R&S®HMC8015
Power Analyzer
SCPI Programmer's Manual

Test & Measurement

SCPI Programmers Manual

Version 01

ROHDE&SCHWARZ

Content

1 Introduction / Basics

This chapter provides basic information on operating an instrument via remote control.

1.1 Remote Control Interfaces

For remote control, LAN / USB (standard interface) or GPIB (optional interface) can be used. The optional GPIB interface has its own interface module slot on the rear panel of the R&S HMC8015.

NOTICE

Within this interface description, the term GPIB is used as a synonym for the IEC/IEEE bus interface.

SCPI (Standard Commands for Programmable Instruments) SCPI commands - messages - are used for remote control. Commands that are not taken from the SCPI standard follow the SCPI syntax rules.

1.1.1 USB Interface

In addition to a LAN interface, the R&S HMC8015 includes a USB device port. For this interface, the user can select if the instrument is accessed via virtual COM port (VCP) or via USB TMC class. The traditional version of the VCP allows the user to communicate with the R&S HMC8015 using any terminal program via SCPI commands once the corresponding Windows drivers have been installed. Naturally, the free software “HMExplorer” is also available for the R&S HMC8015. This Windows application offers the R&S HMC8015 a terminal function or the option to create screenshots.

The modern alternative to the virtual COM port is to remote control the R&S HMC8015 via USB TMC class. TMC stands for “Test & Measurement Class” which indicates that the connected measurement instrument can be recognized without special Windows drivers if VISA drivers are installed and that it can be used directly in corresponding environments. The GPIB interface serves as model to the structure of the TMC design. A major benefit of the USB TMC class is that by sampling specific registers the controlling software can determine if commands have been terminated and if they have been processed correctly. In contrast, the communication via VCP requires analysis and polling mechanisms within the controlling software which may significantly strain the interface of the measurement instruments. The TMC status registers solve this problem with the USB TMC in the same manner as is the case with the GPIB interface for the hardware, namely via corresponding control lines.

If you are using USB VCP you need to install an USB VCP driver, which can be downloaded free of charge from the R&S homepage.

NOTICE

The currently available USB VCP driver is fully tested, functional and released for Windows XP™, Windows Vista™, Windows 7™, Windows 8™ and Windows 10™, both as 32Bit or 64Bit versions.

The R&S HMC8015 USB VCP interface has to be chosen in the HMC SETUP menu and does not need any setting.

NOTICE

If the virtual COM port will be used, you have to install the virtual COM port driver. The virtual COM port (VCP) will be activated in the PC device explorer.

1.1.2 Ethernet (LAN) Interface

The settings of the parameter will be done after selecting the menu item ETHERNET and the soft key PARAMETER. You can set a fix IP adress or a dynamic IP setting via the DHCP function. Please ask your IT department for the correct setting at your network.

IP address
To set up the connection the IP address of the instrument is required. It is part of the resource string used by the program to identify and control the instrument. The resource string has the form:

TCPIP::‹IP_address›::‹IP_port›::SOCKET

The default port number for SCPI socket communication is 5025. IP address and port number are listed In the „Ethernet Settings“ of the R&S®HMC8015, see also: chapter 1.2.2,“Configuring LAN Parameters“.

Example: If the instrument has the IP address 192.1.2.3; the valid resource string is:

TCPIP::192.1.2.3::5025::SOCKET

If the LAN is supported by a DNS server, the host name can be used instead of the IP address. The DNS server (Domain Name System server) translates the host name to the IP address. The resource string has the form:

TCPIP::‹host_name›::‹IP_port›::SOCKET

To assign a host name to the R&S HMC8015, select SETUP button › Misc › Device name.

Example: If the host name is TEST1; the valid resource string is:

TCPIP::TEST1::5025::SOCKET

NOTICE

The end character must be set to linefeed (LF).

1.1.4 GPIB Interface (IEC/IEEE Bus Interface)

In addition to the GPIB functions which are available via USB TMC class, the R&S HMC8015 is optionally available with an integrated GPIB interface. This solution is particularly attractive for customers who already have an existing GPIB environment. With minimum efforts, an old instrument can be replaced by a model of the R&S HMC8015.

To be able to control the instrument via the GPIB bus, the instrument and the controller must be linked by a GPIB bus cable. A GPIB bus card, the card drivers and the program libraries for the programming language must be provided in the controller. The controller addresses the instrument with the GPIB instrument address.

Characteristics

The GPIB interface is described by the following characteristics:

- Up to 15 instruments can be connected
- The total cable length is restricted to a maximum of 15m; the cable length between two instruments should not exceed 2m.
- A wired „OR"-connection is used if several instruments are connected in parallel.

GPIB Instrument Address

In order to operate the instrument via remote control, it must be addressed using the GPIB address. The remote control address is factory-set to 20, but it can be changed in the network environment settings or in the „Setup" menu under „Interface --> Parameter". For remote control, a GPIB address from 0 to 30 are allowed. The GPIB address is maintained after a reset of the instrument settings.

1.2 Setting Up a Network (LAN) Connection

1.2.1 Connecting the Instrument to the Network

The network card can be operated with a 10 Mbps Ethernet IEEE 802.3 or a 100 Mbps Ethernet IEEE 802.3u interface.

NOTICE

Before connecting the instrument to the network or configuring the network, consult your network administrator. Errors may affect the entire network.

NOTICE

To establish a network connection, connect a commercial RJ-45 cable to one of the LAN ports of the instrument and to a PC.

1.2.2 Configuring LAN Parameters

Depending on the network capacities, the TCP/IP address information for the instrument can be obtained in different ways.

- Automatically: DHCP or AutoIP. All address information can be assigned automatically.
- Manually: the address must be set manually.

By default, the instrument is configured to use automatically configuration and obtain all address information automatically. This means that it is safe to establish a physical connection to the LAN without any previous instrument configuration.

NOTICE

If DHCP is used and the system cannot assign an IP address to the R&S HMC8015 (for instance, if no Ethernet cable is connected or the network does not support DHCP), it may take up to three minutes until a timeout allows the interface to be configured again.

Configuring LAN parameters

- Press the SETUP key and then the INTERFACE softkey.
- Press the ETHERNET and then the PARAMETER softkey.

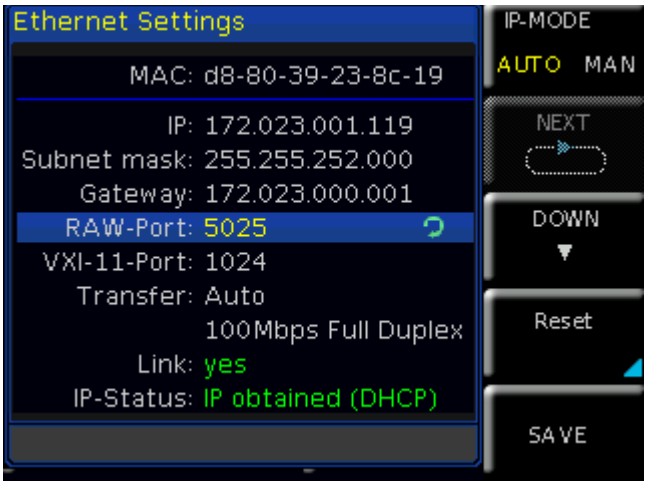

Fig. 1.1: Ethernet Settings dialog box

Some data is displayed for information only and cannot be edited. This includes the „MAC“ (physical) address of the connector and the „Link“ status information.

- Define the IP address of the instrument by entering each of the four blocks individually (manual mode) or choose the automatic IP-Mode.
  a) In manual mode (MAN) define the first block number using the knob.
  b) Press Next to move to the next block and define the number.
  c) When the IP address is complete, press Down to continue with the next setting.

- Define the „Subnetmask“ and „Gateway“ in the same way.

