Power supply
HMP Serie
SCPI Programmers Manual

HAMEG®
Instruments
A Rohde&Schwarz Company

ROHDE&SCHWARZ

SCPI Programmers Manual

Content

1 Basics

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

1.1 Remote Control Interfaces

For remote control, RS-232 / USB (standard interface HO720), Ethernet / USB (optional interface HO730) or GPIB (optional interface HO740) can be used. The optional interfaces replaces the RS-232 / USB HO720 interface module on the rear panel.

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.

NOTICE

The end character must be set to linefeed (LF) or carriage return + linefeed (CR-LF).

1.1.1 RS-232 Interface

If you use classic RS-232 you do not need any driver. In order to set the interface parameter at the HMP, please press the button MENU at the front panel and choose the menu item INTERFACE. Make sure the RS-232 interface is chosen. The menu item SETTINGS opens a menu where you can set and save all parameter for the RS-232 communication. Setting of the RS-232 must fit the setting of the corresponding PC COM port.

In general, there are exist two options for the RS-232 communication: with or without interface handshake. If you are working without handshake you have to integrate appropriate delays between the commands to make sure that all commands are executed correctly (approx. 50ms to 100ms). Without handshake you can have the problem that the interface buffer can overflow (e.g. missing commands). If you are working with interface handshake (set on both sides, HMP and PC) you don‘t need to integrate delays.

In the interface settings of the HMP you can set the interface handshake on/off (button MENU). If you have a look into the menu item INTERFACE you can choose the menu item HANDSHAKE. There you can find the items NONE or CTS/RTS. CTS/RTS means the activation of the interface handshake. Please note that you choose the same settings in your appropriate software.

NOTICE

If you want to use the RS-232 interface we recommend to activate the interface handshake to avoid timing problems.

1.1.2 USB Interface

If you are using USB you need to install a USB driver which is available on the HAMEG Website.

NOTICE

All descriptions regarding the USB interface are true for the HO720 interface card as well as for the optional HO730 USB part. All currently available USB driver are fully tested, functional and released for Windows XP™, Windows Vista™ or Windows 7™ both as 32Bit or 64Bit versions.

The HMP USB interface has to be chosen in the MENU and does not need any setting.

NOTICE

If the virtual COM port will be used, you have to install the virtual COM port part of the HO720 / HO730 USB driver. The virtual COM port (VCP) will be activated in the PC device explorer.

1.1.3 Ethernet (LAN) Interface

The settings of the parameter will be done after selecting the menu item Ethernet. 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 HMP series, 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

1.1.4 GPIB Interface (IEC/IEEE Bus Interface)

An optional GPIB interface is also available for the HMP series. 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 HMP series.

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 by 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 HMP 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

NOTICE

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

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.

- If the network supports dynamic TCP/IP configuration using the Dynamic Host Configuration Protocol (DHCP), and a DHCP server is available, all address information can be assigned automatically.
- Otherwise, the address must be set manually. Automatic Private IP Addressing (APIPA) is not supported.

By default, the instrument is configured to use dynamic TCP/IP 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

Risk of network errors
Connection errors can affect the entire network. If your network does not support DHCP, or if you choose to disable dynamic TCP/IP configuration, you must assign valid address information before connecting the instrument to the LAN. Contact your network administrator to obtain a valid IP address.

Configuring LAN parameters

- Press the MENU key and choose INTERFACE.
- Press the knob, choose ETHERNET (SELECT INTERFACE) and SETTINGS.
- Define the IP and port settings of the instrument.
- Select the IP Port - the port number for SCPI socket communication.

NOTICE

By default the instrument is not set to DHCP. If the instrument is set to DHCP and cannot find a DHCP server, it takes about two minutes until the Ethernet menu is available.

Checking LAN and SCPI connection

- Check the LAN connection using ping: `ping xxx.yyy.zzz.xxx`.
- If the PC can access the instrument, enter the IP address into your PC internet browser:
  http//:xxx.yyy.zzz.xxx
  The „Instrument Home“ page appears. It provides information of 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 via PC, 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 PC 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; 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:** only act on instruments previously addressed as listeners; 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". 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.

Examples:

***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
For demonstration purposes only, assume the existence of the following commands for this section:

- MEASure:CURRent[:DC]?
- MEASure:VOLTage[:DC]?
- FUSE[:STATe] {ON | OFF | 0 | 1}
- FUSE[:STATe]?

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: MEASure:CURRent? is equivalent to MEAS:CURR?

