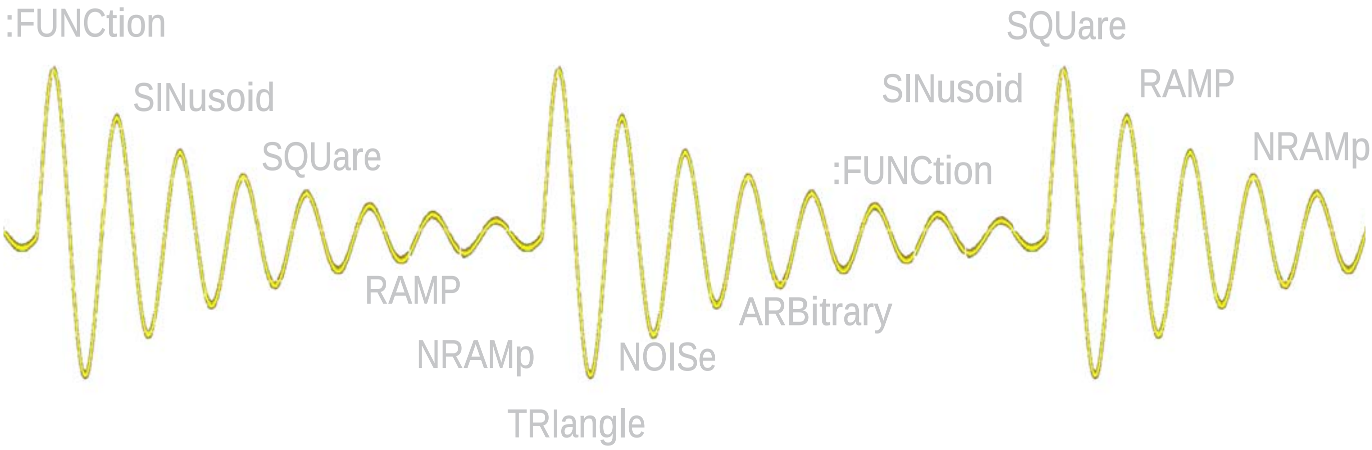
HAMEG Instruments

SCPI-Commands HMF series



Content

1. Introduction

If your HAMEG instrument is equipped with an interface, it can be remote controlled. You get access to nearly all functions that can be called manually via the front panel. HAMEG Instruments offers instrument control via:

HO720	USB and RS232
HO730	USB and Ethernet
HO740	GPIB

1.1 What is SCPI?

SCPI (standard commands for programmable instruments) is an international standard (IEEE-488.2) for external control of measuring instruments etc. It allows you to network complex test setup with host (PC) control. This allows enormous time and manpower reduction.

The SCPI standard contains the programming syntax as well as the main functions of measuring instrument etc. This enables the exchange of comparable instruments from different producers without the need to reprogram the test procedure, as long as these functions and commands have been implemented identically.

1.2 SCPI Syntax

1.2.1 Spelling

Always pay attention that correct spelling is used when composing SCPI commands. All forms of spelling are forbidden except the exact short and long form of a command. Upper and lower case writing can be used. In this document upper case is used for short form, followed by lower case for the long form.

Example: The selection of the menu item 'Measure'

Short form	**:MEAS**
Long form	**:MEASure**
Acceptable spelling	**:MEASURE**
Unacceptable spelling	**:MEASUR**

1.2.2 Path structure

SCPI commands are based on a hierarchical order like a root (tree structure). Each command consists of the declaration of paths, different functions, keywords etc. and the optional allocation of parameters.

1.2.3 Colon

A colon serves as a separator for several key words in the path declaration. Based on the current path a ':' selects a lower hierarchy level. A ':' at the beginning of a command indicates that the following declaration is an element at the ' ROOT' level. The ':' is not applicable, if access is to be made to several elements of the same path. The multiple use of a path is forbidden, if the following command skips to a lower hierarchical level.

1.2.4 Semicolon

A semicolon is used to separate commands from one another.

1.2.5 Comma

If several values can be allocated to a function, they must be separated by commas.

1.2.6 Query

A command forcing the instrument to a direct reply is called a query. They can be used to query system states, parameters and possible border functions. Parameter read out is carried out by question mark (?). Common commands are read without path declaration.

Path and parameters of program command queries have to be specified additionally.

Query without parameter	**:CHAN1:SCAL ?**	**1.0E-01, 100mV**
Query with parameter	**:CHAN1:SCAL ? MIN**	**1.0E-03, 1mV**

1.2.7 Parameter

The transmission of parameters to the instrument is made with the declaration of path and the respective value. The latter is separated from the path by a space character. Please note the different data formats in which values can be assigned.

1.2.8 Instruction termination

The SCPI standard contains so called PMTs (program message terminators) used during instrument control to enable the identification of the end of a command or query by decoding the message bytes. Differentiation is made between ´new line´ (NL) and ´end´ (END). NL (defined as ´ h0a´) will e.g. be transmitted as termination of a command string. Any combination of NL and END is possible. However an instrument has to treat NL, NL+END and END semantically equivalently.

1.2.9 Data Formats

Float

At the input of floating point numbers a ´.´ is used as a decimal separator. Floating point numbers can be delineated in the following ways:

integer	**102**
positive real number	**+10.2**
negative real number	**-10.2**
with exponent	**1.2E-3**
without leading zero	**.123**

The input of the positive leading sign ´+´ is optional.

String

When designating strings as parameters, the string to be transferred is set in quotation marks (“”). The string is defined as a whole value and therefore is separated from the path by a space character.

Character

Character data are text characters which are not set in “”. For example, the activation of channel 1:

Example: **:CHAN1:STAT ON**

In this case ON is the value the function can take over.

