SCPI Programming Guide

Product: AC Power Source - BP-xx-SS-x-xx

Document Release Date : 03/03/2022
Document Version : 1.0.1

Contents

Chapter 1

Introduction

The Standard Commands for Programmable Instruments (SCPI) defines a standard for syntax and commands for use in controlling electronic devices such as measurement devices and power supplies.

The objective of this document is to guide the programmer in interfacing the “BP-xx-SS-x-xx” – from here on indicated as “the product” or “the device” – by presenting the SCPI commands and data structures implemented in the product.

1.1 Communication Interfaces

The product can communicate using the SCPI protocol over two different communication interfaces: RS-232 and Ethernet. To use an interface, simply plug in the corresponding cable. Please note that only one interface at a time can be used. The set-up of communication interfaces and details about them are described in another document (SCPI Communication Set Up Guide).

1.2 Notation

This document applies to the “BP-xx-SS-x-xx”, regardless of the available power and SCPI-unrelated features. To maintain the text as generic as possible, the following placeholders have been used:

V_{min1} Minimum voltage of the first voltage range.

V_{max1} Maximum voltage of the first voltage range.

V_{min2} Minimum voltage of the second voltage range.

V_{max2} Maximum voltage of the second voltage range.

I_{min} Minimum current.

I_{max1} Maximum current of the first voltage range.

I_{max2} Maximum current of the second voltage range.

f_{q1} First quartzed frequency.

f_{q2} Second quartzed frequency.

$f_{var-min}$ Minimum frequency of the variable frequencies range.

$f_{var-max}$ Maximum frequency of the variable frequencies range.

Please refer to the product manual to know the numeric values of the placeholders presented above.

Chapter 2

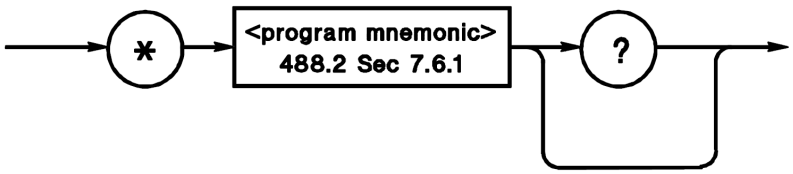
SCPI Basics

This chapter presents briefly the basic terms and notions of the SCPI syntax.

Command Tree SCPI commands are based on a hierarchical structure, allowing the same instrument-control header (or keyword) to be used many times for different purposes [99, Vol. 1, Section 6.2.2]. This hierarchical structure is also known as command tree. The top-level headers identify a specific subsystem.

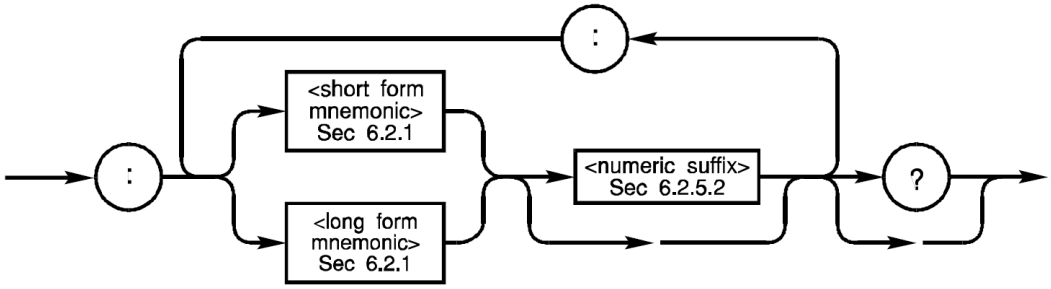
Program Headers - Keywords Program headers are keywords that identify a command and consist of two distinct types, common command headers and instrument-control headers [SCPI, Vol. 1, Section 6].

Common Commands - Mandated Commands Common command and query headers are specified in IEEE 488.2 [SCPI, Vol. 1, Section 6.1]. Their syntax is represented in the following scheme:



Common commands – also known as Mandated Commands – are implemented by all SCPI-compliant instruments [SCPI, Vol. 1, Section 4.1.1]. The <program mnemonic> is 3 characters long (e.g. “*CLS” or “*ESE”).

Instrument-Control Headers Instrument-control headers are used for all other instrument commands [SCPI, Vol. 1, Section 6.2]. Their syntax is represented in the following scheme:



That is,

- If a command is made of more than one header, such headers are separated with a colon.
- A program header must be either exactly equal to the <short form> or to the <long form>.

  - The <short form> is usually made of 3 or 4 characters (e.g. “OUTP” or “MEAS”).
  - The <long form> includes the <short form> and is made of up to 12 characters, (e.g. “OUTPut” or “MEASure”).
  - The <short form> is indicated with uppercase characters inside the <long form>.
- The numeric suffix is a number attached to a header used to differentiate blocks [SCPI, Vol. 2, Section 2.7.2]. Suffixes are not supported by all devices, and they should not be confused with parameters.

Parameters / <PROGRAM DATA> A Parameter (or <PROGRAM DATA>) is a functional element that adds information to the program header [92, Section 7.7]. For example, in the command “FREQuency 50”, “FREQuency” is the program header and “50” is the parameter. Parameters must be separated from the program header(s) by exactly one space.

Program Message Terminator Program Messages are read until the program message terminator is met. The terminator is NL, defined as a single ASCII-encoded byte whose hexadecimal value is 0A [IEEE 488.2, Section 7.5.2].

Program Message Unit A Program Message Unit is a single command, programming data, or query. (e.g. “SOURce:FREQuency 50” or “SYSTem:ERRor?”)