- Select the „RAW Port“ - the port number for SCPI socket communication.

- Select the „VXI-11- Port“ used by the instrument.

- Select the „Transfer“ mode. This mode can either be determined automatically („Auto“ setting), or you can select a combination of a transfer rate and half or full duplex manually.

- Press SAVE to save the LAN parameters.

NOTICE

The „Link“ and „IP-Status“ information at the bottom of the dialog box indicates whether a LAN connection was established successfully.

Checking LAN and SCPI connection

- Check the LAN connection using ping: `ping xxx.yyy.zzz.xxx`.

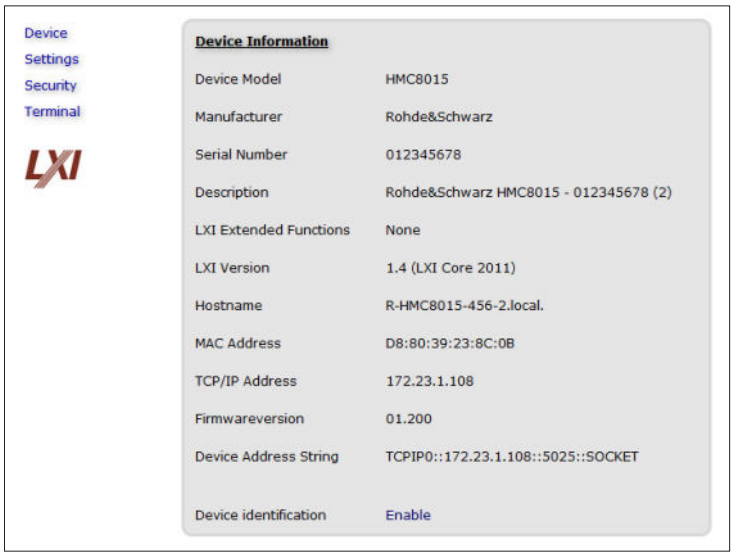

Fig. 1.2: The „Device Information“ page

- If the PC can access the instrument, enter the IP address of the address line of the internet browser on your computer: http//:xxx.yyy.zzz.xxx

- The „Device Information“ page appears. It provides information on the instrument and the LAN connection.

1.3 Switching to Remote Control

When you switch on the instrument, it is always in manual operation state („local" state) and can be operated via the front panel. When you send a command from the control computer, it will be received and executed by the instrument. The display remains on, manual operation via the front panel is always possible.

1.4 Messages and Command Structure

1.4.1 Messages

Instrument messages are employed in the same way for all interfaces, if not indicated otherwise in the description.
See also:

- Structure and syntax of the instrument messages: chapter 1.4.2, „SCPI Command Structure".
- Detailed description of all messages: chapter 2, „Command Reference".

There are different types of instrument messages:

- Commands
- Instrument responses

Commands
Commands (program messages) are messages which the controller sends to the instrument. They operate the instrument functions and request information. The commands are subdivided according to two criteria:

According to the instrument effect:

- Setting commands cause instrument settings such as a reset of the instrument or setting the frequency.
- Queries cause data to be provided for remote control, e.g. for identification of the instrument or polling a parameter value. Queries are formed by appending a question mark to the command header.

According to their definition in standards:

- The function and syntax of the Common commands are precisely defined in standard IEEE 488.2. They are employed identically on all instruments (if implemented). They refer to functions such as management of the standardized status registers, reset and self test.
- Instrument control commands refer to functions depending on the features of the instrument such as voltage settings. Many of these commands have also been standardized by the SCPI committee. These commands are marked as „SCPI compliant" in the command reference chapters. Commands without this SCPI label are device-specific, however, their syntax follows SCPI rules as permitted by the standard.

Instrument responses
Instrument responses (response messages and service requests) are messages which the instrument is sent to the controller after a query. They can contain measurement results, instrument settings and information on the instrument status.

GPIB Interface Messages
Interface messages are transmitted to the instrument on the data lines, with the attention line (ATN) being active (LOW). They are used for communication between the controller and the instrument and can only be sent by a computer which has the function of a GPIB bus controller. GPIB interface messages can be further subdivided into:

- **Universal commands:** act on all instruments connected to the GPIB bus without previous addressing
- **Addressed commands:** only act on instruments previously addressed as listeners

Universal Commands
Universal commands are encoded in the range 10 through 1F hex. They affect all instruments connected to the bus and do not require addressing. Addressed commands are encoded in the range 00 through 0F hex. They only affect instruments addressed as listeners.

1.4.2 SCPI Command Structure

SCPI commands consist of a so-called header and, in most cases, one or more parameters. The header and the parameters are separated by a „white space" (ASCII code 0x20, 32 decimal, e.g. blank). The headers may consist of several mnemonics (keywords). Queries are formed by appending a question mark directly to the header.

The commands can be either device-specific or device-independent (common commands). Common and device-specific commands differ in their syntax.

Syntax for Common Commands
Common (= device-independent) commands consist of a header preceded by an asterisk (*) and possibly one or more parameters.

***RST**	Reset	Resets the instrument.
***ESE**	Event Status Enable	Sets the bits of the event status enable registers.
***ESR?**	Event Status Query	Queries the content of the event status register.
***IDN?**	Identification Query	Queries the instrument identification string.

Table 1.4: Examples of Common Commands

Syntax for Device-Specific Commands

- **Example**: CHANnel<n>:MEASurement:FUNCtions?

Long and short form
The mnemonics feature a long form and a short form. The short form is marked by upper case letters, the long form corresponds to the complete word. Either the short form or the long form can be entered; other abbreviations are not permitted.

Example: CHANnel<n>:MEASurement:DATA? is equivalent to CHAN:MEAS:DATA?

NOTICE

Upper case and lower case notation only serves to distinguish the two forms in the manual, the instrument itself is case-insensitive.

Optional mnemonics

Some command systems permit certain mnemonics to be inserted into the header or omitted. These mnemonics are marked by square brackets. The instrument must recognize the long command to comply with the SCPI standard. Some commands are shortened by these optional mnemonics.

Example:

CHANnel<n>:LIMit<n>[:STATe] {ON}
CHAN:LIM:STAT ON is equivalent to CHAN:LIM ON

NOTICE

Do not omit an optional mnemonic if it includes a numeric suffix that is relevant for the effect of the command.

Special characters

|	**Parameters** A vertical stroke in parameter definitions indicates alternative possibilities in the sense of „or". The effect of the command differs, depending on the used parameter. **Example:** VIEW:NUMeric:PAGE<n>:SIZE {6 | 10} VIEW:NUM:PAGE1:SIZE 6 configures soft menu 1 to 6 measurement values VIEW:NUM:PAGE1:SIZE 10 configures soft menu 1 to 10 measurement values
[ ]	Mnemonics in square brackets are optional and may be inserted into the header or omitted. **Example:** CHANnel<n>:LIMit<n>[:STATe] {ON} CHAN:LIM:STAT ON is equivalent to CHAN:LIM ON
{ }	Parameters in curly brackets are optional and can be inserted once or several times, or omitted. **Example:** INTegrator:DURation {<seconds>| MIN | MAX} The following are valid commands: INT:DUR MAX INT:DUR MIN INT:DUR 2

Table 1.5: Special characters

SCPI Parameters
Many commands are supplemented by a parameter or a list of parameters. The parameters must be separated from the header by a „white space“ (ASCII code 0x20, 32 decimal, e.g. blank). Allowed parameters are:

- Numeric values
- Special numeric values
- Boolean parameters
- Text

The parameters required for each command and the allowed range of values are specified in the command description.