Block

This format is especially used for outputting great amounts of data , e.g. when a signal trace or the current system settings are read out. The structure of a data block is as follows:

<#> <ln> <n> <1bytes data>

#	marking a special data format
ln	length of the number that contains the number of data bytes
n	number of data bytes
data	data bytes (1 .. n)

Example for the data stream caused by a query:

#3600abc … xyz

#	start of block data
3	the number containing number of databytes consists of 3 characters
600	number of subsequent data (456 bytes)
a	value of 1st data byte
b	value of 2nd data byte
z	value of 600th data byte

Special number formats

#H	description in hexadecimal form	#Hxxxxxxxx
#B	description in binary form	#Bxxxxxxxx
#Q	description in octal form	#Qxxxxxxxx

1.3 State and Event

The SCPI standard contains an event handling system for all available interfaces that can be used to be informed about the processes within the oscilloscope. According to the standard the oscilloscope replies only after receiving a query but the event handling enables the device to inform the user that an extraordinary event took place.

SESR - Standard Event State Register

The Standard Event Status Register includes status indicators as well as error messages of the instrument.

Bit 7							Bit 0
R	R	R	R	R	R	R	R
PON	URQ	CME	EXE	DDE	QYE	RQC	OPC

Table 1: SESR - Standard Event Status Register

PON	Power On	The instrument was switched on
URQ	User Request	unused (0)
CME	Command Error	Error during the analysis of a command
EXE	Execution Error	Error during command execution
DDE	Device Dependend Error	An instrument error has appeared
QYE	Query Error	Data got lost or are not available during a query
RQC	Request Control	unused (0)
OPC	Operation Complete	all current operations have ended

SESER Standard Event State Enable Register

Bit 7							Bit 0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
PON	URQ	CME	EXE	DDE	QYE	RQC	OPC

Table 2: SESER - Standard Event Status Enable Register

The SESER determines which events are evaluated.

1	Event will be evaluated
0	Event will be ignored

SBR - State Byte Register

Bit 7							Bit 0
R	R	R	R	R	R	R	R
/	RQS/MSS	ESB	MAV	/	/	/	/

Table 3: SBR - State Byte Register

RQS	Request Service	instrument requests to send
MSS	Master Status Summary	logical sum of ESB and MAV bits
ESB	Event State Bit	an event message is available
MAV	Message Available	an output is available

SRER - Service Request Enable Register

Bit 7							Bit 0
R/W	R	R/W	R/W	R/W	R/W	R/W	R/W
/	/	ESB	MAV	/	/	/	/

Table 4: SRER - Service Request Enable Register

1	output permitted
0	output prohibited

The SRER defines which interfaces may ask for permission to transmit. The GPIB interface is the only one which has a hardware connection to the user for the transmission request (RQS bit); the status of this line will tell whether an event happened.

All other interfaces (RS-232, USB, Ethernet) do not support this. If the user wishes to use the event handling feature, he will have to read the status, e.g. by polling, from the instrument in every case. Eventually, the status byte SBR will yield the desired information.

2 Overview of SCPI Commands

2.1 Status register and common commands

```
*IDN?
*RST
*TST?
*OPC
*OPC?
*WAI
*LRN?
*STB?
*SRE <activation value>
*SRE?
*ESR?
*ESE <activation value>
*ESE?
*CLS
*PSC {0|1}
*PSC?

STATus
      :QUEStionable:CONDition?
      :QUEStionable[:EVENt]?
      :QUEStionable:ENABle <activation value>
      :QUEStionable:ENABle?

SYSTem:ERRor?

STATus:PRESet

SYSTem
      :BEEPer:STATe {OFF|ON|0|1}                         *with release 1.2
      :BEEPer:STATe?
```

2.2 Function and output commands

2.2.1 APPLY Commands

```
APPLy
      :SINusoid [<frequency> [,<amplitude>] [,<offset>]]       with release 1.2
      :SQUare [<frequency> [,<amplitude>] [,<offset>]]
      :RAMP [<frequency> [,<amplitude>] [,<offset>]]
      :PULSe [<frequency> [,<amplitude>] [,<offset>]]
      :ARB [<frequency> [,<amplitude>] [,<offset>]]
APPLy?
```

2.2.2 Configuration of the Output

```
FUNCtion {SINusoid|SQUare|RAMP|PULSe|ARBitrary}
FUNCtion?

FREQuency {<frequency>|MINimum|MAXimum}
FREQuency? [MINimum|MAXimum]
```

```
PERiode {<period in sec.>|MINimum|MAXimum}
PERiode? [MINimum|MAXimum]

VOLTage {<amplitude>|MINimum|MAXimum}
VOLTage? [MINimum|MAXimum]

VOLTage
      :HIGH {<voltage>|MINimum|MAXimum}
      :HIGH? [MINimum|MAXimum]
      :LOW {<voltage>|MINimum|MAXimum}
      :LOW? [MINimum|MAXimum]

VOLTage:OFFSet {<offset>|MINimum|MAXimum}
VOLTage:OFFSet? [MINimum|MAXimum]

FUNCtion:SQUare:DCYCle {<percent>|MINimum|MAXimum}
FUNCtion:SQUare:DCYCle? [MINimum|MAXimum]

FUNCtion:RAMP:SYMMetry {<percent>|MINimum|MAXimum}
FUNCtion:RAMP:SYMMetry? [MINimum|MAXimum]

OUTPut {OFF|ON|0|1}
OUTPut?

OUTPut:LOAD {TERMinated|INFinity}
OUTPut:LOAD?

OUTPut:POLarity {NORMal|INVerted}
OUTPut:POLarity?

OUTPut:OFFSet {OFF|ON|0|1}
OUTPut:OFFSet?

FUNCtion:PULSe
 :WIDTh:HIGH {<seconds>|MINimum|MAXimum}
 :WIDTh:HIGH? [MINimum|MAXimum]

 :WIDTh:LOW {<seconds>|MINimum|MAXimum}
 :WIDTh:LOW? [MINimum|MAXimum]

 :DCYCle {<percent>|MINimum|MAXimum}
 :DCYCle? [MINimum|MAXimum]

 :ETIMe {<seconds>|MINimum|MAXimum}
 :ETIMe? [MINimum|MAXimum]
```

2.2.3 Modulation commands

AM-Commands

```
AM:INTernal
      :FUNCtion {SINusoid|SQUare|RAMP|NRAMp|TRIangle|NOISe|ARBitrary*}
      :FUNCtion?                                          *with release 1.2

AM:INTernal
      :FREQuency {<frequency>|MINimum|MAXimum}
      :FREQuency? [MINimum|MAXimum]
```

```
AM:DEPTh {<modulation in percent>|MINimum|MAXimum}
AM:DEPTh? [MINimum|MAXimum]

AM:SOURce {INTernal|EXTernal}
AM:SOURce?

AM:STATe {OFF|ON|0|1}
AM:STATe?
```