Query The query form of a command is generated by appending a question mark to its last keyword. However, not all commands have a query form, and some commands exist only in the query form [SCPI, Vol. 1, Section 6.2.3]. Replies are sent only for queries, and they do not include the command headers: typically they are just numeric.

Compound Commands Program Message Units can be concatenated in a single Compound Command with the semicolon character [SCPI, Vol. 1, Section 6.2.4][IEEE 488.2, Annex A.1.1].

In brief, the following parsing rules apply:

- The first command is always referenced to the root node. A leading colon is not required.
- If a non-first command is not prepended with the colon, it is referenced to the same tree level as the previous command in a message unit.
- If a non-first command is prepended with the colon, it refers to the root node.
- Common Commands do not belong to a tree, so they must not be prepended with the colon, even if they are the non-first command (e.g “VOLT 72;*OPC?” is correct).
- Common Commands do not alter the application of prefixing rules: their presence do not reset the position of the parser on the command tree.
- If a Compound Command includes multiple queries, replies are returned in a single message, separated with a semicolon [IEEE 488.2, Section 8.4].

Optional Headers Headers written in square brackets are optional, so their presence does not modify the command / query. Such nodes are also called default nodes [SCPI, Vol. 1, Section 5.1]. For example, if a command syntax is indicated as “OUTPut[:STATe]?”, the queries “OUTPut?” and “OUTPut:STATe?” are equivalent. Note that square brackets must not be added to the actual command / query.

Program data and replies formats The following table summarizes the notation used for parameters and replies formats:

Symbol	Description	Example
<NR1>	Base 10 integer number, with or without sign.	123
<NR2>	Base 10 decimal number, with or without sign.	12.34
<NR3>	Base 10 mantissa-exponent number, with or without sign.	12e2
<CHARACTER PROGRAM DATA>	Parameter expressed as alphanumeric string [IEEE 488.2, Section 7.7.1].	MIN, MAX, DEFault, UP, DOWN, NAN, INF, NINF
<DECIMAL NUMERIC PROGRAM DATA> or <NRf>	NR1, NR2, and NR3 [IEEE 488.2, Section 7.7.2].	-
<numeric_value>	<NRf> and <CHARACTER PROGRAM DATA> [SCPI, Vol. 1, Section 7.2.1]	-
<Boolean>	ON|OFF|<NRf> [SCPI, Vol. 1, Section 7.3]	-
<NONDECIMAL NUMERIC PROGRAM DATA>	Number expressed in a base other than 10 [IEEE 488.2, Section 7.7.4]. Includes binary (#B), octal (#Q) and hexadecimal (#H).	#B1010 = 10 (base 10) #Q34 = 28 (base 10) #H5D = 93 (base 10)

Suffix Program Data A Suffix Program Data is used to express units of measure and multipliers [IEEE 488.2, Section 7.7.3].

- Multipliers (e.g. K, M, G) are optional (more on them in table [IEEE 488.2, Table 7.2]).
- When not specified in a command, the default unit of measure and multiplier (specified in the manual) are applied.
- Units of measure and multipliers are not added to query replies [SCPI, Vol. 1, Section 7.5].

MINimum and MAXimum As stated in [SCPI, Vol. 1, Section 7.2.1.2], MINimum and MAXimum values (for commands that support them) can be queried by appending “MINimum” and “MAXimum” to the query itself. For example, to get the value of “MAXimum” for the “SOURce:VOLTage” command, use:

```
Query: SOURce:VOLTage? MAXimum
Reply: 300
```

Case sensitivity As stated in [SCPI, Vol. 1, Section 6], SCPI is not case-sensitive: there is no difference between uppercase and lowercase characters.

Chapter 3

Status Reporting

The first section of this chapter (3.1) presents the Registers Model, which is necessary to better understand the data structures – such as registers and queues – depicted in successive sections of this chapter. This status reporting system is defined as mandatory for compliance in [SCPI, Vol. 1, Section 4.1.3.2].

3.1 The Registers Model

The Register Model allows to summarize multiple events in a single summary message in the Status Byte Register [IEEE 488.2, Section 11.4.2]. The Register Model defines the three register types presented in the next sub-sections.

3.1.1 Condition Registers

[IEEE 488.2, Section 11.4.2.1] A condition is a TRUE/FALSE device state reflected in a Condition Register's bit. Features:

- It is possible to read a Condition Register with device-specific commands without altering its content.
- No device-specific commands can write directly to a Condition Register (they can only reflect in real-time a device condition).
- No device-specific commands can clear directly a Condition Register.

3.1.2 Event Registers

[IEEE 488.2, Section 11.4.2.2] Event Registers capture changes in conditions. Each bit in an Event Register corresponds to a bit in the associated Condition Register. An Event Register's bit becomes TRUE when the associated Condition Register's bit changes. Features:

- It is possible to read an Event Register with device-specific commands. Reading the register clears it.
- No device-specific commands can write directly to an Event Register.
- Event Registers can be cleared with specific commands.

These features ensure that condition changes (events) do not go unnoticed.

3.1.3 Event Enable Registers

[IEEE 488.2, Section 11.4.2.3] Event Enable Registers select which Event Register's bits can cause a TRUE Summary message for the Registers group. Each bit in an Event Enable Register corresponds to a bit in the associated Event Register. Event Enable Registers can be read, written and cleared with device-specific commands.

3.2 The Value of a Register

The value of a register is represented with an integer number: the <NR1> value expressed in base 2 (binary) represents the bits of the register. For example, 130 (decimal) = 10000010 (binary): Bit 7 is TRUE, Bit 7 is FALSE, . . . , Bit 1 is TRUE, Bit 0 is False.