Numeric values
Numeric values can be entered in any form, i.e. with sign, decimal point and exponent. Values exceeding the resolution of the instrument are rounded up or down. The mantissa may comprise up to 255 characters, the exponent must lie inside the value range -32000 to 32000. The exponent is introduced by an „E“ or „e“. Entry of the exponent alone is not allowed.

Example: INT:DUR 1000 = INT:DUR 1e3

Special numeric values
The text listed below are interpreted as special numeric values. In the case of a query, the numeric value is provided.
- MIN/MAX
- MINimum and MAXimum denote the minimum and maximum value.

Example:
INT:DUR MAX
INT:DUR?, Response: 349199

Queries for special numeric values
The numeric values associated to MAXimum/MINimum can be queried by adding the corresponding mnemonics to the command. They must be entered following the quotation mark.

Example:
INT:DUR? MAX
Returns the maximum numeric value.

Boolean Parameters
Boolean parameters represent two states. The „ON“ state (logically true) is represented by „ON“ or a numeric value 1. The „OFF“ state (logically untrue) is represented by „OFF“ or the numeric value 0. The numeric values are provided as the response for a query.

Example:
CHANnel<n>:LIMit<n>[:STATe] ON
CHAN:LIM:[:STATe]?, Response: 1

Text parameters
Text parameters observe the syntactic rules for mnemonics, i.e. they can be entered using a short or long form. Like any parameter, they have to be separated from the header by a white space. In the case of a query, the short form of the text is provided.

Example:
HCOPy:FORMat BMP
HCOPy:FORMat?, Response: BMP

Overview of Syntax Elements
The following table provides an overview of the syntax elements:

:	The colon separates the mnemonics of a command.
,	The comma separates several parameters of a command.
?	The question mark forms a query.
*****	The asterisk marks a common command.
"	Quotation marks introduce a string and terminate it.
	A „white space" (ASCII code 0x20, 32 decimal, e.g. blank) separates the header from the parameters.

Table 1.6: Syntax Elements

Responses to Queries
A query is defined for each setting command. It is formed by adding a question mark to the associated setting command. According to SCPI, the responses to queries are partly subject to stricter rules than in standard IEEE 488.2.

- The requested parameter is transmitted without a header.

Example:
VIEW HARMonics
VIEW?, Response: HARM

- Maximum values, minimum values and all other quantities that are requested via a special text parameter are returned as numeric values.

Example:
INT:DUR? MAX, Response: 349199

- Truth values (Boolean values) are returned as 0 (for OFF) and 1 (for ON).

Example:
CHANnel<n>:LIMit<n>[:STATe] ON
CHAN:LIM:[:STATe]?, Response: 1

1.5 Command Sequence and Synchronization

A sequential command finishes the execution before the next command is starting. In order to make sure that commands are actually carried out in a certain order, each command must be sent in a separate command line.

NOTICE

As a general rule, send commands and queries in different program messages.

1.5.1 Preventing Overlapping Execution

To prevent an overlapping execution of commands, one of the commands *OPC, *OPC? or *WAI can be used. All three commands cause a certain action only to be carried out after the hardware has been set. The controller can be forced to wait for the corresponding action.

Command	Action	Programming the controller
***OPC**	Sets the Operation Complete bit in the ESR after all previous commands have been executed.	• Setting bit 0 in the ESE • Setting bit 5 in the SRE • Waiting for service request (SRQ)
***OPC?**	Stops command processing until 1 is returned. This is only the case after the Operation Complete bit has been set in the ESR. This bit indicates that the previous setting has been completed.	Sending *OPC? directly after the command whose processing should be terminated before other commands can be executed.
***WAI**	Stops further command processing until all commands have been executed before *WAI.	Sending *WAI directly after the command whose processing should be terminated before other commands are executed

Table 1.7: Synchronization using *OPC, *OPC? and *WAI

Command synchronization using *WAI or *OPC? appended to an overlapped command is a good choice if the overlapped command takes time to process. The two synchronization techniques simply block overlapped execution of the command. For time consuming overlapped commands it is usually desirable to allow the controller or the instrument to do other useful work while waiting for command execution. Use one of the following methods:

***OPC with a service request**

- Set the OPC mask bit (bit no. 0) in the ESE: *ESE 1
- Set bit no. 5 in the SRE: *SRE 32 to enable ESB service request.
- Send the overlapped command with *OPC
- Wait for a service request

The service request indicates that the overlapped command has finished.

***OPC? with a service request**

- Set bit no. 4 in the SRE: *SRE 16 to enable MAV service request.
- Send the overlapped command with *OPC?
- Wait for a service request

The service request indicates that the overlapped command has finished.

Event Status Register (ESE)

- Set the OPC mask bit (bit no. 0) in the ESE: *ESE 1
- Send the overlapped command without *OPC, *OPC? or *WAI
- Poll the operation complete state periodically (by means of a timer) using the sequence: *OPC; *ESR?

A return value (LSB) of 1 indicates that the overlapped command has finished.

***OPC? with short timeout**

- Send the overlapped command without *OPC, *OPC? or *WAI
- Poll the operation complete state periodically (by means of a timer) using the sequence: ‹short timeout›; *OPC?
- A return value (LSB) of 1 indicates that the overlapped command has finished. In case of a timeout, the operation is ongoing.
- Reset timeout to former value
- Clear the error queue with SYStem:ERRor? to remove the „-410, Query interrupted“ entries.

Using several threads in the controller application

As an alternative, provided the programming environment of the controller application supports threads, separate threads can be used for the application GUI and for controlling the instrument(s) via SCPI. A thread waiting for a *OPC? thus will not block the GUI or the communication with other instruments.

1.6 Status Reporting System

The status reporting system stores all information on the current operating state of the instrument, and on errors which have occurred. This information is stored in the status registers and in the error queue. Both can be queried via GPIB bus or LAN interface (STATus... commands).

1.6.1 Structure of a SCPI Status Register

Each standard SCPI register consists of 2 or 3 parts (Event, Condition and Enable register). Each part has a width of 16 bits and has different functions. The individual bits are independent of each other, i.e. each hardware status is assigned a bit number which is valid for all 2 or 3 parts. Bit 15 (the most significant bit) is set to zero for all parts. Thus the contents of the register parts can be processed by the controller as positive integers.

NOTICE

Depending on the value of the read register conclusions on the current status of the device can be drawn. For example, when the unit operates in constant voltage, the result of the returned ISUM register is a decimal “2” which corresponds the binary value of “0000000000000010”.

Any part of a status register system can be read by query commands. A decimal value is returned and represents the bit pattern of the requested register. Each SCPI register is 16 bits wide and has various functions. The individual bits are independent, i.e. each hardware status is assigned to a bit number. Bits 11-13 are still „free“ resp. unused (always return a „0“). Certain areas of the registers are not used. The SCPI standard defines only the „basic functions“. Some devices offer an advanced functionality.

