

# PACE

## Pressure Automated Calibration Equipment SCPI Remote Communications Manual

PACE1000



PACE5000



PACE6000





# Introduction

This technical manual provides SCPI protocol instructions for the remote control of the PACE Series indicators and controllers.

## Safety

- The manufacturer has designed this product to be safe when operated using the procedures detailed in this manual. Do not use this product for any other purpose than that stated.
- This publication contains operating and safety instructions that must be followed to make sure of safe operation and to maintain the equipment in a safe condition. The safety instructions are either warnings or cautions issued to protect the user and the equipment from injury or damage.
- Use qualified<sup>1</sup> programming technicians and good engineering practice for all procedures in this publication.

## Pressure

Do not apply pressure greater the maximum safe working pressure to the PACE Series.

## Maintenance

The PACE Series must be maintained using the manufacturer's procedures and should be carried out by authorised service agents or the manufacturer's service departments.

## Technical Advice

For technical advice contact the manufacturer or subsidiary.

## Associated Documents

A beginners Guide To SCPI by Barry Eppler, Published by Addison-Wesley Publishing Company Inc. for Hewlett Packard (ISBN 0-201-56350-9)

## This issue of the manual

This manual details the SCPI commands and queries for the PACE1000, PACE5000 and PACE6000 pressure instruments. This issue removes erroneous code and clarifies the operation status group (3.3).

---

<sup>1</sup> A programming technician must have the necessary specialist knowledge of programming, technical knowledge and documentation to carry out the required work on the PACE Series.

# Abbreviations

The following abbreviations are used in this manual; abbreviations are the same in the singular and plural.

a	Absolute	mV	millivolts
ALT	Altitude	MWP	Maximum working pressure
ASCII	American Standard Code for Information Interchange	No	Number
CAS	Calibrated airspeed	PACE	Pressure automated calibration equipment
e.g.	For example	PDCR	Pressure transducer
Fig.	Figure	PED	Pressure equipment directive
ft	Foot	PTX	Pressure transmitter
g	Gauge	ROC	Rate of climb
GPIB	General purpose interface bus	RS232	Serial communications standard
IDOS	Intelligent digital output sensor (GE product)	Rt CAS	Rate of Calibrated airspeed
i.e.	That is	Rt MACH	Rate of MACH
IEEE	Institute of Electrical and Electronic Engineers	Rx	Receive data
488	standard 488 (for programmable devices with a digital interface)	SCPI	Standard commands for programmable instruments
kg	kilogram	SDS	Sales data sheet
kts	knots	Tx	Transmit data
m	Metre	UUT	Unit under test
mA	milliampere	+ve	Positive
max	Maximum	-ve	Negative
mbar	Millibar	°C	Degrees Celsius
min	Minute or minimum	°F	Degrees Fahrenheit
mm	millimetre		

# Pressure Measurement Units

The following units are used in this manual:

ATM	atmosphere	KG/M2	kilogrammes per square
BAR	bar	KPA	kilo Pascals
CMH2O	centimetres of water at 20°C	LB/FT2	pounds per square foot
CMHG	centimetres of mercury	MH2O	metres of water
FTH2O	feet of water at 20°C	MHG	metres of mercury
FTH2O4	feet of water at 4°C	MMH2O	millimetres of water
HPA	hecto Pascals	MMHG	millimetres of mercury
INH2O	inches of water at 20°C	MPA	mega Pascals
INH2O4	inches of water at 4°C	PA	Pascals
INH2O60	inches of water at 60°F	PSI	pounds per square inch
INHG	inches of mercury	TORR	torr
KG/CM2	kilogrammes per square centimetre	MBAR	millibar

# Code Definitions

The following codes are used in this manual.

ABOR	Abort	LIM	Limit
ADDR	Address	LLO	Local lock out
AVER	Average	LOG	Logical
ALT	Altitude	LPAS	Low pass (filter)
AMPL	Amplitude	MAV	Message available in output queue
ATOD	Analog to Digital	MEAS	Measure
BAR	Barometer	MSS	Summary bit after SRQ
BRID	Bridge	NEGC	Negative Calibration
CAL	Calibration	OFFS	Offset
CAT	Catalogue	OPC	Operational condition
CDIS	Cdisable (calibration disable)	OPER	Operation
CEN	Cenable (calibration enable)	OPT	Option
CLS	Clear	OSB	Summary bit from standard operations status register
COMM	Communicate	OVER	Overshoot
COMP	Compensate	OUTP	Output
COND	Condition	PAR	Parity
CONF	Configuration	PASS	Password
CONT	Controller	PERC	Percent
CONV	Convert	POIN	Points
CORR	Correction	PRES	Preset
COUN	Count	PRES	Pressure
DEF	Define	QUE	Queue
DIAG	Diagnostic	QUES	Questionable
DIOD	Diode	RANG	Range
DISP	Display	REF	Reference
EAV	Error in error queue	RES	Resolution
EFF	Effort	RES	RESet
ENAB	Enable	SAMP	Sample
EOI	End of input	SENS	Sense
ERR	Error	SEPT	Set-point
ESB	Summary bit from standard event	SER	Serial
ESE	Event status enable	SOUR	Source
ESR	Event status register	SPE	Speed
EVEN	Event	SRE	Service request enable
FILT	Filter	SRQ	Service request
FREQ	Frequency	STAR	Start
FULL	Fullscale	STB	Status register query
GTL	Go to local	STAT	State
HEAD	Head	SYST	System
IDN	Identification	TIM	Time to set-point
IMM	Immediate	UNIT	Unit of pressure
INL	In limit	VAL	Value
INP	Input	VALV	Valve
INST	Instrument	VERS	Version
ISOL	Isolation	VOL	Volume
LEV	Level		



# Contents

<b>1.</b>	<b>Introduction</b>	<b>1</b>
1.1	General	1
1.2	System Model	1
1.3	Remote/Local Operation	1
<b>2.</b>	<b>Command Structure</b>	<b>3</b>
2.1	Notation	3
2.2	Message Terminators	3
2.3	Program Headers	4
2.4	SCPI Data Types	5
<b>3.</b>	<b>Status System</b>	<b>7</b>
3.1	Output Queue	9
3.2	Standard Event Group	9
3.3	Operation Status Group	10
3.4	Status Byte Group	11
3.5	Instrument Errors	14
<b>4.</b>	<b>Command and Query Summary</b>	<b>15</b>
4.1	Command Structure	15
	CALibration	17
	:CAL:PRES:POIN	19
	:CAL:PRES:ACC	20
	:CAL:PRES:ABOR	21
	:CAL:PRES:VAL	22
	:CAL:PRES:ZERO:VALV	24
	:CAL:PRES:ZERO:AUTO	26
	:CAL:PRES:ZERO:TIME	27
	:CAL:PRES:ZERO:TIME:STAT	28
	DISPlay	30
	:DISP:WIND	30
	INPut	31
	:INP:LOG	31
	:INP:LOG:STAT	32
	INSTrument	33
	:INST:CAT:ALL	33
	:INST:CAT	34
	:INST:CONT:SENS	35
	:INST:DISP	36
	:INST:LIM	37
	:INST:MAC	39
	:INST:SENS	40
	:INST:SENS:CALD	42

:INST:SENS:FULL	44
:INST:SENS:NEG	46
:INST:SENS:READ	48
:INST:SENS:SN	49
:INST:SENS:ZERO	50
:INST:SN	51
:INST:TASK	53
:INST:TASK:AERO	54
:INST:UNIT	55
:INST:VERS	56
OUTPut	57
:OUTP:STAT	57
:OUTP:LOG	58
:OUTP:ISOL:STAT	60
SENSe	61
:SENS:ALT	61
:SENS:ALT:INL	62
:SENS:ALT:INL:TIME	63
:SENS:ALT:RANG	64
:SENS:ALT:SLEW	65
:SENS:AIRF:QFE	66
:SENS:AIRF:QFF	67
:SENS:AIRF:QNH	68
:SENS:MACH	69
:SENS:MACH:INL	70
:SENS:MACH:INL:TIME	71
:SENS:MACH:RANG	72
:SENS:MACH:SLEW	73
:SENS:PRES	74
:SENS:PRES:AVER	75
:SENS:PRES:AVER:RES	76
:SENS:PRES:AVER:TIME	77
:SENS:PRES:CONT	78
:SENS:PRES:INL	79
:SENS:PRES:INL:TIME	80
:SENS:PRES:SLEW	81
:SENS:PRES:BAR	82
:SENS:PRES:RANG	83
:SENS:PRES:RES	85
:SENS:PRES:CORR:HEAD	86
:SENS:PRES:CORR:HEAD:STAT	88
:SENS:PRES:CORR:OFFS	89
:SENS:PRES:CORR:OFFS:STAT	90
:SENS:PRES:CORR:VOL	91
:SENS:PRES:FILT:LPAS:BAND	92
:SENS:PRES:FILT:LPAS:FREQ	93
:SENS:PRES:FILT:LPAS:STAT	94
:SENS:PRES:MAX	95
:SENS:PRES:MAX:RES	96
:SENS:PRES:MIN	97
:SENS:PRES:MIN:RES	98
:SENS:PRES:PERC	99



:SENS:PRES:PERC:SPAN	100
:SENS:PRES:PERC:STAT	101
:SENS:PRES:PSE	102
:SENS:PRES:REF	103
:SENS:SPE	104
:SENS:SPE:INL	105
:SENS:SPE:INL:TIME	106
:SENS:SPE:RANG	107
:SENS:SPE:SLEW	108
SOURce	109
:SOUR:ALT	109
:SOUR:MACH:REF	110
:SOUR:MACH:REF:MODE	111
:SOUR:MACH:REF:VAL	112
:SOUR:MACH:LEV:IMM:AMPL	113
:SOUR:PRES:COMP	114
:SOUR:PRES:EFF	115
:SOUR:PRES:INL	116
:SOUR:PRES:INL:TIME	117
:SOUR:PRES:LEV:IMM:AMPL	118
:SOUR:PRES:LEV:IMM:AMPL:VENT	119
:SOUR:PRES:RANG	121
:SOUR:PRES:RANG:LOW	122
:SOUR:PRES:SLEW	123
:SOUR:PRES:SLEW:MODE	125
:SOUR:PRES:SLEW:OVER	126
:SOUR:SPE:LEV:IMM:AMPL	127
:SOUR:SPE:LEV:IMM:AMPL:SLEW	128
STATus	129
:STAT:OPER:COND	129
:STAT:OPER:ENAB	130
:STAT:OPER:EVEN	131
:STAT:OPER:PRES:COND	132
:STAT:OPER:PRES:ENAB	133
:STAT:OPER:PRES:EVEN	134
:STAT:PRES	136
:STAT:QUES:COND	137
:STAT:QUES:ENAB	138
:STAT:QUES:EVEN	139
SYSTem	140
:SYST:COMM:USB	140
:SYST:ERR	141
:SYST:DATE	142
:SYST:SET	143
:SYST:TIME	144
:SYST:COMM:SER:CONT:RTS	145
:SYST:COMM:SER:CONT:XONX	146
:SYST:COMM:SER:BAUD	147
:SYST:COMM:SER:TYPE:PAR	148
:SYST:COMM:GPIB:SELF:ADDR	149
:SYST:AREA	150
:SYST:PASS:CDIS	152

	:SYST:PASS:CEN	153
	:SYST:PASS:CEN:STAT	154
	:SYST:VERS	155
	UNIT	156
	:UNIT:ALT	156
	:UNIT:CONV	157
	:UNIT:PRES	158
	:UNIT:PRES:DEF	159
	:UNIT:SPE	161
4.2	Common Commands	162
	*CLS	162
	*ESE	163
	*ESR	164
	*IDN?	165
	*OPC	166
	*SRE	167
	*STB?	168
	*TST?	169
	*WAI	170
4.3	Instrument Control Commands	171
	:GTL	171
	:LLO	172
	:LOC	173
	:REM	174
	:SRQ	175
	:SRQ:ENAB	176
	:SYN	177

## 5. Errors 179

5.1	Error Numbers	179
-----	---------------	-----

# Figures

<b>Figure</b>	<b>Page</b>
1-1 System Model	1
3-1 Status System	8



# Tables

<b>Table</b>	<b>Page</b>
2-1 Supported Multipliers	5
3-1 Standard Event Register	9
3-2 Operation Status Register	11
3-3 Status Byte Register	12
5-1 Errors -100 to -199	179
5-2 Errors -200 to -299	180
5-3 Errors -300 to -399	180
5-4 Errors +201 to +212	180

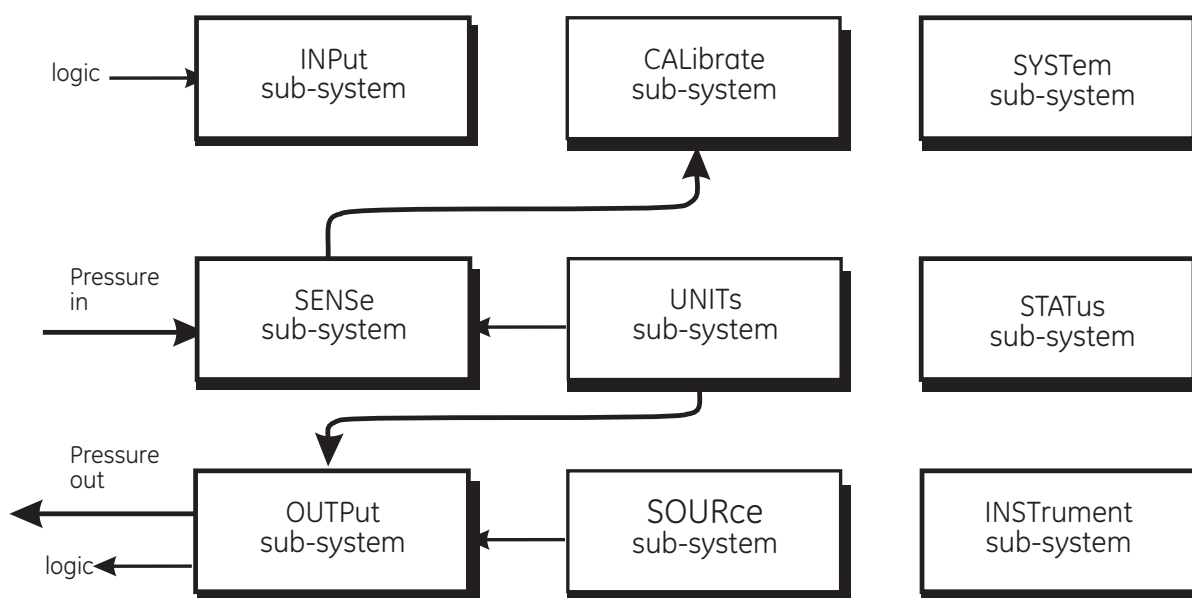


# 1. Introduction

## 1.1 General

The IEEE 488 and RS232 interfaces of the PACE Series provide remote control of the instrument from a suitable computer or controller. The SCPI protocol enables any instrument with a SCPI facility to be controlled using the same commands. The PACE Series use the full SCPI command set and the defined SCPI syntax.

The following sections describe and define each instrument command used by the PACE Series. The commands for the aeronautical option and the sensor calibration module option are described and defined in separate sections. Each section contains a quick reference structure of the relevant commands.



**Figure 1-1: System Model**

## 1.2 System Model

SCPI starts with a high-level block diagram of the measurement functions of the instrument. Each functional block is broken down into smaller block diagrams. SCPI contains a hierarchy of commands called a subsystem that maps directly to the hierarchy of the block diagram.

## 1.3 Remote/Local Operation

Most commands received over the SCPI interface automatically puts the PACE Series into remote control mode and disables the front panel touch-screen. Sending the LOC command returns the PACE Series to local control mode and enables the front panel touch-screen.



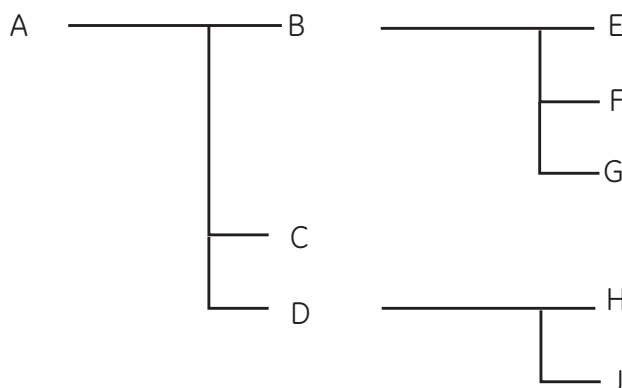


## 2. Command Structure

This section describes the structure of the commands and data sent and received by a PACE Series Controller/Calibrator.

### 2.1 Notation

All SCPI commands are based on a hierarchical tree structure consisting of key words and parameters. Associated commands are grouped together under a common node in the hierarchy.



In the command tree the command A is the root command. A tree pointer is used to decode the SCPI commands. At power-up the pointer goes to the root command.

### 2.2 Message Terminators

All SCPI commands are terminated by line feed i.e., either <newline> (ASCII character, decimal 10), EOI for IEEE. After receiving a termination character the tree pointer returns to the root command.

#### Colon

A colon moves the current path down one level in the command tree, (e.g., the colon in SOURCE:PRESSURE specifies PRESSURE the is one level below SOURCE). When the colon is the first character of the command, it specifies that the next command is a root level command (e.g., :SOURCE specifies that SOURCE is a root level command).

#### Semicolon

A semicolon separates two commands in the same message without changing the tree pointer.

(e.g., with reference to the tree):A:B:E;F;G

This equivalent to sending three messages:

:A:B:E

:A:B:F

:A:B:G

### Commas

If a command requires more than one parameter, separate adjacent parameters by using a comma. A comma does not affect the tree pointer.

(e.g.): SYSTEM:TIME 10,25,30

To execute a command the full path to the command must be specified:

(e.g.): OUTPut:STATe ON

This turns the pressure controller on.

**Note:** There must be a space between the command words and the parameter. In the above example there is a space between: STATE and ON.

SCPI commands are not case sensitive and may have a short form.

(e.g.): OUTP is the short form of OUTPUT.

Some nodes can be the default node and these key words are optional when programming the command. The instrument processes the command, with the same effect, with or without the option node. In this manual [] enclose [default notes].

(e.g.): SOURce[:PRESSure:][:LEVel][:IMMediate][:AMPLitude] 5.0

can be sent as

:SOURce:PRESSure:LEVel 5.0

or

:SOURce 5.0

This sets the set-point to 5.0

## 2.3 Program Headers

Program headers are keywords that identify a command, instruments accept both upper and lower case characters in a program header. There are two types of program header, common command headers and instrument control headers; each header can be a command or a query.

### Common Command and Query Headers

The common command and query program header syntax, specified in IEEE 488.2, are defined as follows:

Command

\*<PROGRAM MNEMONIC>

Query

\*<PROGRAM MNEMONIC>?

### Instrument Control Command and Query Headers

The instrument control command and query program header syntax controls and extracts data from the instrument as follows:

Command

:<MNEMONIC>

:<MNEMONIC> <PARAMETER>

Query

:<MNEMONIC>?

Instrument command headers can have a numeric suffix to identify each of several cases of the same header; the numeric suffix applies to both the long and short forms. All commands headers without a numeric suffix assume the value 1.

e.g.,

:OUTPut:LOGic1?

is the same as

:OUTPut:LOGic:?

### Queries

A query is a program header with an attached question mark character (?). On receiving a query, the current settings are loaded in the output buffer. A query does not affect the operation or set-up of the instrument.

When the parameter of a command contains enumerated character data, both long form and short form are recognized. A query causes the return of data in the short form.

Querying numeric parameters causes the resulting data to be returned in the units selected by the instrument unless specified otherwise.

## 2.4 SCPI Data Types

A variety of data types can be sent to the instrument as parameters or sent out from the instrument as response data.

### Decimal Data

All normal decimal expressions are accepted including optional signs, decimal point and scientific notation.

**Note:** This includes floating point data.

The following are valid:

123

45.67

-2.6

4.6e-10

.76

A suffix multiplier can be added to the numeric value.

:SOUR 100 m

would set the programmable output to 0.1 units (100m units).

The multipliers supported are:

**Table 2-1: Supported Multipliers**

Mnemonic	Multiplier
A	1e-18
G	1e+9
K	1e+3
M	1e-3
T	1e+12

## Chapter 2. Command Structure

---

If a real value is sent to the instrument when an integer is expected, it will be rounded to an integer.

### Integer Data

Integer data are whole numbers (containing no decimal places). A query of an integer value returns numbers containing no decimal places.

**Note:** Integer values can be specified in binary, octal or hexadecimal formats using the suffix letters (upper or lower case) B, Q and H respectively.

e.g., #B1010	binary representation of 10
#Q71	octal representation of 57
#HFA	hexadecimal representation of 250

Hexadecimal digits A-F can be in upper or lower case.

### Enumerated Character Data

Enumerated characters are used for data that has a finite number of values; enumerated parameters use mnemonics to represent each valid setting.

The mnemonics have long and short forms just like command mnemonics.

Example:

```
:SOURce:PRESSure:SLEW:MODE MAXimum
```

*selects the maximum rate mode.*

A query of an enumerated parameter always returns the short form data in upper case.

Example:

```
:SOURce:PRESSure:SLEW:MODE?
```

*queries rate mode, reply:*

```
:SOUR:PREs:SLEW:MODE MAX
```

### Boolean Data

Boolean data can only be one of two conditions; the numbers 1 and 0. Boolean can be “on” or “off”, queries return 1 or 0.

Example:

```
:OUTPut:STATe 1
```

A query of boolean data always returns 1 or 0.

### String Data

String data can contain any of the ASCII characters. A string must start with a double “quote” (ASCII 34) or a single `quote` (ASCII 39) and end with the same character.

**Note:** Characters in a string in either double “quote” or single `quote` are case sensitive.

Example: 1:

```
:SOURCe[:PRESSure]:RANGe '2.00barg'
```

Example: 2:

```
:SOURCe[:PRESSure]:RANGe "2.00barg"
```

*selects the 2 bar g range.*

A query of a string parameter always returns the string in double “quotes”.

---

## 3. Status System

The status reporting system informs the external controller that an event has occurred. This information is in the form of a service request (SRQ) using IEEE 488 or an SRQ message using RS232.

The PACE Series uses status reporting as defined in IEEE 488.2 with the implementation of status registers.

The OPERation status registers have been implemented to comply with the SCPI protocol. These are registers where the individual bits are summary bits of the status of the instrument. Since the SCPI protocol does not include pressure instruments, bit 10 of both these registers are used as a pressure summary bit. This pressure summary bit is expanded to two, 16 bit registers (Bit 15 is not used and is always zero).

The only bit implemented in the Operation status register is bit 10 (summary of the pressure operation status).

A summary bit is the final output of a data structure, it is a single bit that shows the status of one or more related events in the instrument. The basic structure of a summary bit

- Condition register
- Event register
- Enable register
- Logical ANDing of the Event and Enable registers
- Summary bit that summarizes the result using OR logic

### **Condition Register**

This register shows the current status of the device. The condition register is constantly updated - the bits in the register are set or reset showing the current condition.

### **Event Register**

The event register shows an event that occurs in the condition register (a condition bit goes from low to high). This condition change is stored and only reset when the event register is read or the \*CLS command sent.