Note that throughout this document, “Bit 0” is always the LSB (Least Significant Bit).

3.3 Standard Event Status Registers Group

No “Condition Register” is defined for the Standard Event Status Registers group, in fact:

- The Standard Event Status Register (SESR) is an “Event Register” [IEEE 488.2, Section 11.5.1.2.1]
- The Standard Event Status Enable Register (SESER) is an “Event Enable Register”.

3.3.1 Standard Event Status Register (SESR)

This register is required by all devices, and its bits are assigned to specific events in [IEEE 488.2, Section 11.5.1]:

Bit 0 - OPC (Operation Complete) Generated in response to the *OPC command, it indicates that the device has completed all selected pending operations.

Bit 1 - RQC (Request Control) It indicates to the controller that the device is requesting the permission to become the active IEEE 488.1 controller-in-charge.

Bit 2 - QYE (Query Error) A Query Error has occurred.

Bit 3 - DDE (Device Specific Error) A Device Specific Error has occurred.

Bit 4 - E (Execution Error) An Execution Error has occurred.

Bit 5 - CME (Command Error) A Command Error has occurred.

Bit 6 - URQ (User Request) One of a set of local controls (defined by the device designer as a User Request control) has been activated.

Bit 7 - PON (Power ON) An off-to-on transition has occurred in the device’s power supply.

Bits 8-15: Reserved for possible future use by IEEE. These bits are reported as zeros.

The product uses only the following SESR bits: OPC, DDE, E, CME, PON.

3.4 OPERation Status Registers Group

The OPERation status registers contain conditions which are part of the instrument’s normal operation. SCPI defines each of these bits as follows [SCPI, Vol. 1, Section 9.3]:

Bit 0 - CALibrating The instrument is performing a calibration.

Bit 1 - SETTling The instrument is waiting for signals to stabilize in order to begin measurements.

Bit 2 - RANGing The instrument is changing its range.

Bit 3 - SWEeping A sweep is in progress.

Bit 4 - MEASuring The instrument is actively measuring.

Bit 5 - Waiting for TRIG The instrument is in a “wait for trigger” state of the trigger model.

Bit 6 - Waiting for ARM The instrument is in a “wait for arm” state of the trigger model.

Bit 7 - CORRecting The instrument is performing a correction.

Bits 8 - 12 Available to the product designer.

Bit 13 - INSTrument Summary Bit One of n multiple logical instruments is reporting OPERational status.

Bit 14 - PROGram running A user-defined programming is in the run state.

Bit 15 Always zero

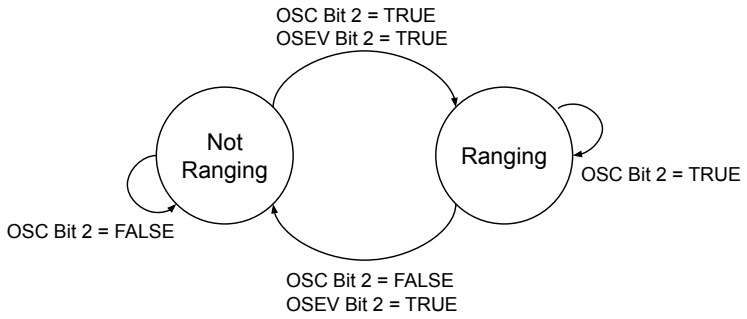
In this group there are a Condition, an Event and an Event Enable Register:

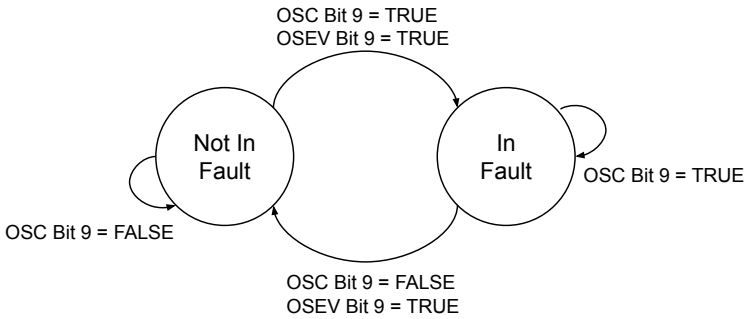
- Operation Status Condition Register (OSC)
- Operation Status Event Register (OSEV)
- Operation Status Enable Register (OSEN)

The product uses only the following OPER bits:

- Bit 2: Set while changing voltage range.
- Bit 9: Set when entering the FAULT condition.

The following schemes represent how bits are changed with respect to states and transitions.





3.5 QUEStionable Status Registers Group

The QUEStionable status register set contains bits which give an indication of the quality of various aspects of the signal. A bit set in the Condition Register indicates that the data being acquired or generated is of questionable quality [SCPI, Vol. 1, Section 9.4].
In this group there are a Condition, an Event and an Event Enable Register:

- Questionable Status Condition Register (QSC)
- Questionable Status Event Register (QSEV)
- Questionable Status Enable Register (QSEN)

> The product does not measure the quality of signals, so QSC and QSEV bits are never set TRUE.