Description of the status register parts

The SCPI standard differs two different status register:

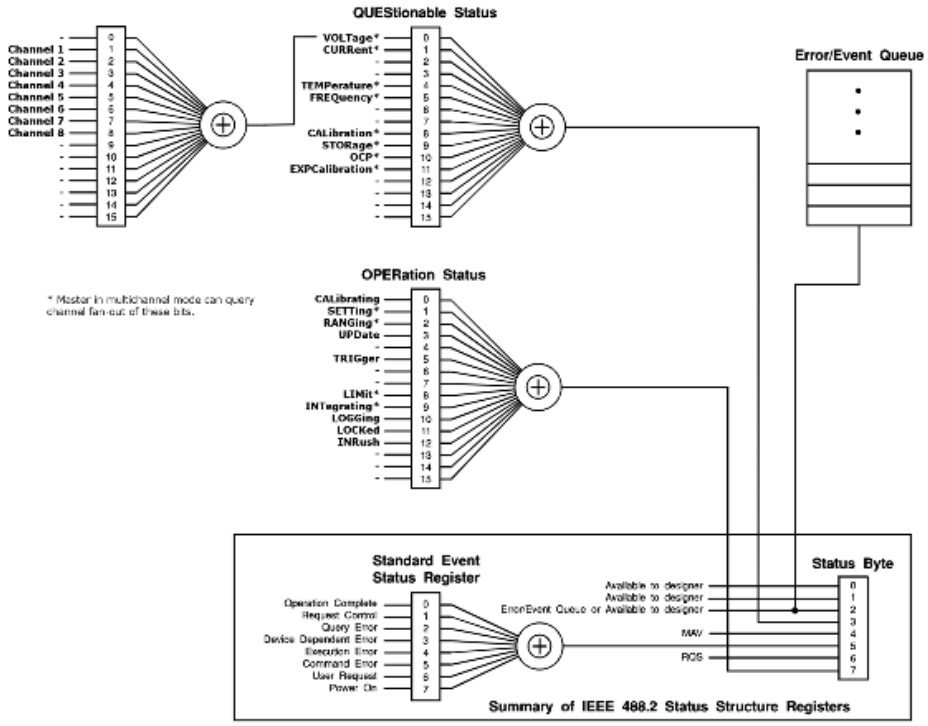

Fig. 1.1: Structure of the status register

CONDition

- The CONDition register queries the actual state of the instrument.

EVENt

- The EVENt status register is set (1) until it is queried. After reading (query) the EVENt status register is set to zero.

NOTICE

The description of registers is only used for general explanation. Due to the complexity we recommend the general accessible SCPI standard document for more detailed information.

For further descriptions of the status register, please refer to chapter 2.

Event Status Register (ESR) and Event Status Enable Register (ESE)
The ESR is defined in IEEE 488.2. It can be compared with the EVENt part of a SCPI register. The event status register can be read out using command *ESR?. The ESE corresponds to the ENABle part of a SCPI register. If a bit is set in the ESE and the associated bit in the ESR changes from 0 to 1, the ESB bit in the STB is set. The ESE register can be set using the command *ESE and read using the command *ESE?.

STATus:QUEStionable Register
This register contains information about different states which may occur. It can be read using the commands STATus:QUEStionable:CONDition? and STATus:QUEStionable[:EVENt]? .

Bit No.	Meaning
0	**VOLTage** This bit shows voltage overrange state.
1	**CURRent** This bit shows current overrange state.
2 to 3	**Not used**
4	**TEMPerature overrange** This bit is set, if an over temperature occurs.
5	**FREQuency** This bit shows frequency overrange state.
6 to 7	**Not used**
8	**CALibrating**
9	**STORage** This bit is set if the device is storing a data.
10	**OCP** This bit indicates the state of over current protection.
11	**EXPCalibration** This bit indicates that calibration is not more valid.
12 to 15	**Not used**

Table 1.8: Bits of the STATus:QUEStionable register (please refer to fig. 1.1)

STATus:OPERation Register

In the CONDition part, the register contains information which operations the instrument is being executing. In the EVENt part, it contains information which operations the instrument has executed since the last reading. It can be read using the commands STATus:OPERation:CONDition? or STATus:OPERation[:EVENt]?. The remote commands for the STATus:OPERation register are described on page 48.

Bit No.	Meaning
0	**Calibrating** (for service department only)
1	**SETTing**
2	**RANGing**
3	**UPDate**
5	**TRIGger** This bit is set while the instrument is waiting for the trigger.
6 to 7	**Not used**
8	**LIMit**
9	**INTegrating**
10	**LOGGing**
11	Instrument Locked (RWLock)
12	**INRush**
13 to 15	**Not used**

Table 1.9: Bits of the STATus:OPERation register (please refer to fig. 1.1)

Query of an instrument status

Each part of any status register can be read using queries.

There are two types of commands:

- The common commands *ESR?, *IDN?, *STB? query the higher-level registers.
- The commands of the STATus system query the SCPI registers (STATus:QUEStionable)

The returned value is always a decimal number that represents the bit pattern of the queried register. This number is evaluated by the controller program.

Decimal representation of a bit pattern (binary weights)

The STB and ESR registers contain 8 bits, the status registers 16 bits. The contents of a status register are specified and transferred as a single decimal number. To make this possible, each bit is assigned a weighted value. The decimal number is calculated as the sum of the weighted values of all bits in the register that are set to 1.

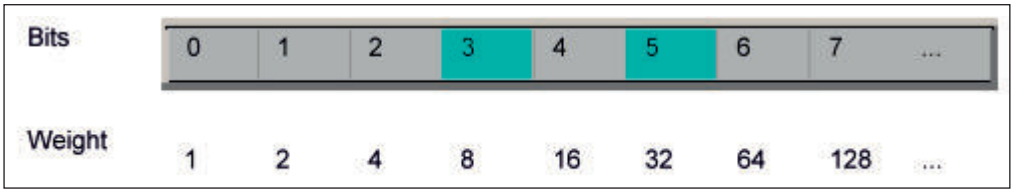

Fig. 1.2: Decimal representation of a bit pattern

Error Queue

Each error state in the instrument leads to an entry in the error queue. The entries of the error queue are detailed plain text error messages that can be looked up in the error log or queried via remote control using SYSTem:ERRor[:NEXT]?. Each call of SYSTem:ERRor[:NEXT]? provides one entry from the error queue. If no error messages are stored, the instrument responds with 0, „No error".

For further description of the error queue and the device error codes, please refer to chapter 2.

2 Command Reference

This chapter provides the description of all remote commands available for the R&S HMC8015 series. The commands are sorted according to the menu structure of the instrument. A list of commands in alphabetical order ist given in the „List of Commands“ at the end of this documentation.

2.1 Common Commands

Common commands are described in the IEEE 488.2 (IEC 625-2) standard. These commands have the same effect and are employed in the same way on different devices.The headers of these commands consist of „*“ followed by three letters. Many common commands are related to the Status Reporting System.

Available common commands:

***CLS**
CLear Status

Sets the status byte (STB), the standard event register (ESR) and the EVENt part of the QUEStionable to zero. The command does not alter the mask and transition parts of the registers. It clears the output buffer.

Usage: Setting only

***ESE <Value>**
Event Status Enable

Sets the event status enable register to the specified value. The query *ESE? returns the contents of the event status enable register in decimal form.

Parameters:
<Value> Range: 0 to 255

***ESR?**
Event Status Read

Returns the contents of the event status register in decimal form and subsequently sets the register to zero.

Return values:
<Contents> Range: 0 to 255

Usage: Query only

***IDN?**
IDeNtification

Returns the instrument identification string.

Return values:
<ID> ROHDE&SCHWARZ,‹device type›,‹serial number›,‹hardware›,‹firmwareversion›

Example: Rohde&Schwarz,HMC8015,000000000,HW42000000,SW01.000

Usage: Query only

***OPC**
OPeration Complete

Sets bit 0 in the event status register when all preceding commands have been executed. This bit can be used to initiate a service request. The query *OPC? writes a „1“ into the output buffer as soon as all preceding commands have been executed. This is used for command synchronization.

NOTICE

The R&S®HMC8015 does not support parallel processing of remote commands. If the query *OPC? returns a „1“, the device is able to process new commands.

***RST**
ReSeT

Sets the instrument to a defined default status.

Usage: Setting only

NOTICE

We recommend to start a program by *RST in order to set the instrument to a defined status prior to starting a program.

***SRE <Contents>**
Service Request Enable

Sets the service request enable register to the indicated value. This command determines under which conditions a service request is triggered. The query *SRE? returns a decimal value of the service request enable register which corresponds to the binary-weighted sum of all bits.

Parameters:
<Contents> Contents of the service request enable register in decimal form.
Bit 6 (MSS mask bit) is always 0.