FM-Commands

```
FM:INTernal
      :FUNCtion {SINusoid|SQUare|RAMP|NRAMp|TRIangle|NOISe|ARBitrary*}
      :FUNCtion?                                                  *with release 1.2

FM:INTernal
      :FREQuency {<frequency>|MINimum|MAXimum}
      :FREQuency? [MINimum|MAXimum]

FM:DEViation {<peak-frequency in Hz>|MINimum|MAXimum}
FM:DEViation? [MINimum|MAXimum]

FM:SOURce {INTernal|EXTernal}
FM:SOURce?

FM:STATe {OFF|ON|0|1}
FM:STATe?
```

PM-Commands

```
PM:INTernal
      :FUNCtion {SINusoid|SQUare|RAMP|NRAMp|TRIangle|NOISe|ARBitrary*}
      :FUNCtion?                                                  *with release 1.2

PM:INTernal
      :FREQuency {<frequency>|MINimum|MAXimum}
      :FREQuency? [MINimum|MAXimum]

PM:DEViation {<phase in degree>|MINimum|MAXimum}
PM:DEViation? [MINimum|MAXimum]

PM:SOURce {INTernal|EXTernal}
PM:SOURce?

PM:STATe {OFF|ON|0|1}
PM:STATe?
```

FSK-Commands

```
FSKey:FREQuency {<frequency>|MINimum|MAXimum}
FSKey:FREQuency? [MINimum|MAXimum]

FSKey:INTernal:RATE {<rate in Hz>|MINimum|MAXimum}
FSKey:INTernal:RATE? [MINimum|MAXimum]

FSK:DCYCle {<duty cycle in %>|MINimum|MAXimum}
FSK:DCYCle? [MINimum|MAXimum]

FSKey:SOURce {INTernal|EXTernal}
FSKey:SOURce?
```

```
FSKey:STATe {OFF|ON|0|1}
FSKey:STATe?
```

PWM-Commands

```
PWM:INTernal:FUNCtion {SINusoid|SQUare*|RAMP*|NRAMp*|TRIangle*|NOISe*|ARBitrary*}
PWM:INTernal:FUNCtion?                                  *with release 1.2

PWM:INTernal:FREQuency {<frequency>|MINimum|MAXimum}
PWM:INTernal:FREQuency? [MINimum|MAXimum]

PWM:DCYCle* {<deviation in %>|MINimum|MAXimum}          *with release 1.2
PWM:DCYCle? [MINimum|MAXimum]

PWM:STATe {OFF|ON|0|1}
PWM:STATe?

PWM:SOURce {INTernal|EXTernal}
PWM:SOURce?
```

2.2.4 Sweep and burst commands

```
FREQuency
:STARt {<frequency>|MINimum|MAXimum}
:STARt? [MINimum|MAXimum]

:STOP {<frequency>|MINimum|MAXimum}
:STOP? [MINimum|MAXimum]

FREQuency
      :CENTer {<frequency>|MINimum|MAXimum}
      :CENTer? [MINimum|MAXimum]

      :SPAN {<frequency>|MINimum|MAXimum}
      :SPAN? [MINimum|MAXimum]

SWEep
      :SPACing {LINear|LOGarithmic}
      :SPACing?

      :TIME {<seconds>|MINimum|MAXimum}
      :TIME? [MINimum|MAXimum]

SWEep:STATe {OFF|ON|0|1}
SWEep:STATe?

MARKer:FREQuency {<frequency>|MINimum|MAXimum}
MARKER:FREQuency? [MINimum|MAXimum]
MARKer {OFF|ON|0|1}
MARKer?

BURSt:MODE {TRIGgered|GATed}
BURSt:MODE?

BURSt:NCYCles {<number of cycles>|MINimum|MAXimum}
BURSt:NCYCles? [MINimum|MAXimum]
```

```
BURSt:INTernal:PERiod {<seconds>|MINimum|MAXimum}
BURSt:INTernal:PERiod? [MINimum|MAXimum]

BURSt:PHASe {<angle>|MINimum|MAXimum}
BURSt:PHASe? [MINimum|MAXimum]

BURSt:STATe {OFF|ON|0|1}
BURSt:STATe?
```

2.2.5 Arbitrary commands

*with release 1.2

```
DATA {<value>,<value>|<binary block>}

DATA:COPY <Arb-name>

DATA:DELete {<Arb-name>}
DATA:DELete:ALL

DATA:SWAP {NORMal|SWAPped}
DATA:SWAP?

FUNCtion:ARBitrary {<Arb-name>}
FUNCtion:ARBitrary?
```

2.2.6 Trigger commands

```
TRIGger                                                     *with release 1.2
*TRG

TRIGger:SOURce {IMMediate|EXTernal}
TRIGger:SOURce?

OUTPut:TRIGger {OFF|ON|0|1}
OUTPut:TRIGger?
```

2.2.7 Save and recall of instrument settings

```
*SAV {0|1|2|3|4} Power off settings are stored at instrument setting 0      *with release 1.2
*RCL {0|1|2|3|4} Settings 1, 2, 3 und 4 are user defined settings.
```

3 Detailed description of SCPI commands

3.1 Status register and common commands

*CLS

Clear the event register in all register groups. This commands also clears the error queue and cancels *OPC operation.

*ESR?

A query of this register returns a decimal value which corresponds to the binary-weighted sum of all bits set in the register.

*ESE < *activation value*>
*ESE?

Enable bits in the Standard Event Status Register to be reported in the Status Byte. The *ESE? query returns a decimal value which corresponds to the binary-weighted sum of all bits enabled by the *ESE command.

*IDN?

Read the function generator's identification string which contains four fields separated by commas.

*LRN?

Query the function generator and return a string of SCPI commands containing the current settings.

*OPC
*OPC?

Set the Operation Complete bit in the Standard Event Register after the previous commands have completed.