NOTICE

Case-insensitivity

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:

FUSE[:STATe] { ON }

FUSE:STAT ON is equivalent to FUSE ON

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:** FUSE:LINK {1 | 2 | 3} FUSE:LINK 1 sets the fuse link CH1 for the selected channel FUSE:LINK 2 sets the fuse link of CH2 for the selected channel
[ ]	Mnemonics in square brackets are optional and may be inserted into the header or omitted. **Example:** FUSE[:STATe] { ON } FUSE:STAT ON is equivalent to FUSE ON
{ }	Parameters in curly brackets are optional and can be inserted once or several times, or omitted. **Example:** VOLTage[:LEVel][:IMMediate][:AMPLitude] {<voltage> I MIN I MAX I UP | DOWN } The following are valid commands: VOLT MAX • VOLT MIN VOLT 10

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 0 to 9, 11 to 32 decimal, e.g. blank). Allowed parameters are:

- Numeric values
- Special numeric values
- Boolean parameters
- Text
- Character strings
- Block data

The required parameters and the allowed value range are specified in the command description.

Numeric values
Numeric values can be entered in the following form. Values exceeding the resolution of the instrument are rounded up or down.

Example:
VOLT 10V = VOLT 10
VOLT 100mV = VOLT 0.1

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:
VOLT MAX
VOLT MAX?, Response: 32.050

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:
VOLT:PROT? 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:
OUTPut[:STATe] ON
OUTPut[:STATe]?, Response: 1

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 0 to 9, 11 to 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:
VOLTage:PROTection:MODE?, Response: measured

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

Example:
VOLT:PROT? MAX, Response: 32.500
- Truth values (Boolean values) are returned as 0 (for OFF) and 1 (for ON).

Example:
OUTPut ON
OUTPut ON?, 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

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

To prevent an overlapping execution of commands 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.

NOTICE

The HMP series does not support parallel processing of remote commands. If OPC? returns a „1“, the device is able to process new commands.

1.6 Status Reporting System

The status reporting system stores all information on the current operating state of the instrument and errors which have occurred. This information is stored in the status registers and in the error queue. Both can be queried via RS-232, USB, GPIB 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.

A STATus:QUEStionable:INSTrument:ISUMmary1 exists as often as device channels are available (e.g. HMP4030 = 3 channels = 3 status register). Accordingly, the description text of the channel information changes in Fig. 1.1 (e.g. instrument 1 = channel 1, instrument 2 = channel 2 etc.).

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".