Range: 0 to 255

NOTICE

The SRE is an enable register. Consequently, there are no denotations about the bits. This register conduce for the „OR" combination of the bits in the status byte.

***STB?**
STatus Byte query

Returns the contents of the status byte in decimal form.

Usage: Query only

***TST?**
self TeST query

Triggers selftests of the instrument and returns an error code in decimal form. „0" indicates no errors occured.

Usage: Query only

***WAI**
WAIt to continue

Prevents servicing of the subsequent commands until all preceding commands have been executed.

Usage: Event

2.2 System related commands

SYSTem:BEEPer:STATe <Mode>
Activates or deactivates the front panel control beeper.

Parameters:
<Mode> ON | OFF

ON: The front panel control beeper will be activated.
OFF: The front panel control beeper will be disabled.

*RST: ON

SYSTem:BEEPer:STATe?
Queries the state of the front panel control beeper. Returns “0” for deactivated (OFF) and “1” for activated (ON) control beeper.

Return values: 0 | 1

0: OFF - Control beeper is deactivated.
1: ON - Control beeper is activated.

*RST: 1

Usage: Query only

SYSTem:BEEPer[:IMMediate]
The instrument returns a single beep immediately.

Usage: Setting only

SYSTem:NAME <String>
Defines an instrument name.

Parameters:
<Name> String with max. 20 characters

SYSTem:NAME?
Queries the instrument name.

Example: „TEST UNIT“

SYSTem:DATE <Year>,<Month>,<Day>
Specifies the internal instrument date.

Parameters:
<Year> Default unit: a

<Month> Range: 1 to 12

<Day> Range: 1 to 31
Default unit: d

Usage: SCPI confirmed

SYSTem:DATE?
Queries the internal instrument date.

Example: 2015,1,1

SYSTem:TIME <Hour>,<Minute>,<Second>
Specifies the internal time for the instrument.

Parameters:

<Hour>	Range: 0 to 23 Default unit: h
<Minute>	Range: 0 to 59 Default unit: min
<Second>	Range: 0 to 59 Default unit: s
Usage:	SCPI confirmed

SYSTem:TIME?
Queries the internal instrument time which is displayed via status bar.

Example: 3,8,41

SYSTem:ERRor[:NEXT]?
Queries an error and removes it from the queue. Positive error numbers are instrument-dependent. Negative error numbers are reserved by the SCPI standard. If the queue is empty the response is 0, "No error".

Return values:	**0**,	"No error"
	-100,	"Command error"
	-102,	"Syntax error"
	-104,	„Data type error"
	-350,	"Queue overflow"
Usage:	Query only	

SYSTem:ERRor:ALL?
Queries the error/event queue for all unread items and removes them from the queue.
The response is a comma separated list of error number and a short description of the error in FIFO order. Positive error numbers are instrument-dependent. Negative error numbers are reserved by the SCPI standard.

Return values:

<Error>	List of: Error/event_number,"Error/event_description>[;Devicedependent info]" If the queue is empty, the response is 0,"No error"
Usage:	Query only SCPI confirmed

SYSTem:ELISt?
Queries the error/event queue for all unread items and removes them from the queue. Please refer to command SYSTem:ERRor:ALL?.

SYSTem:LOCal
Sets the system to front panel control. The front panel control is unlocked. If the front panel control was locked with the SCPI command SYSTem:RWLock, the message box of the locked front panel on the HMC display will be disappeared.

Usage: Setting only

SYSTem:REMote
Sets the system to remote state. The front panel control is locked. By pushing the softkey button UNLOCK KEYS the front panel control will be activated.

Usage: Setting only

SYSTem:RWLock
Sets the system to remote state. The front panel control is locked and a message box is shown on the HMC display. You are only able to unlock the front panel control via SCPI command SYSTem:LOCal.

Usage: Setting only

SYSTem:VERSion?
Returns the firmware version and the version of the SCPI (= Standard Commands for Programmable Instruments) standard.

Usage: Query only

SYSTem:TREE?
Returns a list of implemented remote commands.

Usage: Query only

SYSTem:DEVice?
Queries the instrument type.

Example: HMC8015

Usage: Query only

SYSTem:SHUTdown
Turns off the instrument (standby mode). The device can't be enabled remotely.

Usage: Setting only

SYSTem:SOFTware?
Queries the instrument firmware version.

Example: 00.026

Usage: Query only

SYSTem:HARDware?
Queries the instrument hardware version.

Example: #H40000000

Usage: Query only

SYStem:SNUMber?
Queries the instrument serial number which is stated on the instrument back panel.

Example: 012345678

Usage: Query only

2.3 Display commands

DISPlay:TEXT:CLEar
Clears the text message box on the front display.

Usage: Setting only

DISPlay:TEXT[:DATA] „<String>"
Displays a text message box on the front display.

Example: DISP:TEXT „HMC8015 TEST"

2.4 View Configuration Commands

2.4.1 Numeric

VIEW:NUMeric[:SHOW] {<Index> | MIN | MAX}
Selects the configuration page (1 to 4) in numeric mode.

Parameters:
Index: 1 to 4
MIN: 1
MAX: 4

VIEW:NUMeric[:SHOW]? [MIN | MAX]
Queries the activated configuration page.

Return values:
1 to 4
MIN: 1
MAX: 4

VIEW:NUMeric:PAGE<n>:CELL<m>:FUNCtion <Name>
Selects the measurement function for the appropriate configuration page (1 to 4) and configuration cell (1 to 6 resp. 1 to 10) in numeric mode.

Parameters:

<n> 1...4

<m> Depending on the selected page size (1...6 resp. 1...10)

<Name>

P	Active power P (Watt)
S	Apparent power S (VA)
Q	Reactive power Q (var)
LAMBda	Power factor λ (PF)
PHI	Phase difference Φ (°)
FU	Voltage frequency f_U (V)
FI	Current frequency f_I (A)
URMS	True rms voltage U_{RMS} (V)
UAVG	Voltage average (V)
IRMS	True rms current I_{RMS} (A)
IAVG	Current average (A)
UTHD	Total harmonic distortion of voltage U_{THD} (THD %)
ITHD	Total harmonic distortion of current I_{THD} (THD %)
FPLL	PLL source frequency f_{PLL} (Hz)

TIME	Integration time
WH	Watt hour (Wh)
WHP	Positive watt hour (Wh)
WHM	Negative watt hour (Wh)
AH	Ampere hour (Ah)
AHP	Positive ampere hour (Ah)
AHM	Negative ampere hour (Ah)
URANge	Voltage range
IRANge	Current range
EMPTy	Empty cell

VIEW:NUMeric:PAGE<n>:CELL<m>:FUNCtion?
Queries the measurement function of the appropriate configuration page (1 to 4) and configuration cell (1 to 6 resp. 1 to 10) in numeric mode.

Return values:

P	Active power P (Watt)
S	Apparent power S (VA)
Q	Reactive power Q (var)
LAMBda	Power factor λ (PF)
PHI	Phase difference Φ (°)
FU	Voltage frequency f_U (V)
FI	Current frequency f_I (A)
URMS	True rms voltage U_{RMS} (V)
UAVG	Voltage average (V)
IRMS	True rms current I_{RMS} (A)
IAVG	Current average (A)
UTHD	Total harmonic distortion of voltage U_{THD} (THD %)
ITHD	Total harmonic distortion of current I_{THD} (THD %)
FPLL	PLL source frequency f_{PLL} (Hz)
TIME	Integration time
WH	Watt hour (Wh)
WHP	Positive watt hour (Wh)
WHM	Negative watt hour (Wh)
AH	Ampere hour (Ah)
AHP	Positive ampere hour (Ah)
AHM	Negative ampere hour (Ah)
URANge	Voltage range
IRANge	Current range
EMPTy	Empty cell

VIEW:NUMeric:PAGE<n>:SIZE {6 |10}
Selects the displayed measurement cells for the selected configuration page in numeric mode.

