

# Keysight N774-C & 8162-C Series

N7742C, N7743C, N7744C & N7745C Optical Power Meter

N7747C & N7748C Optical High Performance Power Meter

N7749C Optical Head Interface

81620C, 81623C, 81624C, 81626C & 81628C Optical Power Head

Programming  
Guide

# Notices

© Keysight Technologies 2021

No part of this manual may be reproduced in any form or by any means (including electronic storage and retrieval or translation into a foreign language) without prior agreement and written consent from Keysight Technologies as governed by United States and international copyright laws.

## Manual Part Number

N7740-90C02

## Edition

Edition 2.0, April 2021

Keysight Technologies Deutschland GmbH  
Herrenberger Strasse 130,  
71034 Böblingen, Germany

## Technology Licenses

The hardware and/or software described in this document are furnished under a license and may be used or copied only in accordance with the terms of such license.

## U.S. Government Rights

The Software is “commercial computer software,” as defined by Federal Acquisition Regulation (“FAR”) 2.101. Pursuant to FAR 12.212 and 27.405-3 and Department of Defense FAR Supplement

(“DFARS”) 227.7202, the U.S. government acquires commercial computer software under the same terms by which the software is customarily provided to the public. Accordingly, Keysight provides the Software to U.S. government customers under its standard commercial license, which is embodied in its End User License Agreement (EULA), a copy of which can be found at:

<http://www.keysight.com/find/sweula>.

The license set forth in the EULA represents the exclusive authority by which the U.S.

government may use, modify, distribute, or disclose the Software. The EULA and the license set forth therein, does not require or permit, among other things, that Keysight: (1) Furnish technical information related to commercial computer software or commercial computer software documentation that is not customarily provided to the public; or (2) Relinquish to, or otherwise provide, the government rights in excess of these rights customarily provided to the public to use, modify, reproduce, release, perform, display, or disclose commercial computer software or commercial computer software documentation. No additional government requirements beyond those set forth in the EULA shall apply, except to the extent that those terms, rights, or licenses are explicitly required from all providers of commercial computer software pursuant to the FAR and the DFARS and are set forth specifically in writing elsewhere in the EULA. Keysight shall be under no obligation to update, revise or otherwise modify the Software. With respect to any technical data as defined by FAR 2.101, pursuant to FAR 12.211 and 27.404.2 and DFARS 227.7102, the U.S. government acquires no greater than Limited Rights as defined in FAR 27.401 or DFAR 227.7103-5 (c), as applicable in any technical data.

## Warranty

THE MATERIAL CONTAINED IN THIS DOCUMENT IS PROVIDED “AS IS,” AND IS SUBJECT TO BEING CHANGED, WITHOUT NOTICE, IN FUTURE EDITIONS. FURTHER, TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW, KEYSIGHT DISCLAIMS ALL WARRANTIES, EITHER EXPRESS OR IMPLIED WITH REGARD TO THIS MANUAL AND ANY INFORMATION CONTAINED HEREIN, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. KEYSIGHT SHALL

NOT BE LIABLE FOR ERRORS OR FOR INCIDENTAL OR CONSEQUENTIAL DAMAGES IN CONNECTION WITH THE FURNISHING, USE, OR PERFORMANCE OF THIS DOCUMENT OR ANY INFORMATION CONTAINED HEREIN. SHOULD KEYSIGHT AND THE USER HAVE A SEPARATE WRITTEN AGREEMENT WITH WARRANTY TERMS COVERING THE MATERIAL IN THIS DOCUMENT THAT CONFLICT WITH THESE TERMS, THE WARRANTY TERMS IN THE SEPARATE AGREEMENT WILL CONTROL.

## Safety Notices

### CAUTION

A CAUTION notice denotes a hazard. It calls attention to an operating procedure, practice, or the like that, if not correctly performed or adhered to, could result in damage to the product or loss of important data. Do not proceed beyond a CAUTION notice until the indicated conditions are fully understood and met.

### WARNING

A WARNING notice denotes a hazard. It calls attention to an operating procedure, practice, or the like that, if not correctly performed or adhered to, could result in personal injury or death. Do not proceed beyond a WARNING notice until the indicated conditions are fully understood and met.

## Safety Summary

The following general safety precautions must be observed during all phases of operation of this instrument. Failure to comply with these precautions or with specific warnings or operating instructions in the product manuals violates safety standards of design, manufacture, and intended use of the instrument. Keysight Technologies assumes no liability for the customer's failure to comply with these requirements. Product manuals are provided on the Web. Go to [www.keysight.com](http://www.keysight.com) and type in your product number in the Search field at the top of the page.

General	This product is a Protection Class 1 instrument (provided with a protective earth terminal) and has been manufactured and tested according to international safety standards. The protective features of this product may be impaired if it is used in a manner not specified in the operation instructions.
Environment Conditions	This instrument is intended for indoor use in an Overvoltage Category II, pollution degree 2 environment. It is designed to operate at a maximum relative humidity (RH) of 80%, non-condensing and at altitudes of up to 2000 meters. Refer to the specifications tables for the AC mains voltage requirements and ambient operating temperature range.
Temperature	<p>The instrument should be protected from temperature extremes and changes in temperature that may cause condensation within it.</p> <p>The operating temperature is from:</p> <ul style="list-style-type: none"><li>• +5 °C to +40 °C (N774-C)</li><li>• +0°C to +40°C (81620C/ 81623C/ 81624C)</li><li>• +0°C to +40°C (81623X01C/ 81623X85C/ 81624X01C)</li><li>• +0°C to +35°C (81626C/ 81626X01C, max. 30°C above +20 dBm)</li><li>• +0°C to +40°C (81628C, max. 35°C above +30 dBm)</li></ul> <p>The storage temperature is from -40 °C to +70 °C</p>
Before Applying Power	Verify that all safety precautions are taken. The power cable inlet of the instrument serves as a device to disconnect from the mains in case of hazard. The instrument must be positioned so that the operator can easily access the power cable inlet. When the instrument is rack mounted the rack must be provided with an easily accessible mains switch.
Ground the Instrument	To minimize shock hazard, the instrument chassis and cover must be connected to an electrical protective earth ground. The instrument must be connected to the AC power mains through a grounded power cable, with the ground wire firmly connected to an electrical ground (safety ground) at the power outlet. Any interruption of the protective (grounding) conductor or disconnection of the protective earth terminal will cause a potential shock hazard that could result in personal injury.

Do Not Operate in an  
Explosive Atmosphere










Do not operate the instrument in the presence of flammable gases or fumes.

Do Not Remove the  
Instrument Cover

Operating personnel must not remove instrument covers. Component replacement and internal adjustments must be made only by qualified personnel.

Instruments that appear damaged or defective should be made inoperative and secured against unintended operation until they can be repaired by qualified service personnel.

# Instrument Markings

Instrument Marking	Description
	The instruction manual symbol. The product is marked with this warning symbol when it is necessary for the user to refer to the instructions in the manual.
	Standby supply. Unit is not completely disconnected from AC mains when switch is off.
	The CE mark is a registered trademark of the European Community. This CE mark shows that the product complies with all the relevant European Legal Directives. ICES/NMB-001 - This ISM device complies with the Canadian ICES-001. Cet appareil ISM est conforme a la norme NMB-001 du Canada. ISM GRP 1-A - This is an Industrial Scientific and Medical (ISM) Group 1 Class A product.
	The CSA mark with the 'c' and 'us' subscript indicates the instrument is certified to the applicable Canadian and United States of America standards respectively.
	The RCM mark is a registered trademark of the Australian Communications and Media Authority
	This symbol is a South Korean Class A EMC Declaration, with the product identification code. R - Identification of authorization prefix. R - Identification of basic certification information. Kst - Identification of applicant's information 3E 18526 - N774-C Product identification. 3E 20724 - 8162-C Product identification. This is a Class A instrument suitable for professional use and in electromagnetic environment outside of the home.
	The recycling symbol indicates the general ease with which the instrument can be recycled.
	China Restricted Substance Product Label. The EPUP (environmental protection use period) number in the center indicates the time period during which no hazardous or toxic substances or elements are expected to leak or deteriorate during normal use and generally reflects the expected useful life of the product.
	The UKCA (UK Conformity Assessed) marking is a new UK product marking that is used for goods being placed on the market in Great Britain (England, Wales and Scotland). It covers most goods which previously required the CE marking.

# South Korean Class A EMC Declaration

Information to the user:

This instrument has been conformity assessed for used in business environments. In a residential environment this equipment may caused radio interference.

This EMC statement applies to the equipment only for use in business environment.



사 용 자 안 내 문

이 기 기 는 업 무 용 환 경 에 서 사 용 할 목 적 으 로 적 합 성 평 가 를 받 은 기 기 로 서 가 정 용 환 경 에 서 사 용 하 는 경 우 전 파 간 섭 의 우 려 가 있 습 니 다 .

사용자 안내문은 "업무용 방송통신기자재"에만 적용한다.

## Compliance and Environmental Information

Table 1 Compliance and Environmental Information

Safety Symbol	Description
	<p>This product complies with WEEE Directive (2002/96/EC) marking requirements. The affixed label indicates that you must not discard this electrical/electronic product in domestic household waste.</p> <p>Product Category: With reference to the equipment types in WEEE Directive Annex I, this product is classed as a "Monitoring and Control instrumentation" product.</p> <p>Do not dispose in domestic household waste.</p>
	<p>To return unwanted products, contact your local Keysight office, or see <a href="http://about.keysight.com/en/companyinfo/environment/takeback.shtml">http://about.keysight.com/en/companyinfo/environment/takeback.shtml</a> for more information.</p>

## Declaration of Conformity

Declarations of Conformity for this product and for the Keysight products may be downloaded from the Web. Go to <http://www.keysight.com/go/conformity>.

You can then search by product number to find the latest Declaration of Conformity.

# Contents

- Safety Summary 3
- Instrument Markings 5
- South Korean Class A EMC Declaration 6
- Compliance and Environmental Information 6
- Declaration of Conformity 6

## 1 Introduction to Programming

- Message Queues 10**
  - How the Input Queue Works 10
  - Clearing the Input Queue 10
  - The Output Queue 10
  - The Error Queue 10
- Programming and Syntax Diagram Conventions 12**
  - Short Form and Long Form 12
  - Command and Query Syntax 12
- Common Commands 15**
  - Common Command Summary 15
  - Common Status Information 16
- Status Model 18**
  - Status Registers 18

## 2 Command Summary

- Command Summary 22**

### 3 Instrument Setup and Status

<b>IEEE-Common Commands</b>	30
<b>Status Reporting – The STATus Subsystem</b>	36
<b>Interface/Instrument Behaviour Settings – The SYSTem Subsystem</b>	44
<b>System Communicate – The :SYST:COMMunicate Subsystem</b>	48

### 4 Measurement Operations & Settings

<b>CONFigure Subsystem Commands</b>	60
<b>FETCh Subsystem Commands</b>	63
<b>INITiate Subsystem Commands</b>	66
<b>READ Subsystem Commands</b>	67
<b>Measurement Functions – The SENSE Subsystem</b>	69
<b>Triggering – The TRIGger Subsystem</b>	88

### 5 Error Codes

<b>Error Strings</b>	96
----------------------	----



# 1 Introduction to Programming

Message Queues / 10

Programming and Syntax Diagram Conventions / 12

Common Commands / 15

Status Model / 18

This chapter provides general information on how to control your instrument remotely.

Descriptions for the actual commands for the instruments are given in the following chapters. The information in these chapters is specific to the N774-C multiport power meter instruments.

## Message Queues

The instrument exchanges messages using an input and an output queue. Error messages are kept in a separate error queue.

### How the Input Queue Works

The input queue is a FIFO queue (first-in first-out). Incoming bytes are stored in the input queue. The parser starts if the LF character is received.

### Clearing the Input Queue

Switching the power off, or sending a Device Interface Clear signal, causes commands that are in the input queue, but have not been executed to be lost.

### The Output Queue

The output queue contains responses to query messages. The instrument transmits any data from the output queue when a controller addresses the instrument as a talker.

Each response message ends with a LF (hex 0A). If no query is received, or if the query has an error, the output queue remains empty.

The Message Available bit (MAV, bit 4) is set in the Status Byte register whenever there is data in the output queue.

### The Error Queue

The error queue is 30 errors long. It is a FIFO queue (first-in first-out). That is, the first error read is the oldest error to have occurred. For example:

- 1 If no error has occurred, the error queue contains:  
+ 0, "No error"
- 2 After a command such as wav:pow, the error queue now contains:  
+ 0, "No error"  
-113, "Undefined header"
- 3 If the command is immediately repeated, the error queue now contains:  
+ 0, "No error"  
-113, "Undefined header"  
-113, "Undefined header"

If more than 29 errors are put into the queue, the message:  
-350, "Queue overflow"  
is placed as the last message in the queue.

## Programming and Syntax Diagram Conventions

A program message is a message containing commands or queries that you send to the instruments. The following are a few points about program messages:

- You can use either upper-case or lower-case characters.
- You can send several commands in a single message. Each command must be separated from the next one by a semicolon (;).
- A command message is ended by a line feed character (LF).
- You can use any valid number/unit combination.

In other words, 1500NM, 1.5UM and 1.5E-6M are all equivalent.

If you do not specify a unit, then the default unit is assumed. The default unit for the commands are given with command description in the next chapter.

### Short Form and Long Form

The instrument accepts messages in short or long forms.

For example, the message

```
:STATUS:OPERATION:ENABLE 768
```

is in long form.

The short form of this message is

```
:STAT:OPER:ENAB 768
```

In this manual, the messages are written in a combination of upper and lower case. Upper case characters are used for the short form of the message.

For example, the above command would be written

```
:STATus:OPERation:ENABle
```

The first colon can be left out for the first command or query in your message. That is, the example given above could also be sent as

```
STAT:OPER:ENAB 768
```

### Command and Query Syntax

All characters not between angled brackets must be sent exactly as shown.

The characters between angled brackets (<...>) indicate the kind of data that you should send, or that you get in a response. You do not type the angled brackets in the actual message.

Descriptions of these items follow the syntax description. The following types of data are most commonly used:

string	is ascii data. A string is contained between double quotes ("...") or single quotes ('...').
value	is numeric data in integer (12), decimal (34.5) or exponential format (67.8E-9).
wsp	is a white space.

Other kinds of data are described as required.

The characters between square brackets ([...]) show optional information that you can include with the message.

The bar (|) shows an either-or choice of data, for example, *a|b* means either *a* or *b*, but not both simultaneously.

Extra spaces are ignored, so spaces can be inserted to improve readability.

## Units

Where units are given with a command, usually only the base units are specified. The full sets of units are given in the table below.

**Table 2** Units and allowed Mnemonics

Unit	Default	Allowed Mnemonics
meters	M	PM, NM, UM, MM, M
decibel	DB	MDB, DB
second	S	NS, US, MS, S
decibel/1mW	DBM	MDBM, DBM
Hertz	HZ	HZ, KHZ, MHZ, GHZ, THZ
Watt	Watt	PW, NW, UW, MW, Watt
meters per second	M/S	NM/S, UM/S, MM/S, M/S

## Data Types

With the commands you give parameters to the instrument and receive response values from the instrument. Unless explicitly specified these data are given in ASCII format. The following types of data are used:

- *Boolean* data may only have the values 0 or 1.
- *Integer* range is given for each individual command.
- *Float* variables may be given in decimal or exponential writing (0.123 or 123E-3).  
All *Float* values conform to the 32 bit IEEE Standard, that is, all *Float* values are returned as 32-bit real values.
- A *string* is contained between double quotes ("...") or single quotes ('...').
- When a *register* value is given or returned (for example \*ESE), the *decimal* values for the single bits are added. For example, a value of nine means that bit 0 and bit 3 are set.
- Larger blocks of data are given as *Binary Blocks*, preceded by "`#<H><Len><Block>`"; `<H>` represents the number of digits, `<Len>` represents the number of bytes, and `<Block>` is the data block. For example, for a *Binary Block* with 1 digit and 6 bytes this is: `#16TRACES`. The block represents an array of numbers. Each number has the byte ordering least significant byte first, also called LSBfirst, little-endian or Intel byte ordering.

### NOTE

Note that within your program, calculations with wavelengths may require double-precision 64-bit floats to provide the desired resolution.

## Common Commands

The IEEE 488.2 standard has a list of reserved commands, called common commands. Some of these commands must be implemented by any instrument using the standard, others are optional.

Your instrument implements all the necessary commands, and some optional ones. This section describes the implemented commands.

### Common Command Summary

**Table 3** on page -15 provides a summary of the common commands.

**Table 3 Common Command Summary**

Command	Parameter	Function	Page
*CLS		Clear Status Command	page 30
*ESE		Standard Event Status Enable Command	page 30
*ESE?		Standard Event Status Enable Query	page 31
*ESR?		Standard Event Status Register Query	page 31
*IDN?		Identification Query	page 32
*OPC		Operation Complete Command	page 32
*OPC?		Operation Complete Query	page 33
*OPT?		Options Query	page 33
*RST		Reset Command	page 33
*STB?		Read Status Byte Query	page 34
*TST?		Self Test Query	page 33
*WAI		Wait Command	page 33

### NOTE

These commands are described in more detail in **IEEE-Common Commands** on page 30.