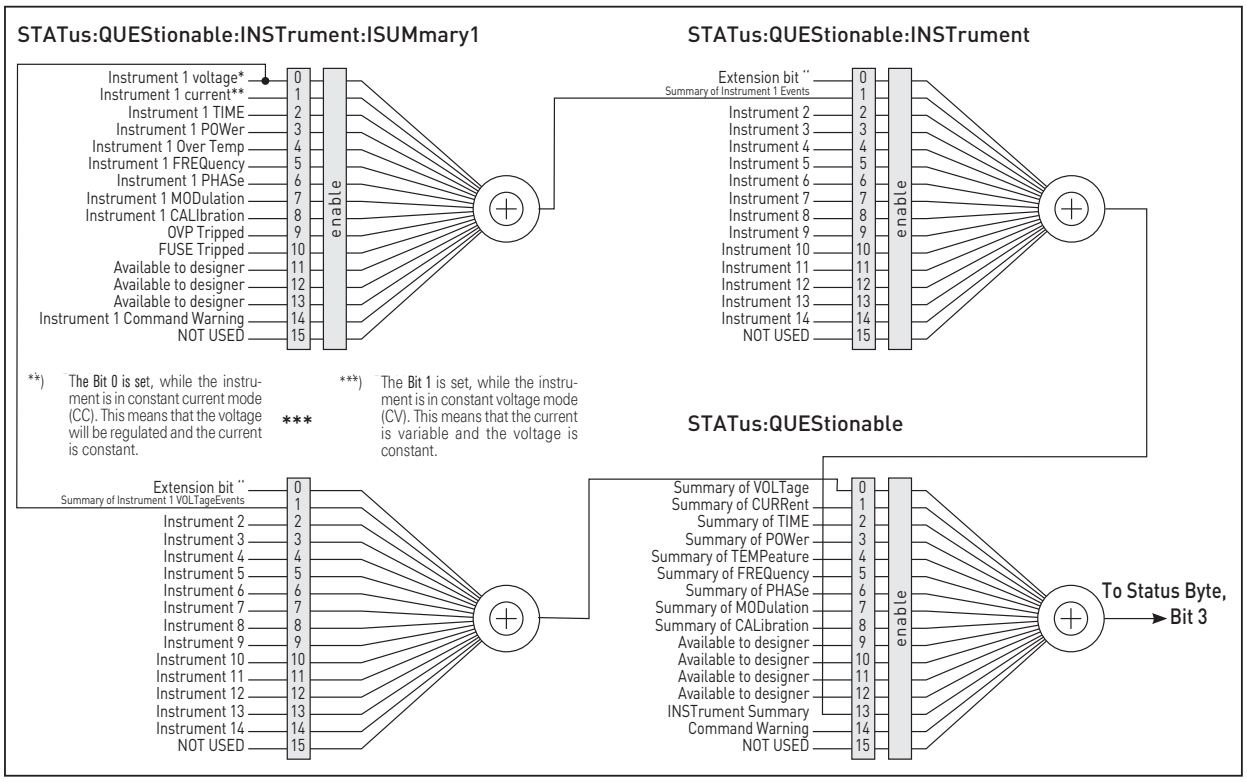


Fig. 1.1: Structure of the status register

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.

Each channel of the power supply is considered as separate „instrument" (SCPI standard definition). Therefore, e.g. the register „Status: Questionable: Instrument: Isumary" of the HMP4030 is also present three times (Isummary1-3).

Description of the status register parts (please refer to fig. 1.1)
The SCPI standard differs two different status register:

CONDition
- The CONDition register queries the actual state of the instrument. If you want to query the constant voltage or current mode, you have to use the CONDition register.

NOTICE

The CONDition register delivers a „1“ (first bit set) in constant current mode (CC) and a „2“ (second bit set) in constant voltage mode (CV).

If the correct channel is selected and the red LED of the channel button lights up (CC mode), the query of the CONDition register has to be deliver a „1“.

Example:
STAT:QUES:ISUM1:COND?

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 is set while the instrument is in constant current mode (CC). This means that the voltage will be regulated and the current is constant.
1	**Current** This bit is set while the instrument is in constant voltage mode (CV). This means that the current is variable and the voltage is constant.
2	**Not used**
3	**Not used**
4	**Temperature overrange** This bit is set if an over temperature occurs.
5-8	**Not used**
9	**OVP Tripped** This bit is set if the over voltage protection has tripped.
10	**Fuse Tripped** This bit is set if the fuse protection has tripped.
11 to 15	**Not used**

Table 1.8: Bits of the STATus:QUEStionable 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.

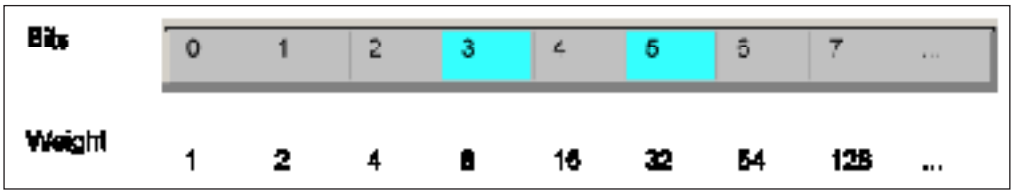


Fig. 1.7: Decimal representation of a bit pattern

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 HMC8012. 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> HAMEG,‹device type›,‹serial number›,‹firmwareversion›

Example: HAMEG,HMP4040,055310003,HW50020001/SW2.41

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 HMP series does not support parallel processing of remote commands. If OPC? returns a „1“, the device is able to process new commands.

***RST**
ReSeT

Sets the instrument to a defined default status. The default settings are indicated in the description of commands.

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

***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.

2.2 System related commands

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

Usage: Setting only

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:
<error> **0**, “No error“
-100, “Command error“
-102, “Syntax error“
-350, “Queue overflow“

Usage: Query only

SYSTem:LOCal
Sets the system to front panel control. The front panel control is unlocked.

Usage: Setting only

SYSTem:REMote
Sets the system to remote state. The front panel control is locked. By pushing the REMOTE button the front panel control will be activated. If the instrument receives a remote command it will be switched into remote control automatically (REMOTE button LED lights up).

Usage: Setting only

SYSTem:RWLock
Sets the system to remote state. The front panel control is locked and can not be unlocked via REMOTE button). You are only able to unlock the front panel control via SCPI command SYSTem:LOCal.

Usage: Setting only

SYSTem:MIX
Sets the system to remote state. The front panel and remote control are possible simultaneously (mixed mode).