Parameters:

<n>	1...4	
<Size>	**6**:	6 measurement cells will be displayed on the screen.
	10:	10 measurement cells will be displayed on the screen.

VIEW:NUMeric:PAGE<n>:SIZE?
Queries the displayed measurement cells for the selected configuration page in numeric mode.

Return values: **6**: 6 measurement cells are displayed on the screen.
10: 10 measurement cells are displayed on the screen.

2.5 Measurement Commands

CHANnel<n>:MEASurement:DATA? [<m>]
Queries the actual measurement value of all activated measurements cells or of a selected measurement cell.

Parameters:

<n> 1 (the numeric suffix is irrelevant)

<m> Selects the measurement cell depending on the selected page size (1...6 resp. 1...10)

Example: CHAN:MEAS:DATA?
Response: 0.0114444,5.12711E-05,-2.34079E-07,NAN,5,0.005,NAN

Usage: Query only

NOTICE

If „NAN“ is displayed (measurement format = ASCII), either the measurement cell was empty (EMPTY) or the measured value could not be displayed due to the chosen settings.

CHANnel<n>:MEASurement:FUNCtions <Function>{,<Function>}
Defines the measurement function list for the CHAN:MEAS:DATA? command. The measurement function list is independent fromt the displayed measurement cells.

Parameters:
<n> 1 (the numeric suffix is irrelevant)

Example: CHAN:MEAS:FUNC P,S,Q,LAMB,PHI

CHANnel<n>:MEASurement:FUNCtions? [<m>]
Queries the measurement function list which was defined with CHAN:MEAS:FUNC command. If parameter <m> is defined, the query only returns the measurement function of parameter <m>.

Return values: Comma-separated measurement function list or measurement function of parameter <m>

Example: CHAN:MEAS:FUNC P,S,Q,LAMB,PHI
CHAN:MEAS:FUNC? 3
Response: Q

CHANnel<n>:MEASurement:FUNCtions:COUNt? [MAX]
Queries the number of measurement functions which are defined with CHAN:MEAS:FUNC command.

Return values: Numeric value
MAX: 250

Example: CHAN:MEAS:FUNC P,S,Q,LAMB,PHI
CHAN:MEAS:FUNC:COUN?
Response: 5

Usage: Query only

CHANnel<n>:MEASurement:FORMat {BINary | ASCii}
Defines the measurement value format.

Parameters:
<n> 1 (the numeric suffix is irrelevant)

BINary: Binary format (4 bytes IEEE float)
ASCii: ASCII format

CHANnel<n>:MEASurement:FORMat?
Queries the measurement value format.

Return values: **BIN**: Binary format (4 bytes IEEE float)
ASC: ASCII format

CHANnel<n>:NAME <String>
Defines the channel name.

Parameters:

<n>	1 (the numeric suffix is irrelevant)
<String>	Up to 8 characters.

CHANnel<n>:NAME?
Queries the defined channel name.

Parameters:

<n>	1 (the numeric suffix is irrelevant)

Return values: String with 8 characters max.

2.6 Acquisition commands

CHANnel<n>[:ACQuisition]:VOLTage:RANGe {<Rms_Value> | MIN | MAX}
Defines the voltage range. If the auto measurement voltage range is activated, the auto measurement voltage range will be deactivated automatically.

Parameters:

<n>	1 (the numeric suffix is irrelevant)
<Rms_Value>	Voltage value in V (5, 15, 30, 60, 150, 300, 600) MIN: 5.00E+00 MAX: 6.000E+02

CHANnel<n>[:ACQuisition]:VOLTage:RANGe? [MIN | MAX]
Queries the voltage range,

Return values: 5.00E+00 to 6.000E+02

MIN: 5.00E+00
MAX: 6.000E+02

CHANnel<n>[:ACQuisition]:VOLTage:RANGe:AUTO {ON | OFF | 1 | 0}
Activates / Deactivates the auto measurement voltage range.

Parameters:
<n> 1 (the numeric suffix is irrelevant)

ON | 1: The auto measurement voltage range will be activated.
OFF | 0: The auto measurement voltage range will be deactivated.

CHANnel<n>[:ACQuisition]:VOLTage:RANGe:AUTO?
Queries the state of the auto measurement voltage range.

Parameters:
<n> 1 (the numeric suffix is irrelevant)

Return values:
1: The auto measurement voltage range is activated.
0: The auto measurement voltage range is deactivated.

CHANnel<n>[:ACQuisition]:VOLTage:CFACtor {3 | 6}
Defines the voltage measurement crest factor.

Parameters:
<n> 1 (the numeric suffix is irrelevant)

3: Crest factor 3
6: Crest factor 6

CHANnel<n>[:ACQuisition]:VOLTage:CFACtor?
Queries the voltage measurement crest factor.

Parameters:
<n> 1 (the numeric suffix is irrelevant)

Return values: Numeric value 3 or 6

CHANnel<n>[:ACQuisition]:VOLTage:INVert {ON | OFF | 1 | 0}
Activates / Deactivates the voltage inversion.

Parameters:
<n> 1 (the numeric suffix is irrelevant)

ON | 1: The voltage inversion will be activated.
OFF | 0: The voltage inversion will be deactivated.

CHANnel<n>[:ACQuisition]:VOLTage:INVert?
Queries the state of the voltage inversion.

Parameters:
<n> 1 (the numeric suffix is irrelevant)

Return values: **1**: The voltage inversion is activated.
0: The voltage inversion is deactivated.

CHANnel<n>[:ACQuisition]:CURRent:RANGe {<Rms_Value> | MIN | MAX}
Defines the current range. If the auto measurement current range is activated, the auto measurement current range will be deactivated automatically.

Parameters:
<n> 1 (the numeric suffix is irrelevant)

<Rms_Value> Current value in mA resp. A
(0.005, 0.01, 0.02, 0.05, 0.1, 0.2, 0.5, 1, 2, 5, 10, 20)

MIN: 5.00E-03
MAX: 2.000E+01

CHANnel<n>[:ACQuisition]:CURRent:RANGe? [MIN | MAX]
Queries the current range,

Return values: 5.00E-03 to 2.000E+01

MIN: 5.00E-03
MAX: 2.000E+01

CHANnel<n>[:ACQuisition]:CURRent:RANGe:AUTO {ON | OFF | 1 | 0}
Activates / Deactivates the auto measurement current range.

Parameters:
<n> 1 (the numeric suffix is irrelevant)

ON | 1: The auto measurement current range will be activated.
OFF | 0: The auto measurement current range will be deactivated.

CHANnel<n>[:ACQuisition]:CURRent:RANGe:AUTO?
Queries the state of the auto measurement current range.

Parameters:
<n> 1 (the numeric suffix is irrelevant)

Return values: **1**: The auto measurement current range is activated.
0: The auto measurement current range is deactivated.

CHANnel<n>[:ACQuisition]:CURRent:CFACtor {3 | 6}
Defines the current measurement crest factor.

Parameters:
<n> 1 (the numeric suffix is irrelevant)

3: Crest factor 3
6: Crest factor 6

CHANnel<n>[:ACQuisition]:CURRent:CFACtor?
Queries the current measurement crest factor.

Parameters:
<n> 1 (the numeric suffix is irrelevant)

Return values: Numeric value 3 or 6

CHANnel<n>[:ACQuisition]:CURRent:INVert {ON | OFF | 1 | 0}
Activates / Deactivates the current inversion.

Parameters:
<n> 1 (the numeric suffix is irrelevant)

ON | 1: The current inversion will be activated.
OFF | 0: The current inversion will be deactivated.