Common Status Information

There are three registers for the status information. Two of these are status-registers and one is an enable-registers. These registers conform to the IEEE Standard 488.2-1987. You can find further descriptions of these registers under **\*ESE**, **\*ESR?**, and **\*STB?**.

**Figure 1** shows how the Standard Event Status Enable Mask (SESEM) and the Standard Event Status Register (SESR) determine the Event Status Bit (ESB) of the Status Byte.

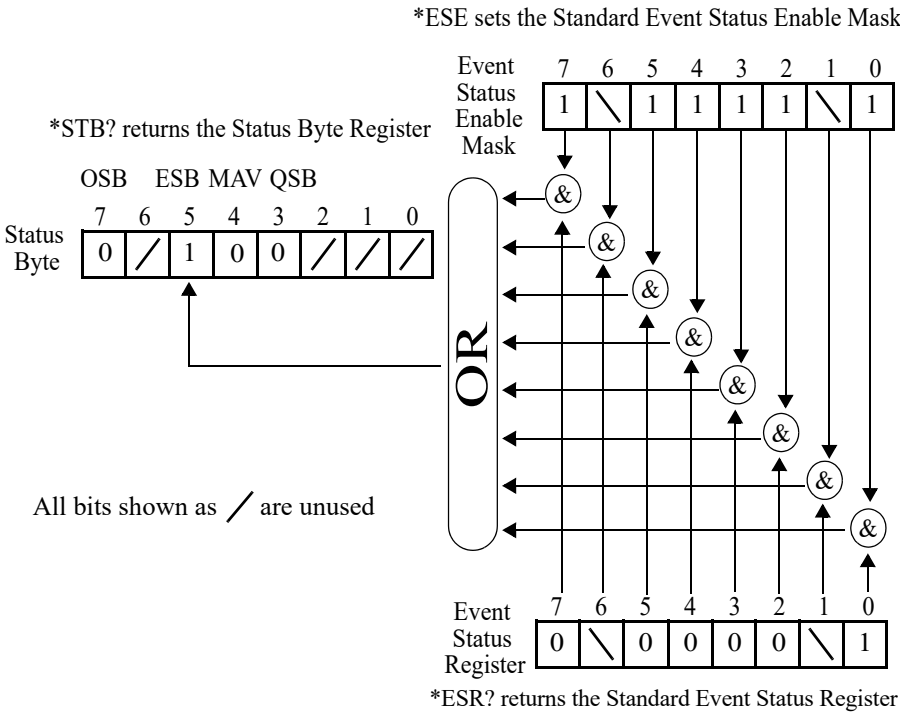


Figure 1 The Event Status Bit

The SESR contains the information about events that are not slot specific. The SESEM allows you to choose the event that may affect the ESB of the Status Byte. If you set a bit of the SESEM to zero, the corresponding event cannot affect the ESB. The default is for all the bits of the SESEM to be set to 0.



The questionable and operation status systems set the Operational Status Bit (OSB) and the Questionable Status Bit (QSB).

**NOTE**

Unused bits in any of the registers change to 0 when you read them.

---

## Status Model

### Status Registers

Each node of the status circuitry has three registers:

A condition register (CONDition), which contains the current status. This register is updated continuously. It is not changed by having its contents read.

The event register (EVENT), which contains details of any positive transitions in the corresponding condition register, that is, when a bit changes from 0 -> 1. The contents of this register are cleared when it is read. The contents of any higher-level registers are affected with regard to the appropriate bit.

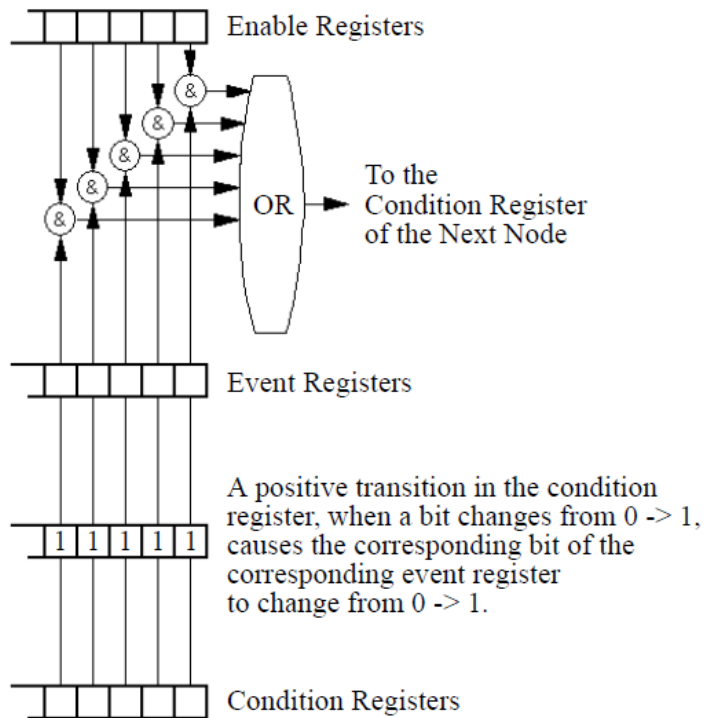
The enable register (ENABLE), which enables changes in the event register to affect the next stage of registers.

#### NOTE

**The event register is the only kind of register that can affect the next stage of registers.**

---

The structures of the Operational and Questionable Status Systems are similar. The following figure describes how the Questionable Status Bit (QSB) and the Operational Status Bit (OSB) of the Status Byte Register are determined.



The Operational/Questionable Slot Status Event Register (OSSER/QSSER) contains the status of a particular module slot. A bit changes from 0 -> 1 when an event occurs, for example, when a laser is switched on. For details of the function of each bit of these registers, see Operation/Questionable Status Summary Register.

The Operational/Questionable Slot Enable Status Mask (OSESMS/QSESMS) allows you to choose the events for each module slot that may affect the Operational/Questionable Status Event Register (see below). If you set a bit of the OSESMS/QSESMS to zero, the occurrence of the corresponding event for this particular module slot cannot affect the Operational/Questionable Status Event Register. The default is for all the bits of the OSESMS/QSESMS to be set to 0.

The Operational/Questionable Status Event Summary Register (OSESER/QSESER) summarizes the status of every module slot of your instrument. If, for any slot, any bit of the QSSER goes from 0 -> 1 AND the corresponding bit of the QSSEM is 1 at the same time, the QSESER bit representing that slot is set to 1.

The Operational/Questionable Status Enable Summary Mask (OSESME/QSESME) allows you to choose the module slots that may affect the OSB/QSB of the Status Byte. If any bit of the QSESER goes from 0 -> 1 AND the corresponding bit of the QSESME is 1 at the same time, the QSB of the Status Byte is set to 1. If you set a bit of the OSESME/QSESME to zero, the corresponding module slot cannot affect the OSB/QSB. The default is for all the bits of the OSESME/QSESME to be set to 0.

# 2 Command Summary

Command Summary / 22

This chapter lists commands relating to the N774-C multiport power meter instruments.

Each of these summaries contains a page reference for more detailed information about the particular command later in this document.

# Command Summary

The commands are ordered in a command tree. Every command belongs to a node in this tree.

The root nodes are also called the subsystems. A subsystem contains all commands belonging to a specific topic. In a subsystem there may be further subnodes.

Table 4 on page 22 gives an overview of the command tree. You see the nodes, the subnodes, and the included commands.

**Table 4      Command Summary**

Command	Page
<b>CONFigure Subsystem</b>	
:CONFigure:MEASurement:SETting:ACTual?	Page 60
:CONFigure:MEASurement:SETting:NUMBer?	Page 60
:CONFigure:MEASurement:SETting:PRESet	Page 60
:CONFigure:MEASurement:SETting:CANCel	Page 61
:CONFigure:MEASurement:SETting:RECall	Page 61
:CONFigure:MEASurement:SETting:SAVE	Page 61
:CONFigure:MEASurement:SETting:ERASe	Page 62

Command	Page
<b>FETCh Subsystem</b>	
:FETCh[n]:POWer?	Page 63
:FETCh[n]:POWer:ALL?	Page 63
:FETCh[n]:POWer:ALL:CSV?	Page 64
:FETCh[n]:POWer:ALL:CONFig?	Page 64
:FETCh[n]:POWer:MAX?	Page 64
:FETCh[n]:POWer:MIN?	Page 65
:FETCh[n]:POWer:EXTRema:RESet	Page 65

Command	Page
<b>INITiate Subsystem</b>	
:INITiate[n][:CHANnel[m]][:IMMediate]	Page 66
:INITiate[n][:CHANnel[m]]:CONTInuous	Page 66
:INITiate[n][:CHANnel[m]]:CONTInuous?	Page 66
:READ[n]:POWer?	Page 68

Command	Page
<b>READ Subsystem</b>	
:READ[n]:POWer:ALL?	Page 67
:READ[n]:POWer:ALL:CSV?	Page 67
:READ[n]:POWer:ALL:CONFig?	Page 67
:READ[n]:POWer?	Page 68

Command	Page
<b>SENSe Subsystem</b>	
:SENSe[n]:CORRection	Page 69
:SENSe[n]:CORRection?	Page 69
:SENSe[n]:CORRection:COLLect:ZERO	Page 69
:SENSe[n]:CORRection:COLLect:ZERO?	Page 70
:SENSe:CORRection:COLLect:ZERO:ALL	Page 70
:SENSe:CORRection:COLLect:ZERO:ALL?	Page 70
:SENSe[n]:CORRection:COLLect:ZERO:QUAD	Page 71
:SENSe[n]:CORRection:COLLect:ZERO:QUAD?	Page 71
:SENSe[n]:FUNctIon:LOOP	Page 72
:SENSe[n]:FUNctIon:LOOP?	Page 72
:SENSe[n]:FUNctIon:PARAmeter:CHECK?	Page 72

Command	Page
:SENSe[n]:FUNCTION:PARAMeter:LOGGing	Page 73
:SENSe[n]:FUNCTION:PARAMeter:LOGGing?	Page 73
:SENSe[n]:FUNCTION:PARAMeter:MINMax	Page 74
:SENSe[n]:FUNCTION:PARAMeter:MINMax?	Page 74
:SENSe[n]:FUNCTION:PARAMeter:STABility	Page 74
:SENSe[n]:FUNCTION:PARAMeter:STABility?	Page 75
:SENSe[n]:FUNCTION:RESult?	Page 75
:SENSe[n]:FUNCTION:RESult:BLOCK?	Page 76
:SENSe[n]:FUNCTION:RESult:BUFA?	Page 76
:SENSe[n]:FUNCTION:RESult:BUFB?	Page 77
:SENSe[n]:FUNCTION:RESult:BLOCK?	Page 77
:SENSe[n]:FUNCTION:RESult:INDex?	Page 77
:SENSe[n]:FUNCTION:RESult:MAXBlockSize?	Page 78
:SENSe[n]:FUNCTION:STATe	Page 78
:SENSe[n]:FUNCTION:STATe?	Page 78
:SENSe[n]:POWER:GAIN:AUTO	Page 79
:SENSe[n]:POWER:GAIN:AUTO?	Page 79
:SENSe[n]:POWER:ATIme	Page 80
:SENSe[n]:POWER:ATIme?	Page 80
:SENSe[n]:POWER:OUTPut	Page 80
:SENSe[n]:POWER:OUTPut?	Page 81
:SENSe[n]:POWER:RANGe:AUTO	Page 81
:SENSe[n]:POWER:RANGe:AUTO?	Page 82
:SENSe[n]:POWER:RANGe	Page 82
:SENSe[n]:POWER:RANGe?	Page 82
:SENSe[n]:POWER:REFerence	Page 83
:SENSe[n]:POWER:REFerence?	Page 83



Command	Page
:SENSe[n]:POWer:REfERENCE:DISPlay	Page 83
:SENSe[n]:POWer:REfERENCE:STATe	Page 84
:SENSe[n]:POWer:REfERENCE:STATe?	Page 84
:SENSe[n]:POWer:REfERENCE:STATe:RATio	Page 84
:SENSe[n]:POWer:REfERENCE:STATe:RATio?	Page 85
:SENSe[n]:POWer:UNIT[:ALL]	Page 85
:SENSe[n]:POWer:UNIT?	Page 86
:SENSe:POWer:UNIT:ALL:CSV?	Page 86
:SENSe[n]:POWer:WAVelength[:ALL]	Page 86
:SENSe[n]:POWer:WAVelength?	Page 87

Command	Page
<b>STATus Subsystem</b>	
:STATus:OPERation[:EVENT]?	Page 36
:STATus:OPERation:CONDition?	Page 36
:STATus[n]:OPERation[:EVENT]?	Page 38
:STATus:OPERation:ENABle	Page 37
:STATus:OPERation:ENABle?	Page 37
:STATus[n]:OPERation:CONDition?	Page 38
:STATus[n]:OPERation:ENABle	Page 39
:STATus[n]:OPERation:ENABle?	Page 39
:STATus:PRESet	Page 39
:STATus:QUEStionable[:EVENT]?	Page 39
:STATus:QUEStionable:CONDition?	Page 40
:STATus[n]:QUEStionable[:EVENT]?	Page 42
:STATus:QUEStionable:ENABle	Page 41
:STATus:QUEStionable:ENABle?	Page 41

Command	Page
:STATus[n]:QUESTionable:CONDition?	Page 42
:STATus[n]:QUESTionable:ENABle	Page 43
:STATus[n]:QUESTionable:ENABle?	Page 43

Command	Page
<b>SYSTem Subsystem</b>	
:SYSTem:DATE	Page 44
:SYSTem:DATE?	Page 44
:SYSTem:HELP:HEADers?	Page 44
:SYSTem:HELP:ERRors?	Page 45
:SYSTem:TIME	Page 45
:SYSTem:PRESet	Page 46
:SYSTem:TIME?	Page 46
:SYSTem:ERRor[:NEXT]?	Page 46
:SYSTem:ERRor:COUNT?	Page 46
:SYSTem:VERSion?	Page 47
:SYSTem:REBoot	Page 47
:SYSTem:COMMunicate:ETHernet:AUTOip:ENABle?	Page 48
:SYSTem:COMMunicate:ETHernet:AUTOip:ENABle	Page 49
:SYSTem:COMMunicate:ETHernet:CANCEL	Page 49
:SYSTem:COMMunicate:ETHernet:DGATeway	Page 49
:SYSTem:COMMunicate:ETHernet:DGATeway?	Page 49
:SYSTem:COMMunicate:ETHernet:DGATeway:CURREnt?	Page 50
:SYSTem:COMMunicate:ETHernet:DHCP:ENABle?	Page 50
:SYSTem:COMMunicate:ETHernet:DHCP:ENABle	Page 50
:SYSTem:COMMunicate:ETHernet:DOMainname?	Page 51
:SYSTem:COMMunicate:ETHernet:DOMainname	Page 51

Command	Page
:SYSTem:COMMunicate:ETHernet:DOMainname:CURRent?	Page 51
:SYSTem:COMMunicate:ETHernet:HOSTname	Page 51
:SYSTem:COMMunicate:ETHernet:HOSTname?	Page 52
:SYSTem:COMMunicate:ETHernet:HOSTname:CURRent?	Page 52
:SYSTem:COMMunicate:ETHernet:NSERver?	Page 52
:SYSTem:COMMunicate:ETHernet:NSERver	Page 53
:SYSTem:COMMunicate:ETHernet:NSERver:CURRent?	Page 53
:SYSTem:COMMunicate:ETHernet:IDN	Page 53
:SYSTem:COMMunicate:ETHernet:IPADdress	Page 53
:SYSTem:COMMunicate:ETHernet:IPADdress?	Page 54
:SYSTem:COMMunicate:ETHernet:IPADdress:CURRent?	Page 54
:SYSTem:COMMunicate:ETHernet:MACaddress?	Page 54
:SYSTem:COMMunicate:ETHernet:NTP:ENABLE?	Page 55
:SYSTem:COMMunicate:ETHernet:NTP:ENABLE	Page 55
:SYSTem:COMMunicate:ETHernet:NTP:SERVer?	Page 55
:SYSTem:COMMunicate:ETHernet:NTP:SERVer	Page 55
:SYSTem:COMMunicate:ETHernet:DESCription?	Page 56
:SYSTem:COMMunicate:ETHernet:DESCription	Page 56
:SYSTem:COMMunicate:ETHernet:RESet	Page 56
:SYSTem:COMMunicate:ETHernet:REStart	Page 57
:SYSTem:COMMunicate:ETHernet:SAVE	Page 57
:SYSTem:COMMunicate:ETHernet:SMASK?	Page 57
:SYSTem:COMMunicate:ETHernet:SMASK	Page 58
:SYSTem:COMMunicate:ETHernet:SMASK:CURRent?	Page 58

Command	Page
<b>TRIGger Subsystem</b>	
:TRIGger	Page 88
:TRIGger[n]:DElay?	Page 88
:TRIGger[n]:DElay	Page 88
:TRIGger[n]:INPut	Page 89
:TRIGger[n]:INPut?	Page 89
:TRIGger[n]:INPut:EDGE?	Page 90
:TRIGger[n][:INPut:EDGE	Page 90
:TRIGger[n]:OFFSet?	Page 91
:TRIGger[n]:OUTPut?	Page 91
:TRIGger[n]:OFFSet	Page 91
:TRIGger[n]:OUTPut	Page 92
:TRIGger:CONFiguration?	Page 92
:TRIGger:CONFiguration	Page 93

# 3 Instrument Setup and Status

IEEE-Common Commands / 30

Status Reporting – The STATus Subsystem / 36

Interface/Instrument Behaviour Settings – The SYSTem Subsystem / 44

System Communicate - The :SYST:COMMunicate Subsystem / 48

This chapter gives descriptions of commands that you can use when setting up your instrument. The commands are split into the following separate subsystems:

- IEEE specific commands that were introduced in [Common Commands](#) on page 15.
- STATus subsystem commands that relate to the status model.
- SYSTem subsystem commands that control the serial interface and internal data.