3.6 Error/Event Queue

[SCPI, Vol. 2, Section 5.1.7] As errors and events are detected, they are placed in a finite size FIFO (First In, First Out) queue. If the queue overflows, the last error/event in the queue is replaced with:

```
-350, "Queue overflow"
```

Any time the queue overflows, the least recent errors/events remain in the queue, and the most recent error/event is discarded. Reading an error/event from the head of the queue removes that error/event from the queue, and opens a position at the tail of the queue for a new error/event, if one is subsequently detected. When all errors/events have been read from the queue, further error/event queries return:

```
0, ‘‘No error’’
```

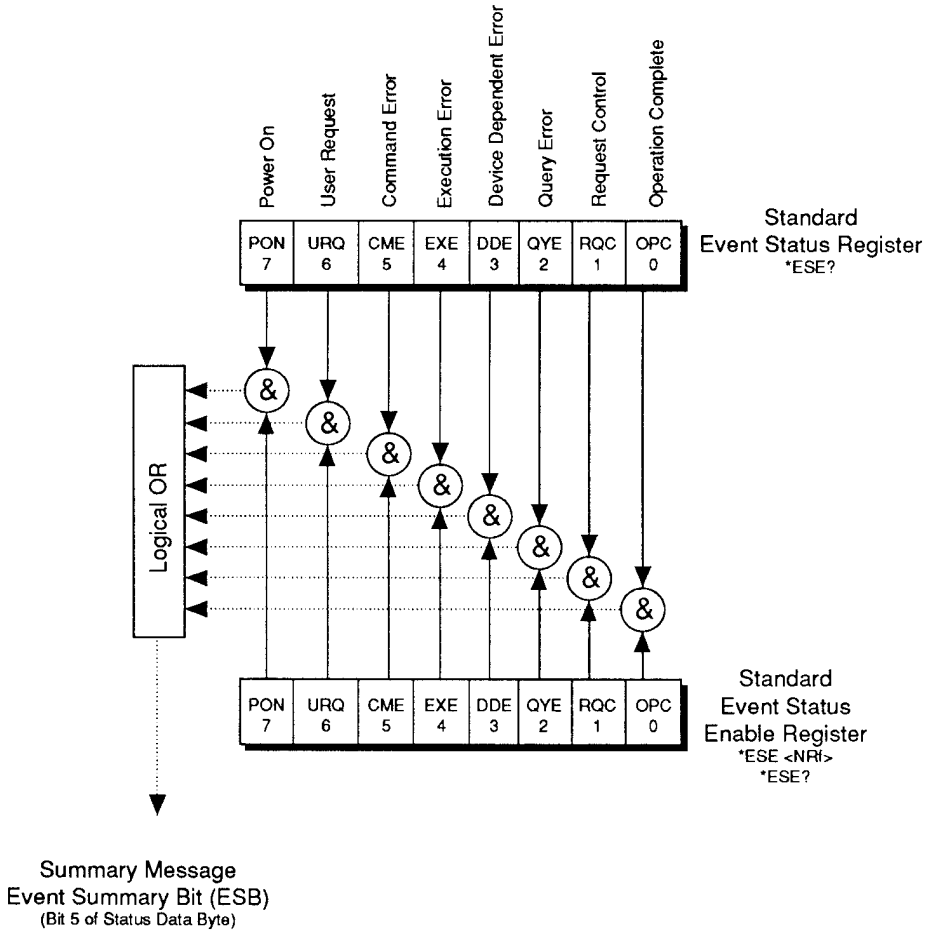
The Error Codes and Descriptions used by the device are reported in a dedicated chapter (6).

> The product implements an Error Queue of length 8.

3.7 Status Byte Register (SBR)

[IEEE 488.2, Section 11.1.1] The Status Byte Register is composed of seven single-bit summary-messages: each summary message summarizes an overlaying Status Data Structure. The summary message for a register group is the “Logical OR” of its bits, where each bit is given by the “Logical AND” between the Event Register and the Event Enable Register of the group.

The following scheme is an example of computation of the summary bit for the Standard Event Status Registers group:



The SBR bits are defined as follows:

Bit 0 - NU (Not Used)

Bit 1 - WAR (Warning Status Register) Summary bit for the Warning Status Register group.

Bit 2 - ERR (Error Event/Queue) TRUE if data is present in the Error queue, FALSE otherwise [IEEE 488.2, Section 11.5.1].

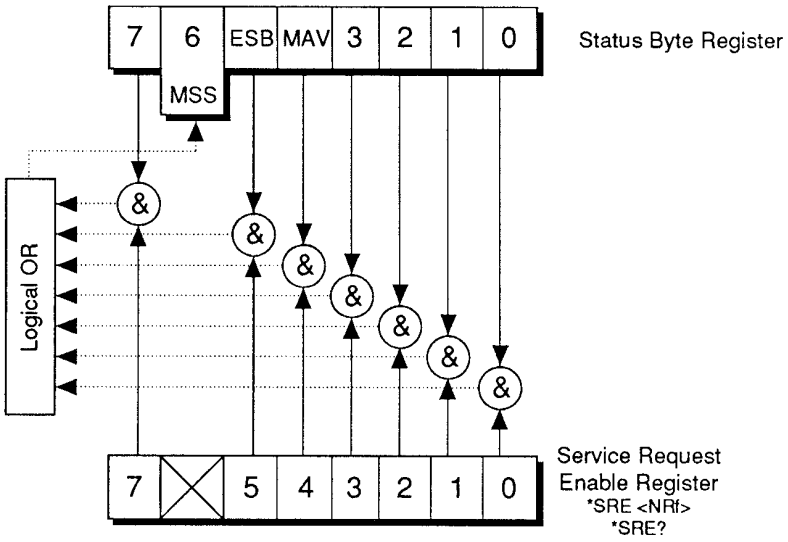
Bit 3 - QUES (Questionable Status Register) Summary bit for the Questionable Status Register group.

Bit 4 - MAV (Message Available) TRUE when there is data in the Output Queue waiting to be read, FALSE otherwise [IEEE 488.2, Section 11.2.1.2].