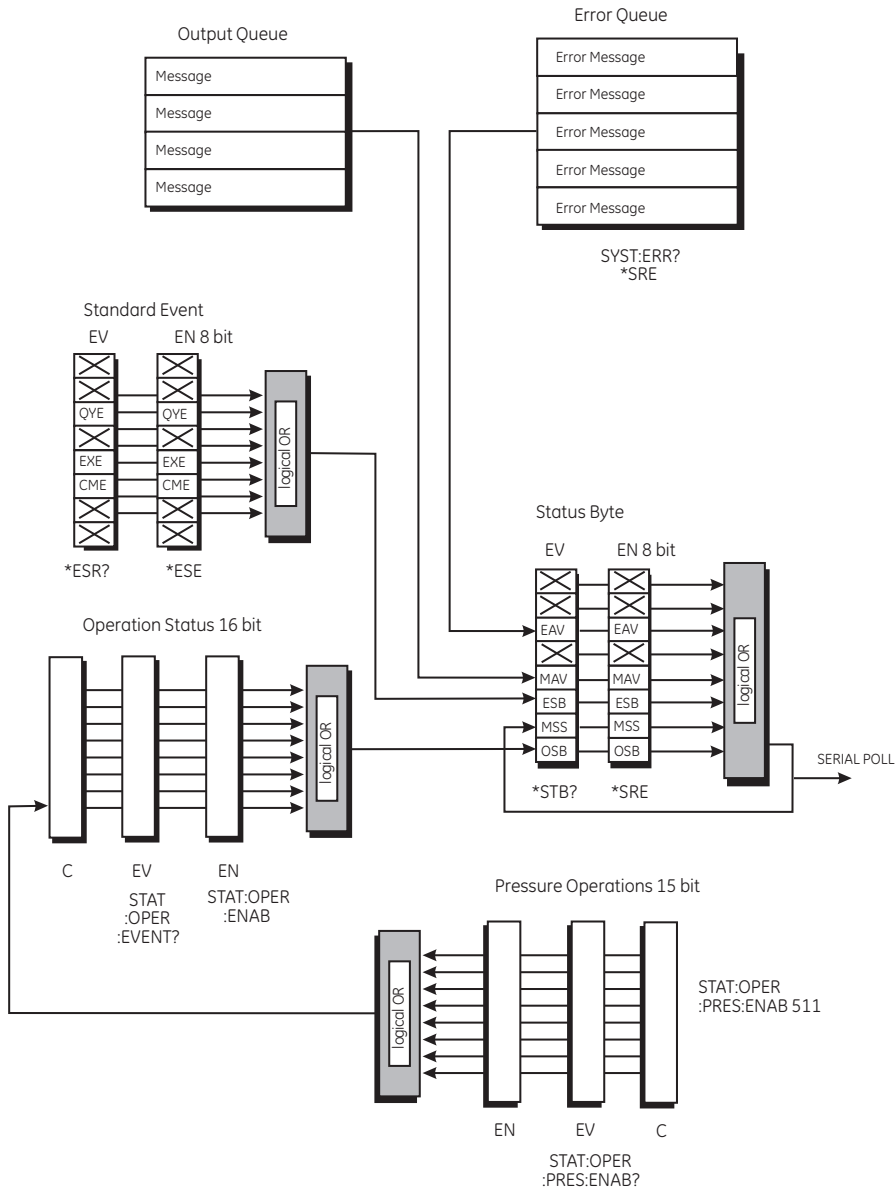
### **Enable Register**

This register allows the results of the event register to pass through to the next cascaded register and enables the user to select the event that should generate the final SRQ event.

# Chapter 3. Status System

The status system implemented in the instrument is shown in the following diagram:

**Note:** Initial values of registers are 0, with the queues empty.



**Figure 3-1: Status System**

- Key:
- C = Condition - variable values
  - EV = Latched values
  - EN = Bit mask

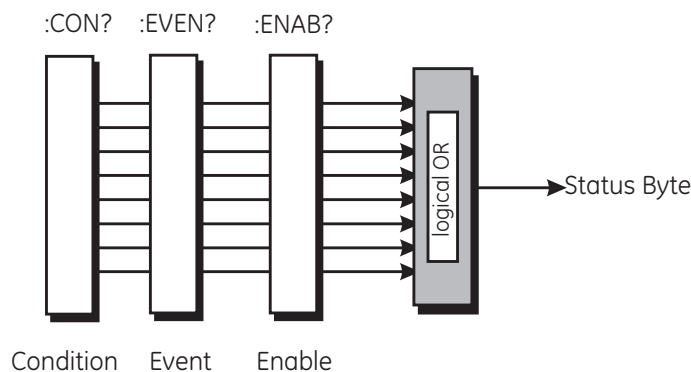
## 3.1 Output Queue

The output queue is a text readable data queue that is read through the IEEE 488 talk command. The queue is cleared by reading all elements in it or by the \*CLS command.

Every time a query has been successfully completed, the response, in a text readable format is placed at the end of the output queue. If the MAV bit in the “Status Byte” was previously cleared it will be set. The output queue can contain up to 256 characters. If there is not enough space in the output queue for a new message, the error -350, “Queue overflow” will be placed into the error queue and the most recent output message will be lost.

## 3.2 Standard Event Group

The standard event group are 8 bit registers that are read by the IEEE 488 standard commands. The event register is cleared by reading it; the event and enable registers are cleared by the \*CLS command.



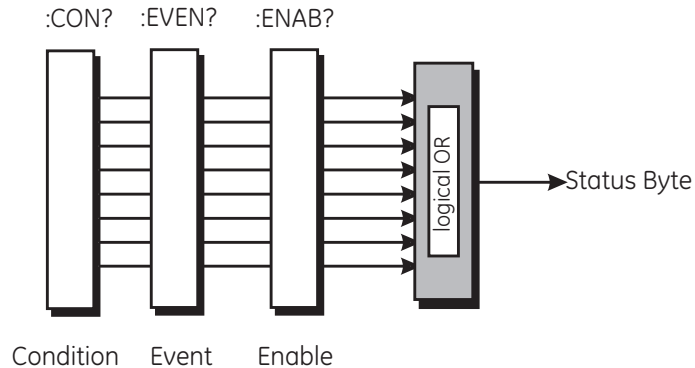
Bits within the standard event condition register are set by system errors and events. In addition to setting the status bits, a text message will be placed in the error queue. The ESB bit in the status byte sets if the associated bit in the event enable register is set. The enable register may be set through the \*ESE command so that selected standard events cause the ESB bit to be set. The system events that set each bit are as follows:

**Table 3-1: Standard Event Register**

Bit	Name	Description	Meaning/data
0	OPC	Not used	Reserved currently returns 0
1	RQC	Not used	Reserved currently returns 0
2	QYE	-400 to -499	Query errors
3	DDE	Not used	Reserved currently returns 0
4	EXE	-200 to -299	Execution errors
5	CME	-100 to -199	Command errors
6	URQ	Not used	Reserved currently returns 0
7	PON	Not used	Reserved currently returns 0

## 3.3 Operation Status Group

The operation status group are 16 bit registers that are read by the STAT:OPER:PRES commands. The event register is cleared by reading it; the event and enable registers are cleared by the \*CLS command.



When a standard operation condition occurs an appropriate bit is set in the condition register (this clears when the condition no longer exists). The bit is then latched in the event register. If the associated bit in the enable register is set, the OPR bit in the status byte sets. The enable register may be set through the STAT:OPER:PRES:ENAB command so that only selected standard operation events cause the OPR bit to set.

Problems can occur with some IEEE 488 controllers reading 16 bit unsigned numbers. All registers in this group do not use bit 15. The enable bit cannot be set and when read returns 0. The condition register is defined as follows:

### Vent complete

This signal occurs when the controller has been requested to vent and the vent has completed or timed out.

### Range change complete

This signal occurs when the controller has been requested to perform a range change and the range change is complete.

### In-Limits reached

This signal is set every time the controlled pressure is within the specified limits. The signal is only generated if the pressure has been within limits for a user defined wait time period.

### Zero complete

This signal is generated when a manual or timed zero is complete. If the zero times out then this signal is also generated.

### Range compare alarm (PACE1000 only)



This signal is generated when the range compare alarm is triggered during the range compare process.

**Table 3-2: Operation Status Register**

Bit (1)	Data (2)	Bit (3)	Data (4)
0	Vent complete	1	Range change complete
2	In-limits reached	3	Zero complete
4	Auto-zero started	5	Fill time, timed-out
6	Reserved - returns 0	7	Range compare alarm
8	Switch contacts changed state	9	Reserved - returns 0
10	Reserved - returns 0	11	Reserved - returns 0
12	Reserved - returns 0	13	Reserved - returns 0
14	Reserved - returns 0	15	Reserved - returns 0

**Auto zero started**

When the controller is in the auto zero mode this signal indicates that the auto zero process has started. The zero complete signal indicates that the zero process has finished.

**Fill timed out**

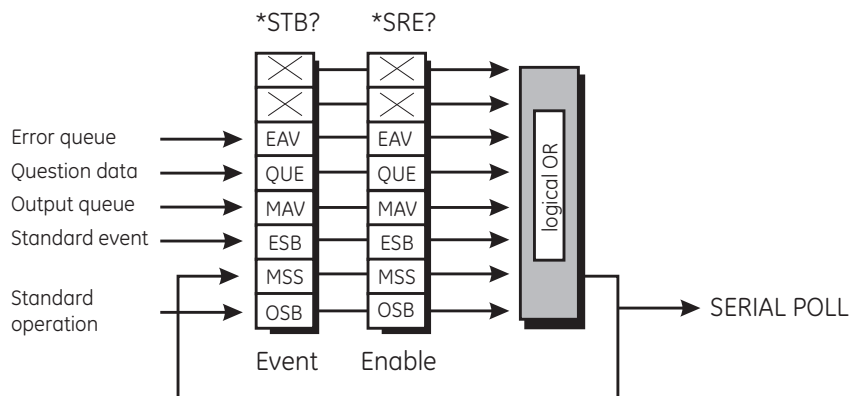
If a set-point has been requested and the set-point cannot be achieved within the fill time-out time, the fill timed out signal is generated.

**Switch contacts changed state**

Every time the switch contacts used for performing a switch test change state this bit is set.

### 3.4 Status Byte Group

The status byte group are 8 bit registers that are read by the IEEE 488 standard commands. The event register is cleared by reading it; the event and enable registers are cleared by the \*CLS command.



Bits within the status byte are a summary of other data structures in the status system. These bits will become set if other parts of the status system indicates that they should do so (i.e., a message in the output queue or error queue or, a condition and enable set in a register pair).

If the associated bit in the status enable register is set, a serial poll is generated and bit 6 is set. The enable register may be set through the \*SRE command so that only selected status bits cause a serial poll.

**Note:** Bit 6 of the enable register is always set to 0.

There are some small differences between \*STB? and serial polling. Either method can be used to read the state of bits 0-5 and bit 7. The reading method is different for bit 6 when using \*STB? and serial poll. In general, use serial polling inside interrupt service routines, not \*STB?

Bit 2 - EAV sets when there is an error in the error queue. The :SYST:ERR? command has to be sent to retrieve the error. The error queue buffers a maximum of five errors. When no more errors are available the message “No Error” is returned.

Bit 4 - MAV sets when there is a message available in the output queue.

Bit 5 - ESB sets when a standard event has occurred in the Standard Event Register.

Bit 6 - MSS sets when an SRQ is generated - SRQ sets when both the Status byte and the Service Request Enable register are at logic 1 (AND function).

**RS232 Specific**

A service request (SRQ) produces the message::SRQ <value>

where:

<value> = the contents of the status summary byte.

The status system data structure sets each bit as follows:

**Table 3-3: Status Byte Register**

Bit	Name	Description
0	-	Reserved currently returns 0
1	-	Reserved currently returns 0
2	EAV	Error in error queue
3	-	Reserved currently returns 0
4	MAV	Messages available in output queue
5	ESB	Summary bit from standard event
6	MSS	Summary bit after service request - SRQ
7	OSB	Summary bit from standard operations status

Example commands using the Status Byte and Status Byte Enable registers:

- \*SRE 16** Generate an SRQ interrupt when messages are available.
- \*SRE?** Find out what events are enabled to generate SRQ interrupts.
- \*STB?** Read and clear the Status Byte Enable register.

**IEEE 488 Specific**

Bit 7 - OSB sets when the pressure operations register bit 10 changes state. The operations register is a 16 bit register only using bit 10. This bit is a summary of the pressure operations register.

**Status reporting register structure**

To set-up the status reporting system.

1. All status registers should be cleared by the command:

\*CLS

2. The Pressure Operations Event register has to be set to enable the Pressure Operations Condition Register to send all the events to be reported; use the command:

:STAT:OPER:PRES:ENAB 511

The enabled events may also be read by the query:

:STAT:OPER:PRES:ENAB?

3. The Operation Status Event register must then be enabled to read bit 10 by the command:

:STAT:OPER:ENAB 1024

The enabled events may also be read by the query:

:STAT:OPER:ENAB?

4. The status request to enable the SRQ must then be set.

To enable only the Operation Status register (OSB) send the command:

\*SRE 128

To enable the Operation Status register (OSB) and the Error Queue (EAV) send the command:

\*SRE 132

This register may also be read by the query:

\*SRE?

An event occurring generates an SRQ, the Status Byte should be queried to find the source of the event.

If bit 2 of the Status Byte Register is set the error queue can be read by the query:

:SYST:ERR?

Keep issuing this query until there are no more errors in the error queue. At this point, bit 2 of the Status Byte Register clears.

If bit 7 of the Status Byte Register is set the Pressure Operations event register can be read by the query:

:STAT:OPER:PRES?

returning the bits of events that have occurred. Reading this register clears it and the associated status bit (bit 7).

At any time the instantaneous status of the pressure system can be read by the query:

:STAT:OPER:PRES:COND?

### 3.5 Instrument Errors

Any instrument error that occurs, either programming errors or execution errors, is stored in an error queue which is separate from the main output queue. The errors can be read by issuing the following command query:

```
:SYST:ERR?
```

The error queue can hold up to five errors. Each time the error queue is queried the instrument responds with the next stored error in the queue. The response consists of an error number followed by a string describing the error. When the error queue is empty the instrument responds with:

```
:SYST:ERR 0, No error
```

Querying the error queue clears the storage location in the error buffer. If more than five errors occur, before being queried, the 'Queue overflow;Error queue overflow' message is placed into the error queue. All subsequent errors are lost until the error queue is cleared.

## 4. Command and Query Summary

The following lists of all the SCPI commands and queries that apply to the instrument.

### 4.1 Command Structure

Some of the commands in the following summary are enabled at specific times and conditions, most can be enabled at any time. The command structure divides into subsystems as follows:

#### **Command sub-system**

:CALibration - calibration commands.

:INPut - switch input of the control module.

:INSTrument - instrument specific commands.

:OUTPut - controls the output pressure and logical outputs.

:SENSe - directs the instrument to measure selected parameters.

:SOURce - the commands that control the pressure outputs.

:STATus - instrument state.

:SYSTem - errors and SCPI version.

:UNIT - sets the units for the instrument.

Common SCPI commands - three letter commands, prefixed by \*.

Instrument control commands - three letter commands, prefixed by :.

**Command and Query Details**

This section describes each command in detail including parameters passed to it and response data returned. The general short form command is shown at the top of each page.

The following information is then given:

- Applicability - A list of instruments that accepts and responds to the command or query.
- Command Syntax** - The upper case represents the short form command.
- Parameter - Type: DECIMAL, INTEGER, ENUMERATED CHARACTER, BOOLEAN or STRING.
- Short form - The short alternative for the command to be effective.
- Function - Basic function of the command.
- Default - The default value or the maximum and minimum values where appropriate.
  
- Query Syntax** - The upper case represents the short form query command.
- Parameter - Type: DECIMAL, INTEGER, ENUMERATED CHARACTER, BOOLEAN or STRING.
- Short form - The short alternative for the query to be effective.
- Function - Basic function of query command.
- Response - Data returned by the instrument following the query command.

**Description**

Details of the command and query with any conditions of use and any related commands.

**Note:** Many of the command descriptions contain an example code: sent (Tx) to the instrument and the data received (Rx) from the instrument.

**Dual module instruments**

Sending and receiving to a module must include either number 1 or 2 after the first part of the command or query. Without the module number, a dual module instrument defaults to module 1.

Example:

Long form	Short form
Tx> :SOURce2:PRESsure:EFFort?	Tx> :SOUR2:PRES:EFF?
Rx> :SOUR2:PRES:EFF -0.2342882	Rx> :SOUR2:PRES:EFF -0.2342882

## CALibration

The CALibration subsystem enables the calibration of the pressure transducers, zero valve open/close/status, pressure zero function and the timed zero function. A pressure range that is not displayed or is displayed can be calibrated. Refer to the user manual for further details. An example calibration process sample is shown below:

<b>Tx</b>	<b>Rx</b>	<b>Remarks</b>
:Inst:Cat1:All?	:INST:CAT:ALL "3.50barg","10.00barg","1.00barg","BAROMETER","4.50bara"	Queries the catalogue of sensors fitted.
:SYST:ERR?	:SYST:ERR 0, No error	Queries the error queue, no errors.
:SOUR1:PRES:RANG?	:SOUR:PRES:RANG "4.50bara"	Queries for the currently selected displayed range.
:SENS1:PRES:RANG?	:SENS:PRES:RANG "3.50barg"	Queries for the currently selected SCPI range.
:SENS1:PRES?	:SENS:PRES -0.0004259	Pressure reading from the currently selected SCPI range ("3.50barg").
:SENS1:PRES:RANG "BAROMETER"	None	Selects the Barometer as the currently selected SCPI range for calibration.
:SENS1:PRES:RANG?	:SENS:PRES:RANG "BAROMETER"	Confirms the Barometer range is selected.
:SENS1:PRES?	:SENS:PRES 1005.7201013	Confirms the correct range is selected by reading the Barometer pressure output.
:SYST:ERR?	:SYST:ERR 0, No error	Queries the error queue, no errors.
:SYST:PASS:CEN:STAT?	:SYST:PASS:CEN:STAT 0	Queries for calibration mode status, calibration mode is disabled.
:SYST:PASS:CEN 2317100	None	Enables calibration mode.
:SYST:PASS:CEN:STAT?	:SYST:PASS:CEN:STAT 1	Confirms calibration mode is enabled.
:SENS1:PRES?	:SENS:PRES 1150.0000	Apply the calibration pressure and check the applied pressure is stable.
:CAL1:PRES:VAL1 1150.0000	None	Sends the 1 <sup>st</sup> calibration pressure of 1150.0000mbar.
:CAL1:PRES:VAL1?	:CAL:PRES:VAL 1150.0000000	Confirms the first calibration point pressure is received.
:SYST:ERR?	:SYST:ERR 0, No error	Queries the error queue, no errors.
:SENS1:PRES?	:SENS:PRES 750.0000	Apply the calibration pressure and check the applied pressure is stable.
:CAL1:PRES:VAL2 750.0000	None	Sends the 2 <sup>nd</sup> calibration pressure of 750.0000mbar.

## Chapter 4. Command and Query Summary

---

<b>Tx</b>	<b>Rx</b>	<b>Remarks</b>
:CAL1:PRES:VAL2?	:CAL:PRES:VAL 750.0000000	Confirms the 2 <sup>nd</sup> calibration point pressure is received.
:SYST:ERR?	:SYST:ERR 0, No error	Queries the error queue, no errors.
:CAL1:PRES:ACC 1	None	Accepts and implements the calibration.
:SYST:PASS:CEN:STAT?	:SYST:PASS:CEN:STAT 0	Calibration mode disabled.
:SYST:ERR?	:SYST:ERR 0, No error	Queries the error queue, no errors.



## :CAL:PRES:POIN

Applicability PACE1000, PACE5000, PACE6000

### Command Syntax

n/a

Parameter:

Short form:

Function:

Defaults:

### Query Syntax

#### :CAL[x]:PRES:POIN?

where  $x = 1 = \text{module 1}$  or  $x = 2 = \text{module 2}$  (if  $x$  is blank the default is module 1).

Normal form: :CAL[x]:PRES:POIN?

Short form: :CAL[x]:POIN?

Function: Gets the number of calibration points

Response:

### Description

Valid only when calibration is enabled, this queries the number of calibration points. To enable calibration mode use the “:SYST:PASS:CED” command. To disable calibration mode use the “:SYST:PASS:CDIS” command. To query the current status of calibration mode use the “:SYST[x]:PASS:CEN:STAT?”. To check for systems errors use the “:SYST:ERR?” query.

x	Tx	Rx	Remarks
1	:CAL[x]:PRES:POIN?	:CAL1:PRES:POIN 3	
2	:CAL[x]2:PRES:POIN?	:CAL2:PRES:POIN 3	
	:SYST:ERR?	:SYST:ERR 0, No error	No errors, error queue empty.
	:SYST:ERR?	:SYST:ERR -203, "Access error; Incorrect password"	Calibration mode not enabled.

## **:CAL:PRES:ACC**

Applicability      PACE1000, PACE5000, PACE6000

### Command Syntax

**:CALibration[x]:[PRESsure]:ACCept<space>[y]**

*where x = 1 = module 1 or x = 2 = module 2 (if x is blank the default is module 1).*

Parameter:          y = Integer = 1 = accept, 0 = reject

Normal form        :CAL[x]:PRES:ACC [y]

Short form:        :CAL[x]:ACC [y]

Function:          Accepts a calibration and exits calibration mode

Defaults:          no default value

### Query Syntax

**n/a**

Short form:

Function:

Response:

### Description

Valid only when calibration is enabled, this command accepts the calibration after entry of the calibration values and measured pressure readings are now adjusted within a few seconds as the new calibration is implemented.

Gauge sensors normally require entry of three calibration values whereas absolute sensors normally require entry of only two calibration values.

To query calibration mode status, enable calibration mode and disable calibration mode refer to commands “:SYST:PASS:CEN:STAT”, “:SYST:PASS:CED” and “:SYST:PASS:CDIS”.

## **:CAL:PRES:ABOR**

Applicability            PACE1000, PACE5000, PACE6000

### **Command Syntax**

#### **:CALibration[x]:[PRESsure]:ABORt**

*where x = 1 = module 1 or x = 2 = module 2 (if x is blank the default is module 1).*

Parameter:            None  
Normal form            :CAL[x]:PRES:ABOR  
Short form:            :CAL[x]:ABOR  
Function:              Aborts calibration values  
Defaults:              no default value

### **Query Syntax**

**n/a**

Short form:  
Function:  
Response:

### **Description**

Aborts the current calibration and exits calibration mode. To query calibration mode status, enable calibration mode and disable calibration mode refer to commands

“:SYST:PASS:CDIS”, “:SYST:PASS:CED” and “:SYST:PASS:CEN:STAT:”.

## :CAL:PRES:VAL

Applicability            PACE1000, PACE5000, PACE6000

### Command Syntax

**:CALibration[x]:[PRESsure]:VALue[y]<space><value>**

where x = 1 = module 1 or x = 2 = module 2 (if x is blank the default is module 1) and y is the calibration data point.

Parameter:            <decimal>

Normal form            :CAL[x]:PRES:VAL[y] value

Short form:            :CAL[x]:VAL[y] value

Function:              Enables calibration value to be entered at each calibration point.

Defaults:

### Query Syntax

**:CAL[x]:PRES:VAL[y]?**

where x = 1 = module 1 or x = 2 = module 2 (if x is blank the default is module 1) and y is the calibration data point.

Normal form            :CAL[x]:PRES:VAL[y]?

Short form:            :CAL[x]:VAL[y]?

Function:              Queries calibration point y value of module x.

Response:              Returns pressure value for VALue[y] of module [x].

### Description

Valid only when calibration is enabled. This command enables a calibration value to be entered during the calibration process. The query gets the calibration value.

Tx	Rx	Remarks
:CAL1:PRES:VAL1 -955.0000	None	1 <sup>st</sup> Calibration point and value.
:CAL1:PRES:VAL1?	:CAL:PRES:VAL - 955.0000000	Confirms the value was entered correctly.
:CAL1:PRES:VAL2 -5.0000	None	2 <sup>nd</sup> Calibration point and value.
:CAL1:PRES:VAL2?	:CAL:PRES:VAL2- 5.0000000	Confirms the value was entered correctly.
:CAL1:PRES:VAL3 245	None	3 <sup>rd</sup> Calibration point and value.

<b>Tx</b>	<b>Rx</b>	<b>Remarks</b>
:CAL1:PRES:VAL3?	:CAL:PRES:VAL3 245.0000000	Incorrect value entered for calibration point 3.
:CAL1:PRES:VAL3 2495.0000	None	Correct value entered for calibration point 3.
:CAL1:PRES:VAL3?	:CAL:PRES:VAL3 2495.0000000	Correct value entered for calibration point 3 validated.
:CAL1:PRES:ACC 1	None	Calibration accepted.