# IEEE-Common Commands

Common Commands on page 15 gave a brief introduction to the IEEE-common commands which can be used with the instruments. This section gives fuller descriptions of each of these commands.

Command:	<b>*CLS</b>		
Syntax:	<b>*CLS</b>		
Description:	The Clear Status (*CLS) command clears the status byte by emptying the error queue and clearing all the event registers (SESR) including the Data Questionable Event Register, the Standard Event Status Register, the Standard Operation Status Register and any other registers that are summarized in the status byte.		
Parameters:	None		
Response:	None		
Example:	<b>*CLS</b>		

Command:	<b>*ESE</b>		
Syntax:	<b>*ESE&lt;wsp&gt;&lt;value&gt;</b> $0 \leq \text{value} \leq 255$		
Description:	The standard Event Status Enable command (*ESE) sets bits in the Standard Event Status Enable Mask (SESEM) that enable the corresponding bits in the standard event status register (SESR). The register is cleared: at power-on, by sending a value of zero. The register is not changed by the *CLS commands.		
Parameters:	The bit value for the register (a 8-bit integer value):		
	Bit	Mnemonic	Decimal Value
	7 (MSB)	Power On	128
	6	Not Used	64
	5	Command Error	32
	4	Execution Error	16
	3	Device Dependent Error	8
	2	Query Error	4
	1	Not Used	2

	0 (LSB)	Operation Complete	1
Response:	None		
Example:	*ESE 255		

Command:	<b>*ESE?</b>		
Syntax:	<b>*ESE?</b>		
Description:	The standard Event Status Enable query *ESE? returns the contents of the Standard Event Status Enable Mask (see *ESE for information on this register).		
Parameters:	None		
Response:	The bit value for the register (a <i>8-bit integer</i> value).		
Example:	*ESE? -> +255		

Command:	<b>*ESR?</b>		
Syntax:	<b>*ESR?</b>		
Description:	The standard Event Status Register query *ESR? returns the contents of the Standard Event Status Register. The register is cleared after being read.		
parameters	None		
response	The bit value for the register (a <i>8-bit integer</i> value):		
	Bit	Mnemonic	Decimal Value
	7 (MSB)	Power On	128
	6	Not used	64
	5	Command Error	32
	4	Execution Error	16
	3	Device Dependent Error	8
	2	Query Error	4

	1	Not used	2
	0 (LSB)	Operation Complete	1
Example:	*ESR? -> 21		

Command:	<b>*IDN?</b>		
Syntax:	<b>*IDN?</b>		
Description:	The IDeNtification query *IDN? gets the instrument identification over the interface.		
Parameters:	None		
Response:	<p>The instrument identification For example:</p> <div> <div> MMMMMMMM  mmmm  ssssssss  rrrrrrrrr </div> <div> manufacturer, for example Keysight Technologies  instrument model number (for example N7745C)  serial number  firmware revision level </div> </div>		
Example:	*IDN? -> Keysight Technologies,N7745C,DE42100168		

Command:	<b>*OPC</b>		
Syntax:	<b>*OPC</b>		
Description:	Generates the OPC message in the standard event status register when all pending overlapped operations have been completed.		
Parameters:	None		
Response:	None		
Example:	*OPC		



Command:	<b>*OPC?</b>
Syntax:	<b>*OPC?</b>
Description:	The Operation Complete query *OPC? parses all program message units in the input queue, sets the operation complete bit in the Standard Event Status register, and places an ASCII '1' in the output queue, when the contents of the input queue have been processed. Taking advantage of this feature, and using *OPC? in a loop to query until the instrument returns 1, can lead to useful gains in program execution efficiency.
Parameters:	None
Response:	<ul style="list-style-type: none"> <li>1 is returned if the instrument is ready to execute a new operation.</li> <li>0 is returned if the instrument is busy.</li> </ul>
Example:	*OPC? -> 1

Command:	<b>*OPT?</b>
Syntax:	<b>*OPT?</b>
Description:	The OPTions query *OPT? returns the options installed in your instrument.
Parameters:	None
Response:	Returns the part number of all installed options, separated by commas. Channels are listed starting with the lowest number, that is, channel 1.
Example:	*OPT? -> N7752A-002, N7752A-002, N7752A-001, N7752A-001,

Command:	<b>*RST</b>
Syntax:	<b>*RST</b>
Description:	<p>The ReSeT command *RST sets the mainframe and all modules to the reset setting (standard setting) stored internally. The instrument is placed in the idle state awaiting a command. The *RST command clears the error queue. The *RST command is equivalent to the *CLS command AND the syst:preset command.</p> <p>The following are not changed:</p> <ul style="list-style-type: none"> <li>Instrument interface address</li> <li>Service request enable register (SRE)</li> <li>Standard Event Status Enable Mask (SESEM)</li> </ul> <p>To prevent this, use the :CONFigure:MEASurement:SETting:PRESet command to keep the previously stored settings in non-volatile RAM.</p>

Parameters:	None
Response:	None
Example:	*RST

Command:	<b>*STB?</b>		
Syntax:	<b>*STB?</b>		
Description:	The SStatus Byte query *STB? returns the contents of the Status Byte register.		
Parameters:	None		
Response:	The bit value for the register (a <i>8-bit signed integer</i> value):		
	Bit	Mnemonic	Decimal Value
	7 (MSB)	Operation Status (OSB)	128
	6	Not used	64
	5	Event Status Bit (ESB)	32
	4	Message Available (MAV)	16
	3	Questionable Status (QSB)	8
	2	Not used	4
	1	Not used	2
	0	Not used	1
Example:	*STB? -> +32		

Command:	<b>*TST?</b>
Syntax:	<b>*TST?</b>
Description:	<p>The self-TeST query *TST? makes the instrument perform a self-test and place the results of the test in the output queue. If the self-test fails, the results are also put in the error queue.</p> <p>We recommend that you read self-test results from the error queue. No further commands are allowed while the test is running. After the self-test the instrument is returned to the setting that was active at the time the self-test query was processed. The self-test does not require operator interaction beyond sending the *TST? query.</p>
Parameters:	None
Response:	<p>Selftest failed 1</p> <p>A value of zero indicates no errors.</p>
Example:	*TST? > 0

Command:	<b>*WAI</b>
Syntax:	<b>*WAI</b>
Description:	<p>The WAI command prevents the instrument from executing any further commands until the current command has finished executing. Some module firmware includes commands that set a "StatNOPC" flag during execution to indicate that the module is busy. *WAI blocks commands until every module hosted by the instrument is no longer busy. All pending operations, are completed during the wait period.</p>
Parameters:	None
Response:	None
Example:	*WAI

# Status Reporting – The STATUS Subsystem

The Status subsystem allows you to return and set details from the Status Model.

Command:	:STATUS:OPERation[:EVENT]?			
Syntax:	:STATUS:OPERation[:EVENT]?			
Description:	Returns the Operational Status Event Summary Register (OSESr).			
Parameters:	None			
Response:	The sum of the results for the Channels (a 16-bit unsigned integer value, where $0 \leq \text{value} \leq 65535$ ):			
	Bits	Mnemonics	Decimal Value	
		N7744C	N7745C	
	9-15	Not used	Not used	
	8	Not used	Channel 8 Summary	256
	7	Not used	Channel 7 Summary	128
	6	Not used	Channel 6 Summary	64
	5	Not used	Channel 5 Summary	32
	4	Channel 4 Summary	Channel 4 Summary	16
	3	Channel 3 Summary	Channel 3 Summary	8
	2	Channel 2 Summary	Channel 2 Summary	4
	1	Channel 1 Summary	Channel 1 Summary	2
	0	Not used	Not used	1
Example:	:stat:oper? -> +0			

Command:	:STATUS:OPERation:CONDition?			
Syntax:	:STATUS:OPERation:CONDition?			
Description:	Reads the Operational Status Condition Summary Register.			
Parameters:	None			

Response:	The sum of the results for the individual Channels (a 16-bit unsigned integer value, where $0 \leq \text{value} \leq 65535$ ):			
	Bits	Mnemonics	Decimal Value	
		N7744C	N7745C	
	9-15	Not used	Not used	
	8	Not used	Channel 8 Summary	256
	7	Not used	Channel 7 Summary	128
	6	Not used	Channel 6 Summary	64
	5	Not used	Channel 5 Summary	32
	4	Channel 4 Summary	Channel 4 Summary	16
	3	Channel 3 Summary	Channel 3 Summary	8
	2	Channel 2 Summary	Channel 2 Summary	4
	1	Channel 1 Summary	Channel 1 Summary	2
	0	Not used	Not used	1
Example:	:stat:oper:cond? -> +0			

Command:	<b>:STATus:OPERation:ENABLE</b>
Syntax:	:STATus:OPERation:ENABLE<wsp><value>
Description:	Sets the bits in the Operational Status Enable Summary Mask (OSESMS) that enable the contents of the OSESR to affect the Status Byte (STB). Setting a bit in this register to 1 enables the corresponding bit in the OSESR to affect bit 7 of the Status Byte.
Parameters:	The bit value for the OSESMS as a 16-bit unsigned integer value (0 .. +65535) The default value is 65535.
Response:	None
Example:	:stat:oper:enab 128

Command:	<b>:STATus:OPERation:ENABLE?</b>
Syntax:	:STATus:OPERation:ENABLE?
Description:	Returns the OSESMS for the OSESR

Parameters:	None
Response:	The bit value for the operation enable mask as a <i>16-bit unsigned integer</i> value (0 .. +65535)
Example:	:stat:oper:enab? -> +128

Command:	<b>:STATus[n]:OPERation[:EVENT]?</b>		
Syntax:	:STATus[n]:OPERation[:EVENT]?		
Description:	Returns the Operational Slot Status Event Register (OSSER) of channel <i>n</i> .		
Parameters:	None		
Response:	The results for the individual channel events (a <i>16-bit unsigned integer</i> value, where $0 \leq \text{value} \leq 65535$ ):		
	Bit	Mnemonic	Decimal Value
	4-15	Not used	
	3	Channel <i>n</i> : Zeroing ongoing	8
	1-2	Not used	
Example:	:stat0:oper? -> +0		

Command:	<b>:STATus[n]:OPERation:CONDition?</b>		
Syntax:	:STATus[n]:OPERation:CONDition?		
Description:	Returns the Operational Slot Status Condition Register of channel <i>n</i> .		
Parameters:	None		
Response:	The results for the individual channel events (a <i>16-bit unsigned integer</i> value, where $0 \leq \text{value} \leq 65535$ ):		
	Bit	Mnemonic	Decimal Value
	4-15	Not used	
	3	Channel <i>n</i> : Zeroing ongoing	8
	1-2	Not used	
Example:	:stat0:oper:cond? -> +0		

Command:	<b>:STATus[n]:OPERation:ENABle</b>
Syntax:	:STATus[n]:OPERation:ENABle<wsp><value>
Description:	Sets the bits in the Operation Slot Status Enable Mask (OSSEM) for channel <i>n</i> that enable the contents of the Operation Slot Status Event Register (OSSER) for channel <i>n</i> to affect the OSESr. Setting a bit in this register to 1 enables the corresponding bit in the OSSER for channel <i>n</i> to affect bit <i>n</i> of the OSESr.
Parameters:	The bit value for the OSSEM as a <i>16-bit unsigned integer</i> value (0 .. +65535)
Response:	None
Example:	:stat0:oper:enab 128

Command:	<b>:STATus[n]:OPERation:ENABle?</b>
Syntax:	:STATus[n]:OPERation:ENABle?
Description:	Returns the OSSEM of channel <i>n</i>
Parameters:	None
Response:	The bit value for the OSSEM as a <i>16-bit unsigned integer</i> value (0 .. +65535)
Example:	:stat0:oper:enab? -> +128

Command:	<b>:STATus:PRESet</b>
Syntax:	:STATus:PRESet
Description:	Presets all bits in all OPERation and QUEStionable status systems to 0.
Parameters:	None
Response:	None
Example:	:stat:pres

Command:	<b>:STATus:QUEStionable[:EVENT]?</b>
Syntax:	:STATus:QUEStionable[:EVENT]?
Description:	Returns the Questionable Status Event Summary Register (QSESr).

Parameters:	None			
Response:	The sum of the results for the QSESR as a <i>16-bit unsigned integer</i> value (0 .. +65535)			
	Bits	Mnemonics	Decimal Value	
		N7744C	N7745C	
	15-9	Not used	Not used	
	8	Not used	Channel 8 Summary	256
	7	Not used	Channel 7 Summary	128
	6	Not used	Channel 6 Summary	64
	5	Not used	Channel 5 Summary	32
	4	Channel 4 Summary	Channel 4 Summary	16
	3	Channel 3 Summary	Channel 3 Summary	8
	2	Channel 2 Summary	Channel 2 Summary	4
	1	Channel 1 Summary	Channel 1 Summary	2
	0	Not used	Not used	1
Example:	:stat:ques? -> +0			

Command:	<b>:STATus:QUEStionable:CONDition?</b>			
Syntax:	:STATus:QUEStionable:CONDition?			
Description:	Returns the Questionable Status Condition Summary Register.			
Parameters:	None			
Response:	The sum of the results for the Questionable Status Condition Summary Register as a <i>16-bit unsigned integer</i> value (0 .. +65535)			
	Bits	Mnemonics	Decimal Value	
		N7744C	N7745C	
	15-9	Not used	Not used	
	8	Not used	Not used	256
	7	Not used	Channel 8 Summary	128
	6	Not used	Channel 7 Summary	64



5	Not used	Channel 6 Summary	32
4	Channel 4 Summary	Channel 5 Summary	16
3	Channel 3 Summary	Channel 4 Summary	8
2	Channel 2 Summary	Channel 3 Summary	4
1	Channel 1 Summary	Channel 2 Summary	2
0	Channel 0 Summary	Channel 1 Summary	1
Example: :stat:ques:cond? -> +0			

Command:	<b>:STATus:QUESTIONable:ENABLE</b>
Syntax:	:STATus:QUESTIONable:ENABLE<wsp><value>
Description:	Sets the bits in the Questionable Status Enable Summary Mask (QSESM) that enable the contents of the QSESR to affect the Status Byte (STB). Setting a bit in this register to 1 enables the corresponding bit in the QSESR to affect bit 3 of the Status Byte.
Parameters:	The bit value for the questionable enable mask as a <i>16-bit unsigned integer</i> value (0 .. +65535) The default value is 65535.
Response:	None
Example:	:stat:ques:enab 128

Command:	<b>:STATus:QUESTIONable:ENABLE?</b>
Syntax:	:STATus:QUESTIONable:ENABLE?
Description:	Returns the QSESM for the event register
Parameters:	None
Response:	The bit value for the QSEM as a <i>16-bit unsigned integer</i> value (0 .. +65535)
Example:	:stat:ques:enab? -> +128

Command:	:STATus[n]:QUESTionable[:EVENT]?		
Syntax:	:STATus[n]:QUESTionable[:EVENT]?		
Description:	Returns the questionable status of channel <i>n</i> - the Questionable Slot Status Event Register (QSSER).		
Parameters:	None		
Response:	The results for the individual channel events (a 16-bit unsigned integer value, where $0 \leq \text{value} \leq 65535$ ):		
	Bit	Mnemonic	Decimal Value
	2-15	Not used	
	1	Channel <i>n</i> : A Zeroing operation has failed	2
	0	Not used	
	Every <i>n</i> th bit is the summary of channel <i>n</i> .		
Example:	:stat0:oper? -> +0		