Bit 5 - ESB (Event Summary Bit) Summary bit for the Standard Event Status Register group [IEEE 488.2, Section 11.2.1.1]

Bit 6 - MSS (Master Summary Status) Summary of the Status Byte Register (SBR) and Service Request Enable Register (SRER). Its computation is represented in the image below [IEEE 488.2, Section 11.2.2.3].

Bit 7 - OPER (Operation Status Register) Summary bit for the Operation Status Register Group.



WAR, QUES, and MAV are not set because the underlying structures are not implemented.

3.8 Service Request Enable Register (SRER)

The SRER is an 8 bit register that enables Status Byte Register (SBR) bits to be reflected in the MSS (Master Summary Status) bit [IEEE 488.2, Section 11.3.2]. Its usage is presented in the previous image.

Chapter 4

Commands List

This chapter presents the device-specific commands implemented by the product, divided in subsystems. Since the product is a Power Source, the command tree is mostly based on [SCPI, Vol. 4, Chapter 7].

4.1 Output Subsystem

The Output Subsystem deals with the Power Source's output power.

Keyword	Parameter Form	Reference
OUTPut		SCPI, Vol. 2, Chapter 15
[:STATe]	<Boolean>	SCPI, Vol. 2, Section 15.12

4.1.1 OUTPut[:STATe]

The machine inverter can be turned ON and OFF with the following command:

```
Command Syntax: OUTPut[:STATe] <Boolean>
```

The state of the machine inverter can be read with the following query:

```
Query Syntax: OUTPut[:STATe]?
Reply: 0|1
```

As specified in [SCPI, Vol. 1, Section 7.3], the reply is "0" (meaning OFF) or "1" (meaning ON).

4.2 Source Subsystem

The source subsystem deals with the output (voltage - current - frequency) of the product.

Keyword	Parameter Form	Reference
[SOURce:]		SCPI, Vol. 2, Chapter 19
FREQuency		SCPI, Vol. 2, Section 19.9
[:IMMediate]	<numeric_value>	MCB
:VARiable	<Boolean>	MCB
CURRent		SCPI, Vol. 2, Section 19.5
[:LEVel]		SCPI, Vol. 2, Section 19.5.4
[:IMMediate]		SCPI, Vol. 2, Section 19.5.4.1
[:AMPLitude]		SCPI, Vol. 2, Section 19.5.4.1.1
[:AC]	<numeric_value>	MCB
VOLTage		SCPI, Vol. 2, Section 19.23
[:LEVel]		SCPI, Vol. 2, Section 19.23.4
[:IMMediate]		SCPI, Vol. 2, Section 19.23.4.1
[:AMPLitude]		SCPI, Vol. 2, Section 19.23.4.1.1
[:AC]	<numeric_value>	MCB
:RANGe	<numeric_value>	SCPI, Vol. 2, Section 19.23.9

4.2.1 [SOURce:]FREQuency[:IMMediate]

The frequency of the output waveform is set with the following command:

```
Command Syntax: [SOURce:]FREQuency[:IMMediate] <numeric_value>
```

- If variable frequency is enabled, accepted values are numbers in the range from [$f_{var-min}$-$f_{var-max}$] Hz. Otherwise, accepted values are f_{q1}, f_{q2}.
- Accepted <CHARACTER PROGRAM DATA>: "MINimum" and "MAXimum".
- Accepted unit of measure: "Hz"

The frequency of the output waveform can be read with the following query:

```
Query Syntax: [SOURce:]FREQuency[:IMMediate]?
Reply: <NR2>
```

Note that MINimum and MAXimum values can be queried – and they change accordingly to variable frequency enabledness.

4.2.2 [SOURce:]FREQuency:VARiable

Variable frequency be turned ON and OFF with the following command:

```
Command Syntax: [SOURce:]FREQuency:VARiable <Boolean>
```

The variable frequency enabledness can be read with the following query:

```
Query Syntax: [SOURce:]FREQuency:VARiable?
Reply: 0|1
```

The reply is “0” (meaning OFF, disabled) or “1” (meaning ON, enabled).

4.2.3 [SOURce:]CURRent[:LEVel][:IMMediate][:AMPLitude][:AC]

This command sets the AC RMS maximum output current of the product:

```
Command Syntax: [SOURce:]CURRent[:LEVel][:IMMediate][:AMPLitude][:AC] <numeric_value>
```

- Accepted value ranges: [I_{min}, I_{max1}] A or [I_{min}, I_{max2}] A depending on the selected voltage range.
- Accepted <CHARACTER PROGRAM DATA>: "MINimum" and "MAXimum".
- Accepted units of measure: "A" and "ARMS"

The AC RMS current sourced by the product can be read with :

```
Query Syntax: [SOURce:]CURRent[:LEVel][:IMMediate][:AMPLitude][:AC]?
Reply: <NR2>
```

The <NR2> reply is in the range [I_{min}, I_{max1}] A or [I_{min}, I_{max2}] A depending on the selected voltage range.
Note that MINimum and MAXimum values can be queried – and they change accordingly to the selected voltage range.