Usage: Setting only

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

Usage: Query only

2.3 Configuration Commands

2.3.1 Channel selection

INSTrument[:SELect] {OUTPut1|OUTPut2|OUTPut3|OUTPut4|OUT1|OUT2|OUT3|OUT4}
Selects a channel. Each channel of the power supply is considered as separate „instrument“, which is required by the SCPI standard.

Parameters:
OUTPut1, OUT1 = channel 1 (CH1)
OUTPut2, OUT2 = channel 2 (CH2)
OUTPut3, OUT3 = channel 3 (CH3)
OUTPut4, OUT4 = channel 4 (CH4)

NOTICE

HMP2020: OUTPut3, OUTPut4, OUT3 and OUT4 are not available
HMP2030 / HMP4030: OUTPut4 and OUT4 are not available

INSTrument[:SELect]?
Queries the channel selection.

Return values:
e.g. OUTP1 (= channel CH1)

Example:
INST OUT1
INST?, Response: OUTP1

INSTrument:NSELect {1|2|3|4}
Numerical selction of a channel. Each channel of the power supply is considered as separate „instrument“, which is required by the SCPI standard.

Parameters:
1 = channel 1 (CH1)
2 = channel 2 (CH2)
3 = channel 3 (CH3)
4 = channel 4 (CH4)

NOTICE

HMP2020: :NSELect {3} and :NSELect {4} are not available
HMP2030 / HMP4030: :NSELect {4} is not available

INSTrument:NSELect?
Queries the numerical channel selection.

Return values:
e.g. 1 (= channel CH1)

Example:
INST:NSEL 1
INST:NSEL?, Response: 1

2.3.2 Voltage setting

[SOURce:]VOLTage[:LEVel][:IMMediate][:AMPLitude] {<voltage>| MIN | MAX}}
Sets the voltage value of the selected channel.

Parameters:

<voltage>	0.000V to 32.050V (adjustable in 1mV steps)
	MIN 0.000V
	MAX 32.050V

Example:
VOLT 10

[SOURce:]VOLTage[:LEVel][:IMMediate][:AMPLitude]? [MIN I MAX]
Queries the voltage value of the selected channel.

Example:
INST OUT1
VOLT 10
VOLT?, Response: 10.000

[SOURce:]VOLTage[:LEVel][:IMMediate][:AMPLitude] {UP I DOWN}
Increases resp. decreases the voltage value of the selected channel. The voltage step size will be defined with the VOLT:STEP command.

Example:
INST OUT1
VOLT:STEP 4
VOLT UP (= channel 1 will be set to 4.000V)

[SOURce:]VOLTage[:LEVel]:STEP[:INCRement) {<numeric value>| DEFault}
Defines the voltage step size for the VOLT UP (VOLT DOWN) command.

Parameters:

<numeric value>	0.000V to 32.050V
	DEF 1.000V

[SOURce:]VOLTage[:LEVel]:STEP[:INCRement)? [Default)
Queries the voltage step size.

Example:
INST OUT1
VOLT:STEP 4
VOLT:STEP?, Response: 4.000

2.3.3 Current setting

[SOURce:]CURRent[:LEVel][:IMMediate][:AMPLitude] {<current>| MIN | MAX}
Sets the current value of the selected channel.

Parameters:

<current>	Current value depending on the instrument type.	
	MIN	0.5mA (5A channel) / 1mA (10A channel)
	MAX	5.000A (HMP2020)
		10.010A (HMP2020 / HMP2030 / HMP4030 / HMP4040)

Example:
CURR 2

[SOURce:]CURRent[:LEVel][:IMMediate][:AMPLitude]? [MIN I MAX]
Queries the current value of the selected channel.

Example:
INST OUT1
CURR 5
CURR?, Response: 5.0000

[SOURce:]CURRent[:LEVel][:IMMediate][:AMPLitude] {UP I DOWN}
Increases resp. decreases the current value of the selected channel. The current step size will be defined with the CURR:STEP command.

Example:
INST OUT1
CURR:STEP 4
CURR UP (= channel 1 will be set to 4.000A)

[SOURce:]CURRent[:LEVel]:STEP[:INCRement) {<numeric value>| DEFault}
Defines the current step size for the CURR UP (CURR DOWN) command.

Parameters:

<numeric value>		0.5mA / 1mA to 10.0100A
	DEF	0.100A

[SOURce:]CURRent[:LEVel]:STEP[:INCRement)? [Default)
Queries the current step size.