If a calibration point is entered incorrectly it can be re-entered assuming the correct applied pressure is still present. Once the calibration is accepted the point values entered adjust the calibration of the range selected for calibration.

Gauge sensors normally require entry of three calibration points whereas absolute sensors normally require entry of only two calibration points.

To query calibration mode status, enable calibration mode and disable calibration mode refer to commands “:SYST:PASS:CDIS”, “:SYST:PASS:CED” and “:SYST:PASS:CEN:STAT:”.

## :CAL:PRES:ZERO:VALV



**CAUTION** Opening the zero valve with high pressure in the system can cause the equipment to malfunction. Reduce the system pressure to zero gauge and make sure the controller is OFF before opening the zero valve.

Applicability            PACE5000, PACE6000

### Command Syntax

**:CALibration[x]:[PRESsure]:ZERO:VALVe[STATe]**

where  $x = 1 = \text{module 1}$  or  $x = 2 = \text{module 2}$  (if  $x$  is blank the default is module 1)

- Parameter:            <boolean> 1 = open, 0 = close
- Normal form            :CAL[x]:PRES:ZERO:VALV <parameter>
- Short form:            :CAL[x]:ZERO:VALV <parameter>
- Function:             Opens and closes zero valve.
- Defaults:             0

### Query Syntax

**:CAL[x]:PRES:ZERO:VALV:STAT?**

where  $x = 1 = \text{module 1}$  or  $x = 2 = \text{module 2}$  (if  $x$  is blank the default is module 1)

- Normal form            :CAL[x]:PRES:ZERO:VALV:STAT?
- Short form:            :CAL[x]:ZERO:VALV?
- Function:             Queries state of valve.
- Response:            1 - valve is open  
                          0 - valve is close

### Description

This command is used to open and close the zero valve in each control module. The query gets the state of the zero valve, open or closed.

### Examples

	Module 1	Module 2	Function
TX>	:CAL1:PRES:ZERO:VALV 1	:CAL2:PRES:ZERO:VALV 1	Opens the zero valve.
TX>	:CAL1:PRES:ZERO:VALV:STAT 1	:CAL2:PRES:ZERO:VALV:STAT 1	Opens the zero valve.

	<b>Module 1</b>	<b>Module 2</b>	<b>Function</b>
	Either of above two commands OPEN the module zero valve.		
TX>	:CAL:PRES:ZERO:VALV:STAT?	:CAL2:PRES:ZERO:VALV:STAT?	Queries the current state of the valve.
RX>	:CAL:PRES:ZERO:VALV:STAT 1	:CAL2:PRES:ZERO:VALV:STAT 1	Query reply = zero valve open.
TX>	:CAL1:PRES:ZERO:VALV 0	:CAL2:PRES:ZERO:VALV 0	Closes the zero valve.
TX>	:CAL1:PRES:ZERO:VALV:STAT 0	:CAL2:PRES:ZERO:VALV:STAT 0	Closes the zero valve.
	Either of above two commands CLOSE the module zero valve.		
TX>	:CAL:PRES:ZERO:VALV:STAT?	:CAL2:PRES:ZERO:VALV:STAT?	Queries the current state of the valve.
RX>	:CAL:PRES:ZERO:VALV:STAT 0	:CAL2:PRES:ZERO:VALV:STAT 0	Query reply = zero valve closed.

## **:CAL:PRES:ZERO:AUTO**

Applicability      PACE5000, PACE6000

### Command Syntax

**:CALibration[x]:[PRESsure]:ZERO:AUTO<space><parameter>**

*where x = 1 = module 1 or x = 2 = module 2 (if x is blank the default is module 1)*

Parameter:          <boolean>  
                         0 - deactivates a zero process  
                         1 - activates a zero process

Normal form         :CAL[x]:PRES:ZERO:AUTO <parameter>

Short form:         :CAL[x]:ZERO:AUTO <parameter>

Function:            Automatic pressure zeroing

Defaults:            0

### Query Syntax

**:CAL[x]:PRES:ZERO:AUTO?**

*where x = 1 = module 1 or x = 2 = module 2 (if x is blank the default is module 1)*

Normal form         :CAL[x]:PRES:ZERO:AUTO?

Short form:         :CAL[x]:ZERO:AUTO?

Function:            Query progress of zero

Response:           0 - Zero complete or not in progress.  
                         1 - Zero in progress.

### Description

This command starts or aborts a zero process. The progress of the zero can be monitored by using the query.

#### Examples:

TX> :CAL1:PRES:ZERO:AUTO?      Queries the zero status

RX> :CAL:PRES:ZERO:AUTO 0      Zero status is complete or not in progress

TX> :CAL:PRES:ZERO:AUTO 1      Initiates an auto-zero process

TX> :CAL1:PRES:ZERO:AUTO?      Queries the zero status

RX> :CAL:PRES:ZERO:AUTO 1      An auto-zero process is in progress



## :CAL:PRES:ZERO:TIME

Applicability PACE5000, PACE6000

### Command Syntax

**:CALibration[x]:[PRESSure]:ZERO:TIME<space><parameter>**

where  $x = 1 = \text{module 1}$  or  $x = 2 = \text{module 2}$  (if  $x$  is blank the default is module 1)

Parameter: Numeric, time value in hours.  
 Normal form :CAL[x]:PRES:ZERO:TIME <parameter>  
 Short form: :CAL[x]:ZERO:TIME <parameter>  
 Function: Sets time period in hours between zero events.  
 Defaults: -

### Query Syntax

**:CAL[x]:PRES:ZERO:TIME?**

where  $x = 1 = \text{module 1}$  or  $x = 2 = \text{module 2}$  (if  $x$  is blank the default is module 1)

Normal form :CAL[x]:PRES:ZERO:TIME?  
 Short form: :CAL[x]:ZERO:TIME?  
 Function: Queries timed zero.  
 Response: Integer number in hours.

### Description

This command sets the time interval between timed zeroing. Queries the current setting and sends a new setting in hours. If the current setting is less than 1hour the query returns 0. Only available in basic task.

Examples:

Tx	Rx	Remarks
:CAL:ZERO:TIME?	:CAL:PRES:ZERO:TIME 0	Queries Module 1 zero time interval = 0
:CAL:ZERO:TIME 3	None	Sets module 1 zero time interval to 3 hours.
:CAL:ZERO:TIME?	:CAL:PRES:ZERO:TIME 3	Queries Module 1 zero time interval = 3
:CAL2:ZERO:TIME?	:CAL:PRES:ZERO:TIME 0	Queries Module 2 zero time interval = 0
:CAL2:ZERO:TIME 2	None	Sets module 2 zero time interval to 2 hours.
:CAL2:ZERO:TIME?	:CAL:PRES:ZERO:TIME 2	Queries Module 2 zero time interval = 2

## :CAL:PRES:ZERO:TIME:STAT

Applicability            PACE5000, PACE6000

### Command Syntax

**:CALibration[x]:[PRESsure]:ZERO:TIME:STATe<space><parameter>**

where x = 1 = module 1 or x = 2 = module 2 (if x is blank the default is module 1)

Parameter:            <<boolean>>  
 Normal form            :CAL[x]:PRES:ZERO:TIME:STAT 1  
 Short form:            :CAL[x]:ZERO:TIME:STAT 1  
 Function:              Sets timed zero on/off.  
 Defaults:              0  
                             (0 - sets timed zero to OFF)  
                             (1 - sets timed zero to ON)

### Query Syntax

**:CALibration[x]:[PRESsure]:ZERO:TIME:STATe?**

where x = 1 = module 1 or x = 2 = module 2 (if x is blank the default is module 1)

Normal form            :CAL[x]:PRES:ZERO:TIME:STAT?  
 Short form:            :CAL[x]:ZERO:TIME:STAT?  
 Function:              Queries the status of the timed zero ON or OFF.

### Description

This command sets the time period for zero on or off. This query gets the status of the timedzero (on or off)

Examples:

Tx	Rx	Remarks
:CAL:PRES:ZERO:TIME:STAT?	:CAL:PRES:ZERO:TIME:STAT 0	Queries Module 1 timed zero status.
:CAL:PRES:ZERO:TIME:STAT 1	None	Activates module 1 timed zero mode.
:CAL1:PRES:ZERO:TIME:STAT?	:CAL:PRES:ZERO:TIME:STAT 1	Queries Module 1 timed zero status.
:CAL1:PRES:ZERO:TIME:STAT 0	None	Deactivates module 1 timed zero mode.
:CAL:PRES:ZERO:TIME:STAT?	:CAL:PRES:ZERO:TIME:STAT 0	Queries Module 1 timed zero status.

Tx	Rx	Remarks
:CAL2:PRES:ZERO:TIME:STAT?	:CAL2:PRES:ZERO:TIME:STAT 0	Queries Module 2 timed zero status.
:CAL2:PRES:ZERO:TIME:STAT 1	None	Activates module 2 timed zero mode.
:CAL2:PRES:ZERO:TIME:STAT?	:CAL2:PRES:ZERO:TIME:STAT 1	Queries Module 2 timed zero status.
:CAL2:PRES:ZERO:TIME:STAT 0	None	Deactivates module 2 timed zero mode.
:CAL2:PRES:ZERO:TIME:STAT?	:CAL2:PRES:ZERO:TIME:STAT 0	Queries Module 2 timed zero status.

## DISPlay

The DISPlay subsystem shows the state of the display window.

### :DISP:WIND

Applicability      PACE1000

#### Command Syntax

**n/a**

Parameter:

Short form:

Function:

Defaults:

#### Query Syntax

### **:DISP[x]:WIND?**

*where: x is the window index 1 to 3. Index 1 is the top window, index 2 is the middle status area window and index 3 is the lower function area window.*

Short form:            :DISP[x]:WIND?

Function:              Asks for window for an allocated range

Response                DISP 1, 2 or 3 and the value in the selected units.

#### **Description**

This query returns the measured reading from the selected display window.

Examples:

TX>            :DISP1:WIND?

RX>            :DISP:WIND "979.44"

TX>            :DISP2:WIND?

RX>            :DISP2:WIND "993.55"

## INPut

The INPut subsystem shows the state of the logical inputs.

### :INP:LOG

Applicability      PACE5000, PACE6000

#### Command Syntax

**n/a**

Parameter:

Short form:

Function:

Defaults:

#### Query Syntax

### :INPut[x]:LOGic?

*where: x (module) = 1 or 2 (default - 1),*

Short form:          :INP:LOG?

Function:            Asks for state of switch input within the controller module.

Response	first parameter	- 0 = OFF, 1 = ON
	second parameter	- measured pressure at the time of switching (snapshot in current pressure units).

#### Description

This query returns the state of the switch input within the module and the pressure at time of switching operations.

Example:

```
TX> :INP:LOG?
```

```
RX> :INP:LOG 0, 0.8321209
```

Current logic OFF, pressure was 0.8321209 in current pressure units when the module was switched to OFF condition.

## **:INP:LOG:STAT**

Applicability PACE1000, PACE5000, PACE6000

**Note:** Only applies to the PACE1000 when the VFC option is fitted.

### Command Syntax

**n/a**

Parameter:

Short form:

Function:

Defaults:

### Query Syntax

**:INPut[x]:LOGic:STATe?**

*where: x (module) = 1 or 2 (default - 1),*

Short form: :INP:LOG:STAT?

Function: Asks for state of switch input.

Response 0 = OFF, 1 = ON

### Description

This query returns the state of the switch input.

## INSTrument

The INSTrument subsystem gets information about the configuration of the instrument.

### :INST:CAT:ALL

Applicability      PACE1000, PACE5000, PACE6000

#### Command Syntax

**n/a**

Parameter:

Short form:

Function:

Defaults:

#### Query Syntax

### :INSTrument:CATalog[x]:[ALL]?

*where: x (module) = 1 or 2 (default - 1)*

Short form:      :INST:CAT[x]:ALL?

Function:      Query all ranges fitted, including source sensor ranges.

Response      A list of comma separated strings of ranges fitted.

#### Description

This query returns all the ranges fitted to the instrument. The reply is a comma separated list of strings representing each range.

Example:

A dual module instrument with a 7.00 barg module and a 3.50barg module. Both have a barometric option fitted.

```
TX>      :INST:CAT1:ALL?
```

```
RX>      :INST:CAT:ALL "3.50barg","10.00barg","1.00barg","BAROMETER","4.50bara"
```

```
TX>      :INST:CAT2:ALL?
```

```
RX>      :INST:CAT2:ALL "2.00barg","3.50barg","1.00barg","BAROMETER","3.00bara"
```

## **:INST:CAT**

Applicability PACE1000, PACE5000, PACE6000

### Command Syntax

**n/a**

Parameter:

Short form:

Function:

Defaults:

### Query Syntax

#### **:INSTrument:CATalog[x]?**

*where: x (module) = 1 or 2 (default - 1)*

Short form: :INST:CAT[x]?

Function: Query all ranges fitted, does not including source sensor ranges.

Response A list of comma separated strings of ranges fitted.

### Description

This query returns all measure ranges fitted to the instrument. The reply is a comma separated list of strings representing each range.

Example:

A dual module instrument with a 7.00 barg module and a 3.50barg module. Both have a barometric option fitted.

TX> :INST:CAT1?

RX> :INST:CAT: "7.00barg","BAROMETER","8.00bara"

TX> :INST:CAT2?

RX> :INST:CAT2: "3.50barg","BAROMETER","4.50bara"



## :INST:CONT:SENS

Applicability PACE6000

### Command Syntax

**n/a**

Parameter:

Short form:

Function:

### Query Syntax

#### **:INSTrument:CONT(y):SEN(z)**

Use the *CONT* index(y) to determine which control module 1 (or blank) or 2. Both *:INST:CONT* and *:INST:CONT1* will address control module 1, whilst *:INST:CONT2* addresses only control module 2.

Use the *SENS* index(z) to determine which sensor as below.

### Description

*:INST:CONT:SENS* or *:INST:CONT:SENS1* "2.00barg" returns Control module 1 full-scale range sensor

*:INST:CONT:SENS2* "7.00barg" returns the +ve port source pressure sensor full-scale

*:INST:CONT:SENS3* "1.00barg" returns the -ve port vacuum pressure sensor full-scale

*:INST:CONT:SENS4* "BAROMETER" if fitted

*:INST:CONT:SENS5* "" Reference sensor range if fitted

*:INST:CONT:SENS6* "" Not used

*:INST:CONT:SENS7* "3.00bara" Returns the pseudo absolute range

*:INST:CONT:SENS8* "" Not used

Example:

TX> :INST:CONT1:SENS2?

RX> :INST:CONT1:SENS2 "7.00barg" 1st control module vacuum sensor

TX> :INST:CONT2:SENS4?

RX> :INST:CONT2:SENS4 "BAROMETER" 2nd control module barometric sensor

Refer to command *:INST:SENS* to address PACE5000 and PACE1000.

## **:INST:DISP**

Applicability      PACE6000

### **Command Syntax**

#### **:INSTrument:DISPlay**

Parameter:          Enumerated character single or dual  
Short form:          :INST:DISP  
Function:            Sets the single or dual display mode.

### **Query Syntax**

#### **:INSTrument:DISPlay?**

Short form:          :INST:DISP?  
Function:            Queries the display setting - single or dual.  
Response            Enumerated character single or dual

### **Description**

This command sets the single or dual display mode. The query returns display setting single or dual display mode.

#### **Example:**

TX>	:INST:DISP?	ask display mode.
RX>	:INST:DISP SING	in single mode.
TX>	:INST:DISP DUAL	change to dual display mode.
TX>	:INST:DISP?	ask display mode.
RX>	INST:DISP DUAL	in dual display mode.
TX>	:INST:DISP SINGLE	change to single display mode.
TX>	:INST:DISP?	ask display mode.
RX>	:INST:DISP SING	change to single display mode.

#### **In the single display mode:**

TX>	:INST:CAT?	
RX>	:INST:CAT "7.00barg","BAROMETER","8.00bara","3.50barg","4.50bara"	
TX>	:INST:CAT2?	
RX>	:INST:CAT2 "7.00barg","BAROMETER","8.00bara","3.50barg","4.50bara"	

**:INST:LIM**

Applicability PACE1000, PACE5000, PACE6000

**Command Syntax**

n/a

Parameter:

Short form:

Function:

**Query Syntax****:INSTrument:[LIMits][x]?**

where:  $x$  (sensor number) = 1, 2... 8 (default - 1).

Short form: :INST?

Function: Queries the upper and lower full-scale limits of the fitted sensor ranges. The index number  $x$  is used to index into the list of available ranges.

<b>x</b>	<b>Sensor PACE5000, PACE6000</b>	<b>Sensor PACE1000</b>
1	Module 1: control sensor	Internal sensor 1
2	Module 1: source pressure +ve	Internal sensor 2
3	Module 1: source pressure -ve	Internal sensor 3
4	Module 1: barometric range (optional)	External IDOS-1
5	Module 2: control sensor	External IDOS-2
6	Module 2: source pressure +ve	External IDOS-3
7	Module 2: source pressure -ve	-
8	Module 2: barometric range (optional)	-

Response A string representing the range, a number representing the upper full-scale and a number representing the lower full-scale.

**Description**

This query returns a string representing the range, a number representing the upper fullscale and lower full-scale.

## Chapter 4. Command and Query Summary

---

### Example:

A dual module instrument, both modules with the barometric option fitted.

```
TX>      :INST:LIM?
RX>      :INST:LIM "7.00barg", 7350.0000000, -1100.0000000
TX>      :INST:LIM1?
RX>      :INST:LIM "7.00barg", 7350.0000000, -1100.0000000
TX>      :INST:LIM2?
RX>      :INST:LIM2 "20.00barg", 21000.0000000, -1100.0000000
TX>      :INST:LIM3?
RX>      :INST:LIM3 "2.00barg", 2100.0000000, -1100.0000000
TX>      :INST:LIM4?
RX>      :INST:LIM4 "BAROMETER", 1207.5000000, 825.0000000
TX>      :INST:LIM5?
RX>      :INST:LIM5 "3.50barg", 3675.0000000, -1100.0000000
TX>      :INST:LIM6?
RX>      :INST:LIM6 "20.00barg", 21000.0000000, -1100.0000000
TX>      :INST:LIM7?
RX>      :INST:LIM7 "10.00barg", 10500.0000000, -1100.0000000
TX>      :INST:LIM8?
RX>      :INST:LIM8 "BAROMETER", 1365.0000000, 38.5000000
```

## **:INST:MAC**

Applicability      PACE1000, PACE5000, PACE6000

### **Command Syntax**

**n/a**

Parameter:

Short form:

Function:

### **Query Syntax**

#### **:INSTrument:MACaddress?**

Short form:      :INST:MAC?

Function:      Queries the MAC internet address.

Response      Address string

### **Description**

This command returns the Media Access Control (MAC) address string.

Example:

TX>      :INST:MAC?

RX>      :INST:MAC "00-D0-1C-0B-1B-1A"

## **:INST:SENS**

Applicability        PACE1000 and PACE5000

### **Command Syntax**

**n/a**

Parameter:

Short form:

Function:

### **Query Syntax**

#### **:INST:SENSor?**

Short form:        :INST:SENS?

Function:         Queries the transducer range (sensor).

Response         Returns sensor range.

### **Description**

This command returns the transducer range. Only module 1 on a PACE6000 can be addressed when using this command, for PACE6000 module 1 and module 2 addressing refer to the command :INST:CONT:SENS

Example:

```
TX>        :INST:SENS?  
RX>        :INST:SENS "7.00barg"  
TX>        :INST:SENS1?  
RX>        :INST:SENS "7.00barg"  
TX>        :INST:SENS2?  
RX>        :INST:SENS2 "20.00barg"  
TX>        :INST:SENS3?  
RX>        :INST:SENS3 "2.00barg"  
TX>        :INST:SENS4?  
RX>        :INST:SENS4 "BAROMETER"  
TX>        :INST:SENS5?  
RX>        :INST:SENS5 "3.50barg"  
TX>        :INST:SENS6?  
RX>        :INST:SENS6 "20.00barg"  
TX>        :INST:SENS7?  
RX>        :INST:SENS7 "10.00barg"
```

TX> :INST:SENS8?  
RX> :INST:SENS8 "BAROMETER"

## :INST:SENS:CALD

Applicability      PACE1000, PACE5000, PACE6000

### Command Syntax

**n/a**

Parameter:

Short form:

Function:

### Query Syntax

#### **:INST:SENS[x]:CALD[y]?**

*where x (sensor number) = 1, 2, . . . .8 (default -1) y = 1, 2, 3, . . . .10 (y times last calibration date)*

Short form:            :INST:SENS:CALD?

Function:             Queries sensor calibration dates in the form yyyy, mm, dd.

Response              Returns sensor [x] with [y] calibration date:

*where:*  
*x is the sensor number, x = 1, 2, . . . .8*  
*y is the yth calibration date for sensor x.*  
*y = 1, 2, 3, . . . .10 calibration dates stored for each sensor.*

<b>x</b>	<b>Sensor PACE5000, PACE6000</b>	<b>Sensor PACE1000</b>
1	Module 1: control sensor	Internal sensor 1
2	Module 1: source pressure +ve	Internal sensor 2
3	Module 1: source pressure -ve	Internal sensor 3
4	Module 1: barometric range (optional)	External IDOS-1
5	Module 2: control sensor	External IDOS-2
6	Module 2: source pressure +ve	External IDOS-3
7	Module 2: source pressure -ve	-
8	Module 2: barometric range (optional)	

### Description

This query returns the following:



### Example

TX> :INST:SENS5:CALD1?

RX> :INST:SENS5:CALD 2009, 11, 21

TX> :INST:SENS5:CALD2?

RX> :INST:SENS5:CALD2 2009, 11, 17

This shows that the module 2 control sensor was last calibrated on 21st November 2009 and the previous calibration was on 17th November 2009.

## :INST:SENS:FULL

Applicability      PACE1000 and PACE5000

### Command Syntax

**n/a**

Parameter:

Short form:

Function:

### Query Syntax

#### **:INSTrument:SENSe[x]:FULLscale?**

where: *x = 1, 2...8 is the sensor number. It defaults to 1.*

Short form:            :INST:SENS[x]:FULL?

Function:             Queries sensor full-scale value.

Response              Returns the full-scale value of the selected sensor.

<b>x</b>	<b>SensorPACE5000, PACE6000</b>	<b>PACE1000</b>
1	Module 1: control sensor	Internal sensor1
2	Module 1: source pressure +ve	Internal sensor 2
3	Module 1: source pressure -ve	Internal sensor 3
4	Module 1: barometric range (optional)	External IDOS-1
5	Module 2: control sensor	External IDOS-2
6	Module 2: source pressure +ve	External IDOS-3
7	Module 2: source pressure -ve	-
8	Module 2: barometric range (optional)	-

**Description**

This query returns the following:

:INST:SENS:FULL?

:INST:SENS:FULL 7.0000000

Default - module 1 control sensor full-scale.

:INST:SENS1:FULL?

:INST:SENS:FULL 7.0000000

Returns module 1 control sensor full-scale.

:INST:SENS2:FULL?

:INST:SENS2:FULL 20.0000000

Returns module 1 +ve source sensor full-scale.

:INST:SENS3:FULL?

:INST:SENS3:FULL 2.0000000

Returns module 1 -ve source sensor full-scale.

:INST:SENS4:FULL?

:INST:SENS4:FULL 1.1500000

Returns module 1 barometric sensor full-scale.

:INST:SENS5:FULL?

:INST:SENS5:FULL 3.5000000

Returns module 2 control sensor full-scale.

:INST:SENS6:FULL?

:INST:SENS2:FULL 20.0000000

Returns module 2 +ve source sensor full-scale.

:INST:SENS7:FULL?

:INST:SENS7:FULL 10.0000000

Returns module 2 -ve source sensor full-scale.