4.2.4 [SOURce:]VOLTage[:LEVel][:IMMediate][:AMPLitude][:AC]

This command sets the AC RMS output voltage of the product:

```
Command Syntax: [SOURce:]VOLTage[:LEVel][:IMMediate][:AMPLitude][:AC] <numeric_value>
```

- Accepted value ranges: [V_{min1}, V_{max1}] V or [V_{min2}, V_{max2}] V, depending on the selected voltage range.
- Accepted <CHARACTER PROGRAM DATA>: "MINimum" and "MAXimum".
- Accepted units of measure: "V" and "VRMS"

The AC RMS voltage sourced by the product can be read with:

```
Query Syntax: [SOURce:]VOLTage[:LEVel][:IMMediate][:AMPLitude][:AC]?
Reply: <NR1>
```

The <NR1> reply is in the range [V_{min1}, V_{max1}] V or [V_{min2}, V_{max2}] V, depending on the selected voltage range.
Note that MINimum and MAXimum values can be queried – and they change accordingly to the selected voltage range.

4.2.5 [SOURce:]VOLTage:RANGe

This command sets the voltage range of the product:

```
Command Syntax: [SOURce:]VOLTage:RANGe <numeric_value>
```

- Accepted values: "V_{max1}V" (the product works in the Voltage Range [V_{min1}, V_{max1}] V) and "V_{max2}V" (the product works in the Voltage Range [V_{min2}, V_{max2}] V)
- Accepted <CHARACTER PROGRAM DATA>: "MINimum" (equivalent to V_{max1}V) and "MAXimum" (equivalent to V_{max2}V)
- Accepted units of measure: "V" and "VRMS"

Note that changing the Voltage Range turns off the inverter and sets the sourced voltage to the minimum voltage of the new range.
The Voltage Range can be read with:

```
Query Syntax: [SOURce:]VOLTage:RANGe?
Reply: <NR1>
```

Note that MINimum and MAXimum values can be queried.

4.3 Measure Subsystem

This query-only subsystem returns the current and voltage measured at the Power Source's output terminals.

Keyword	Parameter Form	Reference
MEASure		SCPI, Vol. 2, Section 3.4
[:SCALar]		SCPI, Vol. 4, Section 7.2.1.1
:CURRent		SCPI, Vol. 4, Section 7.2.1.1
[:AC?]		SCPI, Vol. 4, Section 7.2.1.1
:VOLTage		SCPI, Vol. 4, Section 7.2.1.1
[:AC?]		SCPI, Vol. 4, Section 7.2.1.1

4.3.1 MEASure[:SCALar]:CURRent[:AC]?

This query returns the RMS AC current measured at the output terminal of the Power Source:

```
Query Syntax: MEASure[:SCALar]:CURRent[:AC]? [<expected_value>[,<resolution>]]
Reply: <NR2>
```

In compliance with [SCPI, Vol. 4, Section 7.2.1.1], <expected_value> and <resolution> parameters are included solely for compatibility with the Command Reference: they are accepted but ignored by the device.

The reply is expressed in Ampere (A).

4.3.2 MEASure[:SCALar]:VOLTage[:AC]?

This query returns the AC RMS voltage measured at the output terminal of the Power Source:

```
Query Syntax: MEASure[:SCALar]:VOLTage[:AC]? [<expected_value>[,<resolution>]]
Reply: <NR1>
```

In compliance with [SCPI, Vol. 4, Section 7.2.1.1], <expected_value> and <resolution> parameters are included solely for compatibility with the Command Reference: they are accepted but ignored by the device.

The reply is expressed in Volt (V).

4.4 SCPI Required commands

This section presents the commands defined as mandatory in [SCPI, Vol. 1, Section 4.2.1], plus others belonging to the same subsystems.

Keyword	Parameter Form	Reference
SYSTem		SCPI, Vol. 2, Chapter 21
:ERRor		SCPI, Vol. 2, Section 21.8
[:NEXT]?		SCPI, Vol. 2, Section 21.8
:VERSion?		SCPI, Vol. 2, Section 21.21
:RESet		MCB
STATus		SCPI, Vol. 2, Chapter 20
:OPERation		SCPI, Vol. 2, Section 20.1
[:EVENt]?		SCPI, Vol. 2, Section 20.1.4
:CONDition?		SCPI, Vol. 2, Section 20.1.2
:ENABle	<NRf>|<non-decimal_numeric>	SCPI, Vol. 2, Section 20.1.3
:QUEStionable		SCPI, Vol. 2, Section 20.3
[:EVENt]?		SCPI, Vol. 2, Section 20.3.4
:CONDition?		SCPI, Vol. 2, Section 20.3.2
:ENABle	<NRf>|<non-decimal_numeric>	SCPI, Vol. 2, Section 20.3.3
:PRESet		SCPI, Vol. 2, Section 20.2

4.4.1 SYSTem:ERRor[:NEXT]?

This query pops an element from the Error Queue:

```
Query Syntax: SYStem:ERRor[:NEXT]?
Reply: <Error/event_number>, ‘‘Error/event_description[;Device-dependent Info]’’
```

The <Error/event_number> is a unique integer in the range [-32768, 32767]. The second parameter of the response is a quoted string containing an <Error/event_description> followed by an optional <Device-dependent info> text. Each <Error/event_number> has a unique and fixed <Error/event_description> associated with it. The Error Codes and Descriptions used by the device are reported in a dedicated chapter (6).

4.4.2 SYSTem:VERSion?

This query returns a <NR2> formatted numeric value corresponding to the SCPI version number for which the instrument complies:

```
Query Syntax: SYSTem:VERSion?
Reply: <NR2>
```

The response has the form YYYY.V where the Ys represent the year-version and the V represents the revision number for that year. The product replies with “1999.0”.

4.4.3 SYSTem:RESet

This command resets TCP/IP settings to default values and issues a reboot of the MCU (MicroController Unit):

```
Query Syntax: SYSTem:RESet
```

While executing the command, some text may appear on the serial connection: it shall be ignored by the user. After the execution of the command, registers are cleared and the product is in “Reset Status”, which is defined in (5).