Example:
INST OUT1
CURR:STEP 1
CURR:STEP?, Response: 1.0000

2.3.4 Combined setting of voltage and current

APPLy {<voltage>| DEF | MIN | MAX} [, {<current>| DEF | MIN | MAX}]
Sets the voltage and current value of the selected channel.

Parameters:

<voltage>	**MIN**	0.000V
	MAX	32.050V
	DEF	1.000V
<current>		Minimum / maximum current value depending on the instrument type.
	MIN	0.5mA (5A channel) / 1mA (10A channel)
	MAX	5.000A (HMP2020)
		10.010A (HMP2020 / HMP2030 / HMP4030 / HMP4040)
	DEF	1.000A

Example:
INST OUT1
APPLY 6,2 (= channel 1 will be set to 6V and 2A)

APPLy?
Queries the voltage and current value of the selected channel.

Return values: e.g. **HMP4040**: 32.050,5.0020

NOTICE

The execution of the APPLY command needs more time (approx. 100ms) than a single instrument setting command (e.g. INST OUT1).

2.3.5 Output setting

OUTPut:SELect {OFF | ON | 0 | 1}
Activates or deactivates the previous selected channel. If the channel is activated the channel LED lights up green in CV (constant voltage) mode or red in CC (constant current) mode.

Parameters:
ON | 1 Channel will be activated
OFF | 0 Channel will be deactivated
***RST**: OFF | 0

OUTPut[:STATe] {OFF | ON | 0 | 1}
Activates or deactivates the previous selected channel and turning on the output. The selected channel LED lights up green. If the output will be turned of with OUTP OFF only the previous selected channel will be deactivated. After sending OUTP OFF command the output button is still activated.

Parameters:
ON | 1 Channel and output will be activated
OFF | 0 Channel will be deactivated
***RST**: OFF | 0

Example:
INST OUT1
OUTP ON (= channel 1 and output will be activated; channel and output LED will light up)

OUTPut[:STATe]?
Queries the output state.

Return values:
1 ON - output is activated
0 OFF - output is deactivated

OUTPut:GENeral {OFF | ON | 0 | 1}
Turning on / off all previous selcted channels simultaneously.

Parameters:
ON | 1 Channels and output will be activated
OFF | 0 Channels will be deactivated

Example:
INST OUT1
Volt 12
Curr 0.1
OUTP:SEL ON CH1 LED lights up green

INST OUT2
Volt 12
Curr 0.2
OUTP:SEL ON CH2 LED lights up green

OUTP:GEN ON Channels will be activated simultaneously

2.3.6 Fuse setting

FUSE[:STATe] {ON | OFF | 0 | 1}
Activates or deactivates the fuse for the previous selected channel.

Parameters:
ON | 1 Fuse will be activated
OFF | 0 Fuse will be deactivated
***RST**: OFF | 0

FUSE [:STATe]?
Queries the fuse state of the previous selected channel.

Return values:
1 ON - fuse is activated
0 OFF - fuse is deactivated

Example:
INST OUT1
FUSE ON
FUSE?, Response: 1; fuse of CH1 is activated

FUSE:DELay {<delay>| MIN | MAX}
Defines a fuse delay for the previous selected channel.

Parameters:

<delay>	0ms to 250ms (adjustable in 10ms steps e.g. FUSE:DEL 10 = 10ms)
	MIN 0ms
	MAX 250ms

FUSE:DELay? [MIN I MAX]
Queries the fuse delay time for the previous selected channel.

Example:
INST OUT1
FUSE:DEL 50
FUSE:DEL?, Response: 050

FUSE:LINK {1 | 2 | 3 | 4}
Combines the channel fuses (fuse linking) for the previous selected channel. Each channel fuse can be linked together (depending on the instrument type).

Parameters:
1 = channel CH1
2 = channel CH2
3 = channel CH3
4 = channel CH4

NOTICE

HMP2020: channel 3 (CH3) and channel 4 (CH4) are not available
HMP2030 / HMP4030: channel 4 (CH4) is not available

Example:
INST OUT1
FUSE:LINK 2 (= fuse CH1 is linked with CH2)

FUSE:LINK? {1 | 2 | 3 | 4}
Queries the combined fuses of the previous selected channel. If the fuse of channel 1 is linked with fuse of channel 2, a „1“ is returned; when the fuse of channel 1 is not linked to the fuse of channel 2, it returns a „0“.