*RST

Reset the function generator to its factory default state.

*STB?

Query the summary (condition) register in this register group.

*SRE <*activation value*>
*SRE?

Enable bits in the Status Byte to generate a Service Request. To enable specific bits, you must write a decimal value which corresponds to the binary-weighted sum of the bits in the register. The *SRE? query returns a decimal value which corresponds to the binary-weighted sum of all bits enabled by the *SRE command.

*TST?

Perform a complete self-test of the function generator. Returns "+0" (PASS) or "+1" (FAIL). If the test fails, one or more error messages will be generated to provide additional information on the failure.
Use the SYST:ERR? command to read the error queue.

*WAI

Waits for all pending operations to complete before executing any additional commands over the interface.

*PSC { 0 | 1 }
*PSC?

Power-On Status Clear. Clear the Standard Event enable register and Status Byte condition register at power on (*PSC 1). When *PSC 0 is in effect, these two registers are not cleared at power on. The default is *PSC 1.
The *PSC? query returns the power-on status clear setting and returns "0" (do not clear at power on) or "1" (clear at power on).

STATus
:QUEStionable:CONDition?

Query the condition register in this group. This is a read-only register and bits are not cleared when you read the register. A query of this register returns a decimal value which corresponds to the binaryweighted sum of all bits set in the register.

:QUEStionable[:EVENt]?

Query the event register in this register group. This is a read-only register. Once a bit is set, it remains set until cleared by this command or *CLS (clear status) command. A query of this register returns a decimal value which corresponds to the binary-weighted sum of all bits set in the register.

:QUEStionable:ENABle <*activation value*>
:QUEStionable:ENABle?

Enable bits in the enable register in this register group. The selected bits are then reported to the Status Byte. A *CLS (clear status) will not clear the enable register but it does clear all bits in the event register. The STATus:PRESet command clears all bits in the enable register.
To enable bits in the enable register, you must write a decimal value which corresponds to the binary-weighted sum of the bits you wish to enable in the register.
The QUES:ENAB? query returns a decimal value which corresponds to the binary-weighted sum of all bits enabled by the STAT:QUES:ENAB command.

STATus:PRESet

Clear all bits in the Questionable Data enable register and the Standard Operation enable register.

SYSTem:ERRor?

Reads one error from the error queue.
After reading of an error it will be erased from the register. The next query will read the next error provided there was more than one.

SYSTem
:BEEPer:STATe { OFF | ON | 0 | 1 }

with release 1.2

:BEEPer:STATe?

This command allows to switch the button sounds on/off.
The STAT? query returns the current setting of the button sounds (OFF,ON).

3.2 Function and output commands

3.2.1 Apply commands

APPLy

:SINusoid [<*frequency*> [,<*amplitude*>] [,<*offset*>]]

with release 1.2

:SQUare [<*frequency*> [,<*amplitude*>] [,<*offset*>]]
:RAMP [<*frequency*> [,<*amplitude*>] [,<*offset*>]]
:PULSe [<*frequency*> [,<*amplitude*>] [,<*offset*>]]
:ARB [<*frequency*> [,<*amplitude*>] [,<*offset*>]]

APPLy?

The APPL command provides the most straightforward method to program the function generator over the remote interface. You can select the function, frequency, amplitude, and offset all in one command as shown in the syntax statement above.

Because of the use of optional parameters in the APPLy commands (enclosed in square brackets), you must specify frequency to use the amplitude parameter, and you must specify both frequency and amplitude to use the offset parameter. For example, the following command string is valid (frequency and amplitude are specified but offset is omitted and therefore uses a default value).

```
APPL:SIN 4.0E+2, 5.0
```

However, you cannot specify an amplitude or offset without specifying a frequency.
You can substitute "MIN" or "MAX" in place of specific values for the frequency, amplitude, and offset parameters.
For example, the following statement outputs a 5 V_{pp} sine wave at 50 MHz (the maximum frequency of the HMF2550 for sine) with a -1.2 Volt offset.

APPL:SIN MAX, 5.0, -1.2

3.2.2 Configuration of the output

FUNCtion { **SINus** | **SQUare** | **RAMP** | **PULSe** | **NOISe** | **ARBitrary** }
FUNCtion?
Select the output function. The selected waveform uses the previously selected frequency, amplitude, and offset voltage settings.
The FUNC? query returns the current function (SIN, SQU, RAMP, PULS, NOIS, ARB).

OUTPut { **OFF** | **ON** | 0 | 1 }
OUTPut?
Activates or deactivates the output.
The OUTP? query returns the current status (ON, OFF).

OUTPut:LOAD { **TERMinated** | **INFinity** }
OUTPut:LOAD?
Select the output load. Terminated is defined as 50Ω, infinity is defined as open circuit (without any load).
The LOAD? query returns the current status (terminated, infinity).

OUTPut:POLarity { **NORMal** | **INVerted** } *only valid for Pulse and Arbitrary function*
OUTPut:POLarity?
Inverts the current output waveform (pulse or arbitrary only).
The POL? query returns the current status (normal, inverted).

Sine function

FREQuency { <***frequency***> | **MINimum** | **MAXimum** }
FREQuency?
Set the output frequency. MIN selects the lowest frequency allowed for the sine function and MAX selects the highest frequency allowed. The FREQ? query returns the frequency setting in Hertz for the sine function.

PERiod { <***period***> | **MINimum** | **MAXimum** }
PERiod?
Set the output period. MIN selects the lowest period allowed for the sine function and MAX selects the highest period allowed. The PER? query returns the period setting in sec. for the sine function.

VOLTage { <***amplitude***> | **MINimum** | **MAXimum** }
VOLTage?
Set the output amplitude. MIN selects the lowest amplitude allowed for the sine function and MAX selects the highest amplitude allowed. The VOLT? query returns the amplitude setting in Volt for the sine function.

VOLTage
:HIGH { <***voltage***> | **MINimum** | **MAXimum** }
:HIGH? [**MINimum** | **MAXimum**]
:LOW { <***voltage***> | **MINimum** | **MAXimum** }
:LOW? [**MINimum** | **MAXimum**]
Set the upper and the lower level of the sine function amplitude. MIN selects the lowest amplitude allowed for the sine function and MAX selects the highest amplitude allowed.
This command offers one possibility to specify a sine wave including a offset.