Command:	:STATus[n]:QUESTionable:CONDition?		
Syntax:	:STATus[n]:QUESTionable:CONDition?		
Description:	Returns the Questionable Slot Status Condition Register for channel <i>n</i> .		
Parameters:	None		
Response:	The results for the individual channel events (a 16-bit unsigned integer value, where $0 \leq \text{value} \leq 65535$ ):		
	Bit	Mnemonic	Decimal Value
	2-15	Not used	
	1	Channel <i>n</i> : A Zeroing operation has failed	2
	0	Not used	
	Every <i>n</i> th bit is the summary of channel <i>n</i> .		
Example:	:stat0:ques:cond? -> +0		

Command:	<b>:STATus[n]:QUESTionable:ENABle</b>
Syntax:	:STATus[n]:QUESTionable:ENABle<wsp> <value>
Description:	Sets the bits in the Questionable Slot Status Enable Mask (QSSEM) for channel <i>n</i> that enable the contents of the Questionable Slot Status Register (QSSR) for channel <i>n</i> to affect the QSESr. Setting a bit in this register to 1 enables the corresponding bit in the QSSER for channel <i>n</i> to affect bit <i>n</i> of the QSESr.
Parameters:	The bit value for the QSSEM as a <i>16-bit unsigned integer</i> value (0 .. +65535)
Response:	None
Example:	:stat0:ques:enab 128

Command:	<b>:STATus[n]:QUESTionable:ENABle?</b>
Syntax:	:STATus[n]:QUESTionable:ENABle?
Description:	Returns the QSSEM for channel <i>n</i>
Parameters:	None
Response:	The bit value for the QSSEM as a <i>16-bit integer</i> value (0 .. +65535)
Example:	:stat0:ques:enab? -> +128

# Interface/Instrument Behaviour Settings – The SYSTem Subsystem

The SYSTem subsystem lets you control the instrument's serial interface. You can also control some internal data (like date, time, and so on)

Command:	<b>:SYSTem:DATE</b>
Syntax:	:SYSTem:DATE<wsp><year>,<month>,<day>
Description:	Sets the instrument's internal date.
Parameters:	The date in the format year, month, day
Response:	None
Example:	:syst:date 2019, 1, 26

Command:	<b>:SYSTem:DATE?</b>
Syntax:	:SYSTem:DATE?
Description:	Returns the instrument's internal date.
Parameters:	None
Response:	The date in the format year, month, day
Example:	:syst:date? -> +2019,+1,+26

Command:	<b>:SYSTem:HELP:HEADers?</b>
Syntax:	:SYSTem:HELP:HEADers?
Description:	Returns a list of commands.
Parameters:	None
Response:	Returns a list of commands
Example:	:syst:help:head? -> Returns a list of all commands

Command:	<b>:SYSTem:HELP:ERRors?</b>
Syntax:	:SYSTem:HELP:ERRors?
Description:	Return an overview about all Errorcodes and a short description.
Parameters:	None
Response:	String list of error codes
Example:	:syst:help:err? -> +0,"No error",-100,"Command error",- 101,"Invalid character",-102,"Syntax error",-103,"Invalid separator",-104,"Data type error",-105,"GET not allowed",-108,"Parameter not allowed",...

Command:	<b>:SYSTem:PRESet</b>
Syntax:	:SYSTem:PRESet
Description:	<p>Sets the instrument to the standard settings. The following are not affected by this command:</p> <ul style="list-style-type: none"> <li>▪ the interface address,</li> <li>▪ the output and error queues,</li> <li>▪ the Service Request Enable register (SRE),</li> <li>▪ the Status Byte (STB),</li> <li>▪ the Standard Event Status Enable Mask (SESEM), and</li> <li>▪ the Standard Event Status Register (SESR).</li> </ul> <p>Pressing the "LAN Reset" button for a short time has the same effect. Long pressing of the "LAN Reset" button resets the LAN Parameter.</p>
Parameters:	None
Response:	None
Example:	:SYST:PRES

Command:	<b>:SYSTem:TIME</b>
Syntax:	:SYSTem:TIME<wsp><hour>,<minute>,<second>
Description:	Sets the instrument's internal time.
Parameters:	24-hour time format: hours (0-23), minutes (0-59), seconds (0-59)
Response:	None
Example:	:syst:time 20,15,30

Command:	<b>:SYSTem:TIME?</b>
Syntax:	:SYSTem:TIME?
Description:	Returns the instrument's internal time.
Parameters:	None
Response:	The time in the format hour, minute, second. Hours are counted 0...23 (24 hour time format).
Example:	:syst:time? -> +20,+15,+30

Command:	<b>:SYSTem:ERRor[:NEXT]?</b>
Syntax:	:SYSTem:ERRor[:NEXT]?
Description:	Returns the next error from the error queue.
Parameters:	None
Response:	The number of the latest error, and its meaning. <b>NOTE:</b> Every connection uses its own error queue
Example:	:syst:err? -> -113,"Undefined header"

Command:	<b>:SYSTem:ERRor:COUNt?</b>
Syntax:	:SYSTem:ERRor:COUNt?
Description:	Returns the total no. of errors.
Parameters:	None
Response:	The total count of errors.
Example:	:syst:err:coun? -> 20

Command:	<b>:SYSTem:VERsion?</b>
Syntax:	:SYSTem:VERsion?
Description:	Returns the SCPI revision to which the instrument complies.
Parameters:	None
Response:	The revision year and number.
Example:	:syst:vers? → 1999.0

Command:	<b>:SYSTem:REBoot</b>
Syntax:	:SYSTem:REBoot
Description:	Reboots the instrument.
Parameters:	None
Response:	None
Example:	:syst:reb

## System Communicate - The :SYST:COMMunicate Subsystem

We recommend you change network settings using the local user interface.

**NOTE**

The instrument does not close open connections when restarting the network interface (:SYSTem:COMMunicate:ETHeRnet:REStart). This means the number of possible connections is reduced by the number of previously open connections. However, the instrument does make sure connections are still alive. It should release unused open connections after about two minutes.

Some notes on DHCP/AutoIP/DNS

- If DHCP is enabled but no DHCP server is found, the instrument tries to use AutoIP as a fallback. This may take about 2 minutes.
- Depending on the available network capabilities, the instrument tries to tell the DNS server its host name or read the host and domain named it has been assigned.

MAC address:

The Media Access Control (MAC) number is a unique number associated with each network adapter.

Command:	:SYSTem:COMMunicate:ETHeRnet:AUTOip:ENABle?
Syntax:	:SYSTem:COMMunicate:ETHeRnet:AUTOip:ENABle?
Description:	Check whether Automatic IP addressing is enabled or disabled.
Parameters:	None
Response:	Boolean (0   1)
Example:	:SYST:COMM:ETH:AUTO:ENAB? -> 1



Command:	<b>:SYSTem:COMMunicate:ETHernet:AUTOip:ENABle</b>
Syntax:	:SYSTem:COMMunicate:ETHernet:AUTOip:ENABle
Description:	Enable or disable whether IP addresses can be created automatically by the instrument. Automatic IP addressing is only used if DHCP is enabled, but the instrument cannot find a DHCP server.
Parameters:	Boolean (0   1   OFF   ON)
Response:	None
Example:	:SYST:COMM:ETH:AUTO:ENAB 1

Command:	<b>:SYSTem:COMMunicate:ETHernet:CANCel</b>
Syntax:	:SYSTem:COMMunicate:ETHernet:CANCel
Description:	Undo all changes to the network parameters that have been made since the last save, reboot or ":syst:comm:eth:restart" command.
Parameters:	None
Response:	None
Example:	:SYST:COMM:ETH:CANC

Command:	<b>:SYSTem:COMMunicate:ETHernet:DGATeway</b>
Syntax:	:SYSTem:COMMunicate:ETHernet:DGATeway
Description:	Set the default gateway.
Parameters:	string (Up to four groups of up to 3 digits, groups separated by ".". Groups with leading zeros are interpreted as octal numbers.)
Response:	None
Example:	:syst:comm:eth:dgat "192.168.101.11"

Command:	<b>:SYSTem:COMMunicate:ETHernet:DGATeway?</b>
Syntax:	:SYSTem:COMMunicate:ETHernet:DGATeway?
Description:	Get the default gateway.

Parameters:	None
Response:	String
Example:	:syst:comm:eth:dgat? -> "192.168.101.11"

Command:	<b>:SYSTem:COMMunicate:ETHernet:DGATeway:CURRent?</b>
Syntax:	:SYSTem:COMMunicate:ETHernet:DGATeway:CURRent?
Description:	Get the currently used default gateway.
Parameters:	None
Response:	String
Example:	:syst:comm:eth:dgat:curr? -> "192.168.101.11"

Command:	<b>:SYSTem:COMMunicate:ETHernet:DHCP:ENABLE?</b>
Syntax:	:SYSTem:COMMunicate:ETHernet:DHCP:ENABLE?
Description:	Check whether DHCP is enabled or disabled.
Parameters:	None
Response:	Boolean (0   1)
Example:	:syst:comm:eth:dhcp:enab? -> 1

Command:	<b>:SYSTem:COMMunicate:ETHernet:DHCP:ENABLE</b>
Syntax:	:SYSTem:COMMunicate:ETHernet:DHCP:ENABLE
Description:	Enable or disable DHCP
Parameters:	Boolean (0   1   OFF   ON)
Response:	None
Example:	:syst:comm:eth:dhcp:enab ON

Command:	<b>:SYSTem:COMMunicate:ETHernet:DOMainname?</b>
Syntax:	:SYSTem:COMMunicate:ETHernet:DOMainname?
Description:	Get the domain name.
Parameters:	None
Response:	String
Example:	:syst:comm:eth:dom? -> ".companyname.com"

Command:	<b>:SYSTem:COMMunicate:ETHernet:DOMainname</b>
Syntax:	:SYSTem:COMMunicate:ETHernet:DOMainname
Description:	Set the domain name (used if DHCP is disabled).
Parameters:	String
Response:	None
Example:	:syst:comm:eth:dom ".companyname.com"

Command:	<b>:SYSTem:COMMunicate:ETHernet:DOMainname:CURRent?</b>
Syntax:	:SYSTem:COMMunicate:ETHernet:DOMainname:CURRent?
Description:	Get the currently used domain name.
Parameters:	None
Response:	String
Example:	:syst:comm:eth:dom:curr? -> ".companyame.com"

Command:	<b>:SYSTem:COMMunicate:ETHernet:HOSTName</b>
Syntax:	:SYSTem:COMMunicate:ETHernet:HOSTName
Description:	Set the host name.

Parameters:	string (maximum 19 characters, though not all characters can be used) The default host name is K-P..P-S...S; where P..P is the product Number, and S...S is as many of the last digits of the serial number as it takes to get a 15 character host name. If you set an empty host name (""), the host name will be set to its default value.
Response:	None
Example:	:syst:comm:eth:host "K-N774-C-00001"

Command:	<b>:SYSTem:COMMunicate:ETHernet:HOSTname?</b>
Syntax:	:SYSTem:COMMunicate:ETHernet:HOSTname?
Description:	Get the host name.
Parameters:	None
Response:	String
Example:	:syst:comm:eth:host? -> "K-N774-C-00001"

Command:	<b>:SYSTem:COMMunicate:ETHernet:HOSTname:CURRent?</b>
Syntax:	:SYSTem:COMMunicate:ETHernet:HOSTname:CURRent?
Description:	Get the current host name.
Parameters:	None
Response:	String
Example:	:syst:comm:eth:host:curr? -> "K-N774-C-00001"

Command:	<b>:SYSTem:COMMunicate:ETHernet:NSERver?</b>
Syntax:	:SYSTem:COMMunicate:ETHernet:NSERver?
Description:	Get the defined (DNS) nameserver for name resolution.
Parameters:	None
Response:	IP Address String
Example:	:syst:comm:eth:nser? -> "1.1.1.1", "2.2.2.2"

Command:	<b>:SYSTem:COMMunicate:ETHernet:NSERver</b>
Syntax:	:SYSTem:COMMunicate:ETHernet:NSERver
Description:	Set one or two nameservers for name resolution. (used if DHCP is disabled).
Parameters:	IP Address String
Response:	None
Example:	:syst:comm:eth:nser "1.1.1.1"

Command:	<b>:SYSTem:COMMunicate:ETHernet:NSERver:CURRent?</b>
Syntax:	:SYSTem:COMMunicate:ETHernet:NSERver:CURRent?
Description:	Get the DNS server addresses assigned from your DHCP sever (this is only valide if DHCP is available and enabled.
Parameters:	None
Response:	IP Address String
Example:	:syst:comm:eth:nser:curr? -> "10.127.72.11","10.127.90.11"

Command:	<b>:SYSTem:COMMunicate:ETHernet:IDN</b>
Syntax:	:SYSTem:COMMunicate:ETHernet:IDN
Description:	The LAN LED on the front panel of the instrument flashes for identification.
Parameters:	Boolean (0   1   OFF   ON)
Response:	None
Example:	:syst:comm:eth:ids 1

Command:	<b>:SYSTem:COMMunicate:ETHernet:IPADdress</b>
Syntax:	:SYSTem:COMMunicate:ETHernet:IPADdress
Description:	Set the IP address of the system manually (used if DHCP is disabled).

Parameters:	String (Up to four groups of up to 3 digits, groups separated by ".". Groups with leading zeroes are interpreted as octal numbers.)
Response:	None
Example:	:syst:comm:eth:ipad "192.132.13.2"

Command:	<b>:SYSTem:COMMunicate:ETHernet:IPADdress?</b>
Syntax:	:SYSTem:COMMunicate:ETHernet:IPADdress?
Description:	Get the manually set IP address of the system.
Parameters:	None
Response:	String
Example:	:syst:comm:eth:ipad? -> "192.132.13.2"

Command:	<b>:SYSTem:COMMunicate:ETHernet:IPADdress:CURRent?</b>
Syntax:	:SYSTem:COMMunicate:ETHernet:IPADdress:CURRent?
Description:	Get the current IP address of the instrument.
Parameters:	None
Response:	String
Example:	:syst:comm:eth:ipad:curr? -> "192.132.13.2"

Command:	<b>:SYSTem:COMMunicate:ETHernet:MACaddress?</b>
Syntax:	:SYSTem:COMMunicate:ETHernet:MACaddress?
Description:	Get the MAC address of the network adapter.
Parameters:	None
Response:	String (hexadecimal value).
Example:	:syst:comm:eth:mac? -> "00-07-E0-14-AE-08"

Command:	<b>:SYSTem:COMMunicate:ETHernet:NTP:ENABLE?</b>
Syntax:	:SYSTem:COMMunicate:ETHernet:NTP:ENABLE?
Description:	Returns the usage of a NTP Server
Parameters:	None
Response:	Boolean (0   1)
Example:	:syst:comm:eth:ntp:enab? -> 1

Command:	<b>:SYSTem:COMMunicate:ETHernet:NTP:ENABLE</b>
Syntax:	:SYSTem:COMMunicate:ETHernet:NTP:ENABLE
Description:	Disables or enables instrument's use of NTP. The acronym NTP stands for Network Time Protocol, a protocol for clock synchronization between computer systems.
Parameters:	Boolean (0   1)
Response:	None
Example:	:syst:comm:eth:ntp:enab 1

Command:	<b>:SYSTem:COMMunicate:ETHernet:NTP:SERVer?</b>
Syntax:	:SYSTem:COMMunicate:ETHernet:NTP:SERVer?
Description:	Get the defined Network Time Protocol (NTP) server for clock synchronization.
Parameters:	None
Response:	Address String
Example:	:syst:comm:eth:ntp:serv? -> "pool.ntp.org"

Command:	<b>:SYSTem:COMMunicate:ETHernet:NTP:SERVer</b>
Syntax:	:SYSTem:COMMunicate:ETHernet:NTP:SERVer
Description:	Get the defined Network Time Protocol (NTP) server for clock synchronization.