Return values: **1** = fuse is linked
0 = fuse is not linked

Example:
INST OUT1
FUSE:LINK 2
FUSE:LINK? 2, Response: 1 (= fuse of CH1 is linked with CH2)

FUSE:UNLink {1 | 2 | 3 | 4}
Unlinks the channel fuses (fuse linking) for the previous selected channel.

Parameters: **1** = channel CH1
2 = channel CH2
3 = channel CH3
4 = channel CH4

NOTICE

HMP2020: channel 3 (CH3) and channel 4 (CH4) are not available
HMP2030 / HMP4030: channel 4 (CH4) is not available

Example:
INST OUT1
FUSE:LINK 2 (= fuse CH1 is linked with CH2)
FUSE:UNL 2 (= fuse CH1 is unlinked with CH2)

FUSE:TRIPped?
Queries the fuse trip of the previous selected channel.

Return values **1** = fuse is tripped
0 = fuse is not tripped

Example:
INST OUT1
FUSE:TRIP?, Response: 0 (= fuse of CH1 has not triped)

2.3.6 OVP setting

VOLTage:PROTection[:LEVel] {<voltage> | MIN | MAX }
Sets the OVP value of the previous selected channel.

Parameters:

<voltage>	100mV up to 32.50V (adjustable in 10mV steps)
	MIN 0.100V
	MAX 32.500V

Example:
INST OUT1
VOLT:PROT 5 (= OVP channel 1 will be set to 5V)

VOLTage:PROTection[:LEVel]? [MIN | MAX]
Queries the OVP value of the previous selected channel.

Example:
INST OUT1
VOLT:PROT? MAX, Response: 32.500

VOLTage:PROTection:TRIPped?
Queries the OVP state of the previous selected channel.

Return values	**1** = OVP is tripped
	0 = OVP is not tripped

Example:
INST OUT1
VOLT:PROT:TRIP?, Response: 0 (= OVP of CH1 has not triped)

VOLTage:PROTection:CLEar
Resets the OVP state of the selected channel. If the OVP has tripped the OVP message on the display will be cleared for the selected channel.

VOLTage:PROTection:MODE {MEASured | PROTected}
Sets the OVP mode for the previous selected channel.

Parameters: MEASured | PROTected

MEASured: the OVP switches off if the measured value exceeds the threshold

PROTected: if the adjusted threshold is exceeded the output of the instrument will be not switched on; additionally the measured value is monitored (please also refer to function **MEASured**)

Example:
INST OUT1
VOLT:PROT:MODE PROT (= sets OVP protected mode for CH1)

VOLTage:PROTection:MODE?
Returns the OVP mode for the previous selected channel.

Example:
INST OUT1
VOLT:PROT:MODE PROT
VOLT:PROT:MODE?, Response: protected

2.4 Measurement Commands

MEASure[:SCALar]:CURRent [:DC]?
Returns the measured current value of the previous selected channel.

Return values e.g. 1.0000

MEASure[:SCALar][:VOLTage] [:DC]?
Returns the measured voltage value of the previous selected channel.

Return values e.g. 1.000

2.4 Arbitrary Commands

ARBitrary:DATA <voltage1,current1,time1,voltage2,current2,time2,...>
Defines the arbitrary points for the previous selected channel. Max. 128 arbitrary points can be defined. The dwell time between 2 arbitrary points is specified from10ms to 60s.

Parameters: e.g. ARB:DATA 1,1,1,2,2,1,3,3,1

ARBitrary:TRANsfer {1 | 2 | 3 | 4}
Transfers the previous defined arbitrary points to the selected channel.

Parameters:

1	channel 1 (CH1)
2	channel 2 (CH2)
3	channel 3 (CH3)
4	channel 4 (CH4)

NOTICE

HMP2020: channel 3 (CH3) and channel 4 (CH4) are not available
HMP2030 / HMP4030: channel 4 (CH4) is not available

ARBitrary:STARt {1 | 2 | 3 | 4}
Starts the previous transfered arbitrary points of the selected channel. An arbitrary sign will be shown on the display.

Parameters:

1	channel 1 (CH1)
2	channel 2 (CH2)
3	channel 3 (CH3)
4	channel 4 (CH4)

NOTICE

HMP2020: channel 3 (CH3) and channel 4 (CH4) are not available
HMP2030 / HMP4030: channel 4 (CH4) is not available

ARBitrary:STOP {1 | 2 | 3 | 4}
Stops the previous transfered arbitrary points of the selected channel.