:INST:SENS8:FULL?

:INST:SENS8:FULL 1.1500000

Returns module 2 barometric sensor full-scale.

# :INST:SENS:NEG

Applicability PACE1000, PACE5000, PACE6000

## Command Syntax

n/a

Parameter:

Short form:

Function:

## Query Syntax

### :INSTrument:SENSor[x]:NEGCAl?

where: x = 1, 2...8 is the sensor number. It defaults to 1.

Short form: :INST:SENS:NEG?

Function: Queries sensor negative calibration.

Response <boolean>:

0 - negative calibration not supported

1 - negative calibration supported.

x	SensorPACE5000, PACE6000	PACE1000
1	Module 1: control sensor	Internal sensor1
2	Module 1: source pressure +ve	Internal sensor 2
3	Module 1: source pressure -ve	Internal sensor 3
4	Module 1: barometric range (optional)	External IDOS-1
5	Module 2: control sensor	External IDOS-2
6	Module 2: source pressure +ve	External IDOS-3
7	Module 2: source pressure -ve	-
8	Module 2: barometric range (optional)	-

**Description**

This command query returns sensor negative calibration supported or not supported. Used in DPI515 emulation; for PACE5000/6000 this query always returns '1'.

:INST:SENS:NEGC?

:INST:SENS:NEGC 1

Default - module 1 control sensor NEGcal.

:INST:SENS1:NEGC?

:INST:SENS:NEGC 1

Returns module 1 control sensor NEGcal.

:INST:SENS2:NEGC?

:INST:SENS2:NEGC 1

Returns module 1 +ve source sensor NEGcal.

:INST:SENS3:NEGC?

:INST:SENS3:NEGC 1

Returns module 1 -ve source sensor NEGcal.

:INST:SENS4:NEGC?

:INST:SENS4:NEGC 1

Returns module 1 barometric sensor NEGcal.

:INST:SENS5:NEGC?

:INST:SENS5:NEGC 1

Returns module 2 control sensor NEGcal.

:INST:SENS6:NEGC?

:INST:SENS2:NEGC 1

Returns module 2 +ve source sensor NEGcal.

:INST:SENS7:NEGC?

:INST:SENS7:NEGC 1

Returns module 2 -ve source sensor NEGcal.

:INST:SENS8:NEGC?

:INST:SENS8:NEGC 1

Returns module 2 barometric sensor NEGcal.

## **:INST:SENS:READ**

Applicability PACE1000

### **Command Syntax**

**n/a**

Parameter:

Short form:

Function:

### **Query Syntax**

#### **:INSTrument:SENSor[x]:READing?**

Short form: :INST:SENS:READ?

Function: Queries sensor zero reading.

Response Gets reading of selected sensor [x].

where x is the sensor number.

x = 1, 2, . . . . 8

### **Description**

This query gets the filtered reading followed by the instantaneous reading from the selected sensor.

**:INST:SENS:SN**

Applicability PACE1000, PACE5000, PACE6000

**Command Syntax**

**n/a**

Parameter:

Short form:

Function:

**Query Syntax****:INSTrument:SENSor[x]:Serial Number?**

Short form: :INST:SENS:SN?

Function: Queries the sensor serial number.

Response Gets the serial number of selected sensor [x].  
where x is the sensor number designated in the table below.

<b>x</b>	<b>Description</b>
1 or blank	Internal sensor number 1
2	Internal sensor number 2
3	Internal sensor number 3
4	External IDOS sensor number 1
5	External IDOS sensor number 2
6	External IDOS sensor number 3
7	Pseudo absolute sensor 1*
8	Pseudo absolute sensor 2*
9	Pseudo gauge sensor 1*
10	Pseudo gauge sensor 2*
* Returns the serial number of the main sensor allocated to create a pseudo range (not the barometric sensor).	

**Description**

This query gets the serial number of selected sensor.

To query the serial number of the PACE1000, PACE1000 internal analogue option or VFC option refer to the command INST:SN

## **:INST:SENS:ZERO**

Applicability: PACE1000

### Command Syntax

#### **INSTrument:SENSor[x]:ZERO**

where:  $x = 1, 2 \dots 8$  is the sensor number. It defaults to 1.

Parameter: <integer> 1, 2...8

Short form: :INST:SENS[x]:ZERO

Function: Performs a zero on the selected sensor specified in the sensor index.

### Query Syntax

#### **:INSTrument:SENSor[x]:ZERO?**

where:  $x = 1, 2 \dots 8$  is the sensor number. It defaults to 1.

Short form: :INST:SENS[x]:ZERO?

Function: Queries sensor zero condition.

Response: Gets zero condition [y] of selected sensor [x].  
where x is the sensor number.  
number.

$x = 1, 2, \dots, 8$

y =

0 - Zero ok

1 - Zero in progress

2 - Zero timed out

3 - Zero outside limits

4 - Zero aborted

### Description

The command selects a sensor and performs a zero. The query gets the zero condition after a zero has been performed on the selected sensor.

### Example

TX>	:INST:SENS2:ZERO	Zeroes sensor 2
TX>	:INST:SENS2:ZERO?	Queries sensor 2 status
RX>	:INST:SENS2:ZERO 1	Sensor 2 zero in progress



**:INST:SN**

Applicability: PACE1000, PACE5000, PACE6000

**Command Syntax**

**n/a**

Parameter:

Short form:

Function:

**Query Syntax****:INSTrument:SN[x]?**

where:  $x = 1, 2...7$  is the hardware serial number. If  $x$  is blank, then  $x$  defaults to 1.

Short form: :INST:SN[x]?

Function: Used to query the serial numbers of the instrument hardware and options.

Response: Integers representing serial numbers.

**Description**

This query returns the serial numbers of the hardware installed.

<b>x</b>	<b>Description</b>
1 or blank	PACE1000, PACE5000 or PACE 6000
2	Control module 1 *
3	Control module 2 *
4	Analogue Output 1 †
5	Analogue Output 2 †
6	VFC 1 †
7	VFC 2 †

**Notes:**

1. Depending on future instrument development, additional numbers may be included.
2. \* not applicable to PACE indicators.
3. † option must be fitted.

## Chapter 4. Command and Query Summary

---

Example:

```
TX> :INST:SN?  
RX> :INST:SN 68795  
TX> :INST:SN1?  
RX> :INST:SN 68795  
TX> :INST:SN2?  
RX> :INST:SN2 2803347  
TX> :INST:SN3?  
RX> :INST:SN3 65795  
TX> :INST:SN4?  
RX> :INST:SN4 68884  
TX> :INST:SN5?  
RX> :INST:SN5 0  
TX> :INST:SN6?  
RX> :INST:SN6 88704  
TX> :INST:SN7?  
RX> :INST:SN7 0
```

In this example, 'RX> :INST:SN5 0' and 'RX> :INST:SN7 0' mean the 2<sup>nd</sup> Analogue Output and 2<sup>nd</sup> VFC board are not installed. To query the serial number of PACE1000 sensors refer to the command INST:SENS:SN.

## :INST:TASK

Applicability: PACE1000, PACE5000, PACE6000

### Command Syntax

#### :INSTrument:TASK

Parameter: Enumerated  
 Short form: :INST:TASK  
 Function: Sets the task

### Query Syntax

#### :INSTrument:TASK?

Short form: :INST:TASK?  
 Function: Used to query the task set.  
 Response: Enumerated character representing task.

### Description

This command sets the task and the query returns the task setting.

	PACE1000	PACE5000	PACE6000
AERONAUTICAL	0		
AIRFIELD	0		
AIRSPPEEDLEAK	0		
ALTLEAK	0		
BAROGRAPH	0		
BASIC	0	0	0
BURSTTEST		0	0
DIVIDER		0	0
LEAKTEST	0	0	0
PRESET		0	0
SWITCHTEST		0	0
TESTPROGRAM		0	0

## **:INST:TASK:AERO**

Applicability: PACE1000, PACE6000

### Command Syntax

**:INSTrument:TASK:AERO:[RANGe]**

Parameter:

Short form: :INST:TASK:AERO

Function: Sets the aero ranges: pressure, altitude, speed and MACH.

### Query Syntax

**:INSTrument:TASK:AERO:[RANGe]?**

Short form: :INST:TASK:AERO?

Function: Used to query the task set.

Response: Integers representing task numbers.

### Description

The command sets the aero ranges: pressure, altitude, speed and MACH. This query returns the aero range setting.

## :INST:UNIT

Applicability: PACE1000, PACE5000, PACE6000

### Command Syntax

n/a

Parameter:

Short form:

### Query Syntax

#### :INST:UNIT?

where:  $x = 1, 2 \dots 32$  is the unit number. It defaults to 1.

Short form: :INST:UNIT?

Function: Used to query the units of measurement available. This does not query the current unit in use (refer to the 'UNIT' family of commands/queries).

Response: Enumerated character string.

### Description

This command sets the units of measurement available to the instrument.

TX> :INST:UNIT1? RX> :INST:UNIT MBAR	TX> :INST:UNIT12? RX> :INST:UNIT12 KG/M2	TX> :INST:UNIT23? RX> :INST:UNIT23 INH2O_4
TX> :INST:UNIT2? RX> :INST:UNIT2 BAR	TX> :INST:UNIT13? RX> :INST:UNIT13 MMH2O_4	TX> :INST:UNIT24? RX> :INST:UNIT24 INH2O_20
TX> :INST:UNIT3? RX> :INST:UNIT3 PA	TX> :INST:UNIT14? RX> :INST:UNIT14 CMH2O_4	TX> :INST:UNIT25? RX> :INST:UNIT25 INH2O_60
TX> :INST:UNIT4? RX> :INST:UNIT4 HPA	TX> :INST:UNIT15? RX> :INST:UNIT15 MH2O_4	TX> :INST:UNIT26? RX> :INST:UNIT26 FTH2O_4
TX> :INST:UNIT5? RX> :INST:UNIT5 KPA	TX> :INST:UNIT16? RX> :INST:UNIT16 MMH2O_20	TX> :INST:UNIT27? RX> :INST:UNIT27 FTH2O_20
TX> :INST:UNIT6? RX> :INST:UNIT6 MPA	TX> :INST:UNIT17? RX> :INST:UNIT17 CMH2O_20	TX> :INST:UNIT28? RX> :INST:UNIT28 FTH2O_60
TX> :INST:UNIT7? RX> :INST:UNIT7 MMHG	TX> :INST:UNIT18? RX> :INST:UNIT18 MH2O_20	TX> :INST:UNIT29? RX> :INST:UNIT29 USER1
TX> :INST:UNIT8? RX> :INST:UNIT8 CMHG	TX> :INST:UNIT19? RX> :INST:UNIT19 TORR	TX> :INST:UNIT30? RX> :INST:UNIT30 USER2
TX> :INST:UNIT9? RX> :INST:UNIT9 MHG	TX> :INST:UNIT20? RX> :INST:UNIT20 ATM	TX> :INST:UNIT31? RX> :INST:UNIT31 USER3
TX> :INST:UNIT10?RX> :INST:UNIT10 INHG	TX> :INST:UNIT21? RX> :INST:UNIT21 PSI	TX> :INST:UNIT32? RX> :INST:UNIT32 USER4
TX> :INST:UNIT11? RX> :INST:UNIT11 KG/CM2	TX> :INST:UNIT22? RX> :INST:UNIT22 LB/FT2	

## :INST:VERS

Applicability: PACE1000, PACE5000, PACE6000

### Command Syntax

n/a

Parameter:

Short form:

Function:

### Query Syntax

#### :INSTrument:VERSion[x]?

where:  $x = 1, 2...15$  is the software version. It defaults to 1.

Short form: :INST:VERS[x]?

Function: Queries the software versions of the instrument and options.

Response: Returns strings representing the software version:

### Description

x	Software item	x	Software item
1	Instrument main code	9	Analogue output board 1 boot code
2	Instrument OS build	10	Analogue output board 2 main code
3	Instrument boot ROM	11	Analogue output board 2 boot code
4	Controller 1 main code	12	VFC board 1 main code
5	Controller 1 Boot code	13	VFC board 1 boot code
6	Controller 2 main code	14	VFC board 2 main code
7	Controller 2 Boot code	15	VFC board 2 boot code
8	Analogue output board 1 main code		

Example:

TX> :Inst:Vers?	TX> :Inst:Vers6?	TX> :Inst:Vers12?
RX> :INST:VERS "00.01.09"	RX> :INST:VERS6 "02.00.29"	RX> :INST:VERS12 "00.01.53"
TX> :Inst:Vers1?	TX> :Inst:Vers7?	TX> :Inst:Vers13?
RX> :INST:VERS "00.01.09"	RX> :INST:VERS7 "01.00.00"	RX> :INST:VERS13 "00.00.40"
TX> :Inst:Vers2?	TX> :Inst:Vers8?	TX> :Inst:Vers14?
RX> :INST:VERS2 "01.06.16"	RX> :INST:VERS8 "00.01.53"	RX> :INST:VERS14 ""
TX> :Inst:Vers3?	TX> :Inst:Vers9?	TX> :Inst:Vers15?
RX> :INST:VERS3 "01.01.04"	RX> :INST:VERS9 "00.00.40"	RX> :INST:VERS15 ""
TX> :Inst:Vers4?	TX> :Inst:Vers10?	
RX> :INST:VERS4 "02.00.29"	RX> :INST:VERS10 ""	
TX> :Inst:Vers5?	TX> :Inst:Vers11?	
RX> :INST:VERS5 "01.00.00"	RX> :INST:VERS11 ""	

**Note:** A return of an empty string means the corresponding option is not installed.

## OUTPut

The OUTPut subsystem turns the pressure controller on/off and controls the state of the logical outputs.

### :OUTP:STAT

Applicability: PACE5000, PACE6000

#### Command Syntax

**:OUTPut[x]:STATe <Boolean>**

where: *x (module) = 1 or 2 (default - 1)*

Parameter: <boolean>  
 0 - turn controller off  
 1 - turns controller on

Short form: :OUTP

Function: Turn the pressure controller on/off

Default: 0

#### Query Syntax

**:OUTPut:STATe?**

where: *x (module) = 1 or 2 (default - 1)*

Short form: :OUTP?

Function: Asks for state of pressure controller

Response: 0 - controller off

#### Description

Sets or queries the state of the pressure controller.

Example

TX> :OUTP:STAT?

RX> :OUTP:STAT 0

The module 1 controller currently turned off.

To turn the module 1 controller off:

TX> :OUTP:STAT OFF

To turn the module 1 controller on:

TX> :OUTP:STAT ON

TX> :OUTP:STAT?

RX> :OUTP:STAT 1

## **:OUTP:LOG**

Applicability: PACE1000, PACE5000, PACE6000

### Command Syntax

**:OUTP[x]:LOGic[y]<space><parameter>**

*where: x = VFC 1 or VFC 2 (default = 1) and y = VFC relay 1, VFC relay 2 or VFC relay 3 represents the relay number (default = 1)*

Parameter: <boolean>  
0 - turn relay [y] OFF  
1 - turn relay [y] ON

Form: :OUTP[x]:LOG[y]

Function: Turns relay ON and OFF

Default: VFC 1 of Relay 1

### Query Syntax

**:OUTP[x]:LOGic[y]?**

*where: x = VFC 1 or VFC 2 (default = 1) and y = VFC relay 1, VFC relay 2 or VFC relay 3 represents the relay number (default = 1).*

Form: :OUTP[x]:LOG[y]?

Function: Asks for relay ON/OFF status.

Response: 0 - relay OFF  
1 - relay ON

### Description

The VFC option must be installed and each VFC relay must be set to 'communications' through the front panel measure setup, VFC setup display screen. Alternatively check under Instrument status, Volt-free contact status. If the volt-free contact option is installed but the relay not set to communications' a system error query will return ":SYST:ERR -240,"Hardware error;Incorrect VFC setup".

The three relays can be queried and switched on and off. Omitting [x] or [y], the default of 1 is used so this will address VFC 1 Relay 1.



## Example

<b>Tx</b>	<b>Rx</b>	<b>Remarks</b>
:OUTP1:LOG1?	:OUTP:LOG 0	Query VFC1/R1 Contact NC = Closed, NO = Open
:OUTP1:LOG1 1	None	Set VFC1/R1 contact
:OUTP1:LOG1?	:OUTP:LOG 1	Query VFC1/R1 Contact NC = Open, NO = Closed
:OUTP1:LOG2?	:OUTP:LOG2 0	Query VFC1/R2 Contact NC = Closed, NO = Open
:OUTP1:LOG2 1	None	Set VFC1/R2 contact
:OUTP1:LOG2?	:OUTP:LOG2 1	Query VFC1/R2 Contact NC = Open, NO = Closed
:OUTP1:LOG2 0	None	Set VFC1/R2 contact
:OUTP1:LOG2?	:OUTP:LOG2 0	Query VFC1/R2 Contact NC = Closed, NO = Open
:OUTP1:LOG3?	:OUTP:LOG3 0	Query VFC1/R3 Contact NC = Closed, NO = Open

## :OUTP:ISOL:STAT

Applicability: PACE5000, PACE6000

### Command Syntax

#### :OUTPut[x]:ISOLation:STATe

where  $x = 1 = \text{module 1}$  or  $x = 2 = \text{module 2}$  (if  $x$  is blank the default is module 1)

Parameter: <boolean>  
 0 OFF - turn isolation valve of the module [x] OFF  
 1 ON - turn isolation valve of the module [x] ON

Short form: :OUTP:ISOL:STAT

Function: Turns isolation valve ON and OFF

### Query Syntax

#### :OUTPut[x]:ISOLation:STATe?

where  $x = 1 = \text{module 1}$  or  $x = 2 = \text{module 2}$  (if  $x$  is blank the default is module 1)

Short form: :OUTP:ISOL:STAT?

Function: Asks for isolation valve ON/OFF status.

Response: 0 - isolation valve OFF (Closed/Isolated)  
 1 - isolation valve ON (Open/Non-isolated)

### Description

The output port isolation valve of module 1 or module 2 can be queried and switched on and off.

Example:

	Module 1	Module 2	Remarks
TX>	:OUTP1:ISOL:STAT 1	:OUTP2:ISOL:STAT 1	Opens the output port isolation valve
TX>	:OUTP1:ISOL:STAT?	:OUTP2:ISOL:STAT?	Queries the current state of the valve
RX>	:OUTP:ISOL:STAT 1	:OUTP2:ISOL:STAT 1	Query reply = valve open
TX>	:OUTP1:ISOL:STAT 0	:OUTP2:ISOL:STAT 0	Closes the output port isolation valve
TX>	:OUTP1:ISOL:STAT?	:OUTP2:ISOL:STAT?	Queries the current state of the valve
RX>	:OUTP:ISOL:STAT 0	:OUTP2:ISOL:STAT 0	Query reply = valve closed



**CAUTION** Opening a module output isolation valve with high pressure in the external system can cause the PACE to malfunction. Equalise the external system pressure to zero gauge or use other means to identify the external system pressure before opening the isolation valve.

## SENSe

The SENSe subsystem selects, configures and queries the measurement functions of the instrument.

### :SENS:ALT

Applicability: PACE1000, PACE6000

#### Command Syntax

n/a

Parameter:

Short form:

Function:

#### Query Syntax

### SENSe:ALTitude?

Short form: :SENS:ALT?

Function: This query reads the aeronautical sensor.

Response: A decimal, altitude reading in the current aeronautical units.

#### Description

Queries the altitude reading for the aeronautical sensor in the selected aeronautical units.

**Note:** If the instrument is not in aero mode it returns an error: -221, "Settings conflict".

## **:SENS:ALT:INL**

Applicability: PACE6000

### Command Syntax

**:SENSe[x][:ALTitude]:INLimits <number>**

*where x = 1 or 2 is the module number (default - 1)*

Parameter: <decimal> in-limits value as % full-scale

Short form: :SENS:INL <number>

Function: The instrument has an in-limits indicator. This can generate a service request when the pressure is within limits for a set time period.

Default: 0.01% full-scale

Minimum: 0% full-scale

Maximum: 100% full-scale

### Query Syntax

**:SENSe[x][:ALTitude]:INLimits?**

*where x = 1 or 2 is the module number (default - 1)*

Short form: :SENS:INL?

Function: Query in-limits value

Response: Decimal representing in-limits value as % full-scale.

### Description

Sets the in-limits value.

**Note:** If the instrument is not in aero mode it returns an error: -221. Settings conflict.

## :SENS:ALT:INL:TIME

Applicability: PACE6000

### Command Syntax

**:SENSe[x][:ALTitude]:INLimits:TIME <number>**

where  $x = 1$  or  $2$  is the module number (default - 1)

Parameter: <integer> in-limits time in seconds

Short form: :SENS:INL:TIME <number>

Function: Sets the time that the pressure has to be within limits before generating a service request.

Default: 2 seconds

Minimum: 2 seconds

Maximum: 999 seconds

### Query Syntax

**:SENSe[x][:ALTitude]:INLimits:TIME?**

where  $x = 1$  or  $2$  is the module number (default - 1)

Short form: :SENS:INL:TIME?

Function: Query in-limits timers.

Response: Number representing in-limits time in seconds.

### Description

Sets the in-limits timer value.

**Note:** If the instrument is not in aero mode it returns an error: -221. Settings conflict.

## **:SENS:ALT:RANG**

Applicability: PACE1000, PACE6000

### **Command Syntax**

**n/a**

Parameter:

Short form:

Function:

### **Query Syntax**

#### **SENSe:ALTitude:RANGe?**

Short form: :SENS:ALT:RANG?

Function: This queries the range of the aeronautical sensor.

Response: A string (number), representing altitude range in pressure units.  
*Example: 1.30 bara*

### **Description**

Queries the range of the aeronautical sensor in pressure units.

**Note:** If the instrument is not in aero mode it returns an error: -221. Settings conflict.

## **:SENS:ALT:SLEW**

Applicability: PACE1000, PACE6000

### Command Syntax

#### **SENSE:ALTitude:SLEW**

Parameter: <string>

Short form: :SENS:ALT:SLEW

Function: Sets the rate set-point for altitude slew.

Default: -

### Query Syntax

#### **SENSE:ALTitude:SLEW?**

Short form: :SENS:ALT:SLEW?

Function: Queries the rate set-point for altitude slew.

Response: Decimal number representing the rate set-point for altitude slew (rate of climb) or (vertical speed) in current units.

### Description

This command sets the rate set-point for altitude slew. The query gets the rate set-point for altitude (rate of climb) or (vertical speed) per second or minute.

**Note:** If the instrument is not in aero mode it returns an error: -221. Settings conflict.

## **:SENS:AIRF:QFE**

Applicability: PACE1000

### Command Syntax

**n/a**

Parameter: :SENS:AIRF:QFE?

Short form:

Function:

### Query Syntax

#### **SENSe:AIRField:QFE?**

Short form: :SENS:AIRF:QFE?

Function: This query reads the QFE value when the instrument is in airfield mode.

Response: A decimal reading in selected units of pressure.

### Description

Queries the QFE reading, returns the QFE reading in selected units of pressure when the instrument is in airfield mode.

**Note:** QFE is the mean sea level pressure corrected for temperature and adjusted for a specific site (i.e. a datum such as an airfield).



## **:SENS:AIRF:QFF**

Applicability: PACE1000

### Command Syntax

**n/a**

Parameter:

Short form:

Function:

### Query Syntax

#### **SENSe:AIRField:QFF?**

Short form: :SENS:AIRF:QFF?

Function: This query reads the QFF value when the instrument is in airfield mode.

Response: A decimal reading in selected units of pressure.

### Description

Queries the QFF reading, returns the QFF reading in selected units of pressure when the instrument is in airfield mode.

**Note:** QFF is the mean sea level pressure derived from the barometric pressure calculated, assuming mean long term values of temperature and relative humidity, for a specific site. (i.e. the QFF is the location value plotted on the surface synoptic chart).

## **:SENS:AIRF:QNH**

Applicability: PACE1000

### **Command Syntax**

**n/a**

Parameter:

Short form:

Function:

### **Query Syntax**

#### **SENSe:AIRField:QNH?**

Short form: :SENS:AIRF:QNH?