The HIGH / LOW? query returns the current value in Volt.

VOLTage
:OFFSet { <***offset***> | **MINimum** | **MAXimum** }
:OFFSet?

Set the output offset. MIN selects the lowest offset allowed for the sine function and MAX selects the highest offset allowed. The OFFS? query returns the current offset setting in Volt for the sine function.

OUTPut
:OFFSet { **OFF** | **ON** | 0 | 1 }
:OFFSet?

Set the offset status. Activates or deactivates the offset voltage.
The OFFS? query returns the current status of the offset state (OFF, ON).

Square function

FREQuency { <***frequency***> | **MINimum** | **MAXimum** }
FREQuency?

Set the output frequency. MIN selects the lowest frequency allowed for the square function and MAX selects the highest frequency allowed. The FREQ? query returns the frequency setting in Hertz for the square function.

PERiod { <***period***> | **MINimum** | **MAXimum** }
PERiod?

Set the output period. MIN selects the lowest period allowed for the square function and MAX selects the highest period allowed. The PER? query returns the period setting in sec. for the square function.

VOLTage { <***amplitude***> | **MINimum** | **MAXimum** }
VOLTage?

Set the output amplitude. MIN selects the lowest amplitude allowed for the square function and MAX selects the highest amplitude allowed. The VOLT? query returns the amplitude setting in Volt for the square function.

VOLTage
:HIGH { <***voltage***> | **MINimum** | **MAXimum** }
:HIGH? [**MINimum** | **MAXimum**]
:LOW { <***voltage***> | **MINimum** | **MAXimum** }
:LOW? [**MINimum** | **MAXimum**]

Set the upper and the lower level of the square function amplitude. MIN selects the lowest amplitude allowed for the square function and MAX selects the highest amplitude allowed.
This command offers one possibility to specify a square function including a offset.
The HIGH / LOW? query returns the current value in Volt.

VOLTage
:OFFSet { <***offset***> | **MINimum** | **MAXimum** }
:OFFSet?

Set the output offset. MIN selects the lowest offset allowed for the square function and MAX selects the highest offset allowed. The OFFS? query returns the offset setting in V for the square function.

FUNCtion:SQUare
:DCYCle { <***duty cycle in %***> | **MINimum** | **MAXimum** }
:DCYCle?

Set the output duty cycle. MIN selects the lowest duty cycle allowed for the square function and MAX selects the highest duty cycle allowed. The DCYC? query returns the amplitude setting in % for the square function.

Ramp function

FREQuency { <***frequency***> | **MINimum** | **MAXimum** }
FREQuency?
Set the output frequency. MIN selects the lowest frequency allowed for the ramp function and MAX selects the highest frequency allowed. The FREQ? query returns the frequency setting in Hertz for the ramp function.

PERiod { <***period***> | **MINimum** | **MAXimum** }
PERiod?
Set the output period. MIN selects the lowest period allowed for the ramp function and MAX selects the highest period allowed. The PER? query returns the period setting in sec. for the ramp function.

VOLTage { <***amplitude***> | **MINimum** | **MAXimum** }
VOLTage?
Set the output amplitude. MIN selects the lowest amplitude allowed for the ramp function and MAX selects the highest amplitude allowed. The VOLT? query returns the amplitude setting in Volt for the ramp function.

VOLTage
:HIGH { <***voltage***> | **MINimum** | **MAXimum** }
:HIGH? [**MINimum** | **MAXimum**]
:LOW { <***voltage***> | **MINimum** | **MAXimum** }
:LOW? [**MINimum** | **MAXimum**]
Set the upper and the lower level of the ramp function amplitude. MIN selects the lowest amplitude allowed for the ramp function and MAX selects the highest amplitude allowed.
This command offers one possibility to specify a ramp function including a offset.
The HIGH / LOW? query returns the current value in Volt.

VOLtage
:OFFSet { <***offset***> | **MINimum** | **MAXimum** }
:OFFSet?
Set the output offset. MIN selects the lowest offset allowed for the ramp function and MAX selects the highest offset allowed. The OFFS? query returns the offset setting in Volt for the ramp function.

FUNCtion:RAMP
:SYMmetry { <***value in %*** > | **MINimum** | **MAXimum** }
:SYMmetry?
Set the output symmetry. MIN selects the lowest symmetry allowed for the ramp function and MAX selects the highest symmetry allowed. The SYM? query returns the symmetry setting in % for the square function.

Pulse function

FREQuency { <***frequency***> | **MINimum** | **MAXimum** }
FREQuency?
Set the output frequency. MIN selects the lowest frequency allowed for the pulse function and MAX selects the highest frequency allowed. The RAMP? returns the frequency setting in Hertz for the pulse function.

PERiod { <***period***> | **MINimum** | **MAXimum** }
PERiod?
Set the output period. MIN selects the lowest period allowed for the pulse function and MAX selects the highest period allowed. The PER? query returns the period setting in sec. for the pulse function.

:VOLTage { <***amplitude***> | **MINimum** | **MAXimum** }
:VOLTage?
Set the output amplitude. MIN selects the lowest amplitude allowed for the pulse function and MAX selects the highest amplitude allowed. The VOLT? query returns the amplitude setting in Volt for the pulse function.

VOLTage
:HIGH { <***voltage***> | **MINimum** | **MAXimum** }
:HIGH? [**MINimum** | **MAXimum**]
:LOW { <***voltage***> | **MINimum** | **MAXimum** }
:LOW? [**MINimum** | **MAXimum**]

Set the upper and the lower level of the pulse function amplitude. MIN selects the lowest amplitude allowed for the pulse function and MAX selects the highest amplitude allowed. This command offers one possibility to specify a pulse function including a offset. The HIGH / LOW? query returns the current value in Volt.

VOLTage
:OFFSet { <***offset***> | **MINimum** | **MAXimum** }
:OFFSet?

Set the output offset. MIN selects the lowest offset allowed for the pulse function and MAX selects the highest offset allowed. The OFFS? query returns the offset setting in Volt for the pulse function.