CHANnel<n>[:ACQuisition]:CURRent:INVert?
Queries the state of the current inversion.

Parameters:
<n> 1 (the numeric suffix is irrelevant)

Return values: **1**: The current inversion is activated.
0: The current inversion is deactivated.

CHANnel<n>[:ACQuisition]:CURRent:PROTection?
Queries the overcurrent protection state.

Parameters:
<n> 1 (the numeric suffix is irrelevant)

Return values:
1: The overcurrent protection has tripped.
0: The overcurrent protection has not tripped.

Usage: Query only

CHANnel<n>[:ACQuisition]:CURRent:PROTection:RESet
Resets the overcurrent protection state.

Usage: Event

CHANnel<n>[:ACQuisition]:MODE {AC | DC | AUTO}
Defines the acquisition mode.

Parameters:
<n> 1 (the numeric suffix is irrelevant)

AC: Average value of a voltage / current period (synchronization on period duration).
DC: Measurement of pure DC loads.
AUTO: The acquisition mode will be set automatically.

CHANnel<n>[:ACQuisition]:MODE?
Queries the type of acquisition mode.

Parameters:
<n> 1 (the numeric suffix is irrelevant)

Return values: AC | DC | AUTO

CHANnel<n>[:ACQuisition]:MODE[:AC]:PLL {VOLTage | CURRent}
Defines the reference value of the frequency (Phase Locked Loop Source).

Parameters:
<n> 1 (the numeric suffix is irrelevant)

VOLTage: Reference value voltage.
CURRent: Reference values current.

CHANnel<n>[:ACQuisition]:MODE[:AC]:PLL?
Queries the type of reference value of the frequency.

Parameters:
<n> 1 (the numeric suffix is irrelevant)

Return values: **VOLT**: Reference value voltage.
CURR: Reference values current.

CHANnel<n>[:ACQuisition]:MODE[:FILTer]:FREQuency {ON | OFF | 1 | 0}
Activates / Deactivates the frequency filter functionality.

Parameters:
<n> 1 (the numeric suffix is irrelevant)

ON | 1: The frequency filter function will be activated.
OFF | 0: The frequency filter function will be deactivated.

CHANnel<n>[:ACQuisition]:MODE[:FILTer]:FREQuency?
Queries the state of the frequency filter functionality.

Parameters:
<n> 1 (the numeric suffix is irrelevant)

Return values: **1**: The frequency filter function is activated.
0: The frequency filter function is deactivated.

CHANnel<n>[:ACQuisition]:MODE[:FILTer]:ANALog {ON | OFF | 1 | 0}
Activates / Deactivates the low-pass filter functionality (BWL).

Parameters:
<n> 1 (the numeric suffix is irrelevant)

ON | 1: The low-pass filter function will be activated.
OFF | 0: The low-pass filter function will be deactivated.

CHANnel<n>[:ACQuisition]:MODE[:FILTer]:ANALog?
Queries the state of the low-pass filter functionality.

Parameters:
<n> 1 (the numeric suffix is irrelevant)

Return values: **1**: The low-pass filter function is activated.
0: The low-pass filter function is deactivated.

CHANnel<n>[:ACQuisition]:MODE[:FILTer]:DIGital {ON | OFF | 1 | 0}
Activates / Deactivates the digital filter functionality.

Parameters:
<n> 1 (the numeric suffix is irrelevant)

ON | 1: The digital filter function will be activated (AUTO).
OFF | 0: The digital filter function will be deactivated.

CHANnel<n>[:ACQuisition]:MODE[:FILTer]:DIGital?
Queries the state of the digital filter functionality.

Parameters:
<n> 1 (the numeric suffix is irrelevant)

Return values:
1: The digital filter function is activated.
0: The digital filter function is deactivated.

2.7 Integrator commands

INTegrator[:STATe] {ON | OFF | 1 | 0}
Activates / Deactivates the integrator functionality.

Parameters:
ON | 1: The integrator function will be activated.
OFF | 0: The integrator function will be deactivated.

INTegrator[:STATe]?
Queries the state of the integrator functionality.

Return values:
1: The integrator function is activated.
0: The integrator function is deactivated.

INTegrator:MODE {MANual | DURation | SPAN}
Defines the integrator function mode.

Parameters:
MANual: Manual integration mode (default setting).
DURation: Integration timer value.
SPAN: Integration span mode.

INTegrator:MODE?
Queries the type of the integration mode.

Return values: MAN | DUR | SPAN

INTegrator:DURation {<Seconds> | MIN | MAX}
Defines the integration duration for the span and duration mode.

Parameters: Numeric value in seconds (0s to 96h).

MIN: 0
MAX: 349199

INTegrator:DURation? [MIN | MAX]
Queries the integration duration.

Return values: Numeric value in seconds.

MIN: 0
MAX: 349199

INTegrator:STIMe <Year>,<Month>,<Day>,<Hour>,<Minute>,<Second>
Defines the integration start time.

Parameters:
<Month> Range: 1 to 12

<Day> Range: 1 to 365

<Hour> Range: 0 to 23
Default unit: h

<Minute> Range: 0 to 59
Default unit: min

<Second> Range: 0 to 59
Default unit: s

Example: 2015,1,1,0,0,0

Usage: SCPI confirmed

INTegrator:STIMe?
Queries the integration start time.

Return values: Comma-separated list

INTegrator:STARt
Starts the manual integrator mode, if the integration funtionality is activated with the INTegrator[:STATe] ON command.

Usage: Event

INTegrator:RESet
Resets the manual inetgration time, which is displayed on the display (status line).

Usage: Event

INTegrator:STOP
Starts the manual integrator mode, if the integration funtionality is activated and started with the INTegrator[:STATe] ON and INTegrator:STARt command.

Usage: Event

2.7 Data and File Management

LOG[:STATe] {ON | OFF | 0 | 1}
Turns the data logging function on or off.

Parameters:
ON / 1: Data logging function will be activated.
OFF / 0: Data logging function will be disabled.

*RST: OFF | 0

LOG[:STATe]?
Queries the state of the data logging function.

Return values:
1: ON - Data logging function is activated.
0: OFF - Data logging function is disabled.

LOG:FNAMe {<"File_Name">},[{INT | EXT}]
Defines the file name and storage location for the logging function.

Parameters:
<File_Name> e.g. "Test01.CSV",INT

INT: Internal memory
EXT: USB stick

LOG:FNAMe?
Returns the file name and storage location for the logging function.

Return values: e.g. "/INT/DATA/Test01.CSV"

INT: Internal memory
EXT: USB stick

LOG:MODE {UNLimited | COUNt | DURation | SPAN}
Selects the data logging mode.

Parameters:
UNLimited: Infinite data capture.
COUNt: Number of measurement values to be captured.
DURation: Duration of the measurement values capture.
SPAN: Duration and start time of the measurement values capture.

*RST: UNL

LOG:MODE?
Queries the data logging mode.

Return values: UNL | COUN | TIME

LOG:DURation {<Seconds> | MIN | MAX}
Defines the logging duration for the span and duration mode.

Parameters: Numeric value in seconds (0s to 96h).

MIN: 0
MAX: 349199

LOG:DURation? [MIN | MAX]
Queries the duration of the measurement values capture.

Return values: e.g. 5.00000E+04

LOG:COUNt {<Value> | MIN | MAX}
Sets the number of measurement values to be captured.

Parameters:
<No of samples> 1...100000000

MIN: 1
MAX: 100000000

LOG:COUNt? [MIN | MAX]
Queries the number of measurement values to be captured.

Return values: e.g. 1.00E+01

MIN: 1
MAX: 100000000

LOG:INTerval {<Interval in seconds> | MIN | MAX}
Selects a logging measurement interval. The measurement interval describes the time between the recorded measurements.