4.4.4 STATus:OPERation[:EVENt]?

This query returns the value of the “Operation Status Event Register” (OSEV):

```
Query Syntax: STATus:OPERation:[EVENt]?
Reply: <NR1 NUMERIC RESPONSE DATA>
```

The <NR1 NUMERIC RESPONSE DATA> is in the range [0, 32767].
Note that reading an Event Register clears it.

4.4.5 STATus:OPERation:CONDition?

This query returns the value of the “Operation Status Condition Register” (OSC):

```
Query Syntax:STATus:OPERation:CONDition?
Reply: <NR1 NUMERIC RESPONSE DATA>
```

The <NR1 NUMERIC RESPONSE DATA> is in the range [0, 32767].

4.4.6 STATus:OPERation:ENABle

This command sets the value of the “Operation Status Enable Register” (OSEN):

```
Command Syntax: STATus:OPERation:ENABle <NRf>|<non-decimal numeric>
```

The command accepts parameter values of either format in the range 0 through 65535 (decimal) without error.
The query form reads the value of the register:

```
Query Syntax:STATus:OPERation:ENABle?
Reply: <NR1 NUMERIC RESPONSE DATA>
```

The <NR1 NUMERIC RESPONSE DATA> is in the range [0, 32767].

4.4.7 STATus:QUEStionable[:EVENt]?

This query returns the content of the “Questionable Status Event Register” (QSEV):

```
Query Syntax: STATus:QUEStionable[:EVENt]?
Reply: <NR1 NUMERIC RESPONSE DATA>
```

The <NR1 NUMERIC RESPONSE DATA> is in the range [0, 32767].
Note that reading an Event Register clears it.

4.4.8 STATus:QUEStionable:CONDition?

This query returns the value of the “Questionable Status Condition Register” (QSC):

```
Query Syntax: STATus:QUEStionable:CONDition?
Reply: <NR1 NUMERIC RESPONSE DATA>
```

The <NR1 NUMERIC RESPONSE DATA> is in the range [0, 32767].

4.4.9 STATus:QUEStionable:ENABle

This command sets the value of the “Questionable Status Enable Register” (QSEN):

```
Command Syntax: STATus:QUEStionable:ENABle <NRf>|<non-decimal numeric>
```

The command accepts parameter values of either format in the range 0 through 65535 (decimal) without error. The query form reads the value of the register:

```
Query Syntax: STATus:QUEStionable:ENABle?
Reply: <NR1 NUMERIC RESPONSE DATA>
```

The <NR1 NUMERIC RESPONSE DATA> is in the range [0, 32767].

4.4.10 STATus:PRESet

This command configures the SCPI and device-dependent status data structures such that device-dependent events are reported at a higher level through the status-reporting mechanism. When issued on the device, the following registers are reset to 0:

- Questionable Status Enable register (QSEN)
- Operation Status Enable registers (OSEN)

```
Command Syntax: STATus:PRESet
```

4.5 IEEE 488.2 Mandated Commands

This section presents the Common Commands defined as mandatory in [SCPI, Vol. 1, Section 4.1.1].

Mnemonic	Name	488.2 Section
*CLS	Clear Status Command	10.3
*ESE	Standard Event Status Enable Command	10.10
*ESE?	Standard Event Status Enable Query	10.11
*ESR?	Standard Event Status Register Query	10.12
*IDN?	Identification Query	10.14
*OPC	Operation Complete Command	10.18
*OPC?	Operation Complete Query	10.19
*RST	Reset Command	10.32
*SRE	Service Request Enable Command	10.34
*SRE?	Service Request Enable Query	10.35
*STB?	Read Status Byte Query	10.36
*TST?	Self-Test Query	10.38
*WAI	Wait-to-Continue Command	10.39

4.5.1 *CLS - Clear Status Command

This command clears all Status data structures:

- SESR
- OPERation Status Register (OSC, OSEV)
- QUEStionable Status Register (QSC, QSEV)
- Error/Event Queue

```
Command Syntax: *CLS
```

4.5.2 *ESE - Standard Event Status Enable Command

This command sets the value of the “Standard Event Status Enable Register” (SESER):

```
Command Syntax: *ESE <DECIMAL NUMERIC PROGRAM DATA>
```

The <DECIMAL NUMERIC PROGRAM DATA>, after the eventual conversion to <NR1>, should have a value in the range [0, 255].

4.5.3 *ESE? - Standard Event Status Enable Query

This query returns the value of the “Standard Event Status Enable Register” (SESER):

```
Query Syntax: *ESE?
Reply: <NR1 NUMERIC RESPONSE DATA>
```

The <NR1 NUMERIC RESPONSE DATA> is in the range [0, 255].

4.5.4 *ESR? - Standard Event Status Register

This query returns the value of the “Standard Event Status Register” (SESR):

```
Query Syntax: *ESR?
Reply: <NR1 NUMERIC RESPONSE DATA>
```

The <NR1 NUMERIC RESPONSE DATA> is in the range [0, 255].
Note that reading an Event Register clears it.

4.5.5 *IDN? - Identification Query

This query returns the product identification string, which is composed of the following comma-separated alphanumeric fields:

1. Manufacturer
2. Model
3. Serial Number
4. Firmware version

```
Query Syntax: *IDN?
Reply:<Manufacturer>,<Model>,<Serial Number>,<Firmware version>
```

The length of the reply is less or equal to 72 characters.