FUNCtion:PULSe
WIDTh:HIGH { <***seconds***> | **MINimum** | **MAXimum** }
WIDTh:HIGH? { **MINimum** | **MAXimum** }
WIDTh:LOW { <***seconds***> | **MINimum** | **MAXimum** }
WIDTh:LOW? { **MINimum** | **MAXimum** }

Set the output width for the upper or lower part of the pulse function. MIN selects the lowest width allowed for the pulse function and MAX selects the highest width allowed. The WIDT? query returns the respective width setting in sec. for the upper or lower part of the pulse function.

FUNCtion:PULSe
:DCYCle { <***duty cycle in %*** > | **MINimum** | **MAXimum** }
:DCYCle?

Set the output duty cycle. MIN selects the lowest duty cycle allowed for the pulse function and MAX selects the highest amplitude allowed. The DCYC? query returns the duty cycle setting in % for the pulse function.

FUNCtion:PULSe
:ETIMe { <***seconds***> | (**MINimum** | **MAXimum** }
:ETIMe? { **MINimum** | **MAXimum** }

Set the output edge times. MIN selects the lowest edge time allowed for the pulse function and MAX selects the highest edge time allowed. The ETIM? query returns the current edge time setting in sec. for the pulse function.

3.2.3 Modulations commands

AM modulation

AM:INTernal
:FUNCtion { **SINusoid** | **SQUare** | **RAMP** | **NRAMp** | **TRIangle** | **NOISe** | **ARBitrary*** }
:FUNCtion? *with release 1.2

Set the shape of the AM modulation signal. All above mentioned signal shapes can be selected.
The FUNC? query returns the current setting of the modulation shape.

AM:INTernal
:FREQuency { <***frequency***> | **MINimum** | **MAXimum** }
:FREQuency? [**MINimum** | **MAXimum**]

Set the frequency of the AM modulation signal. MIN selects the lowest frequency allowed for the AM modulation signal and MAX selects the highest frequency allowed. The FREQ? query returns the current frequency setting for the modulation signal in Hertz.

AM:DEPTh { <***modulation in %***> | **MINimum** | **MAXimum** }
AM:DEPTh? [**MINimum** | **MAXimum**]

Set the AM modulation depth in %. MIN selects the lowest modulation depth for the AM modulation signal and MAX selects the highest modulation depth allowed.
The DEPT? query returns the current setting in %.

AM:SOURce { **INTernal** | **EXTernal** }
AM:SOURce?

Set the modulation source. INT selects the internal modulation source for the AM modulation and EXT selects a external source as reference for the modulation. This external source needs to be connected to the rear "modulation input" connector.
The SOUR? query returns the current setting of the modulation source (internal, external).

AM:STATe { **OFF** | **ON** | 0 | 1 }
AM:STATe?

Set the AM modulation status. Activates or deactivates the AM modulation of the main signal.
The STAT? query returns the current status of the AM modulation state (OFF, ON).

FM modulation

FM:INTernal
:FUNCtion { **SINusoid** | **SQUare** | **RAMP** | **NRAMp** | **TRIangle** | **NOISe** | **ARBitrary*** }
:FUNCtion?

*with release 1.2

Set the shape of the FM modulation signal. All above mentioned signal shapes can be selected.
The FUNC? query returns the current setting of the modulation shape.

FM:INTernal
:FREQuency { <***frequency***> | **MINimum** | **MAXimum** }
:FREQuency? [**MINimum** | **MAXimum**]

Set the frequency of the FM modulation signal. MIN selects the lowest frequency allowed for the FM modulation signal and MAX selects the highest frequency allowed. The FREQ? query returns the current frequency setting for the modulation signal in Hertz.

FM:DEViation { <***peak-frequency in Hz***> | **MINimum** | **MAXimum** }
FM:DEViation? [**MINimum** | **MAXimum**]

Set the peak frequency deviation in Hertz. This value represents the variation in frequency of the modulating waveform from the carrier frequency. MIN selects the lowest frequency allowed for the deviation and MAX selects the highest frequency allowed.

FM:SOURce { **INTernal** | **EXTernal** }
FM:SOURce?

Set the modulation source. INT selects the internal modulation source for the FM modulation and EXT selects a external source as reference for the modulation. This external source needs to be connected to the rear "modula-tion input" connector.
The SOUR? query returns the current setting of the modulation source (internal, external).

FM:STATe { **OFF** | **ON** | 0 | 1 }
FM:STATe?

Set the FM modulation status. Activates or deactivates the FM modulation of the main signal.
The STAT? query returns the current status of the AM modulation state (OFF, ON).

PM modulation

PM:INTernal
:FUNCtion { **SINusoid** | **SQUare** | **RAMP** | **NRAMp** | **TRIangle** | **NOISe** | **ARBitrary*** }
:FUNCtion?
*with release 1.2

Set the shape of the PM modulation signal. All above mentioned signal shapes can be selected.
The FUNC? query returns the current setting of the modulation shape.

PM:INTernal
:FREQuency { <***frequency***> | **MINimum** | **MAXimum** }
:FREQuency? [**MINimum** | **MAXimum**]

Set the frequency of the PM modulation signal. MIN selects the lowest frequency allowed for the PM modulation signal and MAX selects the highest frequency allowed. The FREQ? query returns the current frequency setting for the modulation signal in Hertz.

PM:DEViation { <***phase in degree***> | **MINimum** | **MAXimum** }
PM:DEViation? [**MINimum** | **MAXimum**]

Set the deviation of the phase in degree (-180°...+180°). This value represents the variation in degree of the modulating waveform from the carrier waveform. MIN selects the lowest angle allowed for the deviation and MAX selects the highest angle allowed.

PM:SOURce { **INTernal** | **EXTernal** }
PM:SOURce?

Set the modulation source. INT selects the internal modulation source for the PM modulation and EXT selects a external source as reference for the modulation. This external source needs to be connected to the rear "modulation input" connector.
The SOUR? query returns the current setting of the modulation source (internal, external).

PM:STATe { **OFF** | **ON** | 0 | 1 }
PM:STATe?

Set the PM modulation status. Activates or deactivates the PM modulation of the main signal.
The STAT? query returns the current status of the PM modulation state (OFF, ON).