Function: This query reads the QNH value when the instrument is in airfield mode.

Response: A decimal reading in selected units of pressure.

### **Description**

Queries the QNH reading, returns the QNH reading in selected units of pressure when the instrument is in airfield mode.

**Note:** QNH is the pressure measured at a specific site then reduced to mean sea level pressure.

---

## :SENS:MACH

Applicability: PACE1000, PACE6000

### Command Syntax

n/a

Parameter:

Short form:

Function:

### Query Syntax

#### **SENSe:MACH?**

Short form: :SENS:MACH?

Function: This query reads the aeronautical sensors.

Response: A decimal reading in MACH number.

### Description

Queries the MACH reading.

**Note:** If the instrument is not in aero mode it returns an error: -221. Settings conflict.

## **:SENS:MACH:INL**

Applicability: PACE6000

### Command Syntax

**:SENSe[x][:MACH]:INLimits <number>**

*where x = 1 or 2 is the module number (default - 1)*

Parameter: <decimal> MACH in limits value as % full-scale

Short form: :SENS:MACH:INL <number>

Function: The instrument has a MACH in-limits indicator. This can generate a service request when the MACH value is within limits for a set time period.

Default: 0.01% full-scale

Minimum: 0% full-scale

Maximum: 100% full-scale

### Query Syntax

**:SENSe[x][:MACH]:INLimits?**

*where x = 1 or 2 is the module number (default - 1)*

Short form: :SENS:MACH:INL?

Function: Query MACH in-limits value

Response: Decimal representing MACH in-limits value as % full-scale.

### Description

Sets the in-limits value.

**Note:** If the instrument is not in aero mode it returns an error: -221. Settings conflict.

## :SENS:MACH:INL:TIME

Applicability: PACE6000

### Command Syntax

**:SENSe[x][:MACH]:INLimits:TIME <number>**

where  $x = 1$  or  $2$  is the module number (default - 1)

Parameter: <integer> MACH in-limits time in seconds

Short form: :SENS:MACH:INL:TIME <number>

Function: Sets the time that the MACH value has to be within limits before generating a service request.

Default: 2 seconds

Minimum: 2 seconds

Maximum: 999 seconds

### Query Syntax

**:SENSe[x][:MACH]:INLimits:TIME?**

where  $x = 1$  or  $2$  is the module number (default - 1)

Short form: :SENS:MACH:INL:TIME?

Function: Query MACH in-limits timers.

Response: Number representing MACH in-limits time in seconds.

### Description

Sets the MACH in-limits timer value.

**Note:** If the instrument is not in aero mode it returns an error: -221. Settings conflict.

## **:SENS:MACH:RANG**

Applicability: PACE1000, PACE6000

### Command Syntax

#### **SENSe[2]:MACH:RANG**

where  $x = 1$  or  $2$  is the module number (default - 1)

Parameter:

Short form: :SENS:MACH:RANG

Function: Sets the MACH range

### Query Syntax

#### **SENSe:MACH:SLEW?**

Short form: :SENS:MACH:RANG?

Function: This queries the MACH range.

Response: A string (number), representing MACH range.

### Description

Queries the MACH range reading.

**Note:** If the instrument is not in aero mode it returns an error: -221. Settings conflict.

## **:SENS:MACH:SLEW**

Applicability: PACE1000, PACE6000

### Command Syntax

#### **SENSe[2]:MACH:SLEW**

where  $x = 1$  or  $2$  is the module number (default - 1)

Parameter: -

Short form: :SENS:MACH:SLEW

Function: Sets the rate set-point for MACH slew.

### Query Syntax

#### **SENSe:MACH:SLEW?**

Short form: :SENS:MACH:SLEW?

Function: Queries the rate set-point for MACH slew.

Response: A decimal, pressure reading/second in MACH.

### Description

This command sets the rate set-point for MACH slew. The query gets the rate set-point for MACH.

**Note:** If the instrument is not in aero mode it returns an error: -221. Settings conflict.

## :SENS:PRES

Applicability: PACE1000, PACE5000, PACE6000

### Command Syntax

n/a

Parameter:

Short form:

Function:

### Query Syntax

#### :SENSe[:PRESsure]?

where  $x = 1$  or  $2$  is the module number (default - 1)

Short form: :SENS?

Function: This query reads the sensor which has been selected by the :SENS:PRES:RANG command.

Response: A decimal, pressure reading in the current units.

### Description

Queries the pressure reading for the selected sensor in the selected units. The sensor selected can be changed using the :SENSe[:PRESsure]:RANGe command. PACE5000 and PACE6000 for alternative query to :SENS:PRES? that does not require preselection of the sensor using the :SENS:PRES:RANG command refer to :SENS:PRESS:BAR, :SENS:PRES:CONT, :SENS:PRES:PSE and :SENS:PRES:REF and to directly query the pressure values of the +ve and -ve pressure supply source and vacuum sensors refer to the :SOUR:PRES:COMP query.

### Example

```
TX> :SENS1:PRES?
RX> :SENS:PRES 3616.9282227
TX> :SENS2:PRES?
RX> :SENS2:PRES 3617.1921387

TX> :SENS2:PRES:RANG?
RX> :SENS2:PRES:RANG "3.50barg"
Currently module 2 sensing range is in gauge mode.
TX> :SENS2:PRES:RANG "4.50bara"
TX> :SENS2:PRES:RANG?
RX> :SENS2:PRES:RANG "4.50bara"
It has been changed to pseudo-absolute mode.

TX> :SENS2:PRES?
RX> :SENS2:PRES 4615.0807447
The reading is also changed to pseudo-absolute mode.
```

**Note:** :SOURe[:PRESsure]:RANGe command does not change 'SENS?' reading. It changes the front-panel display.



---

## **:SENS:PRES:AVER**

Applicability: PACE1000

### **Command Syntax**

**:SENSe[:PRESsure]:AVERage**

Parameter:

Short form: :SENS:AVE

Function: Enables current average pressure value to be sent.

### **Query Syntax**

**:SENSe[:PRESsure]:AVERage?**

Short form: :SENS:AVE?

Function: Query average pressure value

Response: Current average pressure value.

### **Description**

This command query gets the average pressure value and the current pressure units.

## **:SENS:PRES:AVER:RES**

Applicability: PACE1000

### **Command Syntax**

**:SENSe[:PRESSure]:AVERage:RESet**

Parameter:

Short form: :SENS:AVE:RES

Function: Resets average, minimum and maximum pressure readings.

### **Query Syntax**

**n/a**

Short form:

Function:

Response:

### **Description**

This command resets the maximum, average and minimum pressure readings.

---

## **:SENS:PRES:AVER:TIME**

Applicability: PACE1000

### **Command Syntax**

**:SENSe[:PRESsure]:AVERage:TIME**

Parameter:

Short form: :SENS:AVE:TIME

Function: Sets the averaging function time period.

### **Query Syntax**

**n/a**

Short form:

Function:

Response:

### **Description**

This command sets the averaging function time period.

## **:SENS:PRES:CONT**

Applicability: PACE5000, PACE6000

### Command Syntax

**n/a**

Parameter:

Short form:

Function:

### Query Syntax

#### **:SENSe[x][:PRESsure]:CONTrol?**

*where x = 1 or 2 is the module number (default - 1)*

Normal form: :SENS:PRES:CONT?

Function: Queries the selected module control sensor pressure value.

Response: <number> in the selected units of pressure measurement.

### Description

Returns the selected module pressure measured by the control transducer. An alternative query to :SENS:PRES? that does not require preselection of the sensor using the :SENS:PRES:RANG command. To directly query the pressure values of the +ve and -ve pressure supply source and vacuum sensors refer to the :SOUR:PRES:COMP query.

Example:

TX> :SENS:PRES:CONT?

RX> :SENS:PRES:CONT 982.8430904 (this value in mbar).

## **:SENS:PRES:INL**

Applicability: PACE5000, PACE6000

### Command Syntax

**n/a**

Parameter:

Short form:

Function:

### Query Syntax

#### **:SENSe[x][:PRESsure]:INLimits?**

*where x = 1 or 2 is the module number (default - 1)*

Short form: :SENS:INL?

Function: Query in-limits value

Response: First parameter - current pressure.

Second parameter - in limit:

0 = not in limits

1 = in limits

### Description

This command query gets the in-limits value and the in-limits status:

Example 1:

TX> :SENS:PRES:INL?

RX> :SENS:PRES:INL 990.0527344, 0

Example 2:

TX> :SENS:PRES:INL?

RX> :SENS:PRES:INL 990.0527344, 1

## **:SENS:PRES:INL:TIME**

Applicability: PACE6000

### Command Syntax

**n/a**

Parameter:

Short form:

Function:

### Query Syntax

**:SENSe[x][:PRESsure]:INLimits:TIME?**

*where x = 1 or 2 is the module number (default - 1)*

Short form: :SENS:PRES:INL:TIME?

Function: Query pressure in-limits timers.

Response: Number representing the in-limits time value since last setpoint entry or controller Measure to Control state.

### Description

Sets the pressure in-limits timer value.

## **:SENS:PRES:SLEW**

Applicability: PACE1000, PACE5000, PACE6000

### Command Syntax

**n/a**

Parameter:

Short form:

Function:

### Query Syntax

#### **:SENSe[x][:PRESsure]:SLEW?**

*where x = 1 or 2 is the module number (default - 1)*

Short form: :SENS:SLEW?

Function: Asks for current slew rate.

Response: Decimal number representing slew rate always in current pressure units per second.

### Description

This query gets the slew rate of the input pressure always in current pressure units per second. A constant input pressure gives a slew rate of: 0.0.

Example 1:

```
TX> :SENS:PRES:SLEW?
```

```
RX> :SENS:PRES:SLEW -20.1089802
```

Example 2:

```
TX> :SENS2:PRES:SLEW?
```

```
RX> :SENS2:PRES:SLEW -20.0536633
```

## **:SENS:PRES:BAR**

Applicability: PACE1000, PACE5000, PACE6000

### Command Syntax

**n/a**

Parameter:

Short form:

Function:

### Query Syntax

#### **:SENSe[x][:PRESsure]:BARometer?**

*where x = 1 or 2 is the module number (default - 1)*

Short form: :SENS:BAR?

Function: Queries the barometric pressure value.

Response: <number> in the selected units of pressure measurement.

### Description

Returns the selected module barometric pressure measured by the optional barometric transducer. If the optional barometric transducer is not fitted, the response is zero pressure. PACE5000 and PACE6000 alternative query to :SENS:PRES? that does not require preselection of the sensor using the :SENS:PRES:RANG command. To directly query the pressure values of the +ve and -ve pressure supply source and vacuum sensors refer to the :SOUR:PRES:COMP query.

Example:

TX> :SENS:PRES:BAR?

RX> :SENS:PRES:BAR 982.8430904 (this value in mbar).



## :SENS:PRES:RANG

Applicability: PACE1000, PACE5000, PACE6000

### Command Syntax

**:SENSe[x][:PRESsure]:RANGe <string>**

where  $x = 1 = \text{module 1}$  or  $x = 2 = \text{module 2}$  (if  $x$  is blank the default is module 1)

Parameter: <string> range information

Short form: :SENS:RANG

Function: Selects an internal range.

### Query Syntax

**:SENSe[x][:PRESsure]:RANGe?**

where  $x = 1 = \text{module 1}$  or  $x = 2 = \text{module 2}$  (if  $x$  is blank the default is module 1)

Short form: :SENS:RANG?

Function: Asks for currently sensed range.

Response: A string (number), representing the selected pressure range in pressure units.

*Example: 3.50 barg*

### Description

This command selects an internal pressure range for use by the :SENS[x]:PRES? query to return a displayed or non-displayed pressure reading, and also in calibration mode for selecting the sensor to be calibrated. This command does not affect the pressure range selected for pressure control and measured pressure display purposes, nor does it affect the pressure control function.

Refer to :SOUR:PRES:RANG to change the range used for pressure control and measured pressure display.

Tx	Rx	Remarks
:INST:CAT2:ALL?	:INST:CAT2:ALL "3.50barg","BAROMETER","4.50bara"	Shows module 2 has three ranges.
:SENS2:PRES:RANG?	:SENS2:PRES:RANG "3.50barg"	Queries the current range selected by module 2.
:SENS:PRES:RANG "4.50bara"	None	Selects a different range of module 2 use the command.
:SENS2:PRES:RANG?	:SENS2:PRES:RANG "4.50bara"	Shows the 4.50bara range is selected.

## Chapter 4. Command and Query Summary

---

**Note:** The command parameter, "4.50bara", has to be typed exactly as returned by the :INST:CAT:ALL? command. It is case-sensitive and not typo-error tolerant. The command does not affect the front-panel display only the internal.

Use :SOUR:PRES:RANG to change front-panel range display. (See instrument user's manual).

## :SENS:PRES:RES

Applicability: PACE1000, PACE5000, PACE6000

### Command Syntax

**:SENSe[x][:PRESsure]:RESolution <string>**

where  $x = 1$  or  $2$  is the module number (default - 1)

Parameter: <integer> resolution information

Short form: :SENS:RES

Function: Used to select a resolution to be used.

### Query Syntax

**:SENSe[x][:PRESsure]:RESolution?**

where  $x = 1$  or  $2$  is the module number (default - 1)

Short form: :SENS:RES?

Function: Asks for current resolution.

Response: An integer from 4 to 7 representing selected resolution.

### Description

This command selects the resolution to be used for showing the pressure reading in the front panel display.

- Example 1
- ```
TX> :SENS:PRES:RES?
RX> :SENS:PRES:RES 6
```
- The resolution can be changed by the command, such as:
- ```
TX> :SENS:PRES:RES 4
```
- Example 2
- ```
TX> :SENS:PRES:RES?
RX> :SENS:PRES:RES 4
```
- Query again confirms that the resolution has been changed to '4' :
- Any resolution number outside the 4 to 7 boundary generates error messages, such as:
- ```
TX> :SENS2:PRES:RES 8
```
- Example 3
- ```
TX> :SYST:ERR?
RX> :SYST:ERR -222,"Data out of range; Parameter 1"
TX> :SENS2:PRES:RES 3
TX> :SYST:ERR?
RX> :SYST:ERR -222,"Data out of range; Parameter 1"
```

## **:SENS:PRES:CORR:HEAD**

Applicability: PACE1000, PACE5000, PACE6000

### Command Syntax

**:SENSe[x][:PRESsure]:CORRection:HEAD <enumerated>,<decimal>**

where x = 1 or 2 is the module number (default - 1)

|             |                                        |                                                          |
|-------------|----------------------------------------|----------------------------------------------------------|
| Parameter:  | <enumerated>                           | AIR - Air used as gas<br>NITRogen - Nitrogen used as gas |
|             | <numeric>                              | Height of gas in metres.                                 |
| Short form: | :SENS:CORR:HEAD <enumerated>,<numeric> |                                                          |
| Function:   | Head correction parameters.            |                                                          |
| Default     | Enumerated                             | AIR                                                      |
|             | decimal                                | 0                                                        |

### Query Syntax

**:SENSe[x][:PRESsure]:CORRection:HEAD?**

where x = 1 or 2 is the module number (default - 1)

|             |                                                   |
|-------------|---------------------------------------------------|
| Short form: | :SENS:CORR:HEAD?                                  |
| Function:   | Query gas and height of head correction.          |
| Response:   | AIR/NITRogen and height in metres (+100 to -100). |

### Description

A correction must be made if the unit under test is at a different height from the instrument.

This command programs the gas used and the height difference.

- |           |                                                                                                                       |
|-----------|-----------------------------------------------------------------------------------------------------------------------|
| Example 1 | TX> :SENS:PRES:CORR:HEAD?<br>RX> :SENS:PRES:CORR:HEAD AIR, -0.7500000                                                 |
| Example 2 | Head correction can be set to a new height by the command:<br>TX> :SENS:PRES:CORR:HEAD AIR, 1.2                       |
| Example 3 | Another query will confirm the height change:<br>TX> :SENS:PRES:CORR:HEAD?<br>RX> :SENS:PRES:CORR:HEAD AIR, 1.2000000 |
| Example 4 | Head correction can be set for another gas by:<br>TX> :SENS:PRES:CORR:HEAD NITROGEN, 1.2                              |

Example 5      Another query will confirm the height change:  
TX> :SENS:PRES:CORR:HEAD?  
RX> :SENS:PRES:CORR:HEAD NITR, 1.2000000

**Note:** NITROGEN or NITR is an enumerated data type (not in punctuation marks), not a string and is not case-sensitive and can have a short form.

## **:SENS:PRES:CORR:HEAD:STAT**

Applicability: PACE1000, PACE5000, PACE6000

### Command Syntax

**:SENSe[x][:PRESsure]:CORRection:HEAD:STATe <Boolean>**

where  $x = 1$  or  $2$  is the module number (default - 1)

Parameter: <boolean>  
0 - Disables head correction  
1 - Enables head correction

Short form: :SENS:CORR:HEAD:STAT <Boolean>

Function: Enables/disables head correction.

Default: 0

### Query Syntax

**:SENSe[x][:PRESsure]:CORRection:HEAD:STATe?**

where  $x = 1$  or  $2$  is the module number (default - 1)

Short form: :SENS:CORR:HEAD:STAT?

Function: Query head correction state

Response: 0 - head correction off  
1 - head correction on

### Description

This command enables or disables the head correction compensation.

Example:

```
TX> :SENS:PRES:CORR:HEAD:STATe?
```

```
RX> :SENS:PRES:CORR:HEAD:STAT 0
```

Head correction off

It can be turned on and off by:

```
TX> :SENS:PRES:CORR:HEAD:STATe on
```

```
TX> :SENS:PRES:CORR:HEAD:STATe off
```

Or by:

```
TX> :SENS:PRES:CORR:HEAD:STATe 1
```

```
TX> :SENS:PRES:CORR:HEAD:STATe 0
```

## :SENS:PRES:CORR:OFFS

Applicability: PACE1000, PACE5000, PACE6000

### Command Syntax

#### :SENSe[x][:PRESsure]:CORRection:OFFSet

where  $x = 1$  or  $2$  is the module number (default - 1)

Parameter: <decimal> tare offset value in current pressure units.

Short form: :SENS:OFFS

Function: Subtracts the offset value from the processed reading.

Default: 0

### Query Syntax

#### :SENSe[x][:PRESsure]:CORRection:OFFSet?

where  $x = 1$  or  $2$  is the module number (default - 1)

Short form: :SENS:OFFS?

Function: Asks for the tare value.

Response: Number corresponding to the tare offset value.

### Description

Example:

```
TX> :SENS:PRES:CORR:OFFS?
```

```
RX> :SENS:PRES:CORR:OFFS 0.0
```

Offset is zero.

Offset can be changed to 100 mbar by sending, (instrument must be in mbar):

```
TX> :SENS:PRES:CORR:OFFS 100
```

Send query again to confirm offset has changed:

```
TX> :SENS:PRES:CORR:OFFS?
```

```
RX> :SENS:PRES:CORR:OFFS 100.0000000
```

**Note:** The offset depends on the units of measurement used by the instrument. If changed from mbar to bar, the same query will be returned:

```
TX> :SENS:PRES:CORR:OFFS?
```

```
RX> :SENS:PRES:CORR:OFFS 0.1000000
```

## **:SENS:PRES:CORR:OFFS:STAT**

Applicability: PACE1000, PACE5000, PACE6000

### Command Syntax

**:SENSe[x][:PRESsure]:CORRection:OFFSet:STATe**

where  $x = 1$  or  $2$  is the module number (default - 1)

Parameter: <boolean>  
<boolean>  
0 - disables offset  
1 - enables offset

Short form: :SENS:OFFS:STAT

Function: Enables and disables the offset function.

### Query Syntax

**:SENSe[x][:PRESsure]:CORRection:HEAD:OFFSet:STATe?**

where  $x = 1$  or  $2$  is the module number (default - 1)

Short form: :SENS:OFFS:STAT?

Function: Asks if offset function is on or off.

Response: 1 (on)  
0 (off)

### Description

This command enables and disables the offset function. The query gets the state of the offset (or tare) on or off.

Example:

```
:SENS:PRES:CORR:OFFS:STATe
```

```
TX> :SENS:PRES:CORR:OFFS:STATe?
```

```
RX> :SENS:PRES:CORR:OFFS:STAT 0
```

The offset correction disabled.

Offset can be enabled and disabled by sending:

```
TX> :SENS:PRES:CORR:OFFS:STATe on
```

```
TX> :SENS:PRES:CORR:OFFS:STATe off
```

Or by:

```
TX> :SENS:PRES:CORR:OFFS:STATe 1
```

```
TX> :SENS:PRES:CORR:OFFS:STATe 0
```



## **:SENS:PRES:CORR:VOL**

Applicability: PACE5000, PACE6000

### Command Syntax

**n/a**

Parameter:

Short form:

Function:

### Query Syntax

#### **:SENSe[x][:PRESsure]:CORRection:VOLume?**

where  $x = 1$  or  $2$  is the module number (default - 1)

Short form: :SENS:PRES:CORR:VOL?

Function: Ask for the estimated volume of the system connected to the instrument.

Response: Decimal number in litres corresponding to volume.

### Description

The instrument calculates and reports the volume of the system connected.

Example:

TX> :SENS:PRES:CORR:VOL?

RX> :SENS:PRES:CORR:VOL 0.0150000

## **:SENS:PRES:FILT:LPAS:BAND**

Applicability: PACE1000, PACE5000, PACE6000

### Command Syntax

**:SENSe[x][:PRESsure]:FILTer:[LPASs]:BAND <number>**

where  $x = 1$  or  $2$  is the module number (default - 1)

Parameter: <decimal> filter band response value in % full-scale.

Short form: :SENS:FILT:BAND

Function: Used to set-up the response band component of the filter.

Default: 0

Minimum: 0

Maximum: 100.0

### Query Syntax

**:SENSe[x][:PRESsure]:FILTer:[LPASs]:BAND?**

where  $x = 1$  or  $2$  is the module number (default - 1)

Short form: :SENS:FILT:BAND?

Function: Ask for filter step response band parameter.

Response: Number corresponding to filter step response value in % full-scale.

### Description

The digital low pass filter has a response band configured as percentage of full-scale. e.g., defaults to 0.05%FS. If the reading has changed by more than the configured band response value then the filtering is ignored for that conversion and the pressure goes instantly to the new value.

Example:

```
TX> :SENS:PRES:FILT:LPAS:BAND?
```

```
RX> :SENS:PRES:FILT:LPAS:BAND 50.0000000
```

The current setting for the filter band is 50% of full-scale, i.e., the filter applies only when the change of pressure is within this band.

It can be changed to another number, (for example: 12% of full-scale), by sending:

```
TX> :SENS:PRES:FILT:LPAS:BAND 12
```

```
TX> :SENS:PRES:FILT:LPAS:BAND?
```

```
RX> :SENS:PRES:FILT:LPAS:BAND 12.0000000
```

## :SENS:PRES:FILT:LPAS:FREQ

Applicability: PACE1000, PACE5000, PACE6000

### Command Syntax

**:SENSe[x][:PRESsure]:FILTer:[LPASs]:FREQuency <number>**

where  $x = 1$  or  $2$  is the module number (default - 1)

Parameter: <decimal> filter averaging time in seconds.

Short form: :SENS:FILT:FREQ

Function: Used to set up the averaging component of the filter.

Default: 0

Minimum: 0

Maximum: 20

### Query Syntax

**:SENSe[x][:PRESsure]:FILTer:[LPASs]:FREQuency?**

where  $x = 1$  or  $2$  is the module number (default - 1)

Short form: :SENS:FILT:FREQ?

Function: Ask for filter average parameter.

Response: Decimal number corresponding to filter average time in seconds.

### Description

A digital low pass filter can be applied to the pressure reading. This is a first order low pass filter, the time constant depends on the value set by this command.