Parameters:	**1**	channel 1 (CH1)
	2	channel 2 (CH2)
	3	channel 3 (CH3)
	4	channel 4 (CH4)

NOTICE

HMP2020: channel 3 (CH3) and channel 4 (CH4) are not available
HMP2030 / HMP4030: channel 4 (CH4) is not available

ARBitrary:SAVE {1 | 2 | 3}
Stores the previous defined arbitrary points into the internal memory. Up to 3 arbitrary waveforms can be stored.

ARBitrary:RESTore {1 | 2 | 3}
Loads the stored arbitrary waveform from internal memory.

ARBitrary:REPetitions {0...255}
Defines the repetition rate of the defined arbitrary waveform for the previous selected channel. Up to 255 repetitions are possible. If the repetition rate „0“ is selected the arbitrary waveform of the previous selected channel will be repeated infinitely.

Parameters: e.g. ARB:REP 10

ARBitrary:REPetitions?
Queries the defined repetition rate.

Return values e.g. 4

ARBitrary:CLEar
Clears the previous defined arbitrary waveform data for the selected channel.

2.4.1 Arbitrary example

The following programming example generates an arbitrary sequence for the selected channel CH1 which starts at 1 V and 1 A for 1 sec and will be incremented each second by 1 V. Then this sequence will be transfered to CH1. After activating the HMP output (OUTPUT LED is highlighted) the arbitrary waveform will be repeated 10 times.

```
INST OUT1
ARB:DATA 1,1,1,2,2,1,3,3,1
ARB:REP 10
ARB:TRAN 1
ARB:STAR 1
OUTP ON
```

2.5 Status Reporting

2.5.1 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“
- „STATus:QUEStionable Register“
- Diagram (Fig. 2.1)

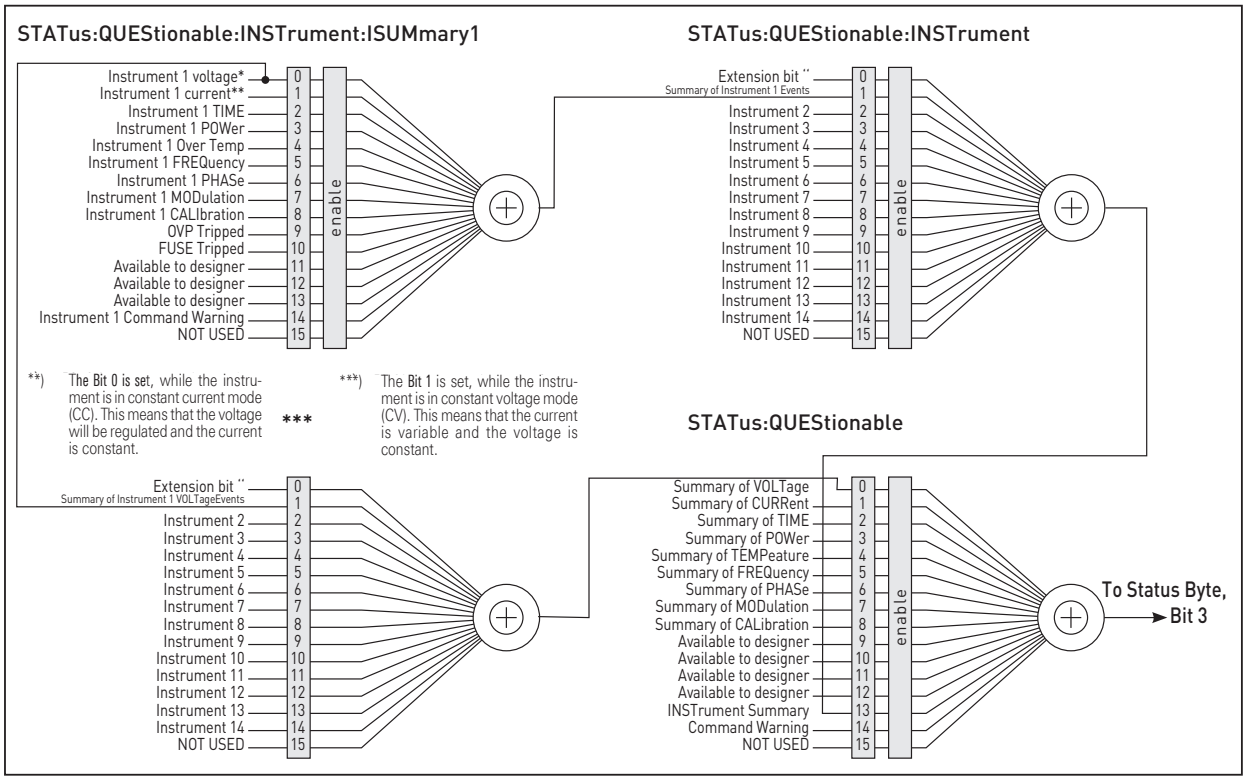


Fig. 2.1: Structure of the status register

The following commands are available:

STATus:QUEStionable[:EVENt]?
Returns 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> Event bits in decimal representation
Range: 0 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: 0 to 65535

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

STATus:QUEStionable:INSTrument[:EVENt]?
Returns 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> Event bits in decimal representation
Range: 0 to 65535

Usage: Query only

STATus:QUEStionable:INSTrument: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: 0 to 65535

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

STATus:QUEStionable:INSTrument:ISUMmary[:EVENt]?
Returns 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> Event bits in decimal representation
Range: 0 to 65535

Usage: Query only

STATus:QUEStionable:INSTrument:ISUMmary:CONDition?
Returns the contents of the CONDition part of the status register to check the actual instrument states. Reading the CONDition registers does not delete the contents.

Return values:
<Condition> Condition bits in decimal representation
Range: 0 to 65535

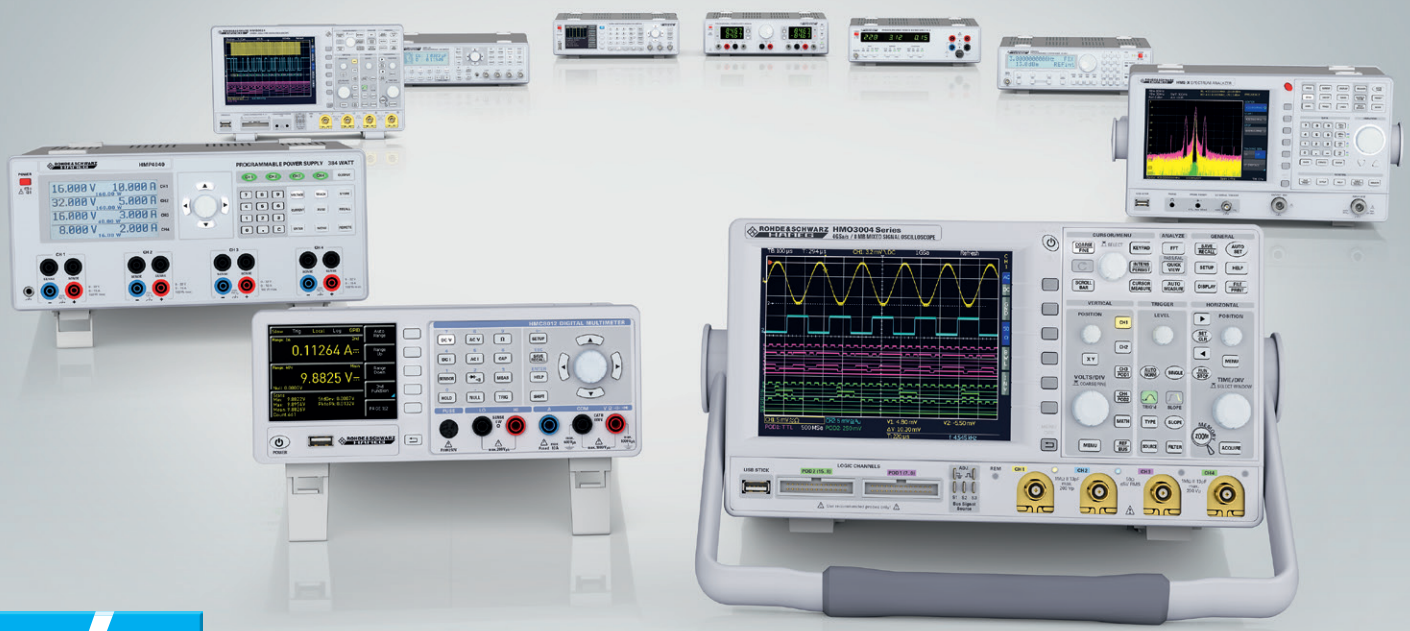
Usage: Query only

STATus:QUEStionable:INSTrument:ISUMmary: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: 0 to 65535

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

HAMEG®
Instruments
A Rohde&Schwarz Company



Value instruments

value-instruments.com

www.hameg.com

HAMEG Instruments GmbH
Industriestr. 6 | 63533 Mainhausen | Germany | Tel +49 (0) 6182 800-500