FSK modulation

FSKey:FREQuency { <***frequency***> | **MINimum** | **MAXimum** }
FSKey:FREQuency? [**MINimum** | **MAXimum**]

Set the frequency of the FSK modulation signal. MIN selects the lowest frequency allowed for the FSK modulation signal and MAX selects the highest frequency allowed. The FREQ? query returns the current frequency setting for the modulation signal in Hertz.

FSKey:INTernal
:RATE { <***rate in Hz***> | **MINimum** | **MAXimum** }
:RATE? [**MINimum** | **MAXimum**]

Set the rate (frequency) of the FSK modulation in order to define the timeframe between the frequency shifts.
MIN selects the lowest rate allowed for the FSK modulation signal and MAX selects the highest rate allowed.
The RATE? query returns the current rate setting for the modulation signal in Hertz.

FSK:DCYCle {<*duty cycle in %*> | MINimum | MAXimum }
FSK:DCYCle? [MINimum | MAXimum]

Set the duty cycle for the FSK modulation.
MIN selects the lowest duty cycle for the FSK modulation and MAX selects the highest duty cycle allowed.
The DCYC? query returns the current setting in %.

FSKey:SOURce { INTernal | EXTernal }
FSKey:SOURce?

Set the modulation source. INT selects the internal modulation source for the FSK modulation and EXT selects a external source as reference for the modulation. This external source needs to be connected to the rear "modulation input" connector.
The SOUR? query returns the current setting of the modulation source (internal, external).

FSKey:STATe { OFF | ON | 0 | 1 }
FSKey:STATe?

Set the FSK modulation status. Activates or deactivates the FSK modulation of the main signal.
The STAT? query returns the current status of the FSK modulation state (OFF, ON).

PWM-Modulation

PWM:INTernal:FUNCtion {SINusoid|SQUare*|RAMP*|NRAMp*|TRIangle*|NOISe*|ARBitrary*}
PWM:INTernal:FUNCtion?

*with release 1.2

Set the shape of the PWM modulation signal. All above mentioned signal shapes can be selected.
The FUNC? query returns the current setting of the modulation shape.

PWM:INTernal:FREQuency {<*frequency*>|MINimum|MAXimum}
PWM:INTernal:FREQuency? [MINimum|MAXimum]

Set the frequency of the PWM modulation signal. MIN selects the lowest frequency allowed for the PWM modulation signal and MAX selects the highest frequency allowed.
The FREQ? query returns the current frequency setting for the modulation signal in Hertz.

PWM:DCYCle* {<*deviation in %*>|MINimum|MAXimum}
PWM:DCYCle? [MINimum|MAXimum]

*with release 1.2

Set the duty cycle deviation for the PWM modulation.
MIN selects the lowest duty cycle deviation for the PWM modulation and MAX selects the highest duty cycle deviation allowed.
The DCYC? query returns the current setting in %.

PWM:STATe {OFF|ON|0|1}
PWM:STATe?

Set the PWM modulation status. Activates or deactivates the PWM modulation of the main signal.
The STAT? query returns the current status of the PWM modulation state (OFF, ON).

PWM:SOURce {INTernal|EXTernal}
PWM:SOURce?

Set the modulation source. INT selects the internal modulation source for the PWM modulation and EXT selects a external source as reference for the modulation. This external source needs to be connected to the rear "modulation input" connector.
The SOUR? query returns the current setting of the modulation source (internal, external).

3.2.4 Sweep and Burst commands

Sweep commands

FREQuency
:STARt { <***frequency***> | **MINimum** | **MAXimum** }
:STARt? [**MINimum** | **MAXimum**]

Set the start frequency of the sweep. MIN selects the lowest frequency allowed for the sweep function and MAX selects the highest frequency allowed. The FREQ? query returns the current start frequency setting for the sweep function in Hertz.

:STOP { <***frequency***> | **MINimum** | **MAXimum** }
:STOP? [**MINimum** | **MAXimum**]

Set the stop frequency of the sweep. MIN selects the lowest stop frequency allowed for the sweep function and MAX selects the highest frequency allowed. The FREQ? query returns the current stop frequency setting for the sweep function in Hertz.

FREQuency
:CENTer { <***frequency***> | **MINimum** | **MAXimum** }
:CENTer? [**MINimum** | **MAXimum**]

Set the center frequency of the sweep. MIN selects the lowest center frequency allowed for the sweep function and MAX selects the highest frequency allowed. The FREQ? query returns the current center frequency setting for the sweep function in Hertz.

:SPAN { <***frequency***> | **MINimum** | **MAXimum** }
:SPAN? [**MINimum** | **MAXimum**]

Set the span frequency of the sweep. MIN selects the lowest span frequency allowed for the sweep function and MAX selects the highest frequency allowed. The FREQ? query returns the current span frequency setting for the sweep function in Hertz.

SWEep
:SPACing { **LINear** | **LOGarithmic** }
:SPACing?

Set the spacing of the sweep to linear or logarithmic.
The SPAC? query returns the current setting of the spacing (linear, logarithmic).

SWEep
:TIME { <***seconds***> | **MINimum** | **MAXimum** }
:TIME? [**MINimum** | **MAXimum**]

Set the time required to sweep from the start frequency to the stop frequency. MIN selects the lowest sweep time allowed for the sweep function and MAX selects the highest sweep time allowed.The TIME? query returns the sweep time in seconds.

SWEep:STATe { **OFF** | **ON** | 0 | 1 }
SWEep:STATe?

Set the sweep status. Activates or deactivates the sweep function of the signal.
The STAT? query returns the current status of the sweep function state (OFF, ON).

MARKer:FREQuency { <***frequency***> | **MINimum** | **MAXimum** }
MARKER:FREQuency? [**MINimum** | **MAXimum**]

Set the frequency of the marker. MIN selects the lowest marker frequency allowed for the marker function and MAX selects the highest frequency allowed.
The FREQ? query returns the current setting of the marker frequency.

MARKer { **OFF** | **ON** | 0 | 1 }
MARKer?
Disable or enable the frequency marker. The MARK? query returns the current setting of the marker function (OFF,ON).