Parameters:	Address String
Response:	None
Example:	:syst:comm:eth:ntp:serv "pool.ntp.org"

Command:	<b>:SYSTem:COMMunicate:ETHernet:DESCription?</b>
Syntax:	:SYSTem:COMMunicate:ETHernet:DESCription?
Description:	Get the desired mDNS service name.
Parameters:	None
Response:	Quoted string of up to 260 characters
Example:	:syst:comm:eth:desc? -> "Keysight N774-C - 00001"

Command:	<b>:SYSTem:COMMunicate:ETHernet:DESCription</b>
Syntax:	:SYSTem:COMMunicate:ETHernet:DESCription
Description:	Set the desired mDNS service name.
Parameters:	Quoted string of up to 260 characters
Response:	None
Example:	:syst:comm:eth:desc "Keysight N774-C - 00001"

Command:	<b>:SYSTem:COMMunicate:ETHernet:RESEt</b>
Syntax:	:SYSTem:COMMunicate:ETHernet:RESEt
Description:	<p>Press the "LAN Reset" Button for a long time has the same effect.  Pressing the "LAN Reset" Button for a short time is the same as system:preset.  DHCP On  AutoIP On  NTP Off  Hostname is a concatenation of product number and serial number.  The password for the web based LAN configuration interface is reset to a blank password.</p>



Parameters:	None
Response:	None
Example:	:syst:comm:eth:res

Command:	<b>:SYSTem:COMMunicate:ETHernet:REStart</b>
Syntax:	:SYSTem:COMMunicate:ETHernet:REStart
Description:	Restart the system's network interface with the new parameters. This command only works if the instrument has a working network connection at the time the command is issued. If not you either have to wait until the instrument decides on an IP address using AutoIP or reboot the instrument.
Parameters:	None
Response:	String
Example:	:syst:comm:eth:rest

Command:	<b>:SYSTem:COMMunicate:ETHernet:SAVE</b>
Syntax:	:SYSTem:COMMunicate:ETHernet:SAVE
Description:	Save the system's network interface parameters.
Parameters:	None
Response:	None
Example:	:syst:comm:eth:save

Command:	<b>:SYSTem:COMMunicate:ETHernet:SMASK?</b>
Syntax:	:SYSTem:COMMunicate:ETHernet:SMASK?
Description:	Get the subnet mask.
Parameters:	None
Response:	String
Example:	:syst:comm:eth:smas? -> "255.255.255.0"

Command:	<b>:SYSTem:COMMunicate:ETHernet:SMASk</b>
Syntax:	:SYSTem:COMMunicate:ETHernet:SMASk
Description:	Set the subnet mask.
Parameters:	String (Up to four groups of up to 3 digits, groups separated by ".". Groups with leading zeroes are interpreted as octal numbers.)
Response:	None
Example:	:syst:comm:eth:smas "255.255.255.0">

Command:	<b>:SYSTem:COMMunicate:ETHernet:SMASk:CURRent?</b>
Syntax:	:SYSTem:COMMunicate:ETHernet:SMASk:CURRent?
Description:	Get the currently used subnet mask.
Parameters:	None
Response:	String
Example:	:syst:comm:eth:smas:curr? -> "255.255.255.0"

# 4 Measurement Operations & Settings

CONFigure Subsystem Commands / 60

FETCh Subsystem Commands / 63

INITiate Subsystem Commands / 66

READ Subsystem Commands / 67

Measurement Functions – The SENSe Subsystem / 69

Triggering – The TRIGger Subsystem / 88

This chapter gives descriptions of commands that you can use when you are setting up or performing measurements. The commands are split up into the following subsystems:

CONFigure subsystem commands that control all instruments.

FETCh, INITiate and READ subsystems let you control measurement parameters for the power meter.

SENSe subsystem commands that control Power Sensors, Optical Head Interface Modules, and Return Loss Modules.

TRIGger subsystem commands that control triggering.

# CONFigure Subsystem Commands

This section provides the description of the following commands.

Command:	<b>:CONFigure:MEASurement:SETting:ACTual?</b>
Syntax:	:CONFigure:MEASurement:SETting:ACTual?
Description:	Get the index of the setting currently being used.
Parameters:	None
Response:	A value > 0 is returned if the setting has been stored in FLASH memory (using :CONFigure:MEASurement:SETting:SAVE), or has been recalled from FLASH memory (using :CONFigure:MEASurement:SETting:RECall), and has not been changed since. 0 is returned if the setting has not yet been stored. 0 is returned if the FLASH setting has been deleted (using:CONFigure:MEASurement:SETting:ERASE) since the last recall or store. -1 is returned if the setting was changed but has not been saved yet.
Example:	:conf:meas:sett:act? → +2

Command:	<b>:CONFigure:MEASurement:SETting:NUMBer?</b>
Syntax:	:CONFigure:MEASurement:SETting:NUMBer?
Description:	Get the number of settings. In addition to the settings spaces in FLASH memory, the working memory can hold a setting.
Parameters:	None
Response:	int
Example:	:conf:meas:sett:numb? → +1

Command:	<b>:CONFigure:MEASurement:SETting:PRESet</b>
Syntax:	:CONFigure:MEASurement:SETting:PRESet
Description:	Resets the setting values in the working memory. In contrast to the *RST and System:Preset commands, the previous stored settings remain in nonvolatile RAM and can be recalled again.
Parameters:	None
Response:	None
Example:	:conf:meas:sett:pres

Command:	<b>:CONFigure:MEASurement:SETTing:CANCel</b>
Syntax:	:CONFigure:MEASurement:SETTing:CANCel
Description:	Discard all the changes to the setting since the last save or recall.
Parameters:	None
Response:	None
Example:	:conf:meas:sett:canc

Command:	<b>:CONFigure:MEASurement:SETTing:RECall</b>
Syntax:	:CONFigure:MEASurement:SETTing:RECall
Description:	Recall a setting from FLASH memory.
Parameters:	Integer
Response:	None
Example:	:conf:meas:sett:rec 1

Command:	<b>:CONFigure:MEASurement:SETTing:SAVE</b>
Syntax:	:CONFigure:MEASurement:SETTing:SAVE
Description:	Save the current setting to FLASH memory.
Parameters:	Integer
Response:	None
Example:	:conf:meas:sett:save 1

Command:	<b>:CONFigure:MEASurement:SETTing:ERASe</b>
Syntax:	:CONFigure:MEASurement:SETTing:ERASe
Description:	Erase a setting from memory.
Parameters:	Integer
Response:	None
Example:	:conf:meas:sett:eras 1

## FETCh Subsystem Commands

This section provides the description of the following commands.

Command:	<b>:FETCh[n]:POWer?</b>
Syntax:	:FETCh[n]:POWer?
Description:	Reads the current power meter value. It does not provide its own triggering and so must be used with either continuous software triggering or a directly preceding immediate software trigger. It returns the value the previous software trigger measured. Any subsequent FETCh command will return the same value, if there is no subsequent software trigger.
Parameters:	None
Response:	The current value as a float value in dBm,W or dB. If the reference state is absolute, units are dBm or W. If the reference state is relative, units are dB.
Example:	fetc1:pow? -> +6.73370400E-04

Command:	<b>:FETCh[n]:POWer:ALL?</b>
Syntax:	:FETCh[n]:POWer:ALL?
Description:	Reads all current power meter values. It does not provide its own triggering and so must be used with either continuous software triggering or a directly preceding immediate software trigger. It returns the value the previous software trigger measured. Any subsequent FETCh command will return the same value, if there is no subsequent software trigger.
Parameters:	None
Response:	4-byte Intel float values in a binary block in Intel byte order. The values are ordered by channel. Data values are always in Watt.
Example:	fetc:pow:all? -> interpreted as +1.33555600E-006 +1.34789100E-006 +1.37456900E-006

Command:	<b>:FETCh[n]:POWer:ALL:CSV?</b>
Syntax:	:FETCh[n]:POWer:ALL:CSV?
Description:	Reads all current power meter values. It does not provide its own triggering and so must be used with either continuous software triggering or a directly preceding immediate software trigger. It returns the value the previous software trigger measured. Any subsequent FETCh command will return the same value, if there is no subsequent software trigger.
Parameters:	None
Response:	String containing the power values from each available channel in a comma separated format. Data values are always in Watt.
Example:	fetc:pow:all:CSV? " -> +1.33555600E-06, +1.34789100E-06, +1.37456900E-06"

Command:	<b>:FETCh[n]:POWer:ALL:CONFig?</b>
Syntax:	:FETCh[n]:POWer:ALL:CONFig?
Description:	Returns the channel and channel numbers for all available power meter channels. Use this command to match returned power values to the appropriate channel and channel number.
Parameters:	None
Response:	A binary block (Intel byte order) consisting of 2-byte unsigned integer value pairs (so each pair has 4 bytes). The first member of the pair represents the channel number, the second value is always 1.
Example:	fetc:pow:all:conf? interpreted as 1,1 2,1 3,1 4,1 This 16-byte block means that there are four powermeters present: Channel 1, Channel 1 Channel 2, Channel 1 Channel 3, Channel 1 Channel 4, Channel 1

Command:	<b>:FETCh[n]:POWer:MAX?</b>
Syntax:	:FETCh[n]:POWer:MAX?
Description:	Returns the maximum power value since the reset.
Parameters:	None
Response:	Maximum power value
Example:	fetc1:pow:max? -> +3.47480224E-03



Command:	<b>:FETCh[n]:POWer:MIN?</b>
Syntax:	:FETCh[n]:POWer:MIN?
Description:	Returns the minimum power value since the reset.
Parameters:	None
Response:	Minimum power value
Example:	fetc1:pow:max? -> +3.47375195E-03

Command:	<b>:FETCh[n]:POWer:EXTRema:RESet</b>
Syntax:	:FETCh[n]:POWer:EXTRema:RESet
Description:	Resets the maximum and minimum power values.
Parameters:	None
Response:	None
Example:	fetc1:pow:extr:res

# INITiate Subsystem Commands

This section provides the description of the following commands.

Command:	<b>:INITiate[n][:CHANnel[m]][:IMMediate]</b>
Syntax:	:INITiate[n][:CHANnel[m]][:IMMediate]
Description:	Initiates the software trigger system and completes one full trigger cycle, that is, one measurement is made for selected [n]. In logging mode it triggers all channels independent from [n].
Parameters:	None
Response:	None
Example:	init1:imm init2:imm

Command:	<b>:INITiate[n][:CHANnel[m]]:CONTinuous</b>
Syntax:	:INITiate[n][:CHANnel[m]]:CONTinuous<wsp><boolean>
Description:	Sets the software trigger system to continuous measurement mode.
Parameters:	A boolean value: <ul style="list-style-type: none"> <li>0 or OFF: do not measure continuously</li> <li>1 or ON: measure continuously</li> </ul>
Response:	None
Example:	init2:cont 1

Command:	<b>:INITiate[n][:CHANnel[m]]:CONTinuous?</b>
Syntax:	:INITiate[n][:CHANnel[m]]:CONTinuous?
Description:	Queries whether the software trigger system operates continuously or not.
Parameters:	None
Response:	A boolean value: <ul style="list-style-type: none"> <li>0 or OFF: do not measure continuously</li> <li>1 or ON: measure continuously</li> </ul>
Example:	init2:cont? -> 1

## READ Subsystem Commands

This section provides the description of the following commands.

Command:	<b>:READ[n]:POWer:ALL?</b>
Syntax:	:READ[n]:POWer:ALL?
Description:	Reads all available power channels. It provides its own software triggering and does not need a triggering command.
Parameters:	None
Response:	4-byte Intel float values in a binary block in Intel byte order. The values are ordered by channel. Data values are always in Watt.
Example:	read:pow:all? -> interpreted as +1.33555600E-006 +1.34789100E-006 +1.37456900E-006

Command:	<b>:READ[n]:POWer:ALL:CSV?</b>
Syntax:	:READ[n]:POWer:ALL:CSV?
Description:	Reads all available power channels. It provides its own software triggering and does not need a triggering command.
Parameters:	None
Response:	4-byte Intel float values in a binary block in Intel byte order. The values are ordered by channel. Data values are always in Watt.
Example:	read:pow:all? -> interpreted as +1.33555600E-006 +1.34789100E-006 +1.37456900E-006

Command:	<b>:READ[n]:POWer:ALL:CONFIg?</b>
Syntax:	:READ[n]:POWer:ALL:CONFIg?
Description:	Returns the channel and channel numbers for all available power meter channels. Use this command to match returned power values to the appropriate channel and channel number.
Parameters:	None
Response:	A binary block (Intel byte order) consisting of 2-byte unsigned integer value pairs (so each pair has 4 bytes). The first member of the pair represents the channel number, the second member of the pair represents the channel number. The channel number is always 1.
Example:	read1:pow:all:conf? -> interpreted as 1 1 2 1 3 1 4 1 This 16-byte block means that there are four power meters present

Command:	:READ[n]:POWer?
Syntax:	:READ[n]:POWer?
Description:	<p>Reads the current power meter value. It provides its own software triggering and does not need a triggering command.</p> <p>If the software trigger system operates continuously, this command is identical to :FETCh[n]:POWer?.</p> <p>If the software trigger system does not operate continuously, this command is identical to generating a software trigger and then reading the power meter value.</p> <p>The power meter must be running for this command to be effective.</p>
Parameters:	None
Response:	<p>The current power meter reading as a float value in dBm, W or dB.</p> <p>If the reference state is absolute, units are dBm or W.</p> <p>If the reference state is relative, units are dB.</p>
Example:	read1:pow? -> +1.33555600E-006

## Measurement Functions – The SENSE Subsystem

The SENSE subsystem lets you control measurement parameters for a Power Sensor, an Optical Head Interface module, or a return loss module.

Command:	<b>:SENSE[n]:CORRection</b>
Syntax:	:SENSE[n]:CORRection<wsp> <value>[DB MDB]
Description:	Enters a calibration value for a module.
Parameters:	The calibration factor as a float value If no unit type is specified, decibels (dB) is implied.
Response:	None
Example:	:sens1:corr 10DB

Command:	<b>:SENSE[n]:CORRection?</b>
Syntax:	:SENSE[n]:CORRection?
Description:	Returns the calibration factor for a module.
Parameters:	None
Response:	The calibration factor as a float value. Units are in dB, although no units are returned in the response message.
Example:	:sens1:corr? → +1.00000000E+000

Command:	<b>:SENSE[n]:CORRection:COLLect:ZERO</b>
Syntax:	:SENSE[n]:CORRection:COLLect:ZERO
Description:	Zeros the electrical offsets for a power meter channel. <b>NOTE:</b> Cover the optical inputs or switch the input source off before starting.
Parameters:	
Response:	None
Example:	:sens1:corr:coll:zero

Command:	<b>:SENSe[n]:CORRection:COLLect:ZERO?</b>
Syntax:	:SENSe[n]:CORRection:COLLect:ZERO?
Description:	Returns the status of the most recent zero command for a power meter channel. The result is backed up in the nonvolatile RAM. <b>NOTE:</b> If a channel fails to zero, it continues to use the result of the last successful zeroing.
Parameters:	None
Response:	<ul style="list-style-type: none"> <li>0: zero succeeded without errors.</li> <li>any other number: remote zeroing failed</li> </ul>
Example:	:sens1:corr:coll:zero? → 0

Command:	<b>:SENSe:CORRection:COLLect:ZERO:ALL</b>
Syntax:	:SENSe:CORRection:COLLect:ZERO:ALL
Description:	Zeros the electrical offsets for all power meter channels. <b>NOTE:</b> Cover the optical inputs or switch the input source off before starting.
Parameters:	
Response:	None
Example:	:sens:corr:coll:zero:all

Command:	<b>:SENSe:CORRection:COLLect:ZERO:ALL?</b>
Syntax:	:SENSe:CORRection:COLLect:ZERO:ALL?
Description:	Returns the status of the most recent zero command for a power meter channel. The result is backed up in the nonvolatile RAM. <b>NOTE:</b> If a channel fails to zero, it continues to use the result of the last successful zeroing.
Parameters:	None
Response:	<p>A hexadecimal integer value which represents the result for all channels. Each hexadecimal digit represents one channel.</p> <ul style="list-style-type: none"> <li>0: zero succeeded without errors.</li> <li>Any other number: remote zeroing failed</li> </ul> <p><b>Example 1:</b> sens:chan:corr:coll:zero:all? → +272 272 decimal = 0x110. This means zeroing failed on channels 2 and 3. All other channels were successful. <b>Example 2:</b> sens:chan:corr:coll:zero:all? → +286326784 286326784 decimal = 0x11110000. This means zeroing failed on channels 5,6,7,8. All other channels were successful.</p>
Example:	:sens:chan:corr:coll:zero:all? → 0

Command:	<b>:SENSe[n]:CORRection:COLLect:ZERO:QUAD</b>
Syntax:	:SENSe[n]:CORRection:COLLect:ZERO:QUAD
Description:	Zeros the electrical offsets for all 4 power meter channels of the selected quad if [n] is between 1 and 4 then the first quad is zeroed if [n] is between 5 and 8 then the second quad is zeroed <b>NOTE:</b> Cover the Optical Inputs or switch the input source off before starting.
Parameters:	None
Response:	None
Example:	:sens1:corr:coll:zero:quad

Command:	<b>:SENSe[n]:CORRection:COLLect:ZERO:QUAD?</b>
Syntax:	:SENSe[n]:CORRection:COLLect:ZERO:QUAD?
Description:	Returns the status of the most recent zero command for all 4 power meter channels of the selected quad. if [n] is between 1 and 4 then the first quad zero result is returned if [n] is between 5 and 8 then the second quad zero result is returned The result is backed up in the nonvolatile RAM. <b>NOTE:</b> If a channel fails to zero, it continues to use the result of the last successful zeroing.
Parameters:	None
Response:	A hexadecimal integer value which represents the result for all 4 power meter channels of the selected quad. Each hexadecimal digit represents one channel . <ul style="list-style-type: none"> <li>0: zero succeeded without errors.</li> <li>Any other number: remote zeroing failed</li> </ul> <b>Example 1:</b> sens1:chan:corr:coll:zero:quad? → +272 272 decimal = 0x110. This means zeroing failed on channels 2 and 3. All other channels were successful. <b>Example 2:</b> sens5:chan:corr:coll:zero:quad? → +272 272 decimal = 0x110. This means zeroing failed on channels 6 and 7. All other channels were successful.
Example:	:sens1:chan:corr:coll:zero:quad? → 0

Command:	<b>:SENSe[n]:FUNctIon:LOOP</b>
Syntax:	:SENSe[n]:FUNctIon:LOOP <wsp> <value>
Description:	Sets the number of logging loops
Parameters:	Number of Loops, an integer value. 0 = Endless Streaming 1 = 1 (Default) 2 = 2 (For 2 Million Points with Buffer A and B) .... n
Response:	None
Example:	:SENS1:FUNC:LOOP 0
<hr/>	
Command:	<b>:SENSe[n]:FUNctIon:LOOP?</b>
Syntax:	:SENSe[n]:FUNctIon:LOOP?
Description:	Gets the number of logging loops. See: :SENSe[n]:FUNctIon:RESult:BUFA? on page 76.
Parameters:	None
Response:	Number of loops: Number of loops is an integer value. 0 = Endless Streaming 1 = 1 (Default) 2 = 2 (For 2 Million Points with Buffer A and B) .... n
Example:	:SENS1:FUNC:LOOP? → 0
<hr/>	
Command:	<b>:SENSe[n]:FUNctIon:PARAmeter:CHECK?</b>
Syntax:	:SENSe[n]:FUNctIon:PARAmeter:CHECK?
Description:	Returns whether the currently settings are consistent. A string with a detailed description of a configuration problem, or "OK" if the sweep os configured correctly. The responses shown below are all the possible configuration problem string. Sum of pre trigger and data points is higher than 1048576. No threshold triggering in relative mode allowed.