4.5.6 *OPC - Operation Complete Command

This command causes the device to generate the operation complete message in the SESR when all pending selected device operations have been finished. However, since only sequential commands are applied, this command sets to TRUE the OPC bit in the SESR [SCPI, Vol. 1, Section 4.1.3.3].

```
Command Syntax: *OPC
```

4.5.7 *OPC? - Operation Complete Query

This query places an ASCII character “1” into the device’s Output Queue when all pending selected device operations have been finished. However, since only sequential commands are applied, this query returns “1” [SCPI, Vol. 1, Section 4.1.3.4], without interacting with the SESR [IEEE 488.2, Section 12.5.3]:

```
Query Syntax: *OPC?
Reply: <NR1 NUMERIC RESPONSE DATA>
```

The <NR1 NUMERIC RESPONSE DATA> is a single ASCII-encoded byte for “1”.

4.5.8 *RST - Reset Command

This command performs a device reset. In other words, it sets the device-specific functions to a known state that is independent of the past-use history of the device.

```
Command Syntax: *RST
```

The machine is set in “Reset Status”, as defined later in the document.

4.5.9 *SRE - Service Request Enable Command

This command sets the value of the “Service Request Enable Register” (SRER):

```
Command Syntax: *SRE <DECIMAL NUMERIC PROGRAM DATA>
```

The <DECIMAL NUMERIC PROGRAM DATA>, after the eventual conversion to <NR1>, should have a value in the range [0, 255].

4.5.10 *SRE? - Service Request Enable Query

This query returns the value of the “Service Request Enable Register” (SRER):

```
Query Syntax: *SRE?
Reply: <NR1 NUMERIC RESPONSE DATA>
```

The <NR1 NUMERIC RESPONSE DATA> is in the range [0, 255].

4.5.11 *STB? - Read Status Byte Query

This query returns the value of the “Status Byte register” (SBR):

```
Query Syntax: *STB?
Reply: <NR1 NUMERIC RESPONSE DATA>
```

The <NR1 NUMERIC RESPONSE DATA> is in the range [0, 255].

4.5.12 *TST? - Self-Test Query

The self-test query causes an internal self-test and places a response into the Output Queue indicating whether the device completed the self-test without any detected errors. Optionally, information on why the self-test was not completed may be contained in the response.

```
Query Syntax: *TST?
Reply: <NR1 NUMERIC RESPONSE DATA>
```

The product does not implement a self-test routine, so the reply is always “0”.

4.5.13 *WAI - Wait-to-Continue Command

The Wait-to-Continue command prevents the device from executing any further commands or queries until the no-operation-pending flag is TRUE. However, since the product implements only sequential commands, such flag is always TRUE and this command does nothing [SCPI, Vol. 1, Section 4.1.3.3]:

```
Command Syntax: *WAI
```

Chapter 5

Reset Status

The product is considered in “Reset Status” when all the following conditions apply:

Inverter: OFF
Output voltage AC: V_{max2}V
Voltage range: [V_{min2}-V_{max2}] V
Output frequency: f_{q1}Hz
Variable frequency enabled: OFF
Variable frequency: $f_{var-min}$Hz
Maximum output current AC: I_{min}A

The product is automatically set to “Reset Status”:

- At start
- After the “*RST” command
- After the “SYSTem:RESet” command

Chapter 6

Error/Event Codes

6.1 Errors Taxonomy

SCPI defines error numbers and descriptions, and groups them by type. Each group of error has a corresponding Error Number range [SCPI, Vol. 2, Section 21.8.2]. The following table reports briefly the defined error types:

Error Type	Cause	Range
Command Error	A syntax error has been detected by the parser.	[-199, -100]
Execution Error	Either: • A <PROGRAM DATA> element following a header was evaluated by the device as outside its legal input range or is otherwise inconsistent with the device's capabilities. • Another valid program message could not be properly executed due to some device condition.	[-299, -200]
Device Specific Error	Some device operations did not properly complete, possibly due to an abnormal hardware or firmware condition	[-399, -300] and [1, 32767]
Query Error	Either: • An attempt is being made to read data from the Output Queue when no output is either present or pending • Data in the Output Queue has been lost.	[-499, -400]

6.2 Errors Issued by the Product

The product may place the following errors in the Error Queue:

Error / Event Number	Error / Event String	Explanation/Description
0	No error	The queue is empty. Every error/event in the queue has been read or the queue was purposely cleared by power-on, *CLS, etc.
-100	Command error	Generic syntax error.
-104	Data type error	The parser recognized a data element different from one allowed.
-108	Parameter not allowed	More parameters were received than expected for the header.
-109	Missing parameter	Fewer parameters were received than required for the header.
-112	Program mnemonic too long	The header contains more than twelve characters.
-113	Undefined header	The header is syntactically correct, but it is undefined for this specific device.
-200	Execution error	Generic execution error.
-220	Parameter error	Generic parameter error.
-222	Data out of range	A legal program data element was parsed but could not be executed because the value is outside the legal range defined by the device.
-224	Illegal parameter value	Used where an exact value, from a list of possibles, was expected.
-350	Queue overflow	The Error Queue is full.

Bibliography

[92] "IEEE Standard Codes, Formats, Protocols, and Common Commands for Use With IEEE Std 488.1-1987, IEEE Standard Digital Interface for Programmable Instrumentation". In: Dec. 1992. DOI: 10.1109/IEEESTD.1992.114468.

[99] "Standard Commands for Programmable Instruments (SCPI)". In: 1999. URL: https://www.ivifoundation.org/docs/scpi-99.pdf.

Feedback

For any comment, question, suggestion or bug report about our SCPI documentation and implementation, please contact us at scpi@mcbelectronics.it.