**Note:** The decimal number used by the command and query does not represent frequency even though 'FREQ' is used.

Example:

|                                                                                                                                                                                                |                                                                                                                                                                               |
|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <pre>:SENS:PRES:FILT? TX&gt; :SENS:PRES:FILT? RX&gt; :SENS:PRES:FILT:LPAS:STAT 0</pre>                                                                                                         | <p>It can be set to another value, such as:</p> <pre>TX&gt; :SENS:PRES:FILT:LPAS:FREQ 1.76 TX&gt; :SENS:PRES:FILT:LPAS:FREQ? RX&gt; :SENS:PRES:FILT:LPAS:FREQ 1.7600000</pre> |
| <pre>:SENS:PRES:FILT:LPAS:FREQ TX&gt; :SENS:PRES:FILT:LPAS:FREQ? RX&gt; :SENS:PRES:FILT:LPAS:FREQ 2.0000000</pre> <p>The current setting for low-pass filter's time constant is 2 seconds.</p> |                                                                                                                                                                               |

## **:SENS:PRES:FILT:LPAS:STAT**

Applicability: PACE1000, PACE5000, PACE6000

### Command Syntax

**:SENSe[x][:PRESsure]:FILTer[:LPASs]:[STATE] <Boolean>**

Parameter: <Boolean>  
0 - Disables low pass filter  
1 - Enables low pass filter

Short form: :SENS:FILT <Boolean>

Function: Sets low pass filter ON or OFF.

Default: OFF

### Query Syntax

**:SENSe[x][:PRESsure]:FILTer[:LPASs]:[STATE]?**

*where x = 1 or 2 is the module number (default - 1)*

Short form: :SENS:FILT?

Function: Query state (on or off) for the low pass filter

Response: 1 (ON) 0 (OFF)

### Description

This command is used to enable or disable the low pass filter for producing a more stable reading. An 'intelligent' filter is implemented so that any noise in the system is filtered while step changes pass straight through the filter.

Example:

TX> :SENS:PRES:FILT:LPAS:STAT?

RX> :SENS:PRES:FILT:LPAS:STAT 0

The filter is currently off.

It can be set on and off by

TX> :SENS:PRES:FILT:LPAS:STAT on

TX> :SENS:PRES:FILT:LPAS:STAT off

Or by

TX> :SENS:PRES:FILT:LPAS:STAT 1

TX> :SENS:PRES:FILT:LPAS:STAT 0

---

## **:SENS:PRES:MAX**

Applicability: PACE1000

### **Command Syntax**

**n/a**

Parameter:

Short form:

Function:

### **Query Syntax**

#### **:SENSe:[PRESsure]:MAXimum?**

Short form: :SENS:MAX?

Function: This query reads the maximum pressure.

Response: A reading of the current set maximum pressure in selected pressure units.

### **Description**

Queries the current maximum setting for pressure.

## **:SENS:PRES:MAX:RES**

Applicability: PACE1000

### Command Syntax

**SENSe:[PRESSure]:MAXimum:RESet**

Parameter:

Short form: :SENS:MAX:RES

Function: Resets maximum, average and minimum pressure readings.

### Query Syntax

**n/a**

Short form:

Function:

Response:

### Description

Command resets the maximum, average and minimum pressure readings.

---

## **:SENS:PRES:MIN**

Applicability: PACE1000

### **Command Syntax**

**n/a**

Parameter:

Short form:

Function:

### **Query Syntax**

#### **SENSe:PRESSure:MINimum?**

Short form: :SENS:MIN?

Function: This query reads the minimum pressure.

Response: A reading of the current set minimum pressure in selected pressure units.

### **Description**

Queries the current minimum setting for pressure.

## **:SENS:PRES:MIN:RES**

Applicability: PACE1000

### Command Syntax

**SENSe:[PRESSsure]:MINimum:RESet**

Parameter:

Short form: :SENS:MIN:RES

Function: Resets minimum, average and maximum pressure readings.

### Query Syntax

**n/a**

Short form:

Function:

Response:

### Description

This command resets minimum, average and maximum pressure readings.



---

## **:SENS:PRES:PERC**

Applicability: PACE1000

### **Command Syntax**

**n/a**

Parameter:

Short form:

Function:

### **Query Syntax**

**SENSe:[PRESsure]:PERCent?**

Short form: :SENS:PERC?

Function: This query reads the percentage pressure.

Response: A reading of the current set percentage pressure.

### **Description**

Queries the current percentage pressure reading.

## **:SENS:PRES:PERC:SPAN**

Applicability: PACE1000

### Command Syntax

#### **SENSe:[PRESsure]:PERCent:SPAN**

Parameter: NONE|FS|SPAN

Short form: SENS:PERC:SPAN

Function: Sets the percentage span.

### Query Syntax

#### **SENSe:[PRESsure]:PERCent:SPAN?**

Short form: SENS:PERC:SPAN?

Function: This query reads the percentage span.

Response: A reading of the current set percentage span.

### Description

Sets the percentage span: NONE|FS|SPAN. Queries the current percentage span setting.

---

## **:SENS:PRES:PERC:STAT**

Applicability: PACE1000

### **Command Syntax**

**n/a**

Parameter:

Short form:

Function:

### **Query Syntax**

#### **SENSe:[PRESsure]:PERCent:STATe?**

Short form: :SENS:PERC:STAT?

Function: This query reads the percentage state.

Response: NONE|FS|SPAN

### **Description**

Queries the current percentage state instrument returns NONE or FS or SPAN.

## **:SENS:PRES:PSE**

Applicability: PACE5000 and PACE6000

### **Command Syntax**

**n/a**

Parameter:

Short form:

Function:

### **Query Syntax**

#### **:SENSe[x][:PRESsure]:PSE?**

*where x = 1 or 2 is the module number (default - 1)*

Normal form: :SENS:PRES:PSE?

Function: Queries the selected module pseudo range pressure value.

Response: <number> in the selected units of pressure measurement.

### **Description**

Returns the selected module pseudo range pressure value. An alternative query to :SENS:PRES? that does not require preselection of the control sensor using the :SENS:PRES:RANG command. To directly query the pressure values of the +ve and -ve pressure supply source and vacuum sensors refer to the :SOUR:PRES:COMP query.

Example:

TX> :SENS:PRES:PSE?

RX> :SENS:PRES:PSE 982.8430904 (this value in mbar).

## **:SENS:PRES:REF**

Applicability: PACE5000 and PACE6000 CM3 modules only

### Command Syntax

**n/a**

Parameter:

Short form:

Function:

### Query Syntax

#### **:SENSe[x][:PRESsure]:REF?**

where  $x = 1$  or  $2$  is the module number (default - 1)

Normal form: :SENS:PRES:REF?

Function: Queries the selected module reference sensor pressure value.

Response: <number> in the selected units of pressure measurement.

### Description

Returns the selected module reference sensor pressure value. An alternative query to :SENS:PRES? that does not require preselection of the sensor using the :SENS:PRES:RANG command. To directly query the pressure values of the +ve and -ve pressure supply source and vacuum sensors refer to the :SOUR:PRES:COMP query.

Example:

TX> :SENS:PRES:REF?

RX> :SENS:PRES:REF 982.8430904 (this value in mbar).

## **:SENS:SPE**

Applicability: PACE1000, PACE6000

### **Command Syntax**

**n/a**

Parameter:

Short form:

Function:

### **Query Syntax**

#### **:SENSe:SPEed?**

Short form: :SENS:SPE?

Function: This query reads the aeronautical sensors.

Response: A decimal reading in currently selected airspeed units.

### **Description**

Queries the airspeed reading.

**Note:** If the instrument is not in aero mode it records an error: -221. Settings conflict.

## :SENS:SPE:INL

Applicability: PACE6000

### Command Syntax

**:SENSe[x][:SPEed]:INLimits <number>**

where  $x = 1$  or  $2$  is the module number (default - 1)

Parameter: <decimal>   airspeed in limits value as % full-scale

Short form:            :SENS:SPE:INL <number>

Function:             The instrument has a airspeed in-limits indicator. This can generate a service request when the airspeed value is within limits for a set time period.

Default:              0.01% full-scale

Minimum:             0% full-scale

Maximum:             100% full-scale

### Query Syntax

**:SENSe[x][:SPEed]:INLimits?**

where  $x = 1$  or  $2$  is the module number (default - 1)

Short form:            :SENS:SPE:INL?

Function:             Query airspeed in-limits value

Response:             Decimal representing airspeed in-limits value as % full-scale.

### Description

Sets the in-limits value.

**Note:** If the instrument is not in aero mode it records an error: -221. Settings conflict.

## **:SENS:SPE:INL:TIME**

Applicability: PACE6000

### Command Syntax

**:SENSe[x][:SPEed]:INLimits:TIME <number>**

Parameter: <integer>   airspeed in-limits time in seconds

Short form: :SENS:SPE:INL:TIME <number>

Function: Sets the time that the airspeed value has to be within limits before generating a service request.

Default:: 2 seconds

Minimum: 2 seconds

Maximum: 999 seconds

### Query Syntax

**:SENSe[x][:SPEed]:INLimits:TIME?**

*where x = 1 or 2 is the module number (default - 1)*

Short form: :SENS:SPE:INL:TIME?

Function: Query airspeed in-limits timers.

Response: Number representing airspeed in-limits time in seconds.

### Description

Sets the airspeed in-limits timer value.

**Note:** If the instrument is not in aero mode it records an error: -221. Settings conflict.



## **:SENS:SPE:RANG**

Applicability: PACE1000, PACE6000

### Command Syntax

#### **:SENSe[2]:SPEEd:RANGe**

where  $x = 1$  or  $2$  is the module number (default - 1)

Parameter:

Short form: :SENS:SPE:RANG

Function: Sets the airspeed range.

### Query Syntax

#### **:SENSe:SPEEd:SLEW?**

Short form: :SENS:SPE:RANG?

Function: This query reads the airspeed range.

Response: A string (number), representing the airspeed range in pressure units.  
example: 3.50 barg

### Description

Queries the airspeed range.

**Note:** If the instrument is not in aero mode it records an error: -221. Settings conflict.

## **:SENS:SPE:SLEW**

Applicability: PACE1000, PACE6000

### Command Syntax

#### **:SENSe[2]:SPEed:SLEW**

where x = 1 or 2 is the module number (default - 1)

Parameter:

Short form: :SENS:SPE:SLEW

Function: Sets the airspeed rate of change.

### Query Syntax

#### **:SENSe:SPE:SLEW?**

Short form: :SENS:SPE:SLEW?

Function: This query reads the airspeed rate of change.

Response: A decimal, airspeed reading/second.

### Description

Queries the airspeed rate of change reading.

**Note:** If the instrument is not in aero mode it records an error: -221. Settings conflict.

## SOURce

The SOURce subsystem controls the pressure output of the instrument.

### :SOUR:ALT

Applicability: PACE6000

#### Command Syntax

##### **:SOURce:ALTitude:LEVel:IMMediate:AMPLtude**

Parameter: <string>

Short form: :SOUR:ALT

Function: Sets the altitude set-point.

Default: -

#### Query Syntax

##### **:SOURce:ALTitude:LEVel:IMMediate:AMPLtude?**

Short form: :SOUR:ALT?

Function: Queries altitude set-point.

Response: Decimal number representing altitude in current units.

#### Description

This command sets the altitude set-point. This query gets the decimal number representing altitude in current units.

**Note:** If the instrument is not in aero mode it returns an error: -221. Settings conflict.

## **:SOUR:MACH:REF**

Applicability: PACE6000

### **Command Syntax**

**n/a**

Parameter:

Short form:

Function:

### **Query Syntax**

#### **:SOURce:MACH:REFerence?**

Short form: :SOUR:MACH:REF

Function: Queries the MACH reference value from either Ps reference or barometric.

Response: Decimal number representing the MACH reference value in currently selected pressure units.

### **Description**

The query gets the MACH reference value in currently selected pressure units.

**Note:** If the instrument is not in aero mode it returns an error: -221. Settings conflict.

## **:SOUR:MACH:REF:MODE**

Applicability: PACE6000

### Command Syntax

#### **:SOURce:MACH:REFerence:MODE**

Parameter: <number>

Short form: :SOUR:MACH:REF:MODE

Function: Enables a MACH reference mode to be set.

Default: -

### Query Syntax

#### **:SOURce:MACH:REFerence:MODE?**

Short form: :SOUR:MACH:REF:MODE?

Function: Queries the MACH reference mode.

Response: Numeric value representing the MACH reference mode.

### Description

This command enables a MACH reference mode to be set. The query gets the MACH reference entered mode.

**Note:** If the instrument is not in aero mode it returns an error: -221. Settings conflict.

## **:SOUR:MACH:REF:VAL**

Applicability: PACE6000

### Command Syntax

#### **:SOURce:MACH:REFerence:VALue**

Parameter: <string>

Short form: :SOUR:MACH:REF:VAL

Function: Sets the MACH reference entered value.

Default: -

### Query Syntax

#### **:SOURce:MACH:REFerence:VALue?**

Short form: :SOUR:MACH:REF:VAL?

Function: Queries the MACH reference entered value from either Ps reference.

Response: Numeric value of the MACH reference.

### Description

This command sets the MACH reference entered value from either Ps reference. The query gets the MACH reference entered value.

**Note:** If the instrument is not in aero mode it returns an error: -221. Settings conflict.

## **:SOUR:MACH:LEV:IMM:AMPL**

Applicability: PACE6000

### Command Syntax

**:SOURce[x]:MACH[:LEVel][:IMMediate][:AMPLitude] <number>**

where  $x = 1$  or  $2$  is the module number (default - 1)

Parameter: <decimal> in MACH number.

Short form: :SOURce:MACH <number>

Function: Sets the MACH set-point

Default: 0.0

### Query Syntax

**:SOURce[x]:MACH[:LEVel][:IMMediate][:AMPLitude] <number>?**

Short form: :SOURce:MACH?

Function: Programmable set-point value

Response: Decimal number representing the set-point in MACH number.

### Description

This command sets the MACH set-point. This query gets the decimal number representing the set-point in MACH number.

**Note:** If the instrument is not in aero mode it returns an error: -221. Settings conflict.





## :SOUR:PRES:EFF

Applicability: PACE5000, PACE6000

### Command Syntax

n/a

Parameter:

Short form:

Function:

Default:

### Query Syntax

#### :SOURce[x][:PRESsure]:EFFort?

where  $x = 1$  or  $2$  is the module number (default - 1)

Short form: :SOUR:EFF?

Function: This query only command returns the effort needed for the controller to achieve the set-point.

Response: Decimal percent number representing controller effort.

### Description

This query only command returns the % effort the controller does to achieve the set-point.

#### Example

```
TX> :SOUR:PRES:EFF?
```

```
RX> :SOUR:PRES:EFF -0.2342882
```

Percentage of effort the supply or vacuum valve makes to maintain the control point. The return number should be -100 to +100. A positive number indicates the supply valve makes more effort and a minus number indicates the vacuum valve makes more effort. If the controller is off:

```
TX> :OUTP off
```

```
TX> :SOUR:PRES:EFF?
```

```
RX> :SOUR:PRES:EFF 0.0
```

The return number is '0.0', since no valve is making any effort.

## **:SOUR:PRES:INL**

Applicability: PACE5000, PACE6000

### Command Syntax

**:SOURce[x][:PRESsure]:INLimits <number>**

where x = 1 or 2 is the module number (default - 1)

Parameter: <decimal> in limits value as % full-scale

Short form: :SOUR:INL <number>

Function: The controller has an in-limits set-point indicator. This can generate a service request when the pressure is within limits for a set time period.

Default: 0.01% full-scale

Minimum: 0.0001% full-scale

Maximum: 10% full-scale

### Query Syntax

**:SOURce[x][:PRESsure]:INLimits?**

where x = 1 or 2 is the module number (default - 1)

Short form: :SOUR:INL?

Function: Query in-limits value

Response: Decimal representing in-limits value as % full-scale.

### Description

Sets the in-limits value.

#### Example

```
TX> :SOUR:PRES:INL?
```

```
RX> :SOUR:PRES:INL 0.0200000
```

The current in-limits set-point is 0.02% full-scale

Can be set to:

```
TX> :SOUR:PRES:INL 0.01
```

```
TX> :SOUR:PRES:INL?
```

```
RX> :SOUR:PRES:INL 0.0100000
```

## :SOUR:PRES:INL:TIME

Applicability: PACE5000, PACE6000

### Command Syntax

**:SOURce[x][:PRESsure]:INLimits:TIME <number>**

where  $x = 1$  or  $2$  is the module number (default - 1)

Parameter: <integer> in-limits time in seconds

Short form: :SOUR:INL:TIME <number>

Function: Sets the time that the pressure has to be within limits before generating a service request.

Default: 1 second

Minimum: 1 second

Maximum: 60 seconds

### Query Syntax

**:SOURce[x][:PRESsure]:INLimits:TIME?**

where  $x = 1$  or  $2$  is the module number (default - 1)

Short form: :SOUR:INL:TIME?

Function: Query in-limits timers.

Response: Number representing in-limits time in seconds.

### Description

Sets the in-limits timer value.

#### Example

TX> :SOUR:PRES:INL:TIME?

RX> :SOUR:PRES:INL:TIME 1

Can be set to 99 by issuing command:

TX> :SOUR:PRES:INL:TIME 9

TX> :SOUR:PRES:INL:TIME?

RX> :SOUR:PRES:INL:TIME 9

Or to 999 by issuing command:

TX> :SOUR:PRES:INL:TIME 59

TX> :SOUR:PRES:INL:TIME?

RX> :SOUR:PRES:INL:TIME 59

## **:SOUR:PRES:LEV:IMM:AMPL**

Applicability: PACE5000, PACE6000

### Command Syntax

**:SOURce[x][:PRESsure][:LEVel][:IMMediate][:AMPLitude] <number>**

where  $x = 1$  or  $2$  is the module number (default - 1)

Parameter: <decimal> Pressure in current units

Short form: :SOUR <number>

Function: Set the pressure set-point

Default: 0.0

### Query Syntax

**:SOURce[x][:PRESsure][:LEVel][:IMMediate][:AMPLitude]?**

where  $x = 1$  or  $2$  is the module number (default - 1)

Short form: :SOUR?

Function: Programmable set-point value

Response: Decimal number representing pressure set-point in current units.

### Description

This command sets the pressure set-point and is the long form of :SOUR:PRES.

#### Example

TX> :SOUR:PRES?

RX> :SOUR:PRES:LEV:IMM:AMPL 0.4000000

The current controller set-point is 0.4 of the current unit.

It can be set to another value, such as:

TX> :SOUR:PRES 0.5

TX> :SOUR:PRES?

RX> :SOUR:PRES:LEV:IMM:AMPL 0.5000000

## **:SOUR:PRES:LEV:IMM:AMPL:VENT**

Applicability: PACE5000, PACE6000

### Command Syntax

**:SOURce[x][:PRESsure][:LEVel][:IMMediate][:AMPLitude]:VENT**<space> <Parameter>

where  $x = 1 = \text{module 1}$  or  $x = 2 = \text{module 2}$  (if  $x$  is blank the default is module 1)

Parameter: <integer>  
 0 - abort vent  
 1 - start vent

Normal form: :SOUR[x]:PRES:LEV:IMM:AMPL:VENT <parameter>

Short form: :SOUR[x]:VENT <parameter>

Function: Vents the user system.

Default: 0

### Query Syntax

**:SOURce[x][:PRESsure][:LEVel][:IMMediate][:AMPLitude]:VENT?**

where  $x = 1 = \text{module 1}$  or  $x = 2 = \text{module 2}$  (if  $x$  is blank the default is module 1)

Normal form: :SOUR[x]:PRES:LEV:IMM:AMPL:VENT?

Short form: :SOUR[x]:VENT?

Function: Query status of vent.

Response: 0 - vent OK  
 1 - vent in progress  
 2 - vent completed

### Description

This command vents the user system; the command should be queried to get the status of the vent.

#### Examples

For an instrument switched on without any vent, a query:

TX> :SOUR:PRES:LEV:IMM:AMPL:VENT?

RX> :SOUR:PRES:LEV:IMM:AMPL:VENT 0 (a return of '0' indicates vent not in progress).

A vent can be started by sending:

TX> :SOUR:PRES:LEV:IMM:AMPL:VENT 1

Immediate query:

TX> :SOUR:PRES:LEV:IMM:AMPL:VENT?

## Chapter 4. Command and Query Summary

---

RX> :SOUR:PRES:LEV:IMM:AMPL:VENT 1

A return of '1' indicates vent in progress

TX> :SOUR:PRES:LEV:IMM:AMPL:VENT?

RX> :SOUR:PRES:LEV:IMM:AMPL:VENT 1

TX> :SOUR:PRES:LEV:IMM:AMPL:VENT?

RX> :SOUR:PRES:LEV:IMM:AMPL:VENT 1

TX> :SOUR:PRES:LEV:IMM:AMPL:VENT?

RX> :SOUR:PRES:LEV:IMM:AMPL:VENT 1

TX> :SOUR:PRES:LEV:IMM:AMPL:VENT?

RX> :SOUR:PRES:LEV:IMM:AMPL:VENT 1

Continuing queries return '1', until the vent finishes and a return of '2':

TX> :SOUR:PRES:LEV:IMM:AMPL:VENT?

RX> :SOUR:PRES:LEV:IMM:AMPL:VENT 2

TX> :SOUR:PRES:LEV:IMM:AMPL:VENT?

RX> :SOUR:PRES:LEV:IMM:AMPL:VENT 2

TX> :SOUR:PRES:LEV:IMM:AMPL:VENT?

RX> :SOUR:PRES:LEV:IMM:AMPL:VENT 2

TX> :SOUR:PRES:LEV:IMM:AMPL:VENT?

RX> :SOUR:PRES:LEV:IMM:AMPL:VENT 2

To abort the vent process, send:

TX> :SOUR:PRES:LEV:IMM:AMPL:VENT 0

## :SOUR:PRES:RANG

Applicability: PACE5000, PACE6000

### Command Syntax

**:SOURce[x][:PRESsure]:RANGe<space><"string">**

where  $x = 1 = \text{module 1}$  or  $x = 2 = \text{module 2}$  (if  $x$  is blank the default is module 1)

Parameter: <string> range information

Short form: :SOUR:RANG

Function: Selects the control range.

### Query Syntax

**:SOURce[x][:PRESsure]:RANGe?**

where  $x = 1 = \text{module 1}$  or  $x = 2 = \text{module 2}$  (if  $x$  is blank the default is module 1)

Normal form: :SOUR[x]:PRES:RANG?

Short form: :SOUR[x]:RANG?

Function: Asks for currently selected control module range. Response: String representing selected sense range.

### Description

This command selects the range to be used for controlling pressure.

e.g. :SOUR:PRES:RANG "2.0bara"

selects the 2 bar absolute range; the pressure units are always in bar.

**Note:** Absolute ranges can be true absolute or pseudo-absolute ranges. Gauge ranges can be true gauge ranges or pseudo-gauge ranges, refer to commands ":INST:CAT:ALL?" and ":INST:CONTx:SENSy?".

Example:

TX> :SOUR:PRES:RANG?

RX> :SOUR:PRES:RANG "3.50barg"

Queries module 1 current displayed measure and control range.

Sets module 2 to another range and then queries the current range selected:

TX> :SOUR2:PRES:RANG "4.50bara"

TX> :SOUR2:PRES:RANG?

RX> :SOUR2:PRES:RANG "4.50bara"

Changes are observed on front panel display. The string parameter is case sensitive.

## **:SOUR:PRES:RANG:LOW**

Applicability: PACE5000, PACE6000

### Command Syntax

**n/a**

Parameter:

Short form:

Function:

Default:

### Query Syntax

#### **:SOURce[x]:PRESsure:RANGe:LOW?**

*where x = 1 or 2 is the module number (default - 1)*

Short form: :SOUR:LOW?

Function: Queries the lower full-scale of the module [x].

Response: Decimal number in current selected units.

### Description

This query gets the lower full-scale of the module [x].