Parameters:	None
Response:	Returns whether the currently settings are consistent. A string with a detailed description of a configuration problem, or "OK" if the sweep os configured correctly. The responses shown below are all the possible configuration problem strings: 0,OK Sum of pre trigger and data points is higher than 1048576. No threshold triggering in relative mode allowed.
Example:	:sens1:func:par:check? → 0,OK

Command:	<b>:SENSe[n]:FUNCTION:PARAMeter:LOGGing</b>
Syntax:	:SENSe[n]:FUNCTION:PARAMeter:LOGGing<wsp> <data points>, <averaging time>[NS US MS S]
Description:	Sets the number of data points and the averaging time for the logging data acquisition function.
Parameters:	<b>Data Points:</b> Data Points is the number of samples that are recorded before the logging mode is completed. Data Points is an integer value. <b>Averaging time:</b> Averaging time is a time value in seconds. There is no time delay between averaging time periods. Use :SENSe[n]:FUNCTION:PARAMeter:STABility? if you want to use delayed measurement. <b>NOTE:</b> Setting parameters for the logging function sets some parameters, including hidden parameters, for the stability and MinMax functions and vice versa. If you specify no units for the averaging time value in your command, seconds are used as the default. See :SENSe[n]:FUNCTION:STATe for information on starting/stopping a data acquisition function. See :SENSe[n]:FUNCTION:RESult? for information on accessing the results of a data acquisition function. <b>NOTE:</b> Before using this command, ensure to stop logging for all available channel. Details can be found in the Application Note "Transient Optical Power Measurements with the N7744C and N7745C". <a href="http://literature.cdn.keysight.com/litweb/pdf/5990-3710EN.pdf">http://literature.cdn.keysight.com/litweb/pdf/5990-3710EN.pdf</a>
Response:	None
Example:	:sens1:func:par:logg 64,1ms

Command:	<b>:SENSe[n]:FUNCTION:PARAMeter:LOGGing?</b>
Syntax:	:SENSe[n]:FUNCTION:PARAMeter:LOGGing?
Description:	Returns the number of data points and the averaging time for the logging data acquisition function.
Parameters:	None
Response:	Returns the number of data points as an integer value and the averaging time, $t_{avg}$ , as a float value in seconds.
Example:	:sens1:func:par:logg? → +64,+1.00000000E-001

Command:	<b>:SENSe[n]:FUNCTION:PARAMeter:MINMax</b>
Syntax:	:SENSe[n]:FUNCTION:PARAMeter:MINMax<wsp> CONTinuous WINDow REFResh,<data points>
Description:	Sets the MinMax mode and the number of data points for the MinMax data acquisition function.
Parameters:	<ul style="list-style-type: none"> <li>CONTinuous: continuous MinMax mode</li> <li>WINDow: window MinMax mode</li> <li>REFResh: refresh MinMax mode</li> </ul> <p><b>NOTE:</b> WINDow mode has the same function as REFResh mode. It is included to ensure compatibility.</p> <p><b>NOTE:</b> The time between samples in MinMax mode is at least the averaging time, but not less than about 2 ms, depending on the interface command rate.</p> <p>Data Points is the number of samples that are recorded in the memory buffer used by the WINDow and REFResh modes. Data Points is an integer value.</p> <p><b>NOTE:</b> Setting parameters for the MinMax function sets some parameters, including hidden parameters, for the stability and Logging functions and vice versa.</p> <p>See :SENSe[n]:FUNCTION:STATe for information on starting/stopping a data acquisition function.</p> <p>See :SENSe[n]:FUNCTION:RESult? for information on accessing the results of a data acquisition function.</p> <p>See Trigger Subsystem Commands for information on how triggering affects data acquisition functions.</p>
Response:	None
Example:	:sens1:func:par:logg 64,1ms

Command:	<b>:SENSe[n]:FUNCTION:PARAMeter:MINMax?</b>
Syntax:	:SENSe[n]:FUNCTION:PARAMeter:MINMax?
Description:	Returns the MinMax mode and the number of data points for the MinMax data acquisition function.
Parameters:	None
Response:	<ul style="list-style-type: none"> <li>CONT: continuous MinMax mode</li> <li>WIND: window MinMax mode</li> <li>REFR: refresh MinMax mode</li> </ul> <p><b>NOTE:</b> WINDow mode has the same function as REFResh mode. It is included to ensure compatibility. The number of data points is returned as an integer value.</p>
Example:	:sens1:func:par:minm? → WIND,+10

Command:	<b>:SENSe[n]:FUNCTION:PARAMeter:STABILITY</b>
Syntax:	:SENSe[n]:FUNCTION:PARAMeter:STABILITY<wsp> <total time>[NS US MS S],<period time>[NS US MS S],<averaging time>[NS US MS S]
Description:	Sets the total time, period time, and averaging time for the stability data acquisition function.

Parameters:	<ul style="list-style-type: none"> <li>▪ Total time: The total time from the start of stability mode until it is completed.</li> <li>▪ Period time: A new measurement is started after the completion of every period time.</li> <li>▪ Averaging time: A measurement is averaged over the averaging time.</li> </ul> <p><b>NOTE:</b> Setting parameters for the Stability function sets some parameters, including hidden parameters, for the MinMax and Logging functions and vice versa.</p> <p>The total time should be longer than the period time.</p> <p>The period time should be longer than the averaging time.</p> <p>The number of data points is equal to the total time divided by the period time.</p> <p>Total time, period time, and averaging time are time values in seconds.</p> <p>If you specify no units in your command, seconds are used as the default.</p> <p>See :SENSe[n]:FUNCTION:PARAmeter:STABility? on page 100 for information on starting/stopping a data acquisition function.</p> <p>See :SENSe[n]:FUNCTION:RESult? on page 100 for information on accessing the results of a data acquisition function.</p> <p>See :TRIGger[n]:CHANnel[m]:INPut for information on how triggering affects data acquisition functions.</p>
-------------	----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Response:	None
-----------	------

Example:	:sens1:func:par:stab 1s,0.1s,0.1s
----------	-----------------------------------

Command:	:SENSe[n]:FUNCTION:PARAmeter:STABility?
----------	-----------------------------------------

Syntax:	:SENSe[n]:FUNCTION:PARAmeter:STABility?
---------	-----------------------------------------

Description:	Returns the total time, period time, and averaging time for the stability data acquisition function.
--------------	------------------------------------------------------------------------------------------------------

Parameters:	None
-------------	------

Response:	Total time, delay time, and averaging time are float values in seconds.
-----------	-------------------------------------------------------------------------

Example:	:sens1:func:par:stab? → +1.00000000E+000, +1.00000000E-001, +1.00000000E-001
----------	------------------------------------------------------------------------------

Command:	:SENSe[n]:FUNCTION:RESult?
----------	----------------------------

Syntax:	:SENSe[n]:FUNCTION:RESult?
---------	----------------------------

Description:	Returns the data array of the last data acquisition function.
--------------	---------------------------------------------------------------

Parameters:	None
Response:	<p>The last data acquisition function's data array as a binary block.            One measurement value is a 4 byte little-endian IEEE 754 single precision value.            For Logging and Stability Data Acquisition functions.            For the MinMax Data Acquisition function, the query returns the minimum, maximum and current power values.            See Data Types for more information on Binary Blocks.            See How to Log Results for information on logging using VISA calls. There are some tips about how to use float format specifiers to convert the binary blocks (32 Bit / IEEE 754 single precision format).</p>
Example:	<p>:sens1:func:res? → #255....            returns a data array for Logging and Stability Data Acquisition functions            :sens1:func:res? → #255            Min: 7.24079E-04, Max: 7.24252E-04, Act: 7.24155E-04            returns the minimum, maximum and current power values for the MinMax Data Acquisition function</p>
Command:	<b>:SENSE[n]:FUNCTION:RESult:BLOCK?</b>
Syntax:	:SENSE[n]:FUNCTION:RESult:BLOCK?<wsp><offset>,<# of data points>
Description:	Returns a specific binary block (Intel byte order) from the data array for the last data acquisition function.
Parameters:	<p>&lt;offset&gt; A zero based offset; the number of data points to ignore.            # data points The number of data points (not bytes!) to return.</p>
Response:	<p>The last stability or logging data acquisition function's data array as a binary block.            This function is not available for min-max measurements.            One measurement value is a 4 byte little-endian IEEE 754 single precision value.</p>
Example:	:sens1:func:res:bloc? #5, 2 → interpreted as 7.24079E-04,7.24252E-04
Command:	<b>:SENSE[n]:FUNCTION:RESult:BUFA?</b>
Syntax:	:SENSE[n]:FUNCTION:RESult:BUFA?
Description:	<p>Returns the data array of the last data acquisition function in Buffer A.            This works only for Logging and Stability Data Acquisition in the loop mode, not for the MinMax Data Acquisition.</p>
Parameters:	None
Response:	<p>The last data acquisition function's data array as a binary block.            For Logging and Stability Data Acquisition functions, one measurement value is a 4 byte little-endian IEEE 754 single precision value. You can read the most recent results out of one buffer while the next logging measurement is filling the other buffer.</p>
Example:	:sens1:func:res:bufa? → #255....

Command:	<b>:SENSe[n]:FUNCTION:RESult:BUFB?</b>
Syntax:	:SENSe[n]:FUNCTION:RESult:BUFB?
Description:	Returns the data array of the last data acquisition function in Buffer B. This works only for Logging and Stability Data Acquisition in the loop mode, not for the MinMax Data Acquisition.
Parameters:	None
Response:	The last data acquisition function's data array as a binary block. For Logging and Stability Data Acquisition functions, one measurement value is a 4 byte little-endian IEEE 754 single precision value. You can read the most recent results out of one buffer while the next logging measurement is filling the other buffer.
Example:	:sens1:func:res:bufb? → #255....

Command:	<b>:SENSe[n]:FUNCTION:RESult:BLOCK?</b>
Syntax:	:SENSe[n]:FUNCTION:RESult:BLOCK? <wsp> <offset>,<# of data points>
Description:	Returns a specific binary block (Intel byte order) from the data array for the last data acquisition function. This function is not available for min-max measurements.
Parameters:	<offset> A zero based offset; the number of data points to ignore. # data points The number of data points (not bytes!) to return.
Response:	The last data acquisition function's data array as a binary block. For Logging and Stability Data Acquisition functions, one measurement value is a 4 byte little-endian IEEE 754 single precision value. See "Data Types" for more information on Binary Blocks. See "How to Log Results" for information on logging using VISA calls. There are some tips about how to use float format specifiers to convert the binary blocks into float values.
Example:	:sens1:func:res:bloc? #5, 2 → interpreted as 7.24079E-04,7.24252E-04

Command:	<b>:SENSe[n]:FUNCTION:RESult:INDEX?</b>
Syntax:	:SENSe[n]:FUNCTION:RESult:INDEX?
Description:	Gets the number of already finished logging loops.
Parameters:	None
Response:	Number of loops: Number of loops is an integer value.
Example:	:sens1:func:res:index? → 1

Command:	<b>:SENSe[n]:FUNCTION:RESult:MAXBlocksize?</b>
Syntax:	:SENSe[n]:FUNCTION:RESult:MAXBlocksize? <wsp> <offset> <# of data points>
Description:	Returns the maximum block size for a single SCPI transfer for power meter data acquisition functions. If your application requires more data points please use SENSE[n]:FUNCTION:RESult:BLOCK? instead of SENSE[n]:FUNCTION:RESult?
Parameters:	None
Response:	An integer value, number of data points. See Data Types for more information on Binary Blocks.
Example:	:sens1:func:res:maxb? → +204050

Command:	<b>:SENSe[n]:FUNCTION:STATe</b>
Syntax:	:SENSe[n]:FUNCTION:STATe<wsp>LOGGing STABility MINMax,STOP START
Description:	Enables/Disables the logging, MinMax, or stability data acquisition function mode.
Parameters:	<ul style="list-style-type: none"> <li>LOGGing: Logging data acquisition function</li> <li>STABility: Stability data acquisition function</li> <li>MINMax: MinMax data acquisition function</li> <li>STOP: Stop data acquisition function</li> <li>START: Start data acquisition function</li> </ul> <p>See :SENSe[n][:CHANnel[m]]:FUNCTION:PARAmeter:LOGGing for more information on the logging data acquisition function. Stop any function before you try to set up a new function. Some parameters cannot be set until you stop the function. Details can be found in the Application Note "Transient Optical Power Measurements with the N7744C and N7745C".  <a href="http://literature.cdn.keysight.com/litweb/pdf/5990-3710EN.pdf">http://literature.cdn.keysight.com/litweb/pdf/5990-3710EN.pdf</a></p>
Response:	None
Example:	:sens1:func:stat logg,star

Command:	<b>:SENSe[n]:FUNCTION:STATe?</b>
Syntax:	:SENSe[n]:FUNCTION:STATe?
Description:	Returns the function mode and the status of the data acquisition function.

Parameters:	None	
Response:	NONE LOGGING_STABILITY MINMAX PROGRESS COMPLETE	No function mode selected Logging or stability data acquisition function MinMax data acquisition function Data acquisition function is in progress Data acquisition function is complete
Example:	:sens1:func:stat? → LOGGING_STABILITY,COMPLETE	

Command:	<b>:SENSe[n]:POWer:GAIN:AUTO</b>	
Syntax:	:SENSe[n]:POWer:GAIN:AUTO<wsp><value>	
Description:	Set the Auto Gain.	
Parameters:	<p>0 = Auto Gain Off. This is the position for best transient response. 1 = Auto Gain On (Default) This is the Position for best dynamic. Auto gain only works for averaging times &gt;=10us. For shorter averaging times, the auto gain is always disabled. The Auto Gain setting works also in the logging and stability modes, where it also increases dynamic or enhances transient response. <b>NOTE:</b> For Logging / Stability, set the Autogain to the same value for all channels. Send the Autogain commands before setting Average Time or configuring Logging/Stability. <b>NOTE:</b> Details are in the Application Note "Transient Optical Power Measurements with the N774x-Series Multiport Power Meter". <a href="http://literature.cdn.keysight.com/litweb/pdf/5990-3710EN.pdf">http://literature.cdn.keysight.com/litweb/pdf/5990-3710EN.pdf</a> <b>NOTE:</b> Disable Auto Gain when modulated Signals are to be measured.</p>	
Response:	None	
Example:	:sens1:pow:gain:auto 1	

Command:	<b>:SENSe[n]:POWer:GAIN:AUTO?</b>	
Syntax:	:SENSe[n]:POWer:GAIN:AUTO?	
Description:	Get the Auto Gain.	
Parameters:	None	
Response:	<p>0 = Auto Gain Off. This is the position for best transient response. 1 = Auto Gain On (Default) This is the Position for best dynamic.</p>	
Example:	:sens1:pow:gain:auto? → 1	

Command:	<b>:SENSe[n]:POWer:ATIMe</b>
Syntax:	:SENSe[n]:POWer:ATIMe<wsp><averaging time>[NS US MS S]
Description:	Sets the averaging time.
Parameters:	The averaging time as a float value in seconds. If you specify no units in your command, seconds are used as the default. <b>NOTE:</b> For N774-C power meters the internal granularity of the averaging time is 1us which allows flexibility.
Response:	None
Example:	:sens1:pow:atim 1s

Command:	<b>:SENSe[n]:POWer:ATIMe?</b>
Syntax:	:SENSe[n]:POWer:ATIMe?
Description:	Returns the averaging time.
Parameters:	None
Response:	The averaging time as a float value in seconds.
Example:	:sens1:pow:atim? → +1.00000000E+000