Parameters:
<Interval in seconds> Numeric value in seconds.

MIN: 0.1s
MAX: 600s

[DATA:]LOG:INTerval? [MIN | MAX]
Queries the selected logging measurement interval.

Return values: e.g. 0.1

LOG:PAGE <Page_Index>
Defines the configuration page which will be logged.

Parameters:
<Page_Index> 1...4

LOG:PAGE?
Queries the configuration page, which will be logged.

Return values: 1...4

LOG:STIMe <Year>,<Month>,<Day>,<Hour>,<Minute>,<Second>
Defines the logging start time.

Parameters:

<n>	1 (the numeric suffix is irrelevant)
<Month>	Range: 1 to 12
<Day>	Range: 1 to 365
<Hour>	Range: 0 to 23 Default unit: h
<Minute>	Range: 0 to 59 Default unit: min
<Second>	Range: 0 to 59 Default unit: s

Example: 2015,1,1,0,0,0

Usage: SCPI confirmed

LOG:STIMe?
Queries the logging start time.

Return values: Comma-separated list

DATA:DATA? {<"Filename">},[{INT | EXT}]
Returns the logging file data values of the selected storage location and file name. If no logging file is found, the message „No Logging Files found" is displayed. If no storage location is selected, the instrument queries the internal memory. Please notice that the logging function has to be activated, if you want to use the manual trigger mode (trigger via TRIG button). Without activating the logging function in manual trigger mode, the instrument is not able to save a logging file internally or on the USB stick.

Return values: e.g. "LOG0029.CSV"

INT: Internal memory
EXT: USB stick

Example: External logging file (USB stick), count = 3
DATA:DATA? „LOG0001.CSV“,EXT

URMS[V];IRMS[A];P[W];FU[Hz];EMPTY;EMPTY;S[VA];Q[var];LAMBDA[]
;UTHD[%];Timestamp
231.27E+00;45.0E-03;6.63E+00;50.0E+00;10.38E+00;7.98E+00
231.38E+00;45.0E-03;6.64E+00;50.0E+00;+00;7.98E+00;639E-03
231.35E+00;45.0E-03;6.63E+00;50.0E+00;10.38E+00;7.98E+00

DATA:DELete {<“Filename“>},[{INT | EXT}]
Deletes the logging file data values of the selected storage location and file name. If no storage location is selected, the instrument uses the internal memory.

Parameters:
<File_Name> e.g. “LOG0001.CSV“

INT: Internal memory
EXT: USB stick

Example: DATA:DEL „LOG0001.CSV“,EXT

DATA:POINts? {<“Filename“>},[{INT | EXT}]
Queries the number of log file values of the selected storage location and file name. If no storage location is selected, the instrument queries the internal memory.

Return values: Depending on the log file.

Example: External logging file (USB stick), count = 5
DATA:POIN? „LOG0001.CSV“,EXT

Query: 5

DATA:LIST? [{INT | EXT}]
Queries all saved logging files of the selected storage location. If no storage location is selected, the instrument queries the internal memory. If you store the logging file on the USB stick, the query returns all files.

Return values:
INT: Internal memory
EXT: USB stick

Example: DATA:LIST? EXT

Query: „LOG0001.CSV“, “LOG0002.CSV“,“LOG0003.CSV“

HCOPy:DATA?
Returns the actual display content (screenshot). The DATA? query responses the screenshot data in binary format.

Usage: Query only

HCOPy:FORMat {BMP | PNG}
Selects the data format of the screenshot.

Parameters: BMP | PNG

BMP: Windows Bitmap Format
PNG: Portable Network Graphic

*RST: BMP

HCOPy:FORMat?
Returns the current setting of the screenshot format.

Return values: BMP | PNG

HCOPy:SIZE:X?
Returns the horizontal expansion of the screenshots.

Usage: Query only

HCOPy:SIZE:Y?
Returns the vertical expansion of the screenshots.

Usage: Query only

***SAV {0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9}**
Stores the current instrument state in the specified storage location. Any state previously stored in the same location is overwritten (no error is generated).

***RCL {0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9}**
Recalls the current instrument state of the specified storage location.

2.8 Status Reporting

2.8.1 STATus:OPERation Register

The commands of the STATus:OPERation subsystem control the status reporting structures of the STATus:OPERation register:
See also:

- chapter 1.6.1, „Structure of a SCPI Status Register“, on page 15
- „STATus:OPERation Register“, on page 17
- Diagram on page 16

The following commands are available:

STATus:OPERation:CONDition?
Returns the of the CONDition part of the operational status register.

Return values: Condition bits in decimal representation.
Range: 1 to 65535

Usage: Query only

STATus:OPERation:ENABle <Enable_value>

Parameters:
<Enable_Value> Range: 1 to 65535

STATus:OPERation:ENABle?
Enables the bits in the enable register for the Standard Operation Register group.

STATus:OPERation[:EVENt]?

Return values: Range: 1 to 65535

Usage: Query only

STATus:OPERation:PTRansition <Value>
STATus:OPERation:PTRansition?

Parameters:
<Value> Range: 1 to 65535

STATus:OPERation:NTRansition <Value>
STATus:OPERation:NTRansition?

Parameters:
<Value> Range: 1 to 65535

2.8.2 STATus:QUEStionable Registers

The commands of the STATus:QUEStionable subsystem control the status reporting structures of the STATus:QUEStionable registers:

See also:
- chapter 1.6.1, „Structure of a SCPI Status Register“, on page 15
- „STATus:OPERation Register“, on page 17
- Diagram on page 16

The following commands are available:

STATus:QUEStionable[:EVENt]?
Queries the contents of the EVENt part of the status register to check whether an event has occurred since the last reading. Reading an EVENt register deletes its contents.

Return values: Event bits in decimal representation
Range: 1 to 65535

Usage: Query only

STATus:QUEStionable:CONDition?
Returns the contents of the CONDition part of the status register to check for questionable instrument or measurement states. Reading the CONDition registers does not delete the contents.

Return values:
<Condition> Condition bits in decimal representation
Range: 1 to 65535

Usage: Query only

STATus:QUEStionable:ENABle <Enable_value>
Sets the enable mask that allows true conditions in the EVENt part to be reported in the summary bit. If a bit in the enable part is set to 1 and its associated event bit transitions to true, a positive transition occurs in the summary bit and is reported to the next higher level.

Parameters:
<Enable_value> Bit mask in decimal representation
Range: 1 to 65535

STATus:QUEStionable:ENABle?
Queries the enable register and returns a decimal value which corresponds to the binary-weighted sum.

STATus:QUEStionable:PTRansition <Value>
STATus:QUEStionable:PTRansition?
Sets / Queries the positive transition filter. If a bit is set, a 0 to 1 transition in the corresponding bit of the condition register causes a 1 to be written in the corresponding bit of the event register.

Parameters:
<Value> Bit mask in decimal representation
Range:1 to 65535

STATus:QUEStionable:NTRansition <Value>
STATus:QUEStionable:NTRansition?
Sets / Queries the negative transition filter. If a bit is set, a 1 to 0 transition in the corresponding bit of the condition register causes a 1 to be written in the corresponding bit of the event register.

Parameters:
<Value> Bit mask in decimal representation
Range: 1 to 65535

3 SCPI Commands

in alphabetic order

Mühldorfstr. 15, 81671 München, Germany
Phone: +49 89 41 29 - 0
Fax: +49 89 41 29 12 164
E-mail: info@rohde-schwarz.com
Internet: www.rohde-schwarz.com
Customer Support: www.customersupport.rohde-schwarz.com
Service: www.service.rohde-schwarz.com

1178.3186.02 | Version 01 | R&S®HMC8015

The following abbreviations are used throughout this manual: R&S®HMC8015 is abbreviated as R&S HMC8015.