Example 1 TX> :SOUR:PRES:RANG:LOW?  
RX> :SOUR:PRES:RANG:LOW "-1000mbar g"

Example 2 TX> :SOUR1:PRES:RANG:LOW?  
RX> :SOUR:PRES:RANG:LOW "-1000mbar g"



## :SOUR:PRES:SLEW

Applicability: PACE5000, PACE6000

### Command Syntax

**:SOURce[x][:PRESsure]:SLEW <number>**

where  $x = 1$  or  $2$  is the module number (default - 1)

Parameter: <decimal> rate in pressure units/second or enumerated string maximum or minimum rate.

Short form: :SOUR:SLEW <number> or maximum or minimum rate

Function: Selects the pressure rate used when value rate is selected.

Default: 100

### Query Syntax

**:SOURce[x][:PRESsure]:SLEW?**

where  $x = 1$  or  $2$  is the module number (default - 1)

Short form: :SOUR:SLEW?

Function: Query rate value

Response: Decimal number representing rate value in selected units/second.

### Description

When the controller rate is selected as value, this command is used to set the controllers rate in selected units/second.

#### Example

```
TX> :SOUR:PRES:SLEW?
```

```
RX> :SOUR:PRES:SLEW 2.0000000
```

Change current unit of mbar to bar and query again:

```
TX> :SOUR:PRES:SLEW?
```

```
RX> :SOUR:PRES:SLEW 0.0020000
```

To set slew rate to other values, such as:

```
TX> :SOUR:PRES:SLEW 4
```

```
TX> :SOUR:PRES:SLEW?
```

```
RX> :SOUR:PRES:SLEW 4.0000000
```

```
TX> :SOUR:PRES:SLEW max
```

```
TX> :SOUR:PRES:SLEW?
```

```
RX> :SOUR:PRES:SLEW 99999999.0000000
```

## Chapter 4. Command and Query Summary

---

TX> :SOUR:PRES:SLEW min  
TX> :SOUR:PRES:SLEW?  
RX> :SOUR:PRES:SLEW 0.0

## :SOUR:PRES:SLEW:MODE

Applicability: PACE5000, PACE6000

### Command Syntax

**:SOURce[x][:PRESsure]:SLEW:MODE <enumerated>**

where  $x = 1$  or  $2$  is the module number (default - 1)

Parameter: <enumerated>  
 MAXimum - maximum rate  
 LINear - user selected linear rate

Short form: :SOUR:SLEW:MODE <enumerated>

Function: Select the rate the controller should use to achieve set-point.

Default: MAXimum

### Query Syntax

**:SOURce[x][:PRESsure]:SLEW:MODE?**

where  $x = 1$  or  $2$  is the module number (default - 1)

Short form: :SOUR:SLEW:MODE?

Function: Query rate mode.

Response: MAX for maximum rate  
 LIN for user defined rate

### Description

The controller can operate in two rate modes - maximum and value. In maximum rate the controller tries to achieve set-point as quickly as possible. In value mode the controller achieves the set-point at a user selected rate.

Example 1 TX> :SOUR:PRES:SLEW:MODE?  
 RX> :SOUR:PRES:SLEW:MODE MAX  
 Can be set to user defined rate:

Example 2 TX> :SOUR:PRES:SLEW:MODE linear  
 TX> :SOUR:PRES:SLEW:MODE?  
 RX> :SOUR:PRES:SLEW:MODE LIN

## **:SOUR:PRES:SLEW:OVER**

Applicability: PACE5000, PACE6000

### Command Syntax

**:SOURce[x][:PRESsure]:SLEW:OVERshoot[:STATe] <Boolean>**

where  $x = 1$  or  $2$  is the module number (default - 1)

Parameter: <boolean>  
0 - overshoot not allowed  
1 - overshoot allowed

Short form: :SOUR:SLEW:OVER <Boolean>

Function: Selects pressure overshoot to 'allowed' or 'not allowed'.

Default: 1 - overshoot allowed

### Query Syntax

**:SOURce[x][:PRESsure]:SLEW:OVERshoot[:STATe]?**

where  $x = 1$  or  $2$  is the module number (default - 1)

Short form: :SOUR:SLEW:OVER?

Function: Query overshoot state

Response: 0 - overshoot not allowed  
1 - overshoot allowed

### Description

The controller can reach the set-point in one of two modes:

**Overshoot 'not allowed'**, the controller changes the pressure to near the set-point. The rate of pressure change slows when approaching the set-point to avoid overshoot.

**Overshoot 'allowed'**, the controller achieves set-point as fast as possible and, when approaching the set-point, may overshoot or undershoot.

Example 1 TX> :SOUR:PRES:SLEW:OVER?  
RX> :SOUR:PRES:SLEW:OVER:STAT 0

Example 2 TX> :SOUR:PRES:SLEW:OVER 1  
RX> :SOUR:PRES:SLEW:OVER:STAT 1

## :SOUR:SPE:LEV:IMM:AMPL

Applicability: PACE6000

### Command Syntax

**:SOURce[x]:SPEed[:LEVel][:IMMediate][:AMPLitude] <number>**

where  $x = 1$  or  $2$  is the module number (default - 1)

Parameter: <decimal>    Airspeed in current units

Short form:            :SOUR:SPE <number>

Function:             Sets the airspeed set-point

Default:              0.0

### Query Syntax

**:SOURce[x]:SPEed[:LEVel][:IMMediate][:AMPLitude]?**

where  $x = 1$  or  $2$  is the module number (default - 1)

Short form:            :SOUR:SPE?

Function:             Programmable set-point value

Response:             Decimal number representing airspeed set-point in current units.

### Description

This command sets the airspeed set-point. This query gets the decimal number representing airspeed set-point in current units.

**Note:** If the instrument is not in aero mode it returns an error: -221. Settings conflict.

## **:SOUR:SPE:LEV:IMM:AMPL:SLEW**

Applicability: PACE6000

### Command Syntax

**:SOURce[x]:SPEed[:LEVel][:IMMediate][:AMPLitude]:SLEW<number>**

*where x = 1 or 2 is the module number (default - 1)*

Parameter: <decimal> Airspeed rate set-point in current units per second.

Short form: :SOUR:SPE:SLEW <number>

Function: Sets the rate set-point for airspeed.

Default: 0.0

### Query Syntax

**:SOURce[x]:SPEed[:LEVel][:IMMediate][:AMPLitude]:SLEW?**

*where x = 1 or 2 is the module number (default - 1)*

Short form: :SOUR:SPE:SLEW?

Function: Queries the rate set-point for airspeed.

Response: Decimal number representing the rate set-point airspeed in currently selected units of speed per second.

### Description

This command sets the airspeed rate set-point. The query gets the rate set-point for airspeed.

**Note:** If the instrument is not in aero mode it returns an error: -221. Settings conflict.

## STATus

The STATus subsystem supports the OPERation status register as defined in SCPI protocol.

### :STAT:OPER:COND

Applicability: PACE1000, PACE5000, PACE6000

#### Command Syntax

n/a

Parameter:

Short form:

Function:

#### Query Syntax

**:STATus:OPERation:CONDition?**

Short form: :STAT:OPER:COND?

Function: Query condition register

Response: Contents of condition register

#### Description

Returns the contents of the 16 bit condition register, see section on status reporting.

## **:STAT:OPER:ENAB**

Applicability: PACE1000, PACE5000, PACE6000

### Command Syntax

**:STATus:OPERation:ENABLE <integer>**

Parameter: <integer> 16 bit value to set enable bits

Short form: STAT:OPER:ENAB <integer>

Function: Controls the status operation enable register.

Default: 0

Minimum: 0

Maximum: 32767

### Query Syntax

**:STATus:OPERation:ENABLE?**

Short form: :STAT:OPER:ENAB?

Function: Query enable register

Response: 16 bit value of enable register.

### Description

Controls the bits that pass through the status reporting system, see status reporting section.



---

## **:STAT:OPER:EVEN**

Applicability: PACE1000, PACE5000, PACE6000

### **Command Syntax**

**n/a**

Parameter:

Short form:

Function:

Default:

### **Query Syntax**

#### **:STATus:OPERation:[EVENT]?**

Short form: :STAT:OPER:EVEN?

Function: Query event register

Response: 16 bit value of event register.

### **Description**

Reads contents of event register, see status reporting section.

## **:STAT:OPER:PRES:COND**

Applicability: PACE1000, PACE5000, PACE6000

### Command Syntax

**n/a**

Parameter:

Short form:

Function:

Default:

### Query Syntax

#### **:STATus:PRESSure:OPERation:CONDition?**

Short form: :STAT:OPER:PRES:COND?

Function: Reads the contents of the pressure condition register.

Response: Contents of pressure condition register.

### Description

Returns the contents of the 16 bit pressure condition register, see section on status reporting.

## **:STAT:OPER:PRES:ENAB**

Applicability: PACE1000, PACE5000, PACE6000

### Command Syntax

**:STATus:OPERation:PRESSure:ENABLE <integer>**

Parameter: <integer> 16 bit value to set pressure enable bits

Short form: STAT:OPER:PRES:ENAB <integer>

Function: Controls the pressure status operation enable register.

Default: 0

Minimum: 0

Maximum: 32767

### Query Syntax

**:STATus:OPERation:PRESSure:ENABLE?**

Short form: :STAT:OPER:PRES:ENAB?

Function: Query enable register.

Response: 16 bit value of pressure enable register.

### Description

Controls the bits that pass through the status reporting system, see status reporting section.

## **:STAT:OPER:PRES:EVEN**

Applicability: PACE5000

### Command Syntax

**n/a**

Parameter:

Short form:

Function:

### Query Syntax

#### **:STATus:OPERation:PRESSure[:EVENT]?**

Short form: :STAT:OPER:PRES?

Function: Reads contents of pressure event register

Response: 16 bit value of pressure event register.

### Description

Reads contents of pressure event register, see status reporting section.

Status System Examples:

Example \*SRE and \*STB

|          | Explanation:                                               |
|----------|------------------------------------------------------------|
| *SRE 255 | To enable every bit of SRE register;                       |
| FRED     | Deliberately type a wrong command, such as;                |
| *STB?    | Now queries Status Byte;                                   |
| *STB 68  | Error Message set EAV and SERIAL POLL set MSS, 22+26 = 68; |
| *STB 0   | Previous read STB clears it.                               |

Refer to Figure 3-1.

Example of Pressure Event generated SRQ

|                                |                                     |
|--------------------------------|-------------------------------------|
| TX> *SRE 128                   | Enable SRE bit 7                    |
| TX> :STAT:OPER:ENAB 1024       | Enable Operation Register, bit 10   |
| TX> :STAT:OPER:PRES:ENAB 32767 | Enable Pressure register all 16 bit |
| TX> :STAT:OPER:PRES:EVEN?      | Query Pressure Event                |
| RX> :STAT:OPER:PRES:EVEN 0     | No any even yet                     |
| TX> :SENS:PRES?                | What's the controller pressure now? |
| RX> :SENS:PRES 1099.9993896    | It is 1100 mbar                     |
| TX> :OUTP 1                    | Switch controller on                |

|     |                        |                                         |
|-----|------------------------|-----------------------------------------|
| TX> | :SOUR:PRES 2000        | To generate an in-limit event           |
| RX> | :SRQ 192               | Receive an SRQ automatically            |
| TX> | :STAT:OPER:PRES:EVEN?  | What happened to Pressure event?        |
| RX> | :STAT:OPER:PRES:EVEN 4 | In-limit event happened, see Table 3-2  |
| TX> | :STAT:OPER:PRES:EVEN?  | Ask again will clear the event register |
| RX> | :STAT:OPER:PRES:EVEN 0 | It is cleared indeed                    |

Example of \*IDN?

|     |                                                      |
|-----|------------------------------------------------------|
| TX> | *IDN?                                                |
| RX> | *IDN GE Druck,Pace5000 User Interface,58784,01.05.04 |

## **:STAT:PRES**

Applicability: PACE1000, PACE5000, PACE6000

### **Command Syntax**

#### **:STATus:PRESet**

Parameter: <integer>

Short form: :STAT:PRES

Function: Enables the preset values to be used.

### **Query Syntax**

**n/a**

Short form:

Function:

Response:

### **Description**

Enables the 25 preset values of set-point to be used.

---

## **:STAT:QUES:COND**

Applicability: PACE1000, PACE5000, PACE6000

### **Command Syntax**

**n/a**

Parameter:

Short form:

Function:

### **Query Syntax**

#### **:STATus:QUEStionable:CONDition?**

Short form: :STAT:QUES:COND?

Function: Query questionable data condition register

Response: Contents of questionable data condition register

### **Description**

Returns the contents of the 16 bit questionable data condition register, see section on status reporting.

## **:STAT:QUES:ENAB**

Applicability: PACE1000, PACE5000, PACE6000

### Command Syntax

**:STATus:QUESTionable:ENABle <integer>**

Parameter: <integer> 16 bit value to set enable bits

Short form: STAT:OPER:ENAB <integer>

Function: Controls the questionable data condition enable register.

Default: 0

Minimum: 0

Maximum: 32767

### Query Syntax

**:STATus:QUESTionable:ENABle?**

Short form: :STAT:OPER:ENAB?

Function: Query questionable data condition enable register.

Response: 16 bit value of enable register.

### Description

Controls the bits that pass through the questionable data condition reporting system, see status reporting section.



## **:STAT:QUES:EVENT**

Applicability: PACE1000, PACE5000, PACE6000

### **Command Syntax**

**n/a**

Parameter:

Short form:

Function:

### **Query Syntax**

#### **:STATus:QUEStionable:[EVENT]?**

Short form: :STAT:QUES?

Function: Query questionable data condition register

Response: 16 bit value of questionable data condition register.

### **Description**

Reads contents of questionable data condition register, see status reporting section.

## **SYSTem**

The SYSTem subsystem consists of general purpose commands.

### **:SYST:COMM:USB**

Applicability: PACE1000

#### **Command Syntax**

##### **:SYSTem:COMMunicate:USB**

Parameter: <boolean>  
                  0 - mass storage  
                  1 - communication

Short form: :SYST:COMM:USB

Function: Switches the communication mode between mass storage to communications.

#### **Query Syntax**

##### **:SYSTem:COMMunicate:USB?**

Short form: :SYST:COMM:USB?  
Function: Gets the communication mode  
Response: 0 - mass storage  
          1 - communication

## :SYST:ERR

Applicability: PACE1000, PACE5000, PACE6000

### Command Syntax

n/a

Parameter:

Short form:

Function:

### Query Syntax

#### :SYStem:ERRor?

Short form: :SYST:ERR?

Function: Gets next error from the error queue.

Response:

### Description

This command queries the system error queue which holds up to five errors. The instrument returns the error code "0" and message "No error" when no more errors are in the queue.

Example:

| Tx               | Rx                                                | Remarks                       |
|------------------|---------------------------------------------------|-------------------------------|
| :SYST:ERR?       | :SYST:ERR 0, No error                             |                               |
| :CAL1:PRES:POIN? | None                                              |                               |
| :SYST:ERR?       | :SYST:ERR -203,"Access error; Incorrect password" | Calibration mode not enabled  |
| :SYST:ERR?       | :SYST:ERR 0, No error                             | Not errors, error queue empty |

To test the system, send a wrong command, such as:

```
TX> :SENS:PRES qwer
```

Ask for error:

```
TX> :SYST:ERR?
```

```
RX> :SYST:ERR -200,"Execution error;Query or command violation"
```

This query empties the error stack, another query returns "No error".

```
TX> :SYST:ERR?
```

## **:SYST:DATE**

Applicability: PACE1000, PACE5000, PACE6000

### **Command Syntax**

#### **:SYSTEM:DATE**

Parameter: Integer in date format.

Short form: :SYST:DATE

Function: Sets the date.

### **Query Syntax**

#### **:SYSTEM:DATE?**

Short form: :SYST:DATE?

Function: Queries date

Response: Returns date setting.

### **Description**

Replies with the date settings in year, month, and day order.

Example:

The system date can be set by:

```
TX> :SYST:DATE 9, 5, 2
```

```
TX> :SYST:DATE?
```

```
RX> :SYST:DATE 9, 5, 2
```

```
TX> :SYST:DATE?
```

```
RX> :SYST:DATE 9, 4, 1
```

It returns current date in year, month, and day order.

## :SYST:SET

Applicability: PACE5000, PACE6000

### Command Syntax

#### :SYSTem:SET[x]

where  $x = 1$  or  $2$  is the module number (default - 1)

Parameter: Enumerated (measure or control) and a decimal number of set-point.

Short form: :SYST:SET

Function: This command only effective at switch-on condition and can set the controller on with a set-point.

### Query Syntax

#### :SYSTem:SET[x]?

where  $x = 1$  or  $2$  is the module number (default - 1)

Short form: :SYST:SET?

Function: Queries current system setting.

Response: Returns current settings in either measure or control with current set-point value.

### Description

#### Example

```
TX> :SYST:SET?
```

```
RX> :SYST:SET MEAS, 0.0
```

At the current system set, when the instrument is switched on, the controller will be in measurement mode and the set-point will be zero.

This command can set the instrument to control mode, immediately after switch-on, with a set-point to 100 mbar, by sending:

```
TX> :SYST:SET CONT, 100.0
```

```
TX> :SYST:SET?
```

```
RX> :SYST:SET CONT, 100.0000000
```

**Note:** The controller will not change mode without a switch on/off process. During normal operations use the following command example:

```
TX> :SOUR 2000
```

```
TX> :OUTP 1
```

This switches on the controller immediately with a set-point of 2000 mbar.

## **:SYST:TIME**

Applicability: PACE1000, PACE5000, PACE6000

### Command Syntax

#### **:SYSTem:TIME**

Parameter: Integer in time format.

Short form: :SYST:TIME

Function: Sets the time.

### Query Syntax

#### **:SYSTem:TIME?**

Short form: :SYST:TIME?

Function: Returns current time in hour, minute and second.

Response: Sends in order: hour, minute and second.

### Description

This command sets the time. The command query returns the current time in hour, minute and second order.

#### Example

TX> :SYST:TIME?

RX> :SYST:TIME 10, 8, 44

It returns current time in hour, minute and second order.

Time can be reset by:

TX> :SYST:TIME 11, 11, 2

TX> :SYST:TIME?

RX> :SYST:TIME 11, 11, 4

## **:SYST:COMM:SER:CONT:RTS**

Applicability: PACE1000, PACE5000, PACE6000

### Command Syntax

#### **:SYSTem:COMMunicate:SERial:CONTrol:RTS**

Parameter: Boolean, ON or OFF

Short form: :SYST:COMM:SER:RTS[x]

Function: Sets the serial communication hardware handshaking where:

[x] =

0: OFF

1: ON

### Query Syntax

#### **:SYSTem:COMMunicate:SERial:CONTrol:RTS?**

Short form: :SYST:COMM:SER:CONT:RTS?

Function: Queries serial communication handshaking.

Response: Returns 0 or 1.

### Description

Selects serial communications hardware handshaking. The command query requests the current serial communication hardware handshaking.

|            |                               |                           |
|------------|-------------------------------|---------------------------|
| Example: 1 | TX> :SYST:COMM:SER:CONT:RTS?  |                           |
|            | RX> :SYST:COMM:SER:CONT:RTS 0 | Hardware handshaking OFF. |
| Example: 2 | TX> :SYST:COMM:SER:CONT:RTS1  | Hardware handshaking ON.  |
|            | TX> :SYST:COMM:SER:CONT:RTS?  | Query again               |
|            | RX> :SYST:COMM:SER:CONT:RTS 1 | Confirmed                 |

## **:SYST:COMM:SER:CONT:XONX**

Applicability: PACE1000, PACE5000, PACE6000

### Command Syntax

#### **:SYSTEM:COMMunicate:SERial:CONTrol:XONXoff**

Parameter: Integer

Short form: :SYST:COMM:SER:XONX

Function: Sets the serial communication handshaking of:

0: NONE

1: XON/XOFF

2: RTS/CTS

### Query Syntax

#### **:SYSTEM:COMMunicate:SERial:CONTrol:XONXoff?**

Short form: :SYST:COMM:SER:CONT:XONX?

Function: Queries serial communication handshaking.

Response: Returns an integer of 0, 1, 2.

### Description

Selects serial communications interface handshaking. The command query requests the current serial communication handshaking.

#### Example

```
TX> :SYST:COMM:SER:CONT:XONX?
```

```
RX> :SYST:COMM:SER:CONT 0
```

The current hardware handshaking is NONE

It can be set to RTS/CTS by sending:

```
TX> :SYST:COMM:SER:CONT 2
```

The integer of 0, 1, 2 represents:

0: NONE

1: XON/XOFF

2: RTS/CTS



## :SYST:COMM:SER:BAUD

Applicability: PACE1000, PACE5000, PACE6000

### Command Syntax

#### :SYSTem:COMMunicate:SERial:BAUD

Parameter: Integer

Short form: :SYST:COMM:SER:BAUD

Function: Sets the baud rate

**Note:** The parameter must be a valid Baud-rate number.

### Query Syntax

#### :SYSTem:COMMunicate:SERial:BAUD?

Short form: :SYST:COMM:SER:BAUD?

Function: Queries current baud setting.

Response: Current baud setting.

### Description

This command instructs the instrument to set the baud rate. The command query returns the current baud rate set in the instrument.

#### Example

TX> :SYST:COMM:SER:BAUD?

RX> :SYST:COMM:SER:BAUD 9600

The current baud-rate is 9600 bit/second.

It can be set to another baud rate (example: 19200) by sending:

TX> :SYST:COMM:SER:BAUD 19200

**Note:** The parameter must be a valid baud rate number:

|      |       |        |        |
|------|-------|--------|--------|
| 2400 | 9600  | 38400  | 115200 |
| 4800 | 19200 | 576000 |        |

**Note:** Changing to a new baud rate causes a loss of communications until resetting the local PC RS232 to the new baud rate.

## **:SYST:COMM:SER:TYPE:PAR**

Applicability: PACE1000, PACE5000, PACE6000

### Command Syntax

#### **:SYSTem:COMMunicate:SERial[:TYPE]:PARity**

Parameter: Enumerate

Short form: SYST:COMM:SER:TYPE:PAR

Function: Sets parity, odd, even or none.

This command breaks the communication between the PC and the PACE instrument. No further query can be made until after resetting the pc to the same setting. The instrument cannot be brought back to local mode.

### Query Syntax

#### **:SYSTem:COMMunicate:SERial:TYPE:PARity?**

Short form: :SYST:COMM:SER:TYPE:PAR?

Function: Queries current parity setting.

Response: Returns odd, even or none.

### Description

This command instructs the instrument to set serial communication parity. The command query gets the current parity setting.

#### Example

TX> :SYST:COMM:SER:TYPE:PAR?

RX> :SYST:COMM:SER:TYPE:PAR NONE

The current setting ignores the parity check.

It can be set to ODD and EVEN by sending:

TX> :SYST:COMM:SER:TYPE:PAR ODD

or

TX> :SYST:COMM:SER:TYPE:PAR EVEN

## **:SYST:COMM:GPIB:SELF:ADDR**

Applicability: PACE1000, PACE5000, PACE6000

### Command Syntax

#### **:SYSTem:COMMunicate:GPIB[:SELF]:ADDRess**

Parameter: Integer

Short form: :SYST:COMM:GPIB:SELF:ADDR

Function: Sets the instrument's GPIB address.

### Query Syntax

#### **:SYSTem:COMMunicate:GPIB:SELF:ADDRess?**

Short form: :SYST:COMM:GPIB:SELF:ADDR?

Function: Queries GPIB address

Response: Returns decimal number representing GPIB address.

### Description

This command instructs the instrument to set a GPIB address. The query command gets the current address.

Example:

```
:TX> :SYST:COMM:GPIB:SELF:ADDR?  
RX> :SYST:COMM:GPIB:SELF:ADDR 1      Current GPIB address is 1  
TX> :SYST:COMM:GPIB:SELF:ADDR 16    Set address to 16  
TX> :SYST:COMM:GPIB:SELF:ADDR?  
RX> :SYST:COMM:GPIB:SELF:ADDR 16
```

## :SYST:AREA

Applicability: PACE1000, PACE5000, PACE6000

### Command Syntax

**:SYSTem:AREA<space><parameter>**

Parameter: String, see below

Form: :SYST:AREA<space><string>

Response: None

Function: Sets a group of parameters for an area of the world.

