in which measurements from input B are averaged more than those from input A. In this case, the averaging bit for the measurement will reset when measurements from **both** input A and input B have reached their respective averaging targets.

Since the averaging bits may change unexpectedly, the GPIB programmer may prefer to use manual set averaging rather than automatic.

MCAL is set whenever sweeps are being performed for a scalar, fault location, or reflection analyzer calibration. For a short/open scalar cal the bit is set for the duration of the short sweeps, and then cleared, set again for the duration of the open sweeps, and then cleared. For a reflection analyzer cal it is set and cleared for each piece to be measured. This removes the need to determine how many sweeps are required for each piece.

QUESTIONABLE STATUS REGISTER



BIT	MNEM	TRANSITION	DESCRIPTION
d_0			Not Used
d_1			Not Used
d_2			Not Used
d_3	POW	POS	Source power unlevelled
d_4	XSTD	POS	External standard not present or standard unlocked
d_5	FREQ	POS	Source unlocked
d_6			Not Used
d_7			Not Used
d_8	CAL	POS	Calibration failure
d_9	LIM11	POS	Out of limits: Chan 1 Meas 1
d_{10}	LIM12	POS	Out of limits: Chan 1 Meas 2
d_{11}	LIM21	POS	Out of limits: Chan 2 Meas 1
d_{12}	LIM22	POS	Out of limits: Chan 2 Meas 2
d_{13}			Not Used
d_{14}			Not Used
d_{15}			Always zero

Notes...

Calibration failure, bit d_8, indicates that calibration data for the power reference output or the internal frequency standard has been corrupted and the instrument is using default calibration data. The bit is reset when both calibrations are again valid.