Burst commands

BURSt:MODE { **TRIGgered** | **GATed** }
BURSt:MODE?
Set the BURST mode. In TRIG mode the signal will be started after the pos. or neg. Slope of the trigger due to current trigger settings. In GAT mode the signal will be started with the pos. slope and will be interrupted by the neg. slope of the gate signal.
The MODE? query returns the current setting of the mode function (triggered, gated).

BURSt:NCYCles { <***number of cycles***> | **MINimum** | **MAXimum** }
BURSt:NCYCles? [**MINimum** | **MAXimum**]
Set the number of repetition cycles of the BURST mode. MIN selects the lowest number of repetitions allowed for the cycle function and MAX selects the highest repetitions allowed.
The NCYC? query returns the current setting of the burst repetition function.

BURSt:INTernal:PERiod { <***seconds***> | **MINimum** | **MAXimum** }
BURSt:INTernal:PERiod? [**MINimum** | **MAXimum**]
Set the internal period time of the BURST mode. MIN selects the lowest period time allowed for the burst function and MAX selects the highest period time allowed.
The PER? query returns the period setting in sec. for the burst function.

BURSt:PHASe { <***angle***> | **MINimum** | **MAXimum** }
BURSt:PHASe? [**MINimum** | **MAXimum**]
Set the phase of the BURST mode in degree. MIN selects the angle allowed for the burst function and MAX selects the highest angle allowed.
The PHAS? query returns the current setting of the phase angle.

BURSt:STATe { **OFF** | **ON** | 0 | 1 }
BURSt:STATe?
Set the burst state. Activates or deactivates the burst function.
The STAT? query returns the current status of the burst function (OFF, ON).

3.2.5 Arbitrary commands

with release 1.2

DATA { <***value***>,<***value***> | <***binary block***> }
Use this function to define a arbitrary waveform. The function generator can handle data either as floating point values or as binary values and is able to distinguish between these two formats automatically.

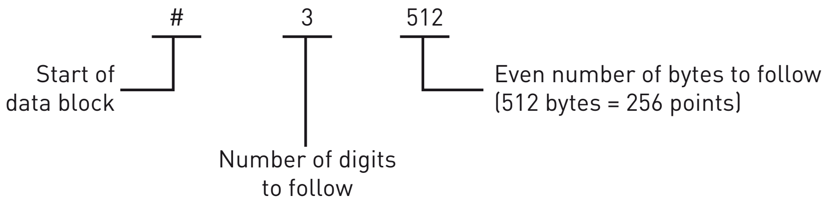
Create floating-point values from -1 to +1.
The values -1 and +1 correspond to the peak values of the waveform (if the offset is 0 volts).
For example, if you set the amplitude to 10 V_{pp} (0 V offset), "+1" corresponds to +5 V and "-1" corresponds to -5 V.

The DATA command will overwrite any previous generated waveform .
Use the DATA:COPY command to copy the waveform to non-volatile memory.

The following example shows how to use the DATA command to create seven arbitrary points using floating-point values:

DATA 1, .67, .33, 0, -.33, -.67, -1

In the binary block format, a block header precedes the waveform data.
The block header has the following format:


#
3
512
Start of data block
Even number of bytes to follow (512 bytes = 256 points)
Number of digits to follow

The function generator represents binary data as 16-bit integers, which are sent as two bytes. Therefore, the total number of bytes is always twice the number of data points in the waveform and must always be an even number.
For example, 512 bytes are required to download a waveform with 256 points.

DATA:COPY <*Arb-name*>

You can output a arbitrary waveform directly from the volatile memory or you can copy the waveform to non-volatile memory using the DATA:COPY command and a unique name for the arbitrary waveform crated. The arbitrary name may contain up to 10 characters.

DATA:DELete { <*Arb-name*> }
DATA:DELete:ALL

Deletes the selected user defined waveform.
DATA:DEL:ALL deletes all stored uder defined waveforms in the function generator.

DATA:SWAP { NORMal | SWAPped }
DATA:SWAP?

Invert a arbitrary waveform. Select whether the function generator provides the user defined waveform or a inverted version of this waveform . The SWAP? query returns the current status of this function (normal, swapped).

FUNCtion:ARBitrary { <*Arb-name*> }
FUNCtion:ARBitrary?

Select one of the built-in arbitrary waveforms or one of user-defined waveforms and load it into the internal memory. The ARB? Query returns the currently selected arbitrary waveform.

3.2.6 Trigger commands

Trigger commands will only effect the wobble and burst mode.

TRIGger
*TRG

with release 1.2

These commands are used to activate a internal trigger signal via remote control.

TRIGger:SOURce { IMMediate | EXTernal }
TRIGger:SOURce?

Set the trigger source. IMM selects the internal source for the trigger and EXT selects a external source as reference for the trigger. This external source needs to be connected to the front "trigger input" connector (TTL).
The SOUR? query returns the current setting of the trigger source (immediate, external).

OUTPut:TRIGger { OFF | ON | 0 | 1 }
OUTPut:TRIGger?

Acitvate or deactivate the trigger function.
The TRIG? query returns the current setting of the trigger (OFF,ON).

3.2.7 Save and recall of instrument settings

SAV** { **0** | **1** | **2** | **3** | **4** } ***Power off settings are stored at instrument setting 0
RCL** { **0** | **1** | **2** | **3** | **4** } ***Settings 1, 2, 3 und 4 are user defined settings. *with release 1.2*

Use this command to save or recall your preferred system settings.

The function generator has five storage locations in non-volatile memory to store the instrument settings. The function generator automatically uses location "0" to hold the state of the instrument at power down.

HAMEG® Instruments
A Rohde & Schwarz Company

Oscilloscopes

Spectrum Analyzer

Power Supplies

Modular System
Series 8000

Programmable Instruments
Series 8100

authorized dealer

www.hameg.com

DQS-Certification: DIN EN ISO 9001:2000
Reg.-Nr.: 071040 QM

HAMEG Instruments GmbH
Industriestraße 6
D-63533 Mainhausen
Tel +49 (0) 61 82 800-0
Fax +49 (0) 61 82 800-100
sales@hameg.com