### Query Syntax

**:SYSTem:AREA?**

Form: :SYST:AREA?

Function: Queries area setting.

Response: :SYST:AREA<space><string>

| String A* | String B* | Remarks           |
|-----------|-----------|-------------------|
| EUROPE    | EUR       |                   |
|           | USA       |                   |
| JAPAN     | JAP       |                   |
| ASIA      |           |                   |
|           | ROW       | Rest of the World |

\* String A or String B may be used for the parameter value.

### Description

This command instructs the instrument to enable a group of default settings for an area of the world.

Example:

| Tx             | Rx             | Remarks                          |
|----------------|----------------|----------------------------------|
| :SYST:AREA?    | :SYST:AREA EUR | Area of use is set to Europe.    |
| :SYST:AREA USA | None           | Sets the area of use to the USA. |
| :SYST:AREA?    | :SYST:AREA USA | Area of use is set to the USA.   |

| <b>Tx</b>           | <b>Rx</b>         | <b>Remarks</b>                                    |
|---------------------|-------------------|---------------------------------------------------|
| :SYST:AREA<br>JAPAN | None              | Sets the area of use to Japan.                    |
| :SYST:AREA?         | :SYST:AREA JAP    | Area of use is set to Japan.                      |
| :SYST:AREA ASIA     | None              | Sets the area of use to Asia.                     |
| :SYST:AREA?         | :SYST:AREA ASIA   | Area of use is set to ASIA.                       |
| :SYST:AREA<br>ROW   | None              | Sets the area of use to the rest of the<br>World. |
| :SYST:AREA?         | :SYST:AREA<br>ROW | Area of use is set to the rest of the<br>World.   |

## **:SYST:PASS:CDIS**

Applicability: PACE1000, PACE5000, PACE6000

### Command Syntax

#### **:SYSTEM:PASSWORD:CDISable**

Parameter: 2317100

Short form: SYST:PASS:CDIS<space>[parameter]

Function: Disables the calibration with a password.

### Query Syntax

**n/a**

Short form:

Function:

Response:

### Description

This command disables calibration mode with the parameter above used as a password. To enable calibration mode refer to commands “:SYST:PASS:CED” and “:SYST:PASS:CEN:STAT:”.

### Example

TX> :SYST:PASS:CEN:STAT?

RX> :SYST:PASS:CEN:STAT 1

TX> :SYST:PASS:CDIS 2317100

RX> None

TX> :SYST:PASS:CEN:STAT?

RX> :SYST:PASS:CEN:STAT 0

## **:SYST:PASS:CEN**

Applicability: PACE1000, PACE5000, PACE6000

### Command Syntax

#### **:SYSTem:PASSword:CENable:**

Parameter: 2317100

Short form: :SYST:PASS:CEN<space>[parameter]

Function: Enables calibration, default condition - disabled (calibration is not allowed).

### Query Syntax

**n/a**

Short form:

Function:

Response:

### Description

This command enables calibration mode with the above parameter used as a password. To disable calibration mode refer to commands “:SYST:PASS:CDIS” and “:SYST:PASS:CEN:STAT?”.

Example:

```
TX> :SYST:PASS:CEN:STAT?
RX> :SYST:PASS:CEN:STAT 0
TX> :SYST:PASS:CEN 2317100
RX> None
TX> :SYST:PASS:CEN:STAT?
RX> :SYST:PASS:CEN:STAT 1
TX> :SYST:PASS:CDIS 2317100
RX> None
TX> :SYST:PASS:CEN:STAT?
RX> :SYST:PASS:CEN:STAT 0
```

## **:SYST:PASS:CEN:STAT**

Applicability: PACE1000, PACE5000, PACE6000

### Command Syntax

**n/a**

Parameter:

Short form:

Function:

### Query Syntax

#### **:SYSTem:PASSword:CENable:STATe?**

Short form: :SYST:PASS:CEN:STAT?

Function: Gets the status of the calibration password.

Response: 0 - disabled  
1 - enabled

### Description

This query asks if the calibration is enabled or disabled.

#### Example

It can be enabled by sending command and password

```
TX> :SYST:PASS:CEN 2317100
TX> :SYST:PASS:CEN:STAT?
RX> :SYST:PASS:CEN:STAT 1
```

When enabled and a 3-point calibration can be carried out.

It can be disabled again by issuing a command and a password, such as:

```
TX> :SYST:PASS:CDIS 2317100
TX> :SYST:PASS:CEN:STAT?
RX> :SYST:PASS:CEN:STAT 0
```



## **:SYST:VERS**

Applicability: PACE1000, PACE5000, PACE6000

### **Command Syntax**

**n/a**

Parameter:

Short form:

Function:

### **Query Syntax**

#### **:SYStem:VERSiOn?**

Short form: :SYST:VERS?

Function: Returns the SCPI version.

Response: version number

### **Description**

Replies with the SCPI version number.

Example

TX> :SYST:VERS?

RX> :SYST:VERS 1995.0

# UNIT

The UNIT sub-system configures the instrument's pressure measurement units.

## :UNIT:ALT

Applicability: PACE1000, PACE6000

### Command Syntax

#### **:UNIT:ALTitude**

Parameter: <name>

Short form: :UNIT:ALT

Function: Selects altitude units.

### Query Syntax

#### **:UNIT:ALTitude?**

Short form: :UNIT:ALT?

Function: Queries what altitude units are selected

Response: SUFFIX DATA

### Description

This command selects the altitude units. The query gets the altitude units are selected.

**Note:** If the instrument is not in aero mode it returns an error: -221, "Settings conflict".

---

## **:UNIT:CONV**

Applicability: PACE1000, PACE6000

### **Command Syntax**

#### **:UNIT:CONVert**

Parameter: <name>

Short form: :UNIT:CONV

Function: Converts units.

### **Query Syntax**

**n/a**

Short form:

Function:

Response:

### **Description**

This command converts units.

## :UNIT:PRES

Applicability: PACE1000, PACE5000, PACE6000

### Command Syntax

**:UNIT[x][:PRESsure] <name>**

where x = 1 or 2 is the module number (default - 1)

Parameter: <name>

Valid units: Refer to :INStrument:UNIT for valid unit list.

**Note:** Take care when entering the units, the letter 'O' can easily be mistaken for the number '0' or vice versa.

Short form: :UNIT <name>

Function: Selects pressure units

### Query Syntax

**:UNIT[x][:PRESsure]?**

where x = 1 or 2 is the module number (default - 1)

Short form: :UNIT?

Function: Query what pressure units are selected

Response: name as above

### Description

This command selects the current pressure units; USER1 to USER4 are the user defined units.

#### Example

TX> :UNIT:PRES?

RX> :UNIT:PRES MBAR

The current unit is mbar

It can be set to another unit, such as bar by sending:

TX> :UNIT:PRES bar

TX> :UNIT:PRES?

RX> :UNIT:PRES BAR

The reply to this query confirms the change, the front panel display can also confirm the change.

## :UNIT:PRES:DEF

Applicability: PACE1000, PACE5000, PACE6000

### Command Syntax

**:UNIT[x][:PRESsure]:DEFine[y] <string>,<number>**

where:  $x = 1$  or  $2$  is the module number (default - 1) and  $y = 1, 2, 3$  or  $4$  is the number of user defined units (default - 1).

Parameter: <string> ASCII representation of unit name.

<number> Conversion factor from Pascals to required units

Short form: :UNIT[x]DEF[y] <string>,<number>

Function: There are four user defined units on the instrument. This command defines the name and conversion factor to use.

Minimum: string ""

number 0.0

Maximum: string 8characters

number 1.0e10

Default: string N/A

number 1000.0

### Query Syntax

**:UNIT[x][:PRESsure]:DEFine[y]?**

where  $x = 1$  or  $2$  is the module number (default - 1) and  
 $y = 1, 2, 3$  or  $4$  is the number of user defined units  
(default - 1).

1 - USER1

2 - USER2

3 - USER3

4 - USER4

Short form: :UNIT[x]DEF[y]?

Function: Query conversion factor

Response: A string representing name and number corresponding to the conversion factor.

### Description

This command defines the name and conversion factor for the special units. This conversion factor is from Pascals to the required units. An index of 1 is for USER1 unit and an index of 2 is for USER2 unit.

## Chapter 4. Command and Query Summary

---

### 1. Example

y = 1, 2, 3 and 4 representing the four user-defined units.

By default:

```
TX> :UNIT:PRES:DEF?           TX> :UNIT:PRES:DEF3?
RX> :UNIT:PRES:DEF "UserUnit1", 1000.0000000  RX> :UNIT:PRES:DEF3 "UserUnit3", 1000.0000000
TX> :UNIT:PRES:DEF1?           TX> :UNIT:PRES:DEF4?
RX> :UNIT:PRES:DEF "UserUnit1", 1000.0000000  RX> :UNIT:PRES:DEF4 "UserUnit4", 1000.0000000
TX> :UNIT:PRES:DEF2?
RX> :UNIT:PRES:DEF2 "UserUnit2", 1000.0000000
```

The first parameter is the unit name and the second the equivalent pressure in Pascals.

### 2. Example - Defining 'MyUnit'

Setting USER4 as 'MyUnit':

```
TX> :UNIT:PRES:DEF4 "MyUnit", 2000.0
TX> :UNIT:PRES:DEF4?
RX> :UNIT:PRES:DEF4 "MyUnit", 2000.0000000
```

Select 'MyUnit' for use:

```
TX> :UNIT:PRES USER4
TX> :UNIT:PRES?
RX> :UNIT:PRES USER4
```

The front panel display shows "MyUnit" and uses it for measurement, set-point, full-scales and source pressures.

### 3. Example

The following query:

```
TX> :SENS:PRES?
RX> :SENS:PRES 5.0000187
```

Returns a number in 'MyUnit'

To confirm:

```
TX> :UNIT:PRES?
RX> :UNIT:PRES USER4
```

To find conversion factor for 'MyUnit'

```
TX> :UNIT:PRES:DEF4?
RX> :UNIT:PRES:DEF4 "MyUnit", 2000.0000000
```

Calculate equivalent in Pascals: 5.0000187 multiplied by 2000.0000000.

---

## :UNIT:SPE

Applicability: PACE1000, PACE6000

### Command Syntax

#### :UNIT:SPEd

Parameter: <name>

Short form: :UNIT:SPE

Function: Selects airspeed units.

### Query Syntax

#### :UNIT:SPEd?

Short form: :UNIT:SPE?

Function: Queries the selected airspeed units.

Response: SUFFIX DATA

### Description

This command selects the airspeed units. The query gets the selected airspeed units.

### 4.2 Common Commands

The commands identified with \* are SCPI common commands.

#### **\*CLS**

##### Command Syntax

###### **\*CLS**

Parameter: None  
Short form: \*CLS  
Function: This command clears the all queues.

##### Query Syntax

**n/a**

Parameter:  
Function:  
Response:

##### **Description**

Clears all event and condition register, see status reporting section.



## **\*ESE**

### **Command Syntax**

#### **\*ESE <integer>**

|             |                                                 |
|-------------|-------------------------------------------------|
| Parameter:  | integer 8 bit value of enable mask              |
| Short form: | *ESE <integer>                                  |
| Function:   | Sets the Standard Event Status enable register. |
| Minimum:    | 0                                               |
| Maximum:    | 255                                             |
| Default:    | 0                                               |

### **Query Syntax**

#### **\*ESE?**

|             |                                                                     |
|-------------|---------------------------------------------------------------------|
| Short form: | *ESE?                                                               |
| Function:   | Query Standard Event Status Enable register.                        |
| Response:   | 8 bit integer of contents of Standard Event Status Enable register. |

### **Description**

See Standard Event Group, Section 3.2 and Figure 3-1.

## **\*ESR**

### **Command Syntax**

**n/a**

Parameter:

Short form:

Function:

### **Query Syntax**

**\*ESR?**

Short form:           \*ESR?

Function:             Queries the Standard Event Status Register

Response:            8 bit integer of contents of Standard Event Status register.

### **Description**

See Standard Event Group, Section 3.2 and Figure 3-1.

## **\*IDN?**

### **Command Syntax**

**n/a**

Parameter:

Short form:

Function:

### **Query Syntax**

**\*IDN?**

Short form:            \*IDN?

Function:             Queries the identification of the instrument.

Response:            A comma separated list containing manufacture, model, serial number and software version.

### **Description**

Return identification

e.g., \*IDN Druck,PACEnnnn,1234,01.00.00

*where: n = PACE model number*

## \*OPC

### Command Syntax

#### \*OPC

Parameter: None

Short form: \*OPC

Function: When all operations have been completed, event 800 is written to the error/event queue (if enabled by STAT:QUE:ENAB). At the same time, this command allows the OPC bit in the standard event register to be set. This can be used to trigger a serial poll, see \*ESR for details. If there are no pending operations these actions are performed immediately. This command should be sent each time the OPC bit is allowed to be set.

Default: -

**Note:** It is intended that the \*OPC command should be the last command in a program message.

### Query Syntax

#### \*OPC?

Short form: \*OPC?

Function: When all operations have been completed a '1' is written to the output queue this can be used to trigger a serial poll, see \*SRE for details.

Response: If there are no pending operations a '1' is written to the queue immediately.

**Note:** It is intended that the \*OPC command should be the last command in a program message.

### Description

This command sets the OPC bit in the standard event register. The query returns '1' when all operations have been completed.

## **\*SRE**

### **Command Syntax**

#### **\*SRE <integer>**

|             |                                           |                            |
|-------------|-------------------------------------------|----------------------------|
| Parameter:  | integer                                   | 8 bit value of enable mask |
| Short form: | *SRE <integer>                            |                            |
| Function:   | Sets the Service Request Enable register. |                            |
| Minimum:    | 0                                         |                            |
| Maximum:    | 255                                       |                            |

### **Query Syntax**

#### **\*SRE?**

|             |                                                               |
|-------------|---------------------------------------------------------------|
| Short form: | *SRE?                                                         |
| Function:   | Query Service Request Enable register.                        |
| Response:   | 8 bit integer of contents of Service Request Enable register. |

### **Description**

See Status Byte Group, Section 3.4 and Figure 3-1.

## **\*STB?**

### **Command Syntax**

**n/a**

Parameter:

Short form:

Function:

### **Query Syntax**

**\*STB?**

Short form:           \*STB?

Function:             Queries the Status Register

Response:            8 bit integer of contents of Status register.

### **Description**

See status reporting section.

## **\*TST?**

### **Command Syntax**

**n/a**

Parameter:

Short form:

Function:

### **Query Syntax**

#### **\*TST?**

Short form:            \*TST?

Function:             Queries the self-test status.

Response:            Returns '0' indicates fail, '1' indicates pass.

### **Description**

On a self-test failure an error message can be stored in the error/event queue. The unit's hardware status cannot be changed during that test.

# **\*WAI**

### **Command Syntax**

#### **\*WAI**

|             |                                                                                                                                                    |
|-------------|----------------------------------------------------------------------------------------------------------------------------------------------------|
| Parameter:  | None                                                                                                                                               |
| Short form: | *WAI                                                                                                                                               |
| Function:   | No further commands are carried out until the completion of all pending operations. This command is ignored if all operations have been completed. |

### **Query Syntax**

#### **n/a**

Short form:

Function:

Response:

### **Description**

This command stops further commands from being carried out until the completion of all pending operations.



## 4.3 Instrument Control Commands

The following commands identified with : are SCPI instrument control commands.

### **:GTL**

Applicability: PACE1000, PACE5000, PACE6000

#### Command Syntax

##### **:GTL**

Parameter: none

Short form: :GTL

Function: Go to local

#### Query Syntax

**n/a**

Short form:

Function:

Response:

#### Description

Takes the instrument out of local lockout mode; the key-pad on the instrument becomes active.

### **:LLO**

Applicability: PACE1000, PACE5000, PACE6000

#### **Command Syntax**

##### **:LOC**

Parameter: none

Short form: :LLO

Function: Local lock-out

#### **Query Syntax**

**n/a**

Short form:

Function:

Response:

#### **Description**

Locks the instrument out of local mode. The instrument will no longer be in local mode and cannot be operated from the front panel.

## **:LOC**

Applicability: PACE1000, PACE5000, PACE6000

### **Command Syntax**

#### **:LOC**

Parameter: none  
Short form: LOC  
Function: Local mode

### **Query Syntax**

**n/a**

Short form:  
Function:  
Response:

### **Description**

Puts the instrument into local mode. The instrument will no longer be in remote mode and can be operated from the front panel.

### **:REM**

Applicability: PACE1000, PACE5000, PACE6000

#### **Command Syntax**

##### **:REM**

Parameter: none

Short form: :REM

Function: Selects remote mode

#### **Query Syntax**

**n/a**

Short form:

Function:

Response:

#### **Description**

Sets the instrument in remote mode. The instrument will no longer be in local mode and cannot be operated from the front panel.

## **:SRQ**

Applicability: PACE1000, PACE5000, PACE6000

### **Command Syntax**

#### **:SRQ**

Parameter: none

Short form: :SRQ

Function: Sends a service request to read messages.

### **Query Syntax**

**n/a**

Short form:

Function:

Response:

### **Description**

A service request causes all messages to be read from the register.

# **:SRQ:ENAB**

Applicability: PACE1000, PACE5000, PACE6000

### Command Syntax

#### **:SRQ:ENABLE**

Parameter: none

Short form: :SRQ:ENAB

Function: Sets a service request to read messages.

### Query Syntax

#### **:SRQ:ENABLE?**

Short form: :SRQ:ENAB?

Function: Gets the messages in the register.

### Description

A service request causes all messages to be read from the register.

---

## :SYN

Applicability: PACE1000, PACE6000

### Command Syntax

**n/a**

Parameter:

Short form:

Function:

### Query Syntax

**:SYN?**

Short form: :SYN?

Function: Queries synchronisation.

Response: <boolean>  
          0 - not synchronised  
          1 - synchronised

### Description

Query for synchronisation purposes.





## 5. Errors

Negative error numbers are used for standard SCPI errors. Positive error numbers are device specific errors. Following the error number, a message describes the error. An error, when detected, is held in the error queue.

When SYST:ERR? is sent any error in the error queue sets the error bit in the event status register.

### 5.1 Error Numbers

Each error detected causes an error number with an error message to be returned as follows:

| Error code | Error message                       |
|------------|-------------------------------------|
| -102       | -102, "Syntax error"                |
| -104       | -104, "Data type error"             |
| -108       | -108, "Parameter not allowed"       |
| -109       | -109, "Missing parameter"           |
| -110       | -110, "Command Header Error"        |
| -111       | -111, "Header Separator Error"      |
| -112       | -112, "Program mnemonic too long"   |
| -113       | -113, "Undefined header"            |
| -114       | -114, "Header suffix out of range"  |
| -120       | -120, "Numeric data error"          |
| -121       | -121, "Invalid character in number" |
| -123       | -123, "Exponent too large"          |
| -124       | -124, "Too many digits"             |
| -128       | -128, "Numeric data not allowed"    |
| -130       | -130, "Suffix error"                |
| -131       | -131, "Invalid suffix"              |
| -134       | -134, "Suffix too long"             |
| -138       | -138, "Suffix not allowed"          |
| -140       | -140, "Character data error"        |
| -141       | -141, "Invalid character data"      |
| -144       | -144, "Character data too long"     |
| -148       | -148, "Character data not allowed"  |
| -150       | -150, "String data error"           |
| -151       | -151, "Invalid string data"         |
| -158       | -158, "String data not allowed"     |

**Table 5-1: Errors -100 to -199**

| Error code | Error message                    |
|------------|----------------------------------|
| -200       | -200, "Execution error"          |
| -201       | -201, "Invalid while in local"   |
| -202       | -202, "Settings lost due to rtl" |
| -203       | -203, "Command protected"        |
| -220       | -220, "Parameter error"          |
| -221       | -221, "Settings conflict"        |
| -222       | -222, "Data out of range"        |
| -223       | -223, "Too much data"            |
| -224       | -224, "Illegal parameter value"  |
| -240       | -240, "Hardware error"           |
| -241       | -241, "Hardware missing"         |

**Table 5-2: Errors -200 to -299**

| Error code | Error message          |
|------------|------------------------|
| -310       | -310, "System error"   |
| -350       | -350, "Queue overflow" |
| -400       | -400, "Query error"    |

**Table 5-3: Errors -300 to -399**

| Error code | Error message                        |
|------------|--------------------------------------|
| 201        | 201, "Query only"                    |
| 202        | 202, "No query allowed"              |
| 203        | 203, "Parameter(s) not expected"     |
| 207        | 207, "Enumerated value not in union" |
| 208        | 208, "Illegal number of parameters"  |
| 210        | 210, "Run out of memory handle"      |
| 211        | 211, "Unit not matched"              |
| 212        | 212, "Unit not required"             |

**Table 5-4: Errors +201 to +212**





## Office Locations

### Australia

**Springfield Central**

Phone: 1300 171 502

Email: [custcare.au@ge.com](mailto:custcare.au@ge.com)

### India

**Bangalore**

Phone: 1-800-301-62632

Email: [cc.ms.india@bhge.com](mailto:cc.ms.india@bhge.com)

### Netherlands

**Hoevelaken**

Phone: +31 334678950

Email: [NL.sensing.sales@bhge.com](mailto:NL.sensing.sales@bhge.com)

### USA

**Boston**

Phone: 1-800-833-9438

Email: [custcareboston@bhge.com](mailto:custcareboston@bhge.com)

### France

**Toulouse**

Phone: +33 562 888 250

Email: [sensing.FR.cc@ge.com](mailto:sensing.FR.cc@ge.com)

### Italy

**Milan**

Phone: +39 02 36 04 28 42

Email: [mariangela.scarati@bhge.com](mailto:mariangela.scarati@bhge.com)

### Russia

**Moscow**

Phone: +7 495 739 6811

Email: [aleksey.khamov@bhge.com](mailto:aleksey.khamov@bhge.com)

### Germany

**Frankfurt**

Phone: +49 (0) 69-22222-973

Email: [sensing.de.cc@ge.com](mailto:sensing.de.cc@ge.com)

### Japan

**Chuo-ku**

Tel: 03-6890-4538

Email: [gesensing.japan@bhge.com](mailto:gesensing.japan@bhge.com)

### UK

**Leicester**

Phone: +44 (0) 116 2317233

Email: [gb.sensing.sales@bhge.com](mailto:gb.sensing.sales@bhge.com)

## Services and Support Locations

### Tech Support

**Global**

Email: [mstechsupport@bhge.com](mailto:mstechsupport@bhge.com)

### France

**Toulouse**

Phone: +33 562 888 250

Email: [sensing.france.services@ge.com](mailto:sensing.france.services@ge.com)

### UAE

**Abu Dhabi**

Phone: +971 2 4079381

Email: [gulfservices@ge.com](mailto:gulfservices@ge.com)

### Brazil

**Campinas**

Phone: +55 11 3958 0098

Email: [mcs.services@ge.com](mailto:mcs.services@ge.com)

### India

**Pune**

Phone: +91-2135-620421 to 425

Email: [mcsindia.inhouseservice@ge.com](mailto:mcsindia.inhouseservice@ge.com)

### UK

**Leicester**

Phone: +44 (0) 116 2317674

Email: [sensing.grobycc@bhge.com](mailto:sensing.grobycc@bhge.com)

### China

**Changzhou**

Phone: +86 (0) 519-83051779-3

Email: [service.mcchina@ge.com](mailto:service.mcchina@ge.com)

### Japan

**Niigata**

Phone: +81 257 45 5509

Email: [kariwa.LScenter@ge.com](mailto:kariwa.LScenter@ge.com)

### USA

**Billerica**

Phone: 1-800-833-9438

Email: [service.boston@ge.com](mailto:service.boston@ge.com)

[bhge.com](http://bhge.com)