Command:	<b>:SENSe[n]:POWer:OUTPut</b>
Syntax:	SENSe[n]:POWer:OUTPut <wsp><mode>
Description:	Enable or Disable the front Front Panel Analog Output BNC Connector of a channel. The analog voltage is always in the range between 0 and 2V.
Parameters:	A string value: <ul style="list-style-type: none"> <li>▪ <b>DISabled:</b> disable the analog output</li> <li>▪ <b>LINear:</b> For linear output, the voltage represents 1 V corresponds to the nominal value of the current range (like 100 µW in the -10 dBm range) and 0 V corresponds to zero signal.</li> <li>▪ <b>LOGarithmic:</b> For logarithmic output, the voltage represents the optical power independent of the current range. For power meters with a maximum range of +10 dBm, 2 V corresponds to +20 dBm, 1.8 V to +10 dBm, 1.6 V to 0 dBm, 1.4 V to -10 dBm,... 0 V to -80 dBm. For power meters with a maximum range of +20 or +30 dBm, the scale is shifted by 10 dB, so 2 V represents +30 dBm and 0 V is -70 dBm. For the 81628C with a +40 dBm maximum range, 2 V is +40 dBm and 0 V is -60 dBm.</li> </ul>



Response:	None
Example:	sens1:pow:outp LIN
Affects	N774-C instruments (except N7744C and N7745)

Command:	<b>:SENSe[n]:POWer:OUTPut?</b>
Syntax:	SENSe[n]:POWer:OUTPut?
Description:	Returns the current analog output state of a channel.
Parameters:	None
Response:	A string value: <ul style="list-style-type: none"> <li>DISabled: disable the analog output</li> <li>LINear: linear voltage output is active</li> <li>LOGarithmic : logarithmic voltage output is active</li> </ul>
Example:	sens1:pow:rang:auto 1
Affects	N774-C instruments (except N7744C and N7745)

Command:	<b>:SENSe[n]:POWer:RANGe:AUTO</b>
Syntax:	:SENSe[n]:POWer:RANGe:AUTO <wsp><boolean>
Description:	<p>Enables or disables automatic power ranging for the channel.</p> <p>If automatic power ranging is enabled, ranging is automatically determined by the instrument. Otherwise, it must be set by the sens[n]:pow:rang command.</p> <p>Automatic ranging while other commands are sent to power meters has lead to timing conflicts in some configurations. Automation programs can often better control the range directly.</p> <p>NOTE: Disable Auto Range when modulated Signals are to be measured. In Logging Mode Auto Range is always OFF.</p>
Parameters:	A boolean value: <ul style="list-style-type: none"> <li>0 or OFF: automatic ranging disabled</li> <li>1 or ON: automatic ranging enabled</li> </ul>
Response:	None
Example:	:sens1:pow:rang:auto 1

Command:	<b>:SENSe[n]:POWer:RANGe:AUTO?</b>	
Syntax:	:SENSe[n]:POWer:RANGe:AUTO?	
Description:	Returns whether automatic power ranging is being used by the channel.	
Parameters:	None	
Response:	A boolean value:	<ul style="list-style-type: none"> <li>0: automatic ranging is not being used</li> <li>1: automatic ranging is being used</li> </ul>
Example:	:sens1:pow:rang:auto? → 1	

Command:	<b>:SENSe[n]:POWer:RANGe</b>	
Syntax:	:SENSe[n]:POWer:RANGe<wsp><value>[DBM]	
Description:	Sets the power range for the channel. The range changes at 10 dBm intervals. The corresponding ranges for linear measurements (measurements in Watts) is given below:	
	Range	Upper Linear Power Limit
	+10 dBm	19.999 mW
	0 dBm	1.9999 mW
	-10 dBm	199.99 µW
	-20 dBm	19.999 µW
	-30 dBm	1.9999 µW
Parameters:	The range as a float value in dBm. The number is rounded to the closest multiple of 10, because the range changes at 10 dBm intervals. Units are in dBm.	
Response:	None	
Example:	:sens1:pow:rang -20DBM	

Command:	<b>:SENSe[n]:POWer:RANGe?</b>	
Syntax:	:SENSe[n]:POWer:RANGe?	
Description:	Returns the range setting for the channel.	
Parameters:	None	
Response:	The range setting as a float value in dBm (-30 to +10).	
Example:	:sens1:pow:rang? → -2.00000000E+001	

Command:	<b>:SENSe[n]:POWer:REFeRence</b>	
Syntax:	:SENSe[n]:[CHANnel[m]]:POWer:REFeRence<wsp>TOMODule TOReF,<value>PW NW UW MW Watt DBM DB MDB	
Description:	Sets the channels reference value.	
Parameters:	TOMODule:	Sets the reference value in dB used if you choose measurement relative to another channel.
	TOReF:	Sets the reference value in Watts or dBm if you choose measurement relative to a constant reference value.
<p>The reference as a float value.          You must append a unit type</p> <ul style="list-style-type: none"> <li>▪ dB if you use TOMODule or</li> <li>▪ Watts or dBm if you use TOReF.</li> </ul>		
<p>The two reference values are completely independent. When you change the reference mode using the command :SENSe[n]:POWer:REFeRence:STATe:RATio, the instrument uses the last reference value entered for the selected reference mode.</p>		
Response:	None	
Example:	:sens1:pow:ref tomod,-40DB	

Command:	<b>:SENSe[n]:POWer:REFeRence?</b>	
Syntax:	:SENSe[n]:POWer:REFeRence?<wsp>TOMODule TOReF	
Description:	Returns the channels reference value.	
Parameters:	TOMODule:	Returns the reference value in dB used if you choose measurement relative to another channel.
	TOReF:	Returns the reference value in Watts or dBm if you choose measurement relative to a constant reference value.
Response:	The reference as a float value.	
Example:	:sens1:pow:ref? toref → +1.00000000E-006	

Command:	<b>:SENSe[n]:POWer:REFeRence:DISPlay</b>	
Syntax:	:SENSe[n]:POWer:REFeRence:DISPlay	
Description:	Takes the current measured input power level value as the reference value. "disp to ref"	

Parameters:	None
Response:	None
Example:	:sens1:pow:ref:disp

Command:	<b>:SENSe[n]:POWer:REFeRence:STATe</b>	
Syntax:	:SENSe[n]:POWer:REFeRence:STATe<wsp><boolean>	
Description:	Sets the measurement units to relative or absolute units.	
Parameters:	A boolean value:	<ul style="list-style-type: none"> <li>▪ 0 or OFF: absolute</li> <li>▪ 1 or ON: relative</li> </ul>
Response:	None	
Example:	:sens1:pow:ref:stat 1	

Command:	<b>:SENSe[n]:POWer:REFeRence:STATe?</b>	
Syntax:	:SENSe[n]:POWer:REFeRence:STATe?	
Parameters:	None	
Response:	A boolean value:	<ul style="list-style-type: none"> <li>▪ 0: absolute</li> <li>▪ 1: relative</li> </ul>
Example:	:sens1:pow:ref:stat? → 1	

Command:	<b>:SENSe[n]:POWer:REFeRence:STATe:RATio</b>	
Syntax:	:SENSe[n]:POWer:REFeRence:STATe:RATio<wsp><channel number> 255 TOREF,<channel number>	
Description:	Selects the reference for the channel.	
Parameters:	Channel number:	an integer value representing the channel number you want to reference
	255 or TOREF:	results are displayed relative to an absolute reference
	Channel number:	an integer value representing the channel number you want to reference

If you want to reference another power sensor channel, use an integer value corresponding to the channel for the first parameter and an integer value corresponding to the channel for the second value.  
 If you want to use an absolute reference, use TOREF as the first parameter and any integer value as the second parameter.  
 NOTE: Note: you have to change the reference mode to "relative" to use this function. See  
 :SENSE[n]:POWer:REFErence:STATe

Response: None

Example: :sens1:pow:ref:stat:rat 2,1      References channel 2.1  
 :sens1:pow:ref:stat:rat TOREF,1      References an absolute reference

Command: :SENSE[n]:POWer:REFErence:STATe:RATio?

Syntax: :SENSE[n]:POWer:REFErence:STATe:RATio?

Description: Returns the reference setting for the channel.

Parameters: None

Response: Results are displayed relative to an absolute reference or to the current power reading from another channel.

Example: :sens1:pow:ref:stat:rat? → +255,+0      results are displayed relative to an absolute reference  
 :sens1:pow:ref:stat:rat? → +2,+1      results are displayed relative to channel 2.1

Command: :SENSE[n]:POWer:UNIT[:ALL]

Syntax: :SENSE[n]:POWer:UNIT<wsp>DBM|0|Watt|1

Description: Sets the sensor power unit of selected channel or of ALL channels.

Parameters: An integer value:      ■ 0: dBm  
                                                                  ■ 1: Watt  
 or DBM or Watt

Response: None

Example: :sens1:pow:unit 1  
 :sens:pow:unit:all 1

Command:	<b>:SENSe[n]:POWer:UNIT?</b>	
Syntax:	:SENSe[n]:POWer:UNIT?	
Description:	Gets the current sensor power unit of selected channel	
Parameters:	None	
Response:	An integer value:	<ul style="list-style-type: none"> <li>0: Current power units are dBm.</li> <li>1: Current power units are Watts.</li> </ul>
Example:	:sens1:pow:unit? → +1	

Command:	<b>:SENSe:POWer:UNIT:ALL:CSV?</b>	
Syntax:	:SENSe:POWer:UNIT:ALL:CSV?	
Description:	Gets the current sensor power units for all channels	
Parameters:	None	
Response:		
Example:	:sens:pow:unit:all:csv? → 1,0,1,1,1,1,1,1	

Command:	<b>:SENSe[n]:POWer:WAVeLength[:ALL]</b>	
Syntax:	:SENSe[n]:POWer:WAVeLength[:ALL]<wsp><value> MIN MAX DEF [PM NM UM MM M]	
Description:	Sets the sensor wavelength. Frequent use of this command can conflict with the timing of autoranging in some configurations. Autorange can be disabled before and enabled after the command if needed.	
Parameters:	The wavelength as a float value in meters.	
	Also allowed are:	<ul style="list-style-type: none"> <li>MIN: minimum programmable value</li> <li>MAX: maximum programmable value</li> <li>DEF: the preset (*RST) default value</li> </ul>
Response:	None	
Example:	:sens1:pow:wav 1550nm :sens:pow:wav:all 1550nm	

Command:	<b>:SENSe[n]:POWer:WAVelength?</b>	
Syntax:	:SENSe[n]:POWer:WAVelength? [<wsp>MIN MAX DEF]	
Description:	Gets the current sensor wavelength.	
Parameters:	None	
Response:	The wavelength as a float value in meters.	Also allowed are: <ul style="list-style-type: none"><li>▪ MIN: minimum programmable value</li><li>▪ MAX: maximum programmable value</li><li>▪ DEF: the preset (*RST) default value</li></ul>
Example:	:sens1:pow:wav? → +1.55000000E-006	

# Triggering – The TRIGger Subsystem

The TRIGger Subsystem allows you to configure how the instrument reacts to incoming or outgoing triggers.

Command:	<b>:TRIGger</b>	
Syntax:	:TRIGger<wsp>NODEA 1 NODEB 2	
Description:	Generates a hardware trigger.	
Parameters:	1 or NODEA: 2 or NODEB:	Is identical to a trigger at the Input Trigger Connector. Generates trigger at the Output Trigger Connector.
	A hardware trigger cannot be effective in the DISabled triggering mode but can be effective in DEFault, PASSthrough or LOOPback triggering modes, see <b>:TRIGger:CONFiguration</b> on page 93 for information on triggering modes.	
Response:	None	
Example:	:trig 1 :trig NODEA	

Command:	<b>:TRIGger[n]:DELAy?</b>	
Syntax:	:TRIGger[n]:DELAy?	
Description:	Returns factor for delay. Effective trigger delay time = factor/32 Mhz	
Parameters:	None	
Response:	Factor	
Example:	:trig1:del? → +0	

Command:	<b>:TRIGger[n]:DELAy</b>	
Syntax:	:TRIGger[n]:DELAy<wsp><value>	
Description:	Defines factor for delay. Effective trigger delay time = factor/32 Mhz. That means, trigger event is excuted after specified time after input trigger.	



Parameters:	Integer number between 0-997 (including limits)
Response:	None
Example:	:trig1:del 10

Command:	<b>:TRIGger[n]:INPut?</b>	
Syntax:	:TRIGger[n]:INPut?	
Description:	Returns the incoming trigger response.	
Parameters:	None	
Response:	IGNore:	Ignore incoming trigger.
	SMEasure:	Start a single measurement. If a measurement function is active, see :SENSe[n]:FUNCTION:STATe on page 78, one sample is performed and the result is stored in the data array, :SENSe[n]:FUNCTION:RESult? on page 75.
	CMEasure:	Start a complete measurement. If a measurement function is active, :SENSe[n]:FUNCTION:STATe on page 78, a complete measurement function is performed.
	MMEasure:	Similar to CME, the first hardware trigger starts the function and if LOOP is not 1, then multiple functions will be performed without waiting for further triggers.
	PREtrigger:	Defines how many samples are stored before trigger event. Not possible to use multiple cycles.
	THReshold:	Similar to PRE, but the starting event is minimum and maximum threshold values. If you don't want both limit, you can write NAN instead of number.
Example:	:trig1:inp? -> IGN	
Dual sensors	Can only be sent to primary channel, secondary channel is also affected.	

Command:	<b>:TRIGger[n]:INPut</b>	
Syntax:	:TRIGger[n]:INPut<wsp><trigger response>	
Description:	Sets the incoming trigger response and arms the module.	

Parameters:	<p>IGNore: Ignore incoming trigger.</p> <p>SMEasure: Start a single measurement. If a measurement function is active, see :SENSe[n]:FUNCTION:STATe on page 78, one sample is performed and the result is stored in the data array, :SENSe[n]:FUNCTION:RESult? on page 75.</p> <p>CMEasure: Start a complete measurement. If a measurement function is active, :SENSe[n]:FUNCTION:STATe on page 78, a complete measurement function is performed.</p> <p>MMEasure: Similar to CME, the first hardware trigger starts the function and if LOOP is not 1, then multiple functions will be performed without waiting for further triggers.</p> <p>PREtrigger: Defines how many samples are stored before trigger event. Not possible to use multiple cycles.</p> <p>THReshold: Similar to PRE, but the starting event is minimum and maximum threshold values. If you don't want both limit, you can write NAN instead of number.</p>
<p><b>NOTE</b> - If a trigger signal arrives at the Input Trigger Connector at the same time that the :SENSe[n]:FUNCTION:STATe on page 78 command is executed, the first measurement value is invalid. You should always discard the first measurement value in this case. The module performs the appropriate action when it is triggered.</p>	
Response:	None
Example:	<pre>:trig1:inp ign :trig1:inp pre,1000 :trig1:inp thr,1000,0.0001,0.0015 :trig1:inp thr,1000,0.1mw,1dBm</pre>
Command:	:TRIGger[n]:INPut:EDGe?
Syntax:	:TRIGger[n]:INPut:EDGe?
Description:	Returns the trigger edge detection of the BNC TTL input.
Parameters:	None
Response:	<ul style="list-style-type: none"> <li>RISing: trigger on the high edge of the BNC TTL input</li> <li>FALLing: trigger on the low edge of the BNC TTL input</li> </ul>
Example:	:TRIG:INP:EDG? → RIS
Command:	:TRIGger[n][:INPut:EDGe
Syntax:	:TRIGger[n][:INPut:EDGe
Description:	Sets the trigger edge detection of the BNC TTL input.

Parameters:	<ul style="list-style-type: none"> <li>▪ 0 or RISING: trigger on the high edge of the BNC TTL input</li> <li>▪ 1 or FALLING: trigger on the low edge of the BNC TTL input</li> </ul>
Response:	None
Example:	:TRIGg:INP:EDG RIS

Command:	<b>:TRIGger[n]:OFFSet?</b>
Syntax:	:TRIGger[n]:OFFSet?
Description:	Returns the number of incoming triggers received before data logging begins.
Parameters:	None
Response:	An integer value.
Example:	:trig1:offs? -> 5

Command:	<b>:TRIGger[n]:OFFSet</b>
Syntax:	:TRIGger[n]:OFFSet <value>
Description:	Sets the number of incoming triggers received before data logging begins.
Parameters:	<value> - an integer value. (maximum possible value is 2147483647)
Response:	None
Example:	:trig1:offs 5

Command:	<b>:TRIGger[n]:OUTPut?</b>
Syntax:	:TRIGger[n]:OUTPut?
Description:	Returns the condition that causes an output trigger.
Parameters:	none

Response:	DISabled: AVGover: MEASure:	Never When averaging time period finishes When averaging time period begins
Example:	:trig1:outp? -> DIS	
Dual sensors	Can only be sent to primary channel, secondary channel parameters are identical.	

Command:	<b>:TRIGger[n]:OUTPut</b>	
Syntax:	:TRIGger[n]:OUTPut	
Description:	Specifies when an output trigger is generated and arms the channel.	
Parameters:	DISabled: AVGover: MEASure:	Never When averaging time period finishes When averaging time period begins
Response:	None	
Example:	:trig1:outp dis	
Dual sensors	Can only be sent to primary channel, secondary channel is also affected. In continuous mode, wav:swe:step:[widt] is used for triggering, see [:SOURce[n]][:CHANnel[m]]:WAVelength:SWEep:STEP:[WIDTH].	

Command:	<b>:TRIGger:CONFiguration?</b>	
Syntax:	:TRIGger:CONFiguration?	
Description:	Returns the hardware trigger configuration.	
Parameters:	None	
Response:	0 or DISabled: 1 or DEFault:  2 or PASSthrough:  3 or LOOPback:	Trigger connectors are disabled. The Input Trigger Connector is activated, the incoming trigger response for each channel. The same as DEFault but a trigger at the Input Trigger Connector generates a trigger at the Output Trigger Connector automatically. The same as DEFault but a trigger at the Output Trigger Connector generates a trigger at the Input Trigger Connector automatically.
Example:	:trig:conf? -> DEF	

Command:	<b>:TRIGger:CONFIguration</b>	
Syntax:	:TRIGger:CONFIguration	
Description:	Sets the hardware trigger configuration with regard to Output and Input Trigger Connectors.	
Parameters:	0 or DISabled: 1 or DEFault:  2 or PASSthrough:  3 or LOOPback:	Trigger connectors are disabled. The Input Trigger Connector is activated, the incoming trigger response for each channel.  The same as DEFault but a trigger at the Input Trigger Connector generates a trigger at the Output Trigger Connector automatically. The same as DEFault but a trigger at the Output Trigger Connector generates a trigger at the Input Trigger Connector automatically.
Response:	none	
Example:	:trig:conf def	



# 5 Error Codes

Error Strings / 96

This chapter gives information about error codes used with the N774-C series instruments.

# Error Strings

Error strings in the range -100 to -183 are defined by the SCPI standard, downloadable from: <http://www.ivifoundation.org/docs/scpi-99.pdf>

**Table 1** Overview for Supported Strings

Error	
Number	String
0	"No error"
-100	"Command Error" [This is the generic syntax error used when a more specific error cannot be detected. This code indicates only that a Command Error as defined in <i>IEEE 488.2</i> , 11.5.1.1.4 has occurred.]
-101	"Invalid character" [A syntactic element contains a character which is invalid for that type; for example, a header containing an ampersand, SETUP. This error might be used in place of error -114 and perhaps some others.]
-102	"Syntax error" [An unrecognized command or data type was encountered; for example, a string was received when the device does not accept strings.]
-103	"Invalid separator" [The parser was expecting a separator and encountered an illegal character; for example, the semicolon was omitted after a program message unit]
-104	"Data type error" [The parser recognized a data element different than one allowed; for example, numeric or string data was expected but block data was encountered.]
-105	"GET not allowed" [A Group Execute Trigger was received within a program message (see <i>IEEE488.2</i> , 7.7).]
-108	"Parameter not allowed" [More parameters were received than expected for the header]
-109	"Missing parameter" [Fewer parameters were received than required for the header]
-110	"Command header error"
-111	"Header separator error"
-112	"Program mnemonic too long" [The header contains more than twelve characters (see <i>IEEE 488.2</i> , 7.6.1.4.1).]



Error	
Number	String
-113	"Undefined header" [The header is syntactically correct, but it is undefined for this specific device; for example, *XYZ is not defined for any device.]
-114	"Header suffix out of range"
-115	"Unexpected number of parameters"
-120	"Numeric data error" [This error, as well as errors -121 through -129, are generated when parsing a data element which appears to be numeric, including the nondecimal numeric types. This error message is used if the device cannot detect a more specific error.]
-121	"Invalid character in number" [An invalid character for the data type being parsed was encountered; for example, an alpha in a decimal numeric]
-123	"Exponent too large" [The magnitude of the exponent was larger than 32000 (see <i>IEEE 488.2, 7.7.2.4.1</i> ).]
-124	"Too many digits" [The mantissa of a decimal numeric data element contained more than 255 digits excluding leading zeros (see <i>IEEE 488.2, 7.7.2.4.1</i> ).]
-128	"Numeric data not allowed" [A legal numeric data element was received, but the device does not accept one in this position for the header.]
-130	"Suffix error"
-131	"Invalid suffix" [The suffix does not follow the syntax described in <i>IEEE 488.2, 7.7.3.2</i> , or the suffix is inappropriate for this device.]
-134	"Suffix too long" [The suffix contained more than 12 characters (see <i>IEEE 488.2, 7.7.3.4</i> ).]
-138	"Suffix not allowed" [A suffix was encountered after a numeric element which does not allow suffixes.]
-140	"Character data error"
-141	"Invalid character data" [Either the character data element contains an invalid character or the particular element received is not valid for the header.]
-144	"Character data too long"
-148	"Character data not allowed" [A legal character data element was encountered where prohibited by the device.]

Error	
Number	String
-150	"String data error" [This error, as well as errors -151 through -159, are generated when parsing a string data element. This error message is used when the device cannot detect a more specific error.]
-151	"Invalid string data" [A string data element was expected, but was invalid for some reason (see <i>IEEE 488.2, 7.7.5.2</i> ); for example, an END message was received before the terminal quote character.]
-158	"String data not allowed" [A string data element was encountered but was not allowed by the device at this point in parsing.]
-160	"Block data error"
-161	"Invalid block data" [A block data element was expected, but was invalid for some reason (see <i>IEEE 488.2, 7.7.6.2</i> ); for example, an END message was received before the length was satisfied.]
-168	"Block data not allowed" [A legal block data element was encountered but was not allowed by the device at this point in parsing.]
-170	"Expression error" [This error, as well as errors -171 through -179, are generated when parsing an expression data element. This particular error message is used when the device cannot detect a more specific error.]
-171	"Invalid expression" [The expression data element was invalid (see <i>IEEE 488.2, 7.7.7.2</i> ); for example, unmatched parentheses or an illegal character.]
-178	"Expression data not allowed" [A legal expression data was encountered but was not allowed by the device at this point in parsing.]
-180	"Macro error"
-181	"Invalid outside macro definition" [Indicates that a macro parameter placeholder (\$<number>) was encountered outside of a macro definition.]
-183	"Invalid inside macro definition" [Indicates that the program message unit sequence, sent with a *DDT or *DMC command, is syntactically invalid (see <i>IEEE 488.2, 10.7.6.3</i> ).]
-184	"Macro parameter error"

Error	
Number	String
-185	<p>"Subop out of range"</p> <p><i>Description:</i> Suboperations are parameters that are passed to refine the destination of a command. They are used to address slots, channels, laser selections and GPIB/SCPI register levels. This error is generated if the parameter is not valid in the current context or system configuration.</p> <p><i>Example:</i> This error occurs if the user queries the status of a summary register and passes an invalid status level.</p> <p><i>Note:</i> Incorrect slots and channels addresses are handled by error code -301</p>
-200	<p>"Execution error (StatExecError)"</p> <p><i>Description:</i> This error occurs when the current function, instrument or module state (or status) prevents the execution of a command. This is a generic error which can occur for a number of reasons.</p> <p><i>Example:</i> When a powermeter has finished a logging application and data is available, the user is not able to reconfigure the logging application parameters. First, the user must stop the logging application.</p>
-201	<p>"Invalid while in local"</p> <p>"Please be patient - GPIB currently locked out"</p> <p><i>Description:</i> Some operations block the complete system. Since no sensible measurements are possible while this is true, the GPIB is locked out.</p> <p><i>Example:</i> When ARA, Lambda zeroing or zeroing is executing on a TLS module, the GPIB is not accessible.</p>
-202	"Settings lost due to rtl"
-203	"Command protected"
-210	"Trigger error"
-211	<p>"Trigger ignored"</p> <p><i>Description:</i> A trigger has been detected but ignored because of timing constraints. (For Example: average time to large).</p>
-212	<p>"Arm ignored"</p> <p><i>Description:</i> The user can set the automatic re-arming option for input and output trigger events (see <b>Error Strings</b> on page 96). When this error occurs, the device ignores the setting because the current module status does not allow the change of trigger settings.</p>
-213	<p>"Init ignored"</p> <p><i>Description:</i> The INIT:IMM command initiates a trigger and completes a full measurement cycle. The continuous measurement must be DISABLED. This error code is generated if the module is still in cont. measurement mode.</p>

Error	
Number	String
-214	"Trigger deadlock"
-215	"Arm deadlock"
-220	"Parameter error (StatParmError)" <i>Description:</i> The user has passed a parameter that cannot be changed in this way. The device cannot detect one of the following more specific errors:
-220	-220, "Parameter error (StatParmOutOfRange)" <i>Description:</i> The user has passed a parameter that exceeds the valid range for this parameter.
-220	"Parameter error (StatParmIllegalVal)" <i>Description:</i> The user has passed a parameter that does not match a value in a list of possible values.
-221	"Settings conflict (StatParmInconsistent)" <i>Description:</i> The user has passed a parameter that conflicts with other already configured parameters. <i>Example:</i> There are constraints for TLS sweep parameters: this error is generated when lambda step size exceeds the difference between start and stop wavelength. If error -221 is returned after you try to start a wavelength sweep, one of the following cases of sweep parameter inconsistency has occurred: Continuous Sweep mode AND I Start is less than I Stop. Continuous Sweep mode AND Sweep Time is too short. Adjust Sweep Speed, I Start, or I Stop. Continuous Sweep mode AND Sweep Time is too long. Adjust Sweep Speed, I Start, or I Stop. Continuous Sweep mode AND Trigger Frequency is too high. Adjust Step Size. Trigger Frequency is the Sweep Speed divided by the Step Size. Stepped Sweep mode AND Lambda Logging Enabled. Continuous Sweep mode AND Lambda Logging Enabled AND Output trigger mode not set to STFinished (Step finished). Continuous Sweep mode AND Lambda Logging is Enabled AND Modulation Source is not set to OFF. Continuous Sweep mode AND Lambda Logging is Enabled AND Sweep Cycles is not set to 1.
-222	"Data out of range (StatParmTooLarge)" <i>Description:</i> The user has passed a continuous parameter that is too large. <i>Example:</i> Wavelength 1800nm when maximum wavelength is 1700nm.
-222	"Data out of range (StatParmTooSmall)" <i>Description:</i> The user has passed a continuous parameter that is too small. <i>Example:</i> Wavelength 700nm when minimum wavelength is 800nm.

Error	
Number	String
-223	<p>"Too much data"</p> <p><i>Description:</i> A function returns more data or the user requests more data than the application is able to handle.</p> <p><i>Example:</i> A tunable laser source produces more data when lambda values of a sweep are stored than the 816x instrument is able to handle. Use the new SENSE:FUNC:RES:BLOCK? command to split the data acquisition into multiple parts.</p>
-224	<p>"Illegal parameter value"</p> <p>[Used where exact value, from a list of possibles, was expected.]</p>
-225	<p>"Out of memory"</p> <p><i>Description:</i> The request application or function cannot be executed because the instrument runs out of memory.</p>
-226	"Lists not same length"
-230	"Data corrupt or stale"
-231	<p>"Data questionable (StatValNYetAcc)"</p> <p><i>Description:</i> The data that is returned is not accurate or reliable. The user should repeat the operation. The reason for this error is unspecific.</p> <p><i>Example:</i> A powermeter configured a long average time has not completed its current measurement cycle when the user queries the current power.</p>
-231	<p>"Data questionable (StatRangeTooLow)"</p> <p><i>Description:</i> As -231 (StatValNYetAcc) but for a more specific reason: The powermeter readout data is not reliable because the currently set (manual) range does not correspond with the input power.</p>
-240	"Hardware error"
-241	"Hardware missing"
-250	"Mass storage error"
-251	"Missing mass storage"
-252	"Missing media"
-253	"Corrupt media"
-254	"Media full"
-255	"Directory full"
-256	"File name not found"

Error	
Number	String
-257	"File name error"
-258	"Media protected"
-260	"Expression error"
-261	<p>"Math error in expression (StatUnitCalculationError)"</p> <p><i>Description:</i> This may occur when the user attempts to transform data in a way that is currently not possible.</p> <p><i>Example:</i> When a powermeter is measuring very small power values in dBm (such as noise power), negative power values in Watt may also be present (such as when the powermeter calibration wavelength does not correspond to the wavelength of input signal). The instrument cannot transform negative Watt values to dBm because the logarithm of a negative value is not defined.</p>
-270	"Macro error"
-271	"Macro syntax error"
-272	<p>"Macro execution error"</p> <p>[Indicates that a syntactically legal macro program data sequence could not be executed due to some error in the macro definition (see IEEE 488.2, 10.7.6.3).]</p>
-273	<p>"Illegal macro label"</p> <p>[Indicates that the macro label defined in the *DMC command was a legal string syntax, but could not be accepted by the device (see IEEE 488.2, 10.7.3 and 10.7.6.2); for example, the label was too long, the same as a common command header, or contained invalid header syntax.]</p>
-274	"Macro parameter error"
-275	"Macro definition too long"
-276	<p>"Macro recursion error"</p> <p>[Indicates that a syntactically legal macro program data sequence could not be executed because the device found it to be recursive (see IEEE 488.2, 10.7.6.6).]</p>
-277	<p>"Macro redefinition not allowed"</p> <p>[Indicates that a syntactically legal macro label in the *DMC command could not be executed because the macro label was already defined (see IEEE 488.2, 10.7.6.4).]</p>
-278	<p>"Macro header not found"</p> <p>[Indicates that a syntactically legal macro label in the *GMC? query could not be executed because the header was not previously defined.]</p>
-280	"Program error"
-281	"Cannot create program"
-282	"Illegal program name"

Error	
Number	String
-283	"Illegal variable name"
-284	<p>"Function currently running (StatModuleBusy)"</p> <p><i>Description:</i> This error is generated when a function is currently running on a module so that it cannot process another command.</p> <p><i>Example:</i> When a powermeter is running a logging application, you are not able to configure the logging application parameters (also see -200).</p>
-285	"Program syntax error"
-286	"Program runtime error"
-290	"Memory use error"
-291	"Out of memory"
-292	"Referenced name does not exist"
-293	"Referenced name already exists"
-294	"Incompatible type"
-300	"Device-specific error"
-303	<p>"Module slot empty or slot / channel invalid"</p> <p><i>Description:</i> The user has send a command to an empty slot.</p>
-310	<p>"System error"</p> <p>[Indicates that some error, termed "system error" by the device, has occurred. This code is device-dependent.]</p>
-311	"Memory error"
-312	"PUD memory lost"
-313	"Calibration memory lost"
-314	"Save/recall memory lost"
-315	"Configuration memory lost"
-320	"Storage fault"
-321	<p>"Out of memory"</p> <p>[An internal operation needed more memory than was available.]</p>

Error	
Number	String
-330	"Self-test failed" <i>Description:</i> You have started the self test, but the module has detected an error while executing it
-340	"Calibration failed"
-350	"Queue overflow" [A specific code entered into the queue in lieu of the code that caused the error. This code indicates that there is no room in the queue and an error occurred but was not recorded.]
-360	"Communication error"
-361	"Parity error in program message"
-362	"Framing error in program message"
-363	"Input buffer overrun"
-365	"Time out error"
-368	"LambdaStop<=LambdaStart"
-369	"sweepTime < min"
-370	"sweepTime > max"
-371	"triggerFreq > max"
-372	"step < min"
-373	"triggerNum > max"
-374	"LambdaLogging = On AND Modulation = On AND ModulationSource! = CoherenceControl"
-375	"LambdaLogging = On AND TriggerOut! = StepFinished"
-376	"Lambda logging in stepped mode"
-377	"step not multiple of 0.1pm"
-378	"triggerFreq < min"
-400	"Query error" [This is the generic query error for devices that cannot detect more specific errors. This code indicates only that a Query Error as defined in <i>IEEE 488.2</i> , 11.5.1.1.7 and 6.3 has occurred.]
-410	"Query INTERRUPTED" [Indicates that a condition causing an INTERRUPTED Query error occurred (see <i>IEEE 488.2</i> , 6.3.2.3); for example, a query followed by DAB or GET before a response was completely sent.]



Error	
Number	String
-420	"Query UNTERMINATED" [Indicates that a condition causing an UNTERMINATED Query error occurred (see <i>IEEE 488.2</i> , 6.3.2.2); for example, the <b>device</b> was addressed to talk and an incomplete program message was received.]
-430	"Query DEADLOCKED" [Indicates that a condition causing an DEADLOCKED Query error occurred (see <i>IEEE 488.2</i> , 6.3.1.7); for example, both input buffer and output buffer are full and the device cannot continue.]
-440	"Query UNTERMINATED after indef resp" [Indicates that a query was received in the same program message after an query requesting an indefinite response was executed (see <i>IEEE 488.2</i> , 6.5.